TMS320F28003x Real-Time Microcontrollers

Technical Reference Manual

Literature Number: SPRUIW9A
OCTOBER 2021 – REVISED MARCH 2022

Table of Contents

List of Figures

List of Tables

About This Manual

This Technical Reference Manual (TRM) details the integration, the environment, the functional description, and the programming models for each peripheral and subsystem in the device.

The TRM should not be considered a substitute for the data manual, rather a companion guide that should be used alongside the device-specific data manual to understand the details to program the device. The primary purpose of the TRM is to abstract the programming details of the device from the data manual. This allows the data manual to outline the high-level features of the device without unnecessary information about register descriptions or programming models.

Notational Conventions

This document uses the following conventions.

- Hexadecimal numbers can be shown with the suffix h or the prefix 0x. For example, the following number is 40 hexadecimal (decimal 64): 40h or 0x40.
- Registers in this document are shown in figures and described in tables.
  - Each register figure shows a rectangle divided into fields that represent the fields of the register. Each field is labeled with its bit name, its beginning and ending bit numbers above, and its read/write properties with default reset value below. A legend explains the notation used for the properties.
  - Reserved bits in a register figure can have one of multiple meanings:
    - Not implemented on the device
    - Reserved for future device expansion
    - Reserved for TI testing
    - Reserved configurations of the device that are not supported
  - Writing nondefault values to the Reserved bits could cause unexpected behavior and should be avoided.

Glossary

TI Glossary This glossary lists and explains terms, acronyms, and definitions.

Related Documentation From Texas Instruments

For a complete listing of related documentation and development-support tools for these devices, visit the Texas Instruments website at www.ti.com.

Additionally, the *TMS320C28x DSP CPU and Instruction Set Reference Guide* and the *TMS320C28x Floating Point Unit and Instruction Set Reference Guide* must be used in conjunction with this TRM.

Support Resources

TI E2E™ support forums are an engineer's go-to source for fast, verified answers and design help — straight from the experts. Search existing answers or ask your own question to get the quick design help you need.

Linked content is provided "AS IS" by the respective contributors. They do not constitute TI specifications and do not necessarily reflect TI's views; see TI's Terms of Use.

Trademarks

TI E2E™, C2000™, Code Composer Studio™, and Texas Instruments™ are trademarks of Texas Instruments. All trademarks are the property of their respective owners.

This page intentionally left blank.

Chapter 1

C2000™ Microcontrollers Software Support

This chapter discusses the C2000™ microcontrollers.

1.1 Introduction

C2000Ware for the C2000™ microcontrollers is a cohesive set of development software and documentation designed to minimize software development time. From device-specific drivers and libraries to device peripheral examples, C2000Ware provides a solid foundation to begin development and evaluation of your product.

C2000Ware can be downloaded from: www.ti.com/tool/C2000WARE

1.2 C2000Ware Structure

The C2000Ware software package is organized into the following directory structure as shown in Table 1-1.

Table 1-1. C2000Ware Root Directories

Directory Name	Description
boards	Contains the hardware design schematics, BOM, Gerber files, and documentation for C2000 controlCARDS.
device_support	Contains all device-specific support files, bit field headers and device development user's guides.
docs	Contains the C2000Ware package user's guides and the HTML index page of all package documentation.
driverlib	Contains the device-specific driver library and driver-based peripheral examples.
libraries	Contains the device-specific and core libraries.

1.3 Documentation

Within C2000Ware, there is an extensive amount of development documentation ranging from board design documentation, to library user's guides, to driver API documentation. The "boards" directory contains all the hardware design, BOM, Gerber files, and more for controlCARDs. To assist with locating the necessary documentation, an HTML page is provided that contains a full list of all the documents in the C2000Ware package. Locate this page in the "docs" directory.

1.4 Devices

C2000Ware contains the necessary software and documentation to jumpstart development for C2000™ microcontrollers. Each device includes device-specific common source files, peripheral example projects, bit field headers, and if available, a device peripheral driver library. Additionally, documentation is provided for each device on how to set up a CCS project, as well as give an overview of all the included example projects and assist with troubleshooting. For devices with a driver library, documentation is also included that details all the peripheral APIs available.

To learn more about C2000™ microcontrollers, visit: www.ti.com/c2000.

1.5 Libraries

The libraries included in C2000Ware range from fixed-point and floating-point math libraries, to specialized DSP libraries, as well as calibration libraries. Each library includes documentation and examples, where applicable. Additionally, the Flash API files and boot ROM source code are located in the "libraries" directory.

1.6 Code Composer Studio™ Integrated Development Environment (IDE)

Code Composer Studio™ is an integrated development environment (IDE) that supports TI's microcontroller and embedded processors portfolio. The Code Composer Studio™ IDE comprises a suite of tools used to develop and debug embedded applications. The latest version of Code Composer Studio™ IDE can be obtained at: www.ti.com/ccstudio

All projects and examples in C2000Ware are built for and tested with the Code Composer Studio™ IDE. Although the Code Composer Studio™ IDE is not included with the C2000Ware installer, it is easily obtainable in a variety of versions.

1.7 SysConfig and PinMUX Tool

To help simplify configuration challenges and accelerate software development, Texas Instruments™ created SysConfig, an intuitive and comprehensive collection of graphical utilities for configuring pins, peripherals, subsystems, and other components. SysConfig helps you manage, expose, and resolve conflicts visually so that you have more time to create differentiated applications.

The tool's output includes C header and code files that can be used with C2000Ware examples or used to configure custom software.

The SysConfig tool automatically selects the pinmux settings that satisfy the entered requirements. The SysConfig tool is delivered integrated in the Code Composer Studio™ IDE, in the C2000Ware GPIO example, as a standalone installer, or can be used by way of the cloud tools portal at: dev.ti.com

This page intentionally left blank.

Chapter 2

C28x Processor

This chapter contains a short description of the C28x processor and extended instruction sets.

Further information can be found in the following documents:

- *TMS320C28x CPU and Instruction Set Reference Guide*
- *TMS320C28x Extended Instruction Sets Technical Reference Manual*
- *Accelerators: Enhancing the Capabilities of the C2000 MCU Family Technical Brief*
- *TMS320C28x FPU Primer Application Report*

2.1 Introduction

The C28x CPU is a 32-bit fixed-point processor. This device draws from the best features of digital signal processing, reduced instruction set computing (RISC), microcontroller architectures, firmware, and tool sets.

For more information on CPU architecture and instruction set, see the *TMS320C28x CPU and Instruction Set Reference Guide*.

2.2 C28X Related Collateral

Foundational Materials

- C2000 Academy - Introduction
- C2000 C28x Optimization Guide
- C2000 Software Guide
- *Enhancing the Computational Performance of the C2000 Microcontroller Family Application Report*

Getting Started Materials

- C2000 Multicore Development User Guide
- C2000Ware - CLAMath
- C2000Ware - FPU Fast RTS
- C2000Ware - FPU Library
- C2000Ware - Fast Integer Division
- C2000Ware - Fixed Point Library
- C2000Ware - IQMath
- C2000Ware - VCU Library
- C2000Ware Libraries Overview
- *CRC Engines in C2000 Devices Application Report*
- *TMS320C28x Extended Instruction Sets Application Report*
- *TMS320C28x FPU Primer Application Report*

Expert Materials

- *Fast Integer Division - A Differentiated Offering From C2000 Product Family Application Report*

2.3 Features

The CPU features include a modified Harvard architecture and circular addressing. The RISC features are single-cycle instruction execution, register-to-register operations, and modified Harvard architecture. The microcontroller features include ease of use through an intuitive instruction set, byte packing and unpacking, and bit manipulation. The modified Harvard architecture of the CPU enables instruction and data fetches to be performed in parallel. The CPU can read instructions and data while it writes data simultaneously to maintain the single-cycle instruction operation across the pipeline.

2.4 Floating-Point Unit

The C28x plus floating-point (C28x+FPU) processor extends the capabilities of the C28x fixed-point CPU by adding registers and instructions to support IEEE single-precision and double-percision floating point operations.

Devices with the C28x+FPU include the standard C28x register set plus an additional set of floating-point unit registers. The additional floating-point unit registers are the following:

- Eight floating-point result registers, RnH (where n = 0–7)
- Floating-point Status Register (STF)
- Repeat Block Register (RB)

All of the floating-point registers, except the repeat block register, are shadowed. This shadowing can be used in high-priority interrupts for fast context save and restore of the floating-point registers.

Fast Integer Division (FINTDIV) supports Truncated, Modulo and Euclidean division formats without cycle penalty and provides results in integer and remainder representation.

For more information, see the *TMS320C28x Extended Instruction Sets Technical Reference Manual*.

2.5 Trigonometric Math Unit

The TMU extends the capabilities of a C28x+FPU by adding instructions and leveraging existing FPU instructions to speed up the execution of common trigonometric and arithmetic operations listed in Table 2-1.

Table 2-1. TMU Supported Instructions

Instruction	C Equivalent Operation	Pipeline Cycles
MPY2PIF32 RaH,RbH	a = b * 2pi	2/3
DIV2PIF32 RaH,RbH	a = b / 2pi	2/3
DIVF32 RaH,RbH,RcH	a = b/c	5
SQRTF32 RaH,RbH	a = sqrt(b)	5
SINPUF32 RaH,RbH	a = sin(b*2pi)	4
COSPUF32 RaH,RbH	a = cos(b*2pi)	4
ATANPUF32 RaH,RbH	a = atan(b)/2pi	4
QUADF32 RaH,RbH,RcH,RdH	Operation to assist in calculating ATANPU2	5

Exponent instruction IEXP2F32 and logarithmic instruction LOG2F32 have been added to support computation of floating-point power function for the non-linear proportional integral derivative control (NLPID) component of the C2000 Digital Control Library. These two added instructions reduce the power function calculations from a typical of 300 cycles using library emulation to less than 10 cycles.

No changes have been made to existing instructions, pipeline or memory bus architecture. All TMU instructions use the existing FPU register set (R0H to R7H) to carry out their operations.

For more information, see the *TMS320C28x Extended Instruction Sets Technical Reference Manual*.

2.6 VCRC Unit

Cyclic redundancy check (CRC) algorithms provide a straightforward method for verifying data integrity over large data blocks, communication packets, or code sections. The C28x+VCRC can perform 8-bit, 16-bit, 24-bit, and 32-bit CRCs. For example, the VCU can compute the CRC for a block length of 10 bytes in 10 cycles. A CRC result register contains the current CRC, which is updated whenever a CRC instruction is executed.

The following are the CRC polynomials used by the CRC calculation logic of VCRC:

- CRC8 polynomial = 0x07
- CRC16 polynomial1 = 0x8005
- CRC16 polynomial2 = 0x1021
- CRC24 polynomial = 0x5d6dcb
- CRC32 polynomial1 = 0x04c11db7
- CRC32 polynomial2 = 0x1edc6f41

This module can calculate CRCs for a byte of data in a single cycle. The CRC calculation for CRC8, CRC16, CRC24 and CRC32 is done byte-wise (instead of computing on a complete 16-bit or 32-bit data read by the C28x core) to match the byte-wise computation requirement mandated by various standards.

The VCRC Unit also allows the user to provide the size (1b-32b) and value of any polynomial to fit custom CRC requirements. The CRC execution time increases to **three cycles** when using a custom polynomial.

For more information, see the *TMS320C28x Extended Instruction Sets Technical Reference Manual*.

This page intentionally left blank.

Chapter 3

System Control and Interrupts

The system-level functionality of this microcontroller configures the clocking, resets, and interrupts of the CPU and peripherals, as well as the operation of the on-chip memories, timers, and security features.

3.1 Introduction

System-level configuration is controlled by a group of submodules that are collectively referred to as the system control module. The system control module provides the following capabilities:

- System-level resets, including power-on and brownout resets
- Clock source selection and PLL configuration
- Missing clock detection
- Clock-gating low-power modes
- Peripheral interrupt handling
- Non-maskable interrupts for certain fault conditions
- Three 32-bit timers
- Windowed watchdog timer, which can generate an interrupt or a reset
- RAM initialization, write protection, and mastership control
- Flash memory ECC, wait state, and cache configuration
- Dual-zone code security module

3.1.1 SYSCTL Related Collateral

Foundational Materials

- *C2000 MCU JTAG Connectivity Debug Application Report*

Getting Started Materials

- C28x Interrupt Nesting
- Debugging JTAG
- *Enhancing Device Security by Using JTAGLOCK Feature Application Report*
- XDS Target Connection Guide

Expert Materials

- *C2000 CPU Memory Built-In Self-Test Application Report*
- *C2000 Memory Power-On Self-Test (M-POST) Application Report*
- *Live Firmware Update With Device Reset on C2000 MCUs Application Report*
- *Live Firmware Update Without Device Reset on C2000 MCUs Application Report*
- *Programming of External Nonvolatile Memory Using SDFlash for TMS320C28x Devices Application Report*
- *Software Phased-Locked Loop (PLL) Design Using C2000 Microcontrollers Application Report*

3.1.2 LOCK Protection on System Configuration Registers

Several system configuration registers are protected from spurious CPU writes by “LOCK” registers. Once these associated LOCK register bits are set, the respective locked registers can no longer be modified by software. See the register descriptions for details.

3.1.3 EALLOW Protection

Some registers in the system are protected from spurious CPU writes by the EALLOW protection mechanism. This uses the special CPU instructions EALLOW and EDIS to enable and disable access to protected registers. The current protection state is given by the EALLOW bit in the CPU ST1 register, as shown in Table 3-1.

Register protection is enabled by default at startup. While protected, all writes to protected registers by the CPU are ignored. Only CPU reads, JTAG reads, and JTAG writes are allowed. If protection is disabled by executing the EALLOW instruction, the CPU is allowed to write freely to protected registers. After modifying registers, they can once again be protected by executing the EDIS instruction to clear the EALLOW bit.

Writes to the clock configuration and peripheral clock enable registers may be disabled until the next reset by writing to special lock registers.

Table 3-1. Access to EALLOW-Protected Registers

EALLOW Bit	CPU Writes	CPU Reads	JTAG Writes	JTAG Reads
0	Ignored	Allowed(1)	Allowed	Allowed
1	Allowed	Allowed	Allowed	Allowed

(1) The EALLOW bit is overridden by way of the JTAG port, allowing full access of protected registers during debug from the Code Composer Studio™ interface.

3.2 Power Management

The TMS320F28003x MCU supports both internal and external VREG selectable using the VREGENZ pin to supply the 1.2v rail. However, not all packages support the external VREG option. For packages that do not support external VREG, the VREGENZ pin is replaced by GPIO39. For more details, see the TMS320F28003x data manual.

3.3 Device Identification and Configuration Registers

The device identification registers and configuration registers provide information on the part number, product family, revision, pin count, qualification status, and feature availability of the device.

All of the device information is part of the DEV_CFG_REGS space. The identification registers are PARTIDL, PARTIDH, and REVID.

A 256-bit Unique ID (UID) is available in UID_REGS. The 256 bits are separated into these registers:

- UID_PSRAND0-5: 192 bits of pseudo-random data
- UID_UNIQUE: 32-bit unique data; the value in this register will be unique across all devices in the same PARTIDH
- UID_CHECKSUM: 32-bit Fletcher checksum of UID_PSRAND0-5 and UID_UNIQUE and calculated as either little- or big-endian during factory testing

3.4 Resets

This section explains the types and effects of the different resets on this device.

3.4.1 Reset Sources

Table 3-2 summarizes the various reset signals and their effect on the device.

Table 3-2. Reset Signals

Reset Source	CPU Core Reset (C28x, FPU, VCU, TMU)	Peripherals Reset	JTAG / Debug Logic Reset	IOs	$\overline{\text{XRS}}$ Output
POR	Yes	Yes	Yes	Hi-Z	Yes
BOR	Yes	Yes	Yes	Hi-Z	Yes
$\overline{\text{XRS}}$ Pin	Yes	Yes	No	Hi-Z	-
$\overline{\text{WDRS}}$	Yes	Yes	No	Hi-Z	Yes
$\overline{\text{NMIWDRS}}$	Yes	Yes	No	Hi-Z	Yes
$\overline{\text{SYSRS}}$ (Debugger Reset)	Yes	Yes	No	Hi-Z	No
$\overline{\text{SCCRESET}}$	Yes	Yes	No	Hi-Z	No
SIMRESET. $\overline{\text{XRS}}$	Yes	Yes	No	Hi-Z	Yes
SIMRESET. $\overline{\text{CPU1RS}}$	Yes	Yes	No	Hi-Z	No
$\overline{\text{HWBISTRS}}$	Yes	No	No	No	No

The resets can be divided into two groups:

- Chip-level resets ($\overline{\text{XRS}}$, POR, BOR, $\overline{\text{WDRS}}$, SIMRESET. $\overline{\text{XRS}}$ and $\overline{\text{NMIWDRS}}$), which reset all or almost all of the device.
- System resets ($\overline{\text{SYSRS}}$, SIMRESET. $\overline{\text{CPU1RS}}$ and $\overline{\text{SCCRESET}}$), which reset a large subset of the device but maintain some system-level configuration.

- Special resets ($\overline{\text{HWBISTRS}}$ and $\overline{\text{TRST}}$), which enable specific device functions.

After a reset, the reset cause register (RESC) is updated with the reset cause. The bits in this register maintain their state across multiple resets. They can only be cleared by a power-on reset (POR) or by writing ones to the RESCCLR register. Some are cleared by the boot ROM as part of its start-up routines.

Many peripheral modules have individual resets accessible through the SOFTPRESx registers. For information about a module's reset state, refer to the chapter for that module.

After any reset, the CPU begins execution from address 0x3FFFC0 (the reset vector), which is in the boot ROM. After running the boot ROM code, the CPU will typically branch to the start of the Flash memory at address 0x80000. For more information on controlling the boot process, see ROM Code and Peripheral Booting.

Note

After a POR, the boot ROMs will clear the M0/M1, LSx, GSx, and message RAMs to ensure that they contain valid ECC.

3.4.2 External Reset ($\overline{\text{XRS}}$)

The external reset ($\overline{\text{XRS}}$) is the main chip-level reset for the device. It resets the CPU, all peripherals and I/O pin configurations, and most of the system control registers. There is a dedicated open-drain pin for $\overline{\text{XRS}}$. This pin may be used to drive reset pins for other ICs in the application, and may itself be driven by an external source. The $\overline{\text{XRS}}$ is driven internally during watchdog, NMI, and power-on resets.

The XRSn bit in the RESC register will be set whenever $\overline{\text{XRS}}$ is driven low for any reason. This bit is then cleared by the boot ROM.

3.4.3 Simulate External Reset (SIMRESET. $\overline{\text{XRS}}$)

In some cases, a user may need to simulate an external reset ($\overline{\text{XRS}}$) in software. This can be done by setting XRSn bit to '1' in SIMRESET register through software. This toggles the $\overline{\text{XRS}}$ pin hence resets the full device (just like an external reset).

After this reset, SIMRESET_XRSn and XRSn bits in the RESC register will be set. Software can read these bits to know the cause of reset and clear the status by writing '1' into corresponding bits in the RESCCLR register.

3.4.4 Power-On Reset (POR)

The power-on reset (POR) circuit creates a clean reset throughout the device during power-up, suppressing glitches on the GPIOs. The $\overline{\text{XRS}}$ pin is held low for the duration of the POR. In most applications, $\overline{\text{XRS}}$ is held low long enough to reset other system ICs, but some applications may require a longer pulse. In these cases, the $\overline{\text{XRS}}$ pin can be driven low externally to provide the correct reset duration. A POR resets everything that $\overline{\text{XRS}}$ does, along with a few other registers – the reset cause register (RESC), the NMI shadow flag register (NMISHDFLG), and the X1 clock counter register (X1CNT). A POR also resets the debug logic used by the JTAG port.

After a POR, the POR and XRSn bits in RESC are set. These bits are then cleared by the boot ROM.

3.4.5 Brown-Out-Reset (BOR)

The brown-out-reset (BOR) is an internal supply voltage supervisor (SVS) circuit which monitors the VDDIO supply for glitches or supply interruptions. If the VDDIO supply voltage drops below operational voltage range, this circuit forces the XRSn pin low until the fault is removed and the supply voltage returns to the minimum operational voltage. A BOR resets everything in the same manner as a POR reset.

The BOR circuit is enabled by default and therefore will always be active during power up or after any type of reset. To disable the BOR circuit, set the BORLVMONDIS bit in the VMONCTL register.

3.4.6 Debugger Reset ($\overline{\text{SYSRS}}$)

During development, it is sometimes necessary to reset the CPU and its peripherals without disconnecting the debugger or disrupting the system-level configuration. To facilitate this, the CPU has its own subsystem reset, which can be triggered by a debugger using Code Composer Studio™ IDE. This reset ($\overline{\text{SYSRS}}$) resets the CPU, its peripherals, many system control registers (including its clock gating and LPM configuration), and all I/O pin configurations.

The $\overline{\text{SYSRS}}$ does not reset the ICEPick debug module, the device capability registers, the clock source and PLL configurations, the missing clock detection state, the PIE vector fetch error handler address, the NMI flags, the analog trims, or anything reset only by a POR (see Section 3.4.4).

3.4.7 Simulate CPU Reset (SIMRESET. $\overline{\text{CPU1RS}}$)

In some cases, a user may need to simulate the CPU reset ($\overline{\text{SYSRS}}$) in software. This can be done by setting CPU1RSn bit to '1' in the SIMRESET register through software. This toggles CPU1.SYSRS signals hence resetting the CPU (just like the debugger reset).

After this reset, SIMRESET_CPU1RSn bit in the RESC register will be set. Software can read this bit to know the cause of reset and clear the status by writing '1' into corresponding bit in RESCCLR register.

3.4.8 Watchdog Reset ($\overline{\text{WDRS}}$)

The device has a watchdog timer that can optionally trigger a reset if it is not serviced by the CPU within a user-specified amount of time. This watchdog reset ($\overline{\text{WDRS}}$) produces an $\overline{\text{XRS}}$ that lasts for 512 INTOSC1 cycles.

After a watchdog reset, the WDRSn and XRSn bits in RESC are set.

3.4.9 Hardware BIST Reset ($\overline{\text{HWBISTRS}}$)

The Hardware Built-In Self-Test (HWBIST) module tests the functionality of the CPU. At the end of the test, it resets the CPU to return it to a working state. This reset ($\overline{\text{HWBISTRS}}$) only affects the CPU itself. The peripherals and system control remain as previously configured. The CPU state is restored in software as part of a special boot ROM flow. For more information on the HWBIST flow, contact your local TI representative.

After a HWBIST reset, the HWBISTn bit in RESC is set. Software can read this bit to know the cause of reset and clear the status by writing '1' into corresponding bit in RESCCLR register.

3.4.10 NMI Watchdog Reset ($\overline{\text{NMIWDRS}}$)

The device has a non-maskable interrupt (NMI) module that detects hardware errors in the system. The NMI module has a watchdog timer that triggers a reset if the CPU does not respond to an error within a user-specified amount of time. This NMI watchdog reset ($\overline{\text{NMIWDRS}}$) produces an $\overline{\text{XRS}}$ that lasts for 512 INTOSC1 cycles.

After an NMI watchdog reset, the NMIWDRSn and XRSn bits in RESC are set.

3.4.11 DCSM Safe Code Copy Reset ($\overline{\text{SCCRESET}}$)

The device has a dual-zone code security module (DCSM) that blocks read access to certain areas of the Flash memory. To facilitate CRC checks and copying of CLA code, TI provides ROM functions to securely access those memory areas. To prevent security breaches, interrupts must be disabled before calling these functions. If a vector fetch occurs in a safe copy or CRC function, the DCSM triggers a reset. This security reset ($\overline{\text{SCCRESET}}$) is similar to a $\overline{\text{SYSRS}}$. However, the security reset also resets the debug logic to deny access to a potential attacker.

After a security reset, the SCCRESETn bit in RESC is set.

3.5 Peripheral Interrupts

This section explains the peripheral interrupt handling on the device. Non-maskable interrupts are covered in Section 3.6. Software interrupts and emulation interrupts are not covered in this document. For information on those, see the *TMS320C28x CPU and Instruction Set Reference Guide*.

3.5.1 Interrupt Concepts

An interrupt is a signal that causes the CPU to pause its current execution and branch to a different piece of code known as an interrupt service routine (ISR). This is a useful mechanism for handling peripheral events, and involves less CPU overhead or program complexity than register polling. However, because interrupts are asynchronous to the program flow, care must be taken to avoid conflicts over resources that are accessed both in interrupts and in the main program code.

Interrupts propagate to the CPU through a series of flag and enable registers. The flag registers store the interrupt until it is processed. The enable registers block the propagation of the interrupt. When an interrupt signal reaches the CPU, the CPU fetches the appropriate ISR address from a list called the vector table.

3.5.2 Interrupt Architecture

The C28x CPU has fourteen peripheral interrupt lines. Two of them (INT13 and INT14) are connected directly to CPU timers 1 and 2, respectively. The remaining twelve are connected to peripheral interrupt signals through the enhanced Peripheral Interrupt Expansion module (ePIE, or PIE as a shortened version). The PIE multiplexes up to sixteen peripheral interrupts into each CPU interrupt line. It also expands the vector table to allow each interrupt to have its own ISR. This allows the CPU to support a large number of peripherals.

An interrupt path is divided into three stages – the peripheral, the PIE, and the CPU. Each stage has its own enable and flag registers. This system allows the CPU to handle one interrupt while others are pending, implement and prioritize nested interrupts in software, and disable interrupts during certain critical tasks.

Figure 3-1 shows the interrupt architecture for this device.

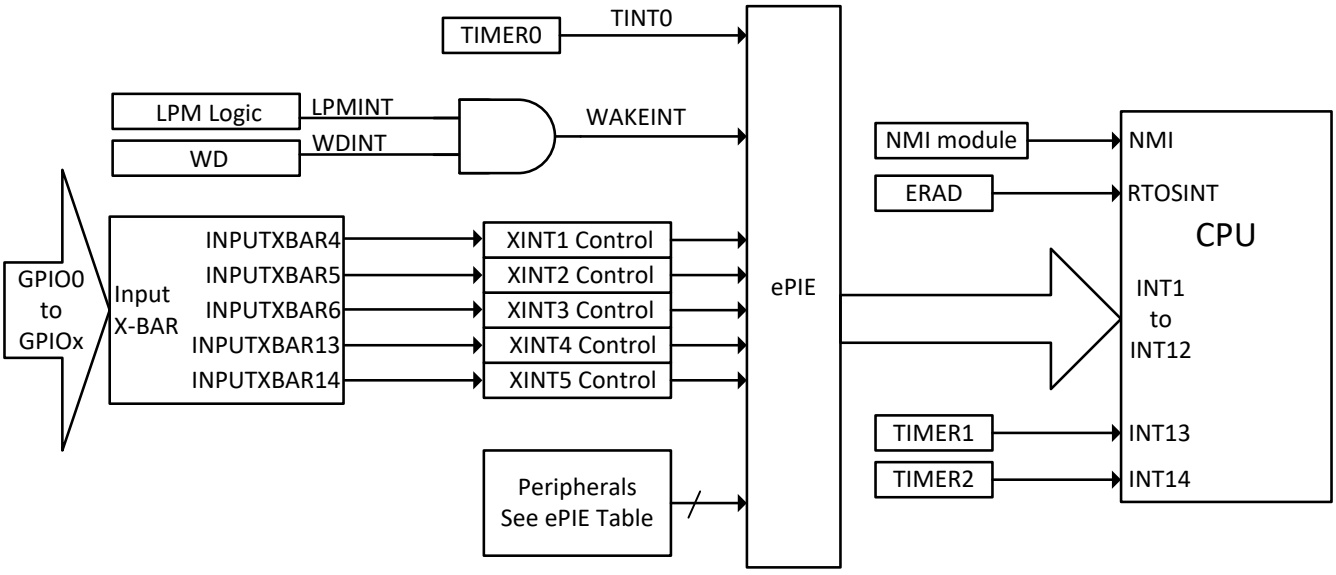


Figure 3-1. Device Interrupt Architecture

3.5.2.1 Peripheral Stage

Each peripheral has its own unique interrupt configuration, which is described in that peripheral's chapter. Some peripherals allow multiple events to trigger the same interrupt signal. For example, a communications peripheral might use the same interrupt to indicate that data has been received or that there has been a transmission error. The cause of the interrupt can be determined by reading the peripheral's status register. Often, the bits in the status register must be cleared manually before another interrupt will be generated.

3.5.2.2 PIE Stage

The PIE provides individual flag and enable register bits for each of the peripheral interrupt signals, which are sometimes called PIE channels. These channels are grouped according to their associated CPU interrupt. Each PIE group has one 16-bit enable register (PIEIERx), one 16-bit flag register (PIEIFRx), and one bit in the PIE acknowledge register (PIEACK). The PIEACK register bit acts as a common interrupt mask for the entire PIE group.

When the CPU receives an interrupt, it fetches the address of the ISR from the PIE. The PIE returns the vector for the lowest-numbered channel in the group that is both flagged and enabled. This gives lower-numbered interrupts a higher priority when multiple interrupts are pending.

If no interrupt is both flagged and enabled, the PIE returns the vector for channel 1. This condition will not happen unless software changes the state of the PIE while an interrupt is propagating. Section 3.5.4 contains procedures for safely modifying the PIE configuration once interrupts have been enabled.

3.5.2.3 CPU Stage

Like the PIE, the CPU provides flag and enable register bits for each of its interrupts. There is one enable register (IER) and one flag register (IFR), both of which are internal CPU registers. There is also a global interrupt mask, which is controlled by the INTM bit in the ST1 register. This mask can be set and cleared using the CPU's SETC and CLRC instructions. In C code, C2000Ware's DINT and EINT macros can be used for this purpose.

Writes to IER and INTM are atomic operations. In particular, if INTM is set, the next instruction in the pipeline will run with interrupts disabled. No software delays are needed.

3.5.3 Interrupt Entry Sequence

Figure 3-2 shows how peripheral interrupts propagate to the CPU.

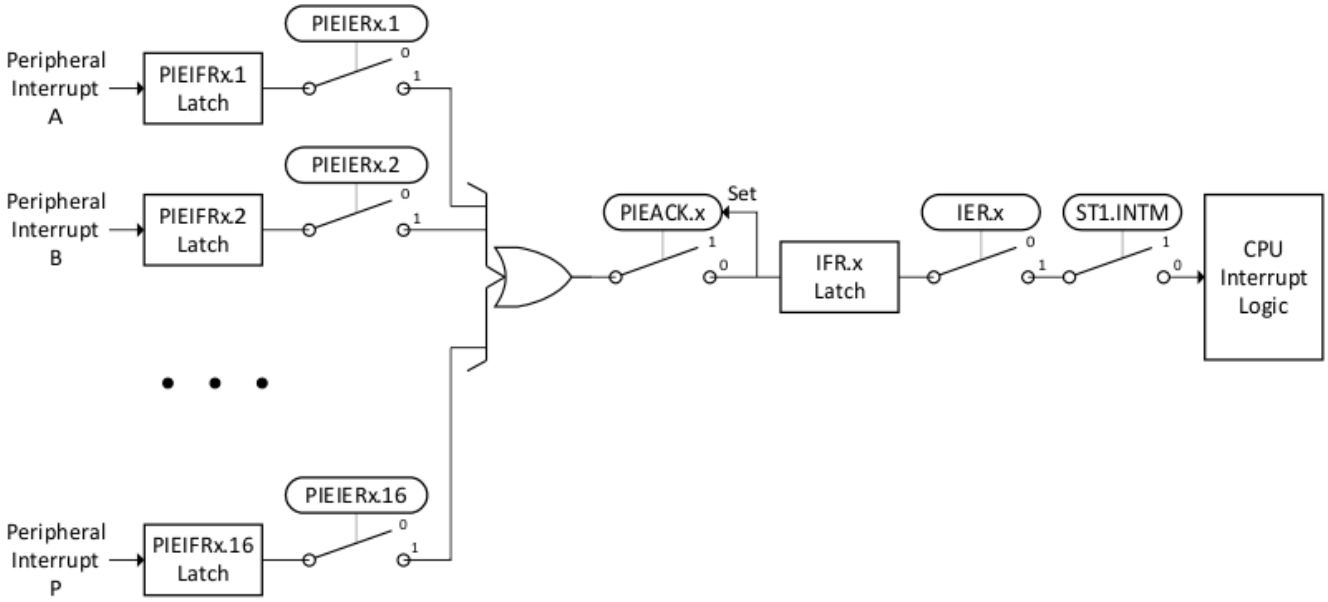


Figure 3-2. Interrupt Propagation Path

When a peripheral generates an interrupt (on PIE group x, channel y), it triggers the following sequence of events:

1. The interrupt is latched in PIEIFRx.y.
2. If PIEIERx.y is set, the interrupt propagates.
3. If PIEACK.x is clear, the interrupt propagates and PIEACK.x is set.
4. The interrupt is latched in IFR.x.
5. If IER.x is set, the interrupt propagates.
6. If INTM is clear, the CPU receives the interrupt.
7. Any instructions in the D2 or later stage of the pipeline are run to completion. Instructions in earlier stages are flushed.
8. The CPU saves its context on the stack.
9. IFR.x and IER.x are cleared. INTM is set. EALLOW is cleared.
10. The CPU fetches the ISR vector from the PIE. PIEIFRx.y is cleared.
11. The CPU branches to the ISR.

The interrupt latency is the time between PIEIFRx.y latching the interrupt and the first ISR instruction entering the execution stage of the CPU pipeline. The minimum interrupt latency is 14 SYSCLK cycles. Wait states on the ISR or stack memories will add to the latency. External interrupts add a minimum of two SYSCLK cycles for GPIO synchronization plus extra time for input qualification (if used). Loops created using the C28x RPT instruction cannot be interrupted.

3.5.4 Configuring and Using Interrupts

At power-up, no interrupts are enabled by default. The PIEIER and IER registers are cleared and INTM is set. The application code is responsible for configuring and enabling all peripheral interrupts.

3.5.4.1 Enabling Interrupts

To enable a peripheral interrupt, perform the following steps:

1. Disable interrupts globally (DINT or SETC INTM).
2. Enable the PIE by setting the ENPIE bit of the PIECTRL register.
3. Write the ISR vector for each interrupt to the appropriate location in the PIE vector table, which can be found in Table 3-3. Note that the vector table is EALLOW-protected.
4. Set the appropriate PIEIERx bit for each interrupt. The PIE group and channel assignments can be found in Table 3-3.
5. Set the CPU IER bit for any PIE group containing enabled interrupts.
6. Enable the interrupt in the peripheral.
7. Enable interrupts globally (EINT or CLRC INTM).

Step 4 does not apply to the Timer1 and Timer2 interrupts, which connect directly to the CPU.

3.5.4.2 Handling Interrupts

ISRs are similar to normal functions, but must do the following:

1. Save and restore the state of certain CPU registers (if used).
2. Clear the PIEACK bit for the interrupt group.
3. Return using the IRET instruction.

Requirements 1 and 3 are handled automatically by the TMS320C28x C compiler if the function is defined using the __interrupt keyword. For information on this keyword, see the Keywords section of the *TMS320C28x Optimizing C/C++ Compiler v6.2.4 User's Guide*. For information on writing assembly code to handle interrupts, see the Standard Operation for Maskable Interrupts section of the *TMS320C28x CPU and Instruction Set Reference Guide*.

The PIEACK bit for the interrupt group must be cleared manually in user code. This is normally done at the end of the ISR. If the PIEACK bit is not cleared, the CPU will not receive any further interrupts from that group. This does not apply to the Timer1 and Timer2 interrupts, which do not go through the PIE.

3.5.4.3 Disabling Interrupts

To disable all interrupts, set the CPU's global interrupt mask via DINT or SETC INTM. It is not necessary to add NOPs after setting INTM or modifying IER – the next instruction will execute with interrupts disabled.

Individual interrupts can be disabled using the PIEIERx registers, but care must be taken to avoid race conditions. If an interrupt signal is already propagating when the PIEIER write completes, it may reach the CPU and trigger a spurious interrupt condition. To avoid this, use the following procedure:

1. Disable interrupts globally (DINT or SETC INTM).
2. Clear the PIEIER bit for the interrupt.
3. Wait 5 cycles to make sure that any propagating interrupt has reached the CPU IFR register.
4. Clear the CPU IFR bit for the interrupt's PIE group.
5. Clear the PIEACK bit for the interrupt's PIE group.
6. Enable interrupts globally (EINT or CLRC INTM).

Interrupt groups can be disabled using the CPU IER register. This cannot cause a race condition, so no special procedure is needed.

PIEIFR bits must never be cleared in software since the read/modify/write operation may cause incoming interrupts to be lost. The only safe way to clear a PIEIFR bit is to have the CPU take the interrupt. The following procedure can be used to bypass the normal ISR:

1. Disable interrupts globally (DINT or SETC INTM).
2. Modify the PIE vector table to map the PIEIFR bit's interrupt vector to an empty ISR. This ISR will only contain a return from interrupt instruction (IRET).
3. Disable the interrupt in the peripheral registers.
4. Enable interrupts globally (EINT or CLRC INTM).
5. Wait for the pending interrupt to be serviced by the empty ISR.
6. Disable interrupts globally.
7. Modify the PIE vector table to map the interrupt vector back to its original ISR.
8. Clear the PIEACK bit for the interrupt's PIE group.
9. Enable interrupts globally.

3.5.4.4 Nesting Interrupts

By default, interrupts do not nest. It is possible to nest and prioritize interrupts via software control of the IER and PIEIERx registers. Example code can be found in C2000Ware and documentation is available at .

3.5.4.5 Vector Address Validity Check

There are two copies of the ePIE vector table. The primary vector table is located at addresses 0xD00 - 0xEFF. The redundant vector table is located at addresses 0x01000D00 - 0x01000EFF. A write to a primary vector address writes to both tables, while a write to a redundant vector address only writes to the redundant table. Both tables are read independently.

During a vector fetch, the ePIE performs a hardware comparison of both vector table outputs. If there is a mismatch between the two vector tables, the CPU branches to the address in the PIEVERRADDR register and the ePIE sends trip signals to the PWMs. If the PIEVERRADDR register value has not been set, the default boot ROM handler at address 0x003FFFBE is used.

3.5.5 PIE Channel Mapping

Table 3-3 shows the PIE group and channel assignments for each peripheral interrupt. Each row is a group, and each column is a channel within that group. When multiple interrupts are pending, the lowest-numbered channel is the lowest-numbered group is serviced first. Thus, the interrupts at the top of the table have the highest priority, and the interrupts at the bottom have the lowest priority.

Table 3-3. Pie Channel Mapping

	INTx.1	INTx.2	INTx.3	INTx.4	INTx.5	INTx.6	INTx.7	INTx.8	INTx.9	INTx.10	INTx.11	INTx.12	INTx.13	INTx.14	INTx.15	INTx.16
INT1.y	ADCA1	ADCB1	ADCC1	XINT1	XINT2	-	TIMER0	WAKE / WDOG	-	SYS_ERR	-	-	-	-	-	-
INT2.y	EPWM1_ TZ	EPWM2_ TZ	EPWM3_ TZ	EPWM4_ TZ	EPWM5_ TZ	EPWM6_ TZ	EPWM7_ TZ	EPWM8_ TZ	-	-	-	-	-	-	-	-
INT3.y	EPWM1	EPWM2	EPWM3	EPWM4	EPWM5	EPWM6	EPWM7	EPWM8	-	-	-	-	-	-	-	-
INT4.y	ECAP1	ECAP2	ECAP3	-	-	-	-	-	-	-	ECAP3INT2	-	-	-	-	-
INT5.y	EQEP1	EQEP2	-	-	CLB1	CLB2	CLB3	CLB4	SDFM1	SDFM2	-	-	SDFM1DR1	SDFM1DR2	SDFM1DR3	SDFM1DR4
INT6.y	SPIA_RX	SPIA_TX	SPIB_RX	SPIB_TX	-	-	-	-	-	-	-	-	SDFM2DR1	SDFM2DR2	SDFM2DR3	SDFM2DR4
INT7.y	DMA_ CH1	DMA_ CH2	DMA_ CH3	DMA_ CH4	DMA_ CH5	DMA_ CH6	-	-	-	-	FSITX_ INT1	FSITX_ INT2	FSIRX_ INT1	FSIRX_ INT2	-	DCC0
INT8.y	I2CA	I2CA_ FIFO	I2CB	I2CB_ FIFO	-	-	-	-	LINA_0	LINA_1	LINB_0	LINB_1	PMBUSA	-	-	DCC1
INT9.y	SCIA_RX	SCIA_TX	SCIB_RX	SCIB_TX	DCANA_0	DCANA_1	-	-	MCAN_0	MCAN_1	MCAN_ECC	MCAN_ WAKE	BGCRC_ CPU	-	-	HICA
INT10.y	ADCA_ EVT	ADCA2	ADCA3	ADCA4	ADCB_EVT	ADCB2	ADCB3	ADCB4	ADCC_ EVT	ADCC2	ADCC3	ADCC4	-	-	-	-
INT11.y	CLA1_1	CLA1_2	CLA1_3	CLA1_4	CLA1_5	CLA1_6	CLA1_7	CLA1_8	-	-	-	-	-	-	-	-
INT12.y	XINT3	XINT4	XINT5	MPOST	FMC	-	FPU_OVER FLOW	FPU_ UNDER FLOW	-	RAM_ CORR_ ERR	FLASH_ CORR_ ERR	RAM_ACC_ VIOLATION	AES_SIN_ TREQ	BGCRC_ CLA1	CLA_OVER FLOW	CLA_ UNDER FLOW

3.5.5.1 PIE Interrupt Priority

3.5.5.1.1 Channel Priority

For every PIE group, the low number channels in the group have the highest priority. For instance in PIE group 1, channel 1.1 has priority over channel 1.3. If those two enabled interrupts occurred simultaneously, channel 1.1 will be serviced first with channel 1.3 left pending. Once the ISR for channel 1.1 completes and provided there are no other enabled and pending interrupts for PIE group 1, channel 1.3 will be serviced. However, for the CPU to service any more interrupts from a PIE group, PIEACK for the group must be cleared. For this specific example, in order for channel 1.3 to be serviced, channel 1.1's ISR has to clear PIEACK for group 1.

The following example describes an alternative scenario: channel 1.1 is currently being serviced by the CPU, channel 1.3 is pending and before channel 1.1's ISR completes, channel 1.2 which is enabled also comes in. Since channel 1.2 has a higher priority than channel 1.3, the CPU will service channel 1.2 and channel 1.3 will still be left pending. Using the steps from the Interrupt Entry Sequence (Section 3.5.3), channel 1.2 interrupt can happen as late as step 10 (The CPU fetches the ISR vector from the PIE. PIEIFRx.y is cleared) and it will still be serviced ahead of channel 1.3.

3.5.5.1.2 Group Priority

Generally, the lowest channel in the lowest PIE group has the highest priority. An example of this is channels 1.1 and 2.1. Those two channels have the highest priority in their respective groups. If the interrupts for those two enabled channels happened simultaneously and provided there are no other enabled and pending interrupts, channel 1.1 will be serviced first by the CPU with channel 2.1 left pending.

However, there are cases where channel priority supersedes group priority. This special case happens depending on which step the CPU is currently at in the Interrupt Entry Sequence (Section 3.5.3).

The following illustrates an example of this special case.

The CPU is about to service channel 2.3 and is currently going through the steps in the Interrupt Entry Sequence (Section 3.5.3).

1. As the CPU reaches step 10 (The CPU fetches the ISR vector from the PIE. PIEIFRx.y is cleared), two enabled interrupts: channel 1.1 and channel 2.1 come in.
2. Due to channel priority, channel 2.1 will be serviced ahead of channel 2.3. However, group priority dictates that channel 1.1 be serviced ahead of channels 2.1 and 2.3.
3. Channel priority supersedes here and channel 2.1 will be serviced ahead of 1.1 and 2.3.
4. After channel 2.1 completes, channel 1.1 is serviced followed by channel 2.3.

Group priority is only guaranteed if no interrupts are currently being serviced, that is, the Interrupt Entry Sequence (Section 3.5.3) is not executing.

3.5.6 Vector Tables

Table 3-4 shows the CPU interrupt vector table. The vectors for INT1 – INT12 are not used in this device. The reset vector is fetched from the boot ROM instead of from this table. All vectors are EALLOW-protected.

Table 3-5 shows the pie vector table.

Table 3-4. CPU Interrupt Vectors

Name	Vector ID	Address	Size (x16)	Description	Core Priority	ePIE Group Priority
Reset	0	0x0000 0D00	2	Reset is always fetched from location 0x003F_FFC0 in Boot ROM	1 (Highest)	-
INT1	1	0x0000 0D02	2	Not used. See PIE Group 1	5	-
INT2	2	0x0000 0D04	2	Not used. See PIE Group 2	6	-
INT3	3	0x0000 0D06	2	Not used. See PIE Group 3	7	-
INT4	4	0x0000 0D08	2	Not used. See PIE Group 4	8	-
INT5	5	0x0000 0D0A	2	Not used. See PIE Group 5	9	-
INT6	6	0x0000 0D0C	2	Not used. See PIE Group 6	10	-
INT7	7	0x0000 0D0E	2	Not used. See PIE Group 7	11	-
INT8	8	0x0000 0D10	2	Not used. See PIE Group 8	12	-
INT9	9	0x0000 0D12	2	Not used. See PIE Group 9	13	-
INT10	10	0x0000 0D14	2	Not used. See PIE Group 10	14	-
INT11	11	0x0000 0D16	2	Not used. See PIE Group 11	15	-
INT12	12	0x0000 0D18	2	Not used. See PIE Group 12	16	-
INT13	13	0x0000 0D1A	2	CPU TIMER1 Interrupt	17	-
INT14	14	0x0000 0D1C	2	CPU TIMER2 Interrupt	18	-
DATALOG	15	0x0000 0D1E	2	CPU Data Logging Interrupt	19 (lowest)	-
RTOSINT	16	0x0000 0D20	2	CPU Real-Time OS Interrupt	4	-
EMUINT	17	0x0000 0D22	2	CPU Emulation Interrupt	2	-
NMI	18	0x0000 0D24	2	Non-Maskable Interrupt	3	-
ILLEGAL	19	0x0000 0D26	2	Illegal Instruction (ITRAP)	-	-
USER 1	20	0x0000 0D28	2	User-Defined Trap	-	-
USER 2	21	0x0000 0D2A	2	User-Defined Trap	-	-
USER 3	22	0x0000 0D2C	2	User-Defined Trap	-	-
USER 4	23	0x0000 0D2E	2	User-Defined Trap	-	-
USER 5	24	0x0000 0D30	2	User-Defined Trap	-	-
USER 6	25	0x0000 0D32	2	User-Defined Trap	-	-
USER 7	26	0x0000 0D34	2	User-Defined Trap	-	-
USER 8	27	0x0000 0D36	2	User-Defined Trap	-	-
USER 9	28	0x0000 0D38	2	User-Defined Trap	-	-
USER 10	29	0x0000 0D3A	2	User-Defined Trap	-	-
USER 11	30	0x0000 0D3C	2	User-Defined Trap	-	-
USER 12	31	0x0000 0D3E	2	User-Defined Trap	-	-

Table 3-5. PIE Interrupt Vectors

Name	Vector ID	Address	Size (x16)	Description	Core Priority	ePIE Group Priority
PIE Group 1 Vectors - Muxed into CPU INT1						
INT1.1	32	0x0000 0D40	2	ADCA1 interrupt	5	1 (Highest)
INT1.2	33	0x0000 0D42	2	ADCB1 interrupt	5	2
INT1.3	34	0x0000 0D44	2	ADCC1 interrupt	5	3
INT1.4	35	0x0000 0D46	2	XINT1 interrupt	5	4
INT1.5	36	0x0000 0D48	2	XINT2 interrupt	5	5
INT1.6	37	0x0000 0D4A	2	Reserved	5	6
INT1.7	38	0x0000 0D4C	2	TIMER0 interrupt	5	7
INT1.8	39	0x0000 0D4E	2	WAKE interrupt	5	8
INT1.9	128	0x0000 0E00	2	Reserved	5	9
INT1.10	129	0x0000 0E02	2	SYS_ERR interrupt	5	10
INT1.11	130	0x0000 0E04	2	Reserved	5	11
INT1.12	131	0x0000 0E06	2	Reserved	5	12
INT1.13	132	0x0000 0E08	2	Reserved	5	13
INT1.14	133	0x0000 0E0A	2	Reserved	5	14
INT1.15	134	0x0000 0E0C	2	Reserved	5	15
INT1.16	135	0x0000 0E0E	2	Reserved	5	16 (Lowest)
PIE Group 2 Vectors - Muxed into CPU INT2						
INT2.1	40	0x0000 0D50	2	EPWM1 trip zone interrupt	6	1 (Highest)
INT2.2	41	0x0000 0D52	2	EPWM2 trip zone interrupt	6	2
INT2.3	42	0x0000 0D54	2	EPWM3 trip zone interrupt	6	3
INT2.4	43	0x0000 0D56	2	EPWM4 trip zone interrupt	6	4
INT2.5	44	0x0000 0D58	2	EPWM5 trip zone interrupt	6	5
INT2.6	45	0x0000 0D5A	2	EPWM6 trip zone interrupt	6	6
INT2.7	46	0x0000 0D5C	2	EPWM7 trip zone interrupt	6	7
INT2.8	47	0x0000 0D5E	2	EPWM8 trip zone interrupt	6	8
INT2.9	136	0x0000 0E10	2	Reserved	6	9
INT2.10	137	0x0000 0E12	2	Reserved	6	10
INT2.11	138	0x0000 0E14	2	Reserved	6	11
INT2.12	139	0x0000 0E16	2	Reserved	6	12
INT2.13	140	0x0000 0E18	2	Reserved	6	13
INT2.14	141	0x0000 0E1A	2	Reserved	6	14
INT2.15	142	0x0000 0E1C	2	Reserved	6	15
INT2.16	143	0x0000 0E1E	2	Reserved	6	16 (Lowest)
PIE Group 3 Vectors - Muxed into CPU INT3						
INT3.1	48	0x0000 0D60	2	EPWM1 interrupt	7	1 (Highest)
INT3.2	49	0x0000 0D62	2	EPWM2 interrupt	7	2
INT3.3	50	0x0000 0D64	2	EPWM3 interrupt	7	3
INT3.4	51	0x0000 0D66	2	EPWM4 interrupt	7	4

Table 3-5. PIE Interrupt Vectors (continued)

Name	Vector ID	Address	Size (x16)	Description	Core Priority	ePIE Group Priority
INT3.5	52	0x0000 0D68	2	EPWM5 interrupt	7	5
INT3.6	53	0x0000 0D6A	2	EPWM6 interrupt	7	6
INT3.7	54	0x0000 0D6C	2	EPWM7 interrupt	7	7
INT3.8	55	0x0000 0D6E	2	EPWM8 interrupt	7	8
INT3.9	144	0x0000 0E20	2	Reserved	7	9
INT3.10	145	0x0000 0E22	2	Reserved	7	10
INT3.11	146	0x0000 0E24	2	Reserved	7	11
INT3.12	147	0x0000 0E26	2	Reserved	7	12
INT3.13	148	0x0000 0E28	2	Reserved	7	13
INT3.14	149	0x0000 0E2A	2	Reserved	7	14
INT3.15	150	0x0000 0E2C	2	Reserved	7	15
INT3.16	151	0x0000 0E2E	2	Reserved	7	16 (Lowest)
PIE Group 4 Vectors - Muxed into CPU INT4						
INT4.1	56	0x0000 0D70	2	ECAP1 interrupt	8	1 (Highest)
INT4.2	57	0x0000 0D72	2	ECAP2 interrupt	8	2
INT4.3	58	0x0000 0D74	2	ECAP3 interrupt	8	3
INT4.4	59	0x0000 0D76	2	Reserved	8	4
INT4.5	60	0x0000 0D78	2	Reserved	8	5
INT4.6	61	0x0000 0D7A	2	Reserved	8	6
INT4.7	62	0x0000 0D7C	2	Reserved	8	7
INT4.8	63	0x0000 0D7E	2	Reserved	8	8
INT4.9	152	0x0000 0E30	2	Reserved	8	9
INT4.10	153	0x0000 0E32	2	Reserved	8	10
INT4.11	154	0x0000 0E34	2	ECAP3 interrupt 2	8	11
INT4.12	155	0x0000 0E36	2	Reserved	8	12
INT4.13	156	0x0000 0E38	2	Reserved	8	13
INT4.14	157	0x0000 0E3A	2	Reserved	8	14
INT4.15	158	0x0000 0E3C	2	Reserved	8	15
INT4.16	159	0x0000 0E3E	2	Reserved	8	16 (Lowest)
PIE Group 5 Vectors - Muxed into CPU INT5						
INT5.1	64	0x0000 0D80	2	EQEP1 interrupt	9	1 (Highest)
INT5.2	65	0x0000 0D82	2	EQEP2 interrupt	9	2
INT5.3	66	0x0000 0D84	2	Reserved	9	3
INT5.4	67	0x0000 0D86	2	Reserved	9	4
INT5.5	68	0x0000 0D88	2	CLB1 interrupt	9	5
INT5.6	69	0x0000 0D8A	2	CLB2 interrupt	9	6
INT5.7	70	0x0000 0D8C	2	CLB3 interrupt	9	7
INT5.8	71	0x0000 0D8E	2	CLB4 interrupt	9	8
INT5.9	160	0x0000 0E40	2	SDFM1 interrupt	9	9
INT5.10	161	0x0000 0E42	2	SDFM2 interrupt	9	10
INT5.11	162	0x0000 0E44	2	Reserved	9	11
INT5.12	163	0x0000 0E46	2	Reserved	9	12
INT5.13	164	0x0000 0E48	2	SDFM1DR1 interrupt	9	13
INT5.14	165	0x0000 0E4A	2	SDFM1DR2 interrupt	9	14

Table 3-5. PIE Interrupt Vectors (continued)

Name	Vector ID	Address	Size (x16)	Description	Core Priority	ePIE Group Priority
INT5.15	166	0x0000 0E4C	2	SDFM1DR3 interrupt	9	15
INT5.16	167	0x0000 0E4E	2	SDFM1DR4 interrupt	9	16 (Lowest)
PIE Group 6 Vectors - Muxed into CPU INT6						
INT6.1	72	0x0000 0D90	2	SPIA_RX interrupt	10	1 (Highest)
INT6.2	73	0x0000 0D92	2	SPIA_TX interrupt	10	2
INT6.3	74	0x0000 0D94	2	SPIB_RX interrupt	10	3
INT6.4	75	0x0000 0D96	2	SPIB_TX interrupt	10	4
INT6.5	76	0x0000 0D98	2	Reserved	10	5
INT6.6	77	0x0000 0D9A	2	Reserved	10	6
INT6.7	78	0x0000 0D9C	2	Reserved	10	7
INT6.8	79	0x0000 0D9E	2	Reserved	10	8
INT6.9	168	0x0000 0E50	2	Reserved	10	9
INT6.10	169	0x0000 0E52	2	Reserved	10	10
INT6.11	170	0x0000 0E54	2	Reserved	10	11
INT6.12	171	0x0000 0E56	2	Reserved	10	12
INT6.13	172	0x0000 0E58	2	SDFM2DR1 interrupt	10	13
INT6.14	173	0x0000 0E5A	2	SDFM2DR2 interrupt	10	14
INT6.15	174	0x0000 0E5C	2	SDFM2DR3 interrupt	10	15
INT6.16	175	0x0000 0E5E	2	SDFM2DR4 interrupt	10	16 (Lowest)
PIE Group 7 Vectors - Muxed into CPU INT7						
INT7.1	80	0x0000 0DA0	2	DMA_CH1 interrupt	11	1 (Highest)
INT7.2	81	0x0000 0DA2	2	DMA_CH2 interrupt	11	2
INT7.3	82	0x0000 0DA4	2	DMA_CH3 interrupt	11	3
INT7.4	83	0x0000 0DA6	2	DMA_CH4 interrupt	11	4
INT7.5	84	0x0000 0DA8	2	DMA_CH5 interrupt	11	5
INT7.6	85	0x0000 0DAA	2	DMA_CH6 interrupt	11	6
INT7.7	86	0x0000 0DAC	2	Reserved	11	7
INT7.8	87	0x0000 0DAE	2	Reserved	11	8
INT7.9	176	0x0000 0E60	2	Reserved	11	9
INT7.10	177	0x0000 0E62	2	Reserved	11	10
INT7.11	178	0x0000 0E64	2	FSITX_INT1	11	11
INT7.12	179	0x0000 0E66	2	FSITX_INT2	11	12
INT7.13	180	0x0000 0E68	2	FSIRX_INT1	11	13
INT7.14	181	0x0000 0E6A	2	FSIRX_INT2	11	14
INT7.15	182	0x0000 0E6C	2	Reserved	11	15
INT7.16	183	0x0000 0E6E	2	DCC0 interrupt	11	16 (Lowest)
PIE Group 8 Vectors - Muxed into CPU INT8						
INT8.1	88	0x0000 0DB0	2	I2CA interrupt	12	1 (Highest)
INT8.2	89	0x0000 0DB2	2	I2CA FIFO interrupt	12	2
INT8.3	90	0x0000 0DB4	2	I2CB interrupt	12	3
INT8.4	91	0x0000 0DB6	2	I2CB FIFO interrupt	12	4
INT8.5	92	0x0000 0DB8	2	Reserved	12	5
INT8.6	93	0x0000 0DBA	2	Reserved	12	6
INT8.7	94	0x0000 0DBC	2	Reserved	12	7

Table 3-5. PIE Interrupt Vectors (continued)

Name	Vector ID	Address	Size (x16)	Description	Core Priority	ePIE Group Priority
INT8.8	95	0x0000 0DBE	2	Reserved	12	8
INT8.9	184	0x0000 0E70	2	LINA interrupt 0	12	9
INT8.10	185	0x0000 0E72	2	LINA interrupt 1	12	10
INT8.11	186	0x0000 0E74	2	LINB interrupt 0	12	11
INT8.12	187	0x0000 0E76	2	LINB interrupt 1	12	12
INT8.13	188	0x0000 0E78	2	PMBUSA interrupt	12	13
INT8.14	189	0x0000 0E7A	2	Reserved	12	14
INT8.15	190	0x0000 0E7C	2	Reserved	12	15
INT8.16	191	0x0000 0E7E	2	DCC1 interrupt	12	16 (Lowest)
PIE Group 9 Vectors - Muxed into CPU INT9						
INT9.1	96	0x0000 0DC0	2	SCIA RX interrupt	13	1 (Highest)
INT9.2	97	0x0000 0DC2	2	SCIA TX interrupt	13	2
INT9.3	98	0x0000 0DC4	2	SCIB RX interrupt	13	3
INT9.4	99	0x0000 0DC6	2	SCIB TX interrupt	13	4
INT9.5	100	0x0000 0DC8	2	DCANA interrupt 0	13	5
INT9.6	101	0x0000 0DCA	2	DCANA interrupt 1	13	6
INT9.7	102	0x0000 0DCC	2	Reserved	13	7
INT9.8	103	0x0000 0DCE	2	Reserved	13	8
INT9.9	192	0x0000 0E80	2	MCAN interrupt 0	13	9
INT9.10	193	0x0000 0E82	2	MCAN interrupt 1	13	10
INT9.11	194	0x0000 0E84	2	MCAN_ECC interrupt	13	11
INT9.12	195	0x0000 0E86	2	MCAN_WAKE interrupt	13	12
INT9.13	196	0x0000 0E88	2	BGCRC_CPU interrupt	13	13
INT9.14	197	0x0000 0E8A	2	Reserved	13	14
INT9.15	198	0x0000 0E8C	2	Reserved	13	15
INT9.16	199	0x0000 0E8E	2	HICA interrupt	13	16 (Lowest)
PIE Group 10 Vectors - Muxed into CPU INT10						
INT10.1	104	0x0000 0DD0	2	ADCA event interrupt	14	1 (Highest)
INT10.2	105	0x0000 0DD2	2	ADCA2 interrupt	14	2
INT10.3	106	0x0000 0DD4	2	ADCA3 interrupt	14	3
INT10.4	107	0x0000 0DD6	2	ADCA4 interrupt	14	4
INT10.5	108	0x0000 0DD8	2	ADCB event interrupt	14	5
INT10.6	109	0x0000 0DDA	2	ADCB2 interrupt	14	6
INT10.7	110	0x0000 0DDC	2	ADCB3 interrupt	14	7
INT10.8	111	0x0000 0DDE	2	ADCB4 interrupt	14	8
INT10.9	200	0x0000 0E90	2	ADCC event interrupt	14	9
INT10.10	201	0x0000 0E92	2	ADCC2 interrupt	14	10
INT10.11	202	0x0000 0E94	2	ADCC3 interrupt	14	11
INT10.12	203	0x0000 0E96	2	ADCC4 interrupt	14	12
INT10.13	204	0x0000 0E98	2	Reserved	14	13
INT10.14	205	0x0000 0E9A	2	Reserved	14	14
INT10.15	206	0x0000 0E9C	2	Reserved	14	15
INT10.16	207	0x0000 0E9E	2	Reserved	14	16 (Lowest)

Table 3-5. PIE Interrupt Vectors (continued)

Name	Vector ID	Address	Size (x16)	Description	Core Priority	ePIE Group Priority
PIE Group 11 Vectors - Muxed into CPU INT11						
INT11.1	112	0x0000 0DE0	2	CLA1_1 interrupt	15	1 (Highest)
INT11.2	113	0x0000 0DE2	2	CLA1_2 interrupt	15	2
INT11.3	114	0x0000 0DE4	2	CLA1_3 interrupt	15	3
INT11.4	115	0x0000 0DE6	2	CLA1_4 interrupt	15	4
INT11.5	116	0x0000 0DE8	2	CLA1_5 interrupt	15	5
INT11.6	117	0x0000 0DEA	2	CLA1_6 interrupt	15	6
INT11.7	118	0x0000 0DEC	2	CLA1_7 interrupt	15	7
INT11.8	119	0x0000 0DEE	2	CLA1_8 interrupt	15	8
INT11.9	208	0x0000 0EA0	2	Reserved	15	9
INT11.10	209	0x0000 0EA2	2	Reserved	15	10
INT11.11	210	0x0000 0EA4	2	Reserved	15	11
INT11.12	211	0x0000 0EA6	2	Reserved	15	12
INT11.13	212	0x0000 0EA8	2	Reserved	15	13
INT11.14	213	0x0000 0EAA	2	Reserved	15	14
INT11.15	214	0x0000 0EAC	2	Reserved	15	15
INT11.16	215	0x0000 0EAE	2	Reserved	15	16 (Lowest)
PIE Group 12 Vectors - Muxed into CPU INT12						
INT12.1	120	0x0000 0DF0	2	XINT3 interrupt	16	1 (Highest)
INT12.2	121	0x0000 0DF2	2	XINT4 interrupt	16	2
INT12.3	122	0x0000 0DF4	2	XINT5 interrupt	16	3
INT12.4	123	0x0000 0DF6	2	MPOST interrupt	16	4
INT12.5	124	0x0000 0DF8	2	Flash Wrapper Operation done interrupt	16	5
INT12.6	125	0x0000 0DFA	2	Reserved	16	6
INT12.7	126	0x0000 0DFC	2	FPU overflow interrupt	16	7
INT12.8	127	0x0000 0DFE	2	FPU underflow interrupt	16	8
INT12.9	216	0x0000 0EB0	2	Reserved	16	9
INT12.10	217	0x0000 0EB2	2	RAM correctable error interrupt	16	10
INT12.11	218	0x0000 0EB4	2	Flash correctable error interrupt	16	11
INT12.12	219	0x0000 0EB6	2	RAM access violation interrupt	16	12
INT12.13	220	0x0000 0EB8	2	AES_SIN_TREQ interrupt	16	13
INT12.14	221	0x0000 0EBA	2	BGCRC_CLA1 interrupt	16	14
INT12.15	222	0x0000 0EBC	2	CLA OVERFLOW interrupt	16	15
INT12.16	223	0x0000 0EBE	2	CLA UNDERFLOW interrupt	16	16 (Lowest)

3.6 Exceptions and Non-Maskable Interrupts

This section describes system-level error conditions that can trigger a non-maskable interrupt (NMI). The interrupt allows the application to respond to the error.

3.6.1 Configuring and Using NMIs

An incoming NMI sets a status bit in the NMIFLG register and starts the NMI watchdog counter. This counter is clocked by the SYSCLK, and if it reaches the value in the NMIWDPRD register, it triggers an NMI watchdog reset (NMIWDRS). To prevent this, the NMI handler must clear the flag bit using the NMIFLGCLR register. Once all flag bits are clear, the NMIINT bit in the NMIFLG register may also be cleared to allow future NMIs to be taken.

The NMI module is enabled by the boot ROM during the startup process. To respond to NMIs, an NMI handler vector must be written to the PIE vector table.

3.6.2 Emulation Considerations

The NMI watchdog counter behaves as follows under debug conditions:

CPU Suspended	When the CPU is suspended, the NMI watchdog counter will be suspended.
Run-Free Mode	When the CPU is placed in run-free mode, the NMI watchdog counter will resume operation as normal.
Real-Time Single-Step Mode	When the CPU is in real-time single-step mode, the NMI watchdog counter will be suspended. The counter remains suspended even within real-time interrupts.
Real-Time Run-Free Mode	When the CPU is in real-time run-free mode, the NMI watchdog counter operates as normal.

3.6.3 NMI Sources

There are several types of hardware errors that can trigger an NMI. Additional information about the error is usually available from the module that detects it.

3.6.3.1 Missing Clock Detection

The missing clock detection logic monitors OSCCLK for failure. If the OSCCLK source stops, the PLL is bypassed, OSCCLK is connected to INTOSC1, and an NMI is fired to the CPU. For more information on missing clock detection, see Section 3.7.12.1.

3.6.3.2 RAM Uncorrectable Error

A double-bit ECC data error, or single-bit ECC address error in a RAM read will trigger an NMI. This applies to CPU and DMA reads. Single-bit ECC data errors do not trigger an NMI, but can optionally trigger a normal peripheral interrupt. For more information on RAM error detection, see Section 3.11.8.

3.6.3.3 Flash Uncorrectable ECC Error

A double-bit ECC data error or single-bit ECC address error in a Flash read triggers an NMI. Single-bit ECC data errors do not trigger an NMI, but can optionally trigger a normal peripheral interrupt. For more information on Flash error detection, see Section 3.11.8.

3.6.3.4 CPU HWBIST Error

If the Hardware Built-in self (HWBIST) module detects a fault in the CPU, an NMI is generated.

3.6.3.5 Software-Forced Error

There is a special NMI source that can only be triggered by writing to the SWERR bit in the NMIFLGFRC register. Since the SWERR flag is never set by a real hardware fail, it can be used to implement a self-test mode for the NMI subsystem.

3.6.4 CRC Fail

A CRC fail result from the Background CRC (BGCRC) module can generate NMI. By default, this NMI is enabled. To disable this feature, the NMIDIS configuration field in the BGCRC_CTRL1 register has to be written with "1010".

3.6.5 ERAD NMI

The ERAD module can generate NMI based on different events. This is configurable in the GLBL_NMI_CTL register.

3.6.6 Illegal Instruction Trap (ITRAP)

If the CPU tries to execute an illegal instruction, it generates a special interrupt called an illegal instruction trap (ITRAP). This interrupt is non-maskable and has its own vector in the PIE vector table. For more information about ITRAPs, see the Illegal-Instruction Trap section of the *TMS320C28x DSP CPU and Instruction Set Reference Guide*.

Note

A RAM fetch access violation will trigger an ITRAP in addition to the normal peripheral interrupt for RAM access violations. The CPU will handle the ITRAP first.

3.6.7 Error Pin

A signal called ERRORSTS can be output to GPIO24, GPIO28, GPIO29 or GPIO55. This signal goes low when any bit is set in the NMI shadow flag register (NMISHDFLG). It can be used to alert an external system to a problem in the microcontroller. Since the state of ERRORSTS is based on the shadow flags, ERRORSTS will remain low until the flags are cleared by the CPU or a power-on reset occurs.

All GPIO pins are inputs on power-up. If the state of the chosen ERRORSTS pin during power-up is important, an external pull-down should be connected to the pin.

3.7 Clocking

This section explains the clock sources and clock domains on this device, and how to configure them for application use. Figure 3-3 and Figure 3-4 provide an overview of the device's clocking system.

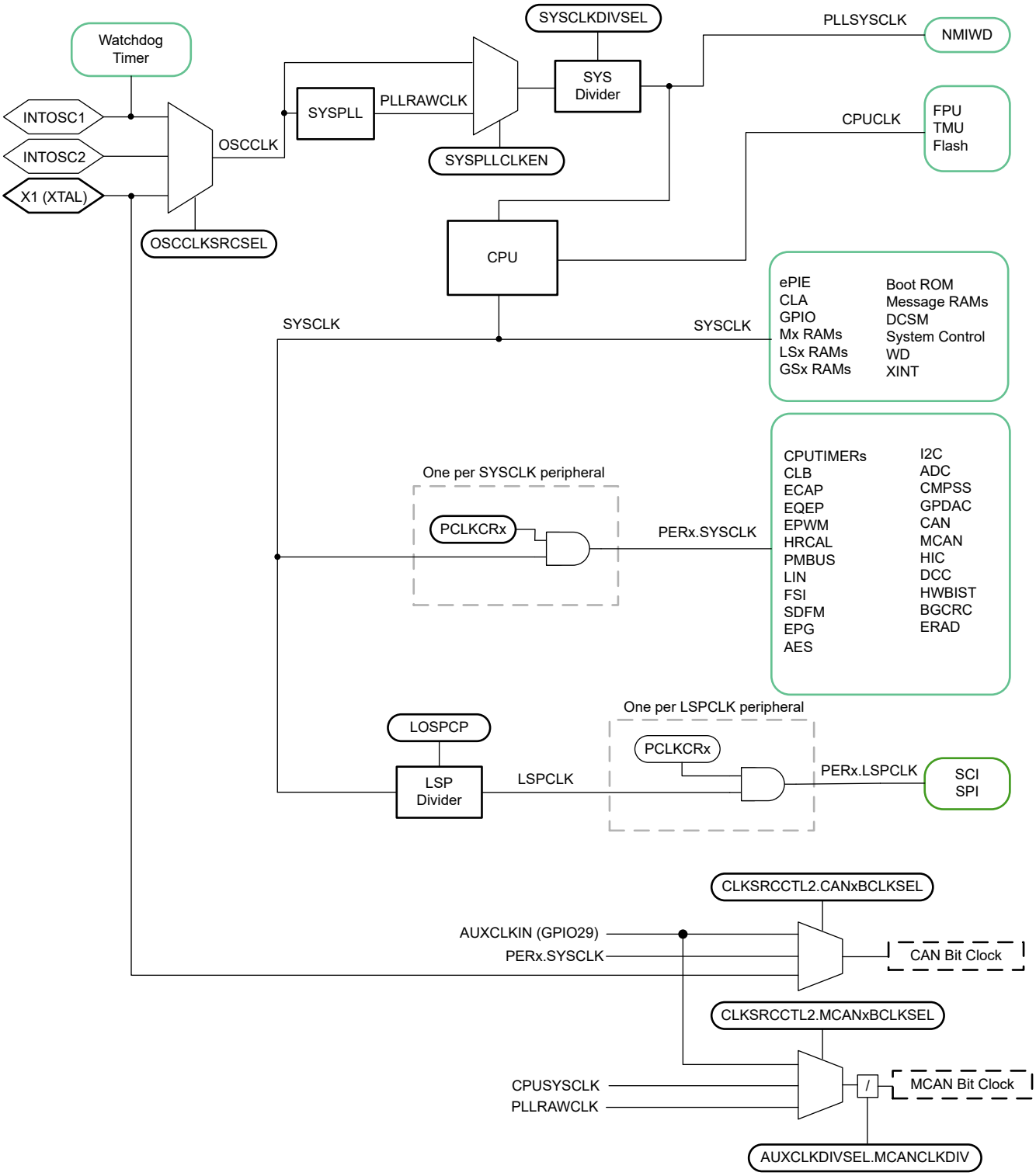


Figure 3-3. Clocking System

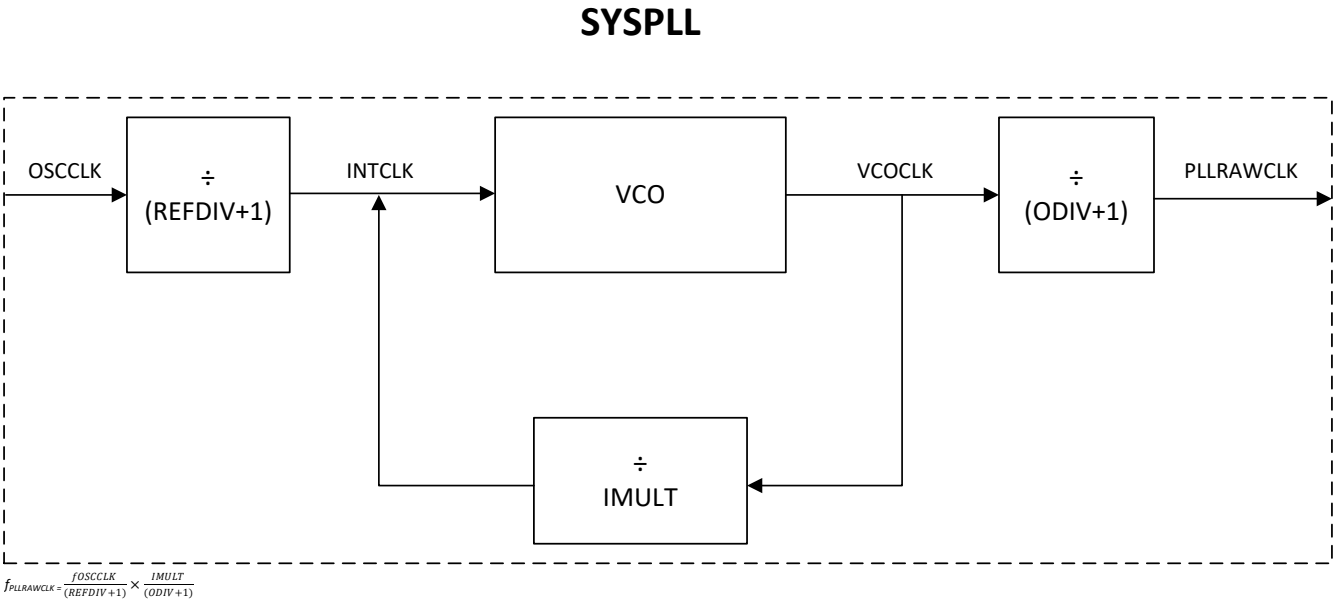


Figure 3-4. System PLL

3.7.1 Clock Sources

All of the clocks in the device are derived from one of four clock sources.

3.7.1.1 Primary Internal Oscillator (INTOSC2)

At power-up, the device is clocked from an on-chip 10-MHz oscillator (INTOSC2). INTOSC2 is the primary internal clock source and is the default system clock at reset. It is used to run the boot ROM and can be used as the system clock source for the application. Note that the INTOSC2 frequency tolerance is too loose to meet the timing requirements for CAN. Use of the CAN modules requires an external oscillator. When INTOSC2 is used as the system clock source, GPIO19 (X1) and GPIO18 (X2) are available as GPIO pins.

3.7.1.2 Backup Internal Oscillator (INTOSC1)

The device also includes a redundant on-chip 10 MHz oscillator (INTOSC1). INTOSC1 is a backup clock source that normally only clocks the watchdog timers and missing clock detection circuit (MCD). If MCD is enabled and a missing system clock is detected, the system PLL is bypassed and all system clocks are connected to INTOSC1 automatically. INTOSC1 may also be manually selected as the system clock source for debug purposes.

3.7.1.3 Auxiliary Clock Input (AUXCLKIN)

An additional external clock source is supported on GPIO29 (AUXCLKIN). This must be a single-ended 3.3V external clock as shown in Figure 3-5. It can be used as the clock source for DCAN and MCAN. Frequency limits and timing requirements are found in the *TMS320F28003x Real-Time Microcontrollers Data Sheet*. The external clock should be connected directly to the GPIO29 pin.

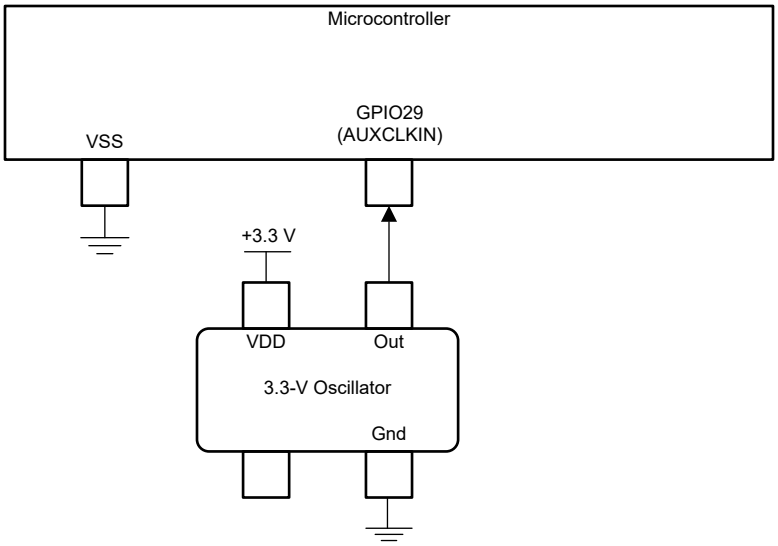


Figure 3-5. AUXCLKIN

3.7.1.4 External Oscillator (XTAL)

The device supports an external clock source (XTAL), which can be used as the main system and CAN bit clock source. Frequency limits and timing requirements are found in the *TMS320F28003x Real-Time Microcontrollers Data Sheet*. External clock sources use the X1/GPIO19 and X2/GPIO18 pins. After power-up, the X1 and X2 pin functionality can be enabled by following the procedure in Section 3.7.6.

Three types of external clock sources are supported:

- A single-ended 3.3V external clock. The clock signal should be connected to X1, as shown in Figure 3-6.

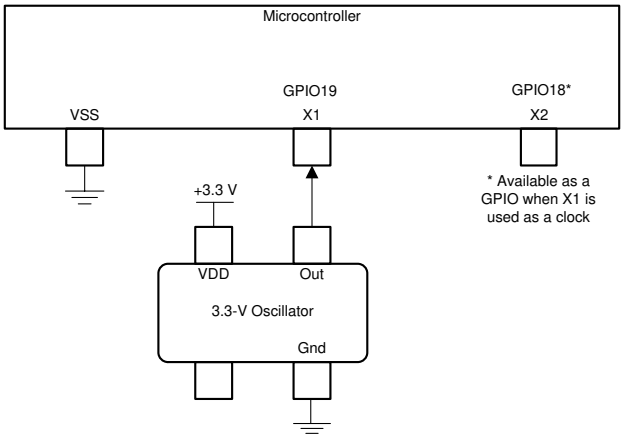


Figure 3-6. Single-ended 3.3V External Clock

- An external crystal. The crystal should be connected across X1 and X2 with its load capacitors connected to VSS as shown in Figure 3-7.

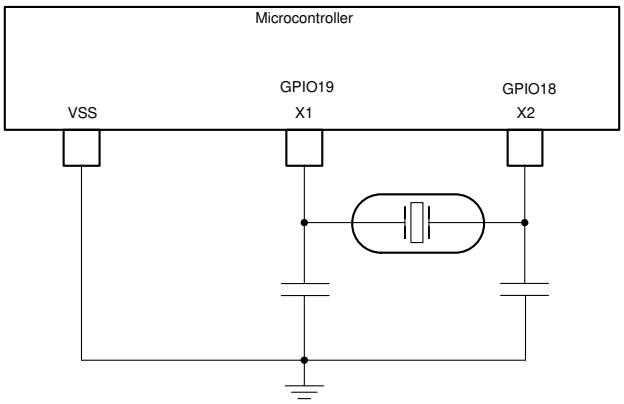


Figure 3-7. External Crystal

- An external resonator. The resonator should be connected across X1 and X2 with its ground connected to VSS as shown in Figure 3-8.

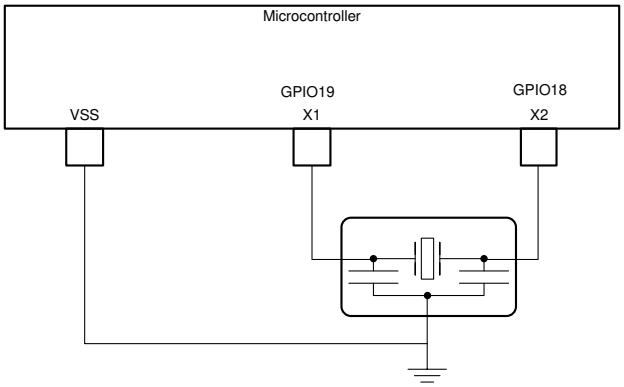


Figure 3-8. External Resonator

Table 3-6. ALT Modes

XTALCR Bit(1)		Operating Mode	GPIO19 Available on X1?	GPIO18 Available on X2?
OSCOFF	SE			
0	0	Crystal Mode: Quartz crystal connected to X1/X2	No	No
0	1	Single-Ended Mode: External clock on X1	No	Yes
1	0	Oscillator off	Yes	Yes
1	1	Single-Ended Mode: External clock on X1	No	Yes

(1) OSCOFF and SE determine the ALT mode of GPIO18 and GPIO19.

3.7.2 Derived Clocks

The clock sources discussed in the previous section can be multiplied (via PLL) and divided down to produce the desired clock frequencies for the application. This process produces a set of derived clocks, which are described in this section.

3.7.2.1 Oscillator Clock (OSCCLK)

One of INTOSC2, XTAL, or INTOSC1 must be chosen to be the master reference clock (OSCCLK) for the CPU and most of the peripherals. OSCCLK may be used directly or fed through the system PLL to reach a higher frequency. At reset, OSCCLK is the default system clock, and is connected to INTOSC2.

3.7.2.2 System PLL Output Clock (PLLRAWCLK)

The system PLL allows the device to run at its maximum rated operating frequency, and in most applications will generate the main system clock. This PLL uses OSCCLK as a reference. PLLRAWCLK is the output of the PLL's voltage-controlled oscillator (VCO). For configuration instructions, see Section 3.7.6.

3.7.3 Device Clock Domains

The device clock domains feed the clock inputs of the various modules in the device. They are connected to the derived clocks, either directly or through an additional divider.

3.7.3.1 System Clock (PLLSYSCLK)

The NMI watchdog timer has its own clock domain (PLLSYSCLK). Despite the name, PLLSYSCLK may be connected to the system PLL (PLLRAWCLK) or to OSCCLK. The chosen clock source is run through a frequency divider, which is configured via the SYSCLKDIVSEL register. PLLSYSCLK is gated in HALT mode.

3.7.3.2 CPU Clock (CPUCLK)

The CPU has its own clock (CPUCLK) that is used to clock the CPU and Flash wrapper. This clock is identical to PLLSYSCLK, but is gated when the CPU enters IDLE or HALT mode.

3.7.3.3 CPU Subsystem Clock (SYSCLK and PERx.SYSCLK)

Each peripheral clock has its own independent clock gating which is controlled by the PCLKCRx registers.

Note

Application needs to wait for 5 SYSCLK cycles after enabling clock to the peripherals when using PCLKCRx.

3.7.3.4 Low-Speed Peripheral Clock (LSPCLK and PERx.LSPCLK)

The SCI and SPI modules can communicate at bit rates that are much slower than the CPU frequency. These modules are connected to a shared clock divider, which generates a low-speed peripheral clock (LSPCLK) derived from SYSCLK. LSPCLK uses a /4 divider by default, but the ratio can be changed via the LOSPCP register. Each SCI and SPI module's clock (PERx.LSPCLK) can be gated independently via the PCLKCRx registers.

3.7.3.5 CAN Bit Clock

The required frequency tolerance for the DCAN and MCAN bit clock depends on the bit timing setup and network configuration, and can be as tight as 0.1%. Since the main system clock (in the form of SYSCLK) may not be precise enough, the bit clock can also be connected to XTAL, AUXCLKIN and PLLRAWCLK via the CLKSRCCTL2 register. There is an independent selection for each CAN module. See the CLKSRCCTL2 register for the valid options for the MCAN and DCAN.

To guarantee correct operation, the frequency of the CAN bit clock must be less than or equal to the SYSCLK frequency.

3.7.3.6 CPU Timer2 Clock (TIMER2CLK)

CPU timers 0 and 1 are connected to PERx.SYSCLK. Timer 2 is connected to PERx.SYSCLK by default, but may also be connected to INTOSC1, INTOSC2, or XTAL via the TMR2CLKCTL register. This register also provides a separate prescale divider for timer 2. If a non-SYSCLK source is used, it must be divided down to no more than half the SYSCLK frequency.

The main reason to use a non-SYSCLK source would be for internal frequency measurement. In most applications, timer 2 will run off of SYSCLK.

3.7.4 XCLKOUT

It is sometimes necessary to observe a clock directly for debug and testing purposes. The external clock output (XCLKOUT) feature supports this by connecting a clock to an external pin, which can be GPIO16 or GPIO18. The available clock sources are PLLSYSCLK, PLLRAWCLK, SYSCLK, INTOSC1, INTOSC2, and XTAL.

To use XCLKOUT, first select the clock source via the CLKSRCCTL3 register. Next, select the desired output divider via the XCLKOUTDIVSEL register. Finally, connect GPIO16 or GPIO18 to mux channel 11 using the GPIO configuration registers.

3.7.5 Clock Connectivity

Table 3-7 shows the clock connections sorted by the clock domain.

Table 3-7. Clock Connections Sorted by Clock Domain

Clock Domain	Module Name
CPUCLK	FPU
	TMU
	Flash
SYSCLK	CLA
	ePIE
	Mx RAMs
	LSx RAMs
	GSx RAMs
	Message RAMs
	Boot ROM
	GPIO Input Sync and Qual
	WD
	XINT
	DCSM
PLLSYSCLK	CPU
	NMIWD
PERx.SYSCLK	AES
	CLB
	Timer0 - 2
	DCC0 - 1
	FSI
	ePWM1 - 8
	eCAP1 - 3
	eQEP1 - 2
	ADCA, B, C
	GPDACA, B
	CMPSS1 - 4
	DCAN
	MCAN
	I2CA - B
	PMBUSA
	LINA - B
	HICA
	HWBIST
	BGCRC
	HRCAL
	EPG
	SDFM
	ERAD
PERx.LSPCLK	SCIA - B
	SPIA - B
CAN Bit Clock	DCAN, MCAN
WDCLK (INTOSC1)	Watchdog Timer

3.7.6 Clock Source and PLL Setup

The needs of the application are what ultimately determine the clock configuration. Specific concerns such as application performance, power consumption, total system cost, and EMC are beyond the scope of this document, but they should provide answers to the following questions:

1. What is the desired CPU frequency?
2. Is CAN required?
3. What types of external oscillators or clock sources are available?

If CAN will be used in the appilcation, it is recommended to use an external clock source with a precise frequency as the reference clock. Otherwise, it may be possible to use only INTOSC2 and avoid the need for more external components.

3.7.7 Using an External Crystal or Resonator

The X1 and X2 pins double as GPIO19 and GPIO18. At power-up, these pins are in GPIO mode and the on-chip crystal oscillator is powered off. The following procedure can be used to switch the pins to X1 and X2 mode and enable the oscillator:

1. Clear the XTALCR.OSCOFF bit.
2. Wait for the crystal to power up. 1ms is the typical wait time but this depends on the crystal that is being used.
3. Clear the X1 counter by writing a 1 to X1CNT.CLR and keep clearing until the X1 counter value in the X1CNT register is no longer saturated 2047 (0x7ff).
4. Wait for the X1 counter value in the X1CNT register to reach 2047 (0x7ff).
5. Repeat steps 3-4 three additional times.
6. Select XTAL as the OSCCLK source by writing a 1 to CLKSRCCTL1.OSCCLKSRCSEL.
7. Check the MCLKSTS bit in the MCDCR register. If it's set, the oscillator has not finished powering up, and more time is required:
   a. Clear the missing clock status by writing a 1 to MCDCR.MCLKCLR.
   b. Repeat steps 2-7. Do not reset the device. Doing so will power down the oscillator, which requires the procedure to be restarted from step 1.
   c. If the oscillator has not finished powering up in 10 milliseconds, there is a real clock failure.
8. If MCDCR.MCLKSTS is clear, the oscillator startup is a success. The system clock is now derived from XTAL.

3.7.7.1 X1/X2 Precondition Circuit

The GPIO19/18 alternate functionality on X1/X2 can be used to speed up the start-up time of the crystal by as much as 30% if needed. This functionality is achieved by preconditioning the load capacitors CL1 and CL2 to a known state before the XTAL is turned on.

The steps below outline the procedure to precondition X1/X2 before turning on the XTAL:

1. ClkCfgRegs.XTALCR2.bit.XIF = 1; // Precondition X1 to High
2. ClkCfgRegs.XTALCR2.bit.XOF = 1; // Precondition X2 to High
3. ClkCfgRegs.XTALCR2.bit.FEN = 1; // Enable X1/X2 Precondition
4. DEVICE_DELAY_US(1);
5. ClkCfgRegs.XTALCR.bit.OSCOFF = 0; // Removes Precondition and Turns on the XTAL
6. ClkCfgRegs.XTALCR2.bit.FEN = 0; // Disables X1/X2 Precondition

3.7.8 Using an External Oscillator

The procedure for using an external oscillator connected to the X1 pin is similar to the procedure for using a crystal or resonator:

1. Clear the XTALCR.OSCOFF bit.
2. Set the XTALCR.SE bit to enable single-ended mode.
3. Clear the X1 counter by writing a 1 to X1CNT.CLR and keep clearing until the X1 counter value in the X1CNT register is no longer saturated 2047 (0x7ff).

4. Wait for the X1 counter value in the X1CNT register to reach 2047 (0x7ff).
5. Repeat steps 3-4 three additional times.
6. Select XTAL as the OSCCLK source by writing a 1 to CLKSRCCTL1.OSCCLKSRCSEL.
7. Check the MCLKSTS bit in the MCDCR register. If it's set, either the external oscillator or the device has failed.
8. If MCLKSTS is clear, the switch to the external clock is a success. The system clock is now derived from XTAL.

3.7.9 Choosing PLL Settings

The equation shown in Figure 3-4 should be used to configure the PLL.

IMULT is the integer value of the multiplier.

REFDIV is the reference divider for the OSCCLK.

ODIV is the output divider of the PLLRAWCLK.

PLLSYSCLKDIV is the system clock divider.

For the permissible values of the multipliers and dividers, see the documentation for their respective registers.

Many combinations of multiplier and divider can produce the same output frequency. However, the product of the reference clock frequency and the multiplier (known as the VCO frequency) must be in the range specified in the *TMS320F28003x Real-Time Microcontrollers Data Sheet*.

Note

The system clock frequency (PLLSYSCLK) may not exceed the limit specified in the *TMS320F28003x Real-Time Microcontrollers Data Sheet*. This limit does not allow for oscillator tolerance.

3.7.10 System Clock Setup

Once the application requirements are understood, a specific clock configuration can be determined. The default configuration is for INTOSC2 to be used as the system clock (PLLSYSCLK) with a divider of 1. The following procedure should be used to set up the desired application configuration:

Refer to your device SysCtl_setClock() function inside C2000Ware installation for an example.

Recommended sequence to set up the system PLL:

1. Bypass the PLL by clearing SYSPLLCTL1[PLLCLKEN]. Allow at least 60 NOP instructions for this to take effect.
2. Power down the PLL by writing to SYSPLLCTL1.PLLEN=0 and allow at least 60 NOP instructions for this to take effect.
3. Select the reference clock source (OSCCLK) by writing to CLKSRCCTL1.OSCCLKSRCSEL. Allow at least 300 NOP instructions for this to take effect.
4. Set the system clock divider to "/1" to ensure the fastest PLL configuration by clearing SYSCLKDIVSEL[PLLSYSCLKDIV].
5. Set the IMULT, REFDIV, and ODIV simultaneously by writing 32-bit value in SYSPLLMULT at once. This will automatically enable the PLL. Be sure the settings for the multiplier and dividers do not violate the frequency specifications defined in the *TMS320F28003x Real-Time Microcontrollers Data Sheet*.
6. Wait for PLL to lock by polling for lock status bit to go high, that is, SYSPLLSTS.LOCKS=1
7. Configure DCC with reference clock as OSCCLK and clock under measurement as PLLRAWCLK, and verify the frequency of the PLL. If the frequency is out of range, do not enable PLLRAWCLK as SYSCLK, stop here and troubleshoot. Refer to DCC chapter for more information on its configuration and usage.
8. Switch to the PLL as the system clock by setting SYSPLLCTL1[PLLCLKEN].

Note

1. SYSPLL must be bypassed and powered down manually before changing the OSCCLK source.
2. At least 60 CPU clock cycles delay is needed after bypassing PLL, that is, SYSPLLCTL1.PLLCLKEN=0.
3. At least 60 CPU clock cycles delay is needed after PLL is powered down, that is, SYSPLLCTL1.PLLEN=0.
4. At least 300 CPU clock cycles delay is needed after OSSCLK source is changed.
5. PLL SLIP bit is not supported. DCC should be used to check the validity of the PLL clock. This feature is included as part of SysCtl_setClock() function inside C2000Ware.

3.7.11 SYS PLL Bypass

If the application requires the PLL clock to be bypassed from the system, then it needs to configure SYSPLLCTL1.PLLCLKEN=0. It takes up to 60 CPU clock cycles before the bypass is effective. In the meantime, if PLLSYSCLKDIV is reduced to a lower value (for example from /2 to /1 or /4 to /2) the device may be clocked above the maximum rated frequency and can lead to unpredictable device behavior. Hence, a delay of 60 CPU clock cycles is required after bypassing the PLL from enable state, that is, going from PLLCLKEN=1 to PLLCLKEN=0.

3.7.12 Clock (OSCCLK) Failure Detection

In order to achieve safety diagnostic there are two circuits available, which can detect failure of PLL reference clock.

1. Missing Clock Detection (MCD)
2. PLL Reference Clock Detection

Table 3-8 lists the details of both circuits.

Table 3-8. Clock Source (OSCCLK) Failure Detection

Clock Failure Detection Circuitry	Clocks Detected	Time for Detection (in Cycles)	Limitations
Missing Clock Detection (MCD)	INTOSC2, XTAL/X1	8192 INTOSC1 cycles	Cannot detect INTOSC1 clock failure.
PLL Reference Clock Detection	INTOSC1, INTOSC2, XTAL/X1	10 OSCCLK cycles	Requires PLL to be enabled.

3.7.12.1 Missing Clock Detection

The missing clock detect (MCD) logic detects OSCCLK failure, using INTOSC1 as the reference clock source. This circuit only detects complete loss of OSCCLK and doesn't do any detection of frequency drift on the OSCCLK.

This circuit monitors the OSCLK (primary clock) using the 10 MHz clock provided by the INTOSC1 (secondary clock) as a backup clock. This circuit functions as below:

1. The primary clock (OSCCLK) clock keeps ticking a 7-bit counter (named as MCDPCNT). This counter is asynchronously reset with XRSn.
2. The secondary clock (INTOSC1) clock keeps ticking a 13-bit counter (named as MCDSCNT). This counter is asynchronously reset with XRSn.
3. Each time MCDPCNT overflows, the MCDSCNT counter is reset. Thus, if OSCCLK is present or not slower than INTOSC1 by a factor of 64, MCDSCNT will never overflow.
4. If OSCCLK stops for some reason, or is slower than INTOSC1 by at least a factor of 64, the MCDSCNT will overflow and a missing clock condition will be detected on OSCCLK.
5. The above check is continuously active, unless the MCD is disabled using MCDCR register (by making the MCLKOFF bit 1)

6. If the circuit ever detects a missing OSCCLK, the following occurs:
   - The MCDSTS flag is set
   - The MCDSCNT counter is frozen to prevent further missing clock detection
   - The CLOCKFAIL signal goes high, which generates TRIP events to PWM modules and fires NMIs to CPU1.NMIWD.
   - PLL is forcefully bypassed and OSCCLK source is switched to INTOSC1 (New, System Clock Frequency = INTOSC1 Freq 10 MHz)/SYSDIV). In the meantime, when clock switches to INTOSC1 System runs on PLL limp Clock.
   - SYSPLLMULT.IMULT is zeroed out automatically in this case.
   - While the MCDSTS bit is set, the OSCCLKSRCSEL bits have no effect and OSCCLK is forcefully connected to INTOSC1.
   - PLLRAWCLK going to the system is switched to INTOSC1 automatically
7. If the MCLKCLR bit is written (this is a W=1 bit), MCDSTS bit will be cleared and OSCCLK source will be decided by the OSCCLKSRCSEL bits. Writing to MCLKCLR will also clear the MCDPCNT and MCDSCNT counters to allow the circuit re-evaluate missing clock detection. If user wants to lock the PLL after missing clock detection, he needs to first switch the clock source to INTOSC1 (using OSCCLKSRCSEL register), do a MCLKCLR and re-lock the PLL.
8. The MCD is enabled at power up.

Figure 3-9 shows the missing clock logic functional flow.

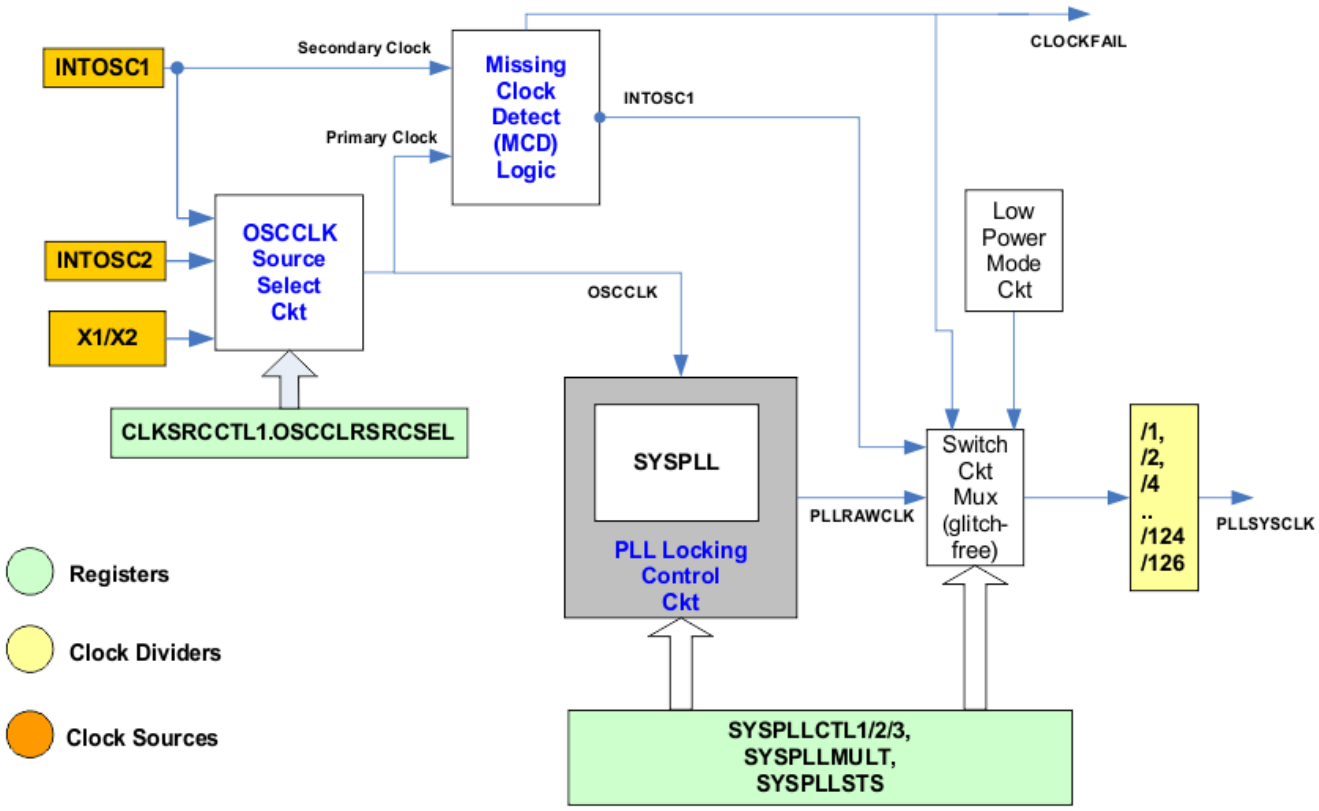


Figure 3-9. Missing Clock Detection Logic

Note

On a complete clock failure when OSCCLK is dead, it may take a maximum time of 8192 INTOSC1 cycles (that is, 0.8192 ms) before the CLOCKFAIL signal goes high, after which:

- NMI is generated
- OSCCLK is switched to INTOSC1
- PWM Trip happens

3.7.12.2 PLL Reference Clock Lost Detection

This circuit monitors reference clock to PLL (OSCCLK) and triggers Clock failure only on a complete loss of OSCCLK and not on OSCCLK frequency drift. Unlike MCD this circuit requires PLL to be configured and running. Features of PLL Reference Clock Lost Detection:

1. Response time for clock failure detection is much faster compared to MCD.
2. It can detect clock failure for any clock sources used, that is, INTOSC1, INTOSC2, and XTAL/X1. If INTOSC1 is used as OSCCLK, the steps for configuration are different and explained below in this section.

3.7.12.2.1 When OSCCLK = INTOSC2 or XTAL/X1

Recommended steps to configure PLL Reference Clock Lost Detection circuit:

1. Configure PLL as per the recommended sequence described in Section 3.7.10.
2. Enable Reference Clock Lost to trigger a CLOCKFAIL event by configuring MCDCR.SYSREF_LOST_MCD_EN=1.

When PLL Reference Clock Lost Detection subsystem detects that OSCCLK is missing:

- PLL is bypassed and OSCCLK is connected to INTOSC1 (after the PLLSYSCLK divider). In the meantime when clock switches to INTOSC1, system runs on PLL Limp Clock.
- SYSPLLMULT.IMULT is zeroed out automatically.
- The MCDCR.SYSREF_LOSTS and SYSPLLSTS.REF_LOSTS flag is set. While this flag is set, the OSCCLKSRCSEL bits have no effect.
- The CLOCKFAIL signal goes high, which generates trip events to the PWM modules and triggers an NMI.

To clear a Reference Clock Lost condition, write a 1 to the MCDCR.SYSREF_LOSTSCLR bit, this will restore the functionality of the OSCCLKSRCSEL. To re-lock the PLL during the loss of Reference Clock Lost, switch the OSCCLK source to INTOSC1 using the OSCCLKSRCSEL bits, clear the missing clock condition, then write to the PLL registers.

3.7.12.2.2 When OSCCLK = INTOSC1

Recommended steps to configure PLL Reference Clock Lost Detection when OSCCLK= INTOSC1:

1. Configure PLL as per the recommended sequence described in Section 3.7.10.
2. Disable MCD by configuring MCDCR.MCLKOFF=1.
3. Disable Reference Clock Lost to trigger a CLOCKFAIL event by configuring MCDCR.SYSREF_LOST_MCD_EN=0.

When PLL Reference Clock Lost Detection subsystem detects that OSCCLK is missing:

- System runs on PLL Limp Clock.
- SYSPLLMULT.IMULT stays as it is and does not get zeroed out automatically.
- The SYSPLLSTS.REF_LOSTS flag is set. While this flag is set, the OSCCLKSRCSEL bits have no effect.
- Since MCD is disabled it does not generate CLOCKFAIL signal, so it does not trigger NMI interrupt nor generate EPWM trip event. Application SW needs to monitor SYSPLLSTS.REF_LOSTS flag to determine clock failure and then generate trip event for EPWM.

To clear a Reference Clock Lost condition, write a 1 to the MCDCR.SYSREF_LOSTSCLR bit, this will restore the functionality of the OSCCLKSRCSEL. To re-lock the PLL during the loss of Reference Clock Lost, switch the OSCCLK source to INTOSC2 or XTAL/X1 using the OSCCLKSRCSEL bits, then write to the PLL registers.

Note

- It is recommended to use OSCCLK=INTOSC2 or XTAL/X1 and not INTOSC1. INTOSC1 should be treated as a fallback clock ONLY.
- If clock failure happens when OSCCLK=INTOSC1 and MCD is not turned OFF then SYSCLK will be dead and system will hang up.

3.8 32-Bit CPU Timers 0/1/2

This section describes the three 32-bit CPU timers (TIMER0/1/2) shown in Figure 3-10.

Timer0 and Timer1 can be used in user applications. Timer2 is reserved for real-time operating system uses (for example, TI-RTOS). If the application is not using an operating system that utilizes this timer, then Timer2 can be used in the application. timer interrupt signals (TINT0, TINT1, TINT2) are connected as shown in Figure 3-11.

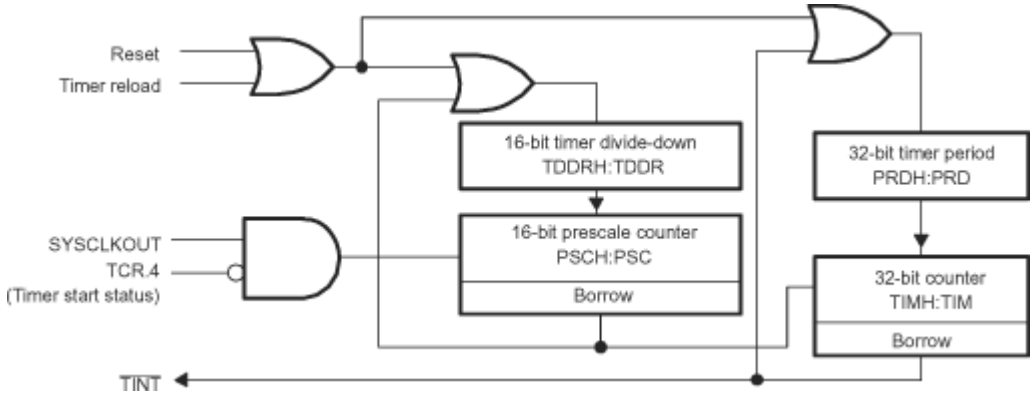
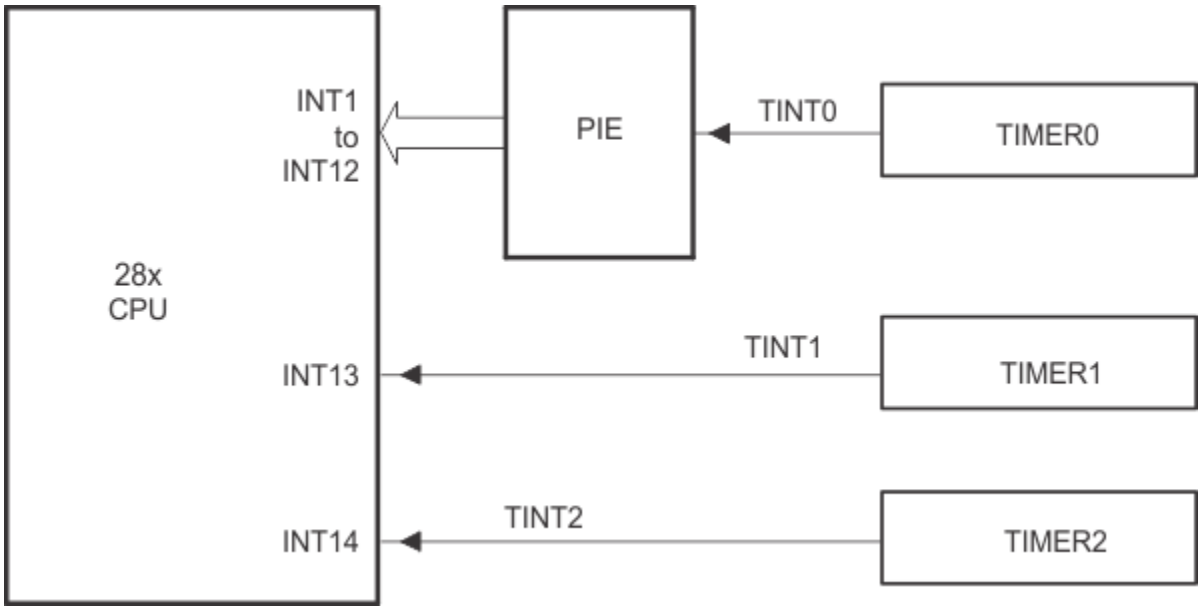


Figure 3-10. CPU Timers



A. The timer registers are connected to the memory bus of the C28x processor.
B. The CPU Timers are synchronized to SYSCLKOUT.

Figure 3-11. CPU Timer Interrupt Signals and Output Signal

The general operation of a CPU timer is as follows:

- The 32-bit counter register, TIMH:TIM, is loaded with the value in the period register PRDH:PRD
- The counter decrements once every (TPR[TDDRH:TDDR]+1) SYSCLK cycles, where TDDRH:TDDR is the timer divider.
- When the counter reaches 0, a timer interrupt output signal generates an interrupt pulse.

The registers listed in Section 3.15 are used to configure the timers.

3.9 Watchdog Timer

The watchdog module consists of an 8-bit counter fed by a prescaled clock (WDCLK, which is connected to INTOSC1). When the counter reaches its maximum value, the module generates an output pulse 512 WDCLKs wide. This pulse can generate an interrupt or a reset. The CPU must periodically write a 0x55 + 0xAA sequence into the watchdog key register to reset the watchdog counter. The counter can also be disabled.

The counter's clock is divided down from WDCLK by two dividers. The prescaler is adjustable from /1 to /64 in powers of two. The pre-divider defaults to /512 for backwards compatibility, but is adjustable from /2 to /4096 in powers of two. This allows a wide range of timeout values for safety-critical applications.

Figure 3-12 shows the various functional blocks within the watchdog module.

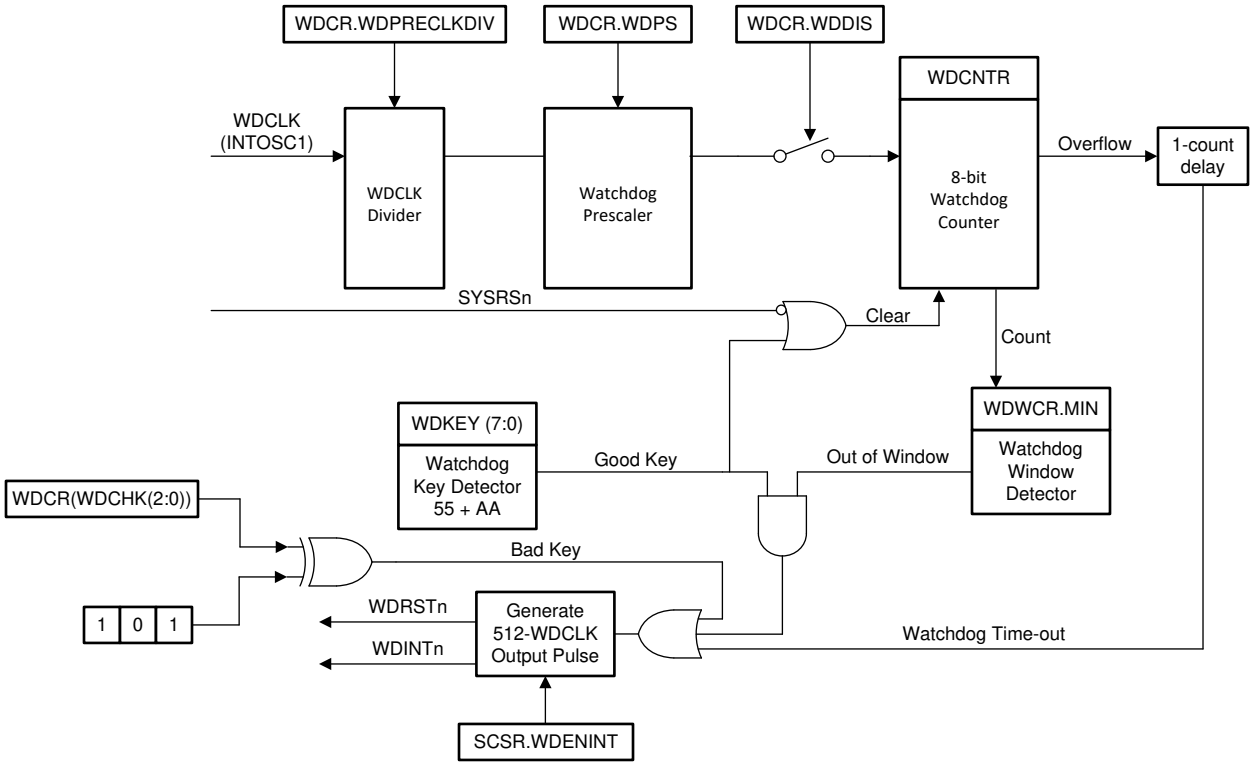


Figure 3-12. Watchdog Timer Module

3.9.1 Servicing the Watchdog Timer

The watchdog counter (WDCNTR) is reset when the proper sequence is written to the WDKEY register before the 8-bit watchdog counter overflows. The WDCNTR is reset-enabled when a value of 0x55 is written to the WDKEY. When the next value written to the WDKEY register is 0xAA, then the WDCNTR is reset. Any value written to the WDKEY other than 0x55 or 0xAA causes no action. Any sequence of 0x55 and 0xAA values can be written to the WDKEY without causing a system reset; only a write of 0x55 followed by a write of 0xAA to the WDKEY resets the WDCNTR.

The first action that enables the WDCNTR to be reset is shown in Step 3 in Table 3-9. The WDCNTR is not actually reset until step 6. Step 8 again re-enables the WDCNTR to be reset and step 9 resets the WDCNTR. Step 10 again re-enables the WDCNTR to be reset. Writing the wrong key value to the WDKEY in step 11 causes no action, however the WDCNTR is no longer enabled to be reset and the 0xAA in step 12 now has no effect.

If the watchdog is configured to reset the device, then a WDCR overflow or writing the incorrect value to the WDCR[WDCHK] bits will reset the device and set the watchdog flag (WDRSn) in the reset cause register (RESC). After a reset, the program can read the state of this flag to determine whether the reset was caused by the watchdog. After doing this, the program should clear WDRSn to allow subsequent watchdog resets to be detected. Watchdog resets are not prevented when the flag is set.

Table 3-9. Example Watchdog Key Sequences

Step	Value Written to WDKEY	Result
1	0xAA	No action
2	0xAA	No action
3	0x55	WDCNTR is enabled to be reset if next value is 0xAA.
4	0x55	WDCNTR is enabled to be reset if next value is 0xAA.
5	0x55	WDCNTR is enabled to be reset if next value is 0xAA.
6	0xAA	WDCNTR is reset.
7	0xAA	No action
8	0x55	WDCNTR is enabled to be reset if next value is 0xAA.
9	0xAA	WDCNTR is reset.
10	0x55	WDCNTR is enabled to be reset if next value is 0xAA.
11	0x32	Improper value written to WDKEY. No action, WDCNTR no longer enabled to be reset by next 0xAA.
12	0xAA	No action due to previous invalid value.
13	0x55	WDCNTR is enabled to be reset if next value is 0xAA.
14	0xAA	WDCNTR is reset.

3.9.2 Minimum Window Check

To complement the timeout mechanism, the watchdog also contains an optional "windowing" feature that requires a minimum delay between counter resets. This can help protect against error conditions that bypass large parts of the normal program flow but still include watchdog handling.

To set the window minimum, write the desired minimum watchdog count to the WDWCR register. This value will take effect after the next WDKEY sequence. From then on, any attempt to service the watchdog when WDCNTR is less than WDWCR will trigger a watchdog interrupt or reset. When WDCNTR is greater than or equal to WDWCR, the watchdog can be serviced normally.

At reset, the window minimum is zero, which disables the windowing feature.

3.9.3 Watchdog Reset or Watchdog Interrupt Mode

The watchdog can be configured in the SCSR register to either reset the device ($\overline{\text{WDRST}}$) or assert an interrupt ($\overline{\text{WDINT}}$) if the watchdog counter reaches its maximum value. The behavior of each condition is:

- **Reset mode:**

  If the watchdog is configured to reset the device, then the $\overline{\text{WDRST}}$ signal will pull the device reset ($\overline{\text{XRS}}$) pin low for 512 OSCCLK cycles when the watchdog counter reaches its maximum value.
- **Interrupt mode:**

  When the watchdog counter expires, it will assert an interrupt by driving the $\overline{\text{WDINT}}$ signal low for 512 OSCCLK cycles. The falling edge of $\overline{\text{WDINT}}$ triggers a WAKEINT interrupt in the PIE if it is enabled. Because the PIE is edge-triggered, re-enabling the WAKEINT while $\overline{\text{WDINT}}$ is active will not produce a duplicate interrupt.

  To avoid unexpected behavior, software should not change the configuration of the watchdog while $\overline{\text{WDINT}}$ is active. For example, changing from interrupt mode to reset mode while $\overline{\text{WDINT}}$ is active will immediately reset the device. Disabling the watchdog while $\overline{\text{WDINT}}$ is active will cause a duplicate interrupt if the watchdog is later re-enabled. If a debug reset is issued while $\overline{\text{WDINT}}$ is active, the reset cause register (RESC) will show a watchdog reset. The WDINTS bit in the SCSR register can be read to determine the current state of $\overline{\text{WDINT}}$.

3.9.4 Watchdog Operation in Low Power Modes

In IDLE mode, the watchdog interrupt ($\overline{\text{WDINT}}$) signal can generate an interrupt to the CPU to take the CPU out of IDLE mode. As with any other peripheral, the watchdog interrupt will trigger a WAKE interrupt in the PIE during IDLE mode. User software must determine which peripheral caused the interrupt.

Note: If the watchdog interrupt is used to wake-up from an IDLE low power mode condition, software must make sure that the $\overline{\text{WDINT}}$ signal goes back high before attempting to reenter the IDLE mode. The $\overline{\text{WDINT}}$ signal will be held low for 512 OSCCLK cycles when the watchdog interrupt is generated. The current state of $\overline{\text{WDINT}}$ can be determined by reading the watchdog interrupt status bit (WDINTS) bit in the SCSR register. WDINTS follows the state of $\overline{\text{WDINT}}$ by two SYSCLKOUT cycles.

In HALT mode, the internal oscillators and watchdog timer are kept active if the user sets CLKSRCCTL1.WDHALTI = 1. A watchdog reset can wake the system from HALT mode, but a watchdog interrupt cannot.

3.9.5 Emulation Considerations

The watchdog module behaves as follows under various debug conditions:

CPU Suspended	When the CPU is suspended, the watchdog clock (WDCLK) is suspended.
Run-Free Mode	When the CPU is placed in run-free mode, then the watchdog module resumes operation as normal.
Real-Time Single-Step Mode	When the CPU is in real-time single-step mode, the watchdog clock (WDCLK) is suspended. The watchdog remains suspended even within real-time interrupts.
Real-Time Run-Free Mode	When the CPU is in real-time run-free mode, the watchdog operates as normal.

3.10 Low Power Modes

This device has HALT, IDLE, and STANDBY as clock-gating low-power modes.

All low-power modes are entered by setting the LPMCR register and executing the IDLE instruction. More information about this instruction can be found in the *TMS320C28x CPU and Instruction Set Reference Guide*.

Low-power modes should not be entered into while a Flash program or erase operation is ongoing. Entering HALT will stop all CPU and peripheral activities. This includes active transmissions and control algorithms. When preparing to enter HALT mode, the application should ensure that the system is prepared to enter a period of inactivity.

Before entering HALT mode, check the value of the GPIODAT register of the pin selected for HALT wake-up (GPIOLPMSEL0/1) prior to entering the low-power mode to ensure that the wake event has not already been asserted.

3.10.1 Clock-Gating Low-Power Modes

IDLE and HALT modes on this device are similar to those on other C28x devices. Table 3-10 describes the effect on the system when any of the clock-gating low-power modes are entered.

Table 3-10. Effect of Clock-Gating Low-Power Modes on the Device

Modules/ Clock Domain	IDLE	STANDBY	HALT
SYSCLK	Active	Gated	Gated
CPUCLK	Gated	Gated	Gated
Clock to modules connected to PERx.SYSCLK	Active	Gated	Gated
WDCLK	Active	Active	Gated if CLKSRCCTL1.WDHALTI = 0
PLL	Powered	Powered	Software must power down PLL before entering HALT.
INTOSC1	Powered	Powered	Powered down if CLKSRCCTL1.WDHALTI = 0
INTOSC2	Powered	Powered	Powered down if CLKSRCCTL1.WDHALTI = 0
Flash(1)	Powered	Powered	Powered
XTAL(2)	Powered	Powered	Powered

(1) The Flash module is not powered down by hardware in any LPM. It may be powered down using software if required by the application. For more information, see the *Flash Module* chapter.
(2) The XTAL is not powered down by hardware in any LPM. It may be powered down by software setting the XTALCR.OSCOFF bit to 1. This can be done at any time during the application if the XTAL is not required.

3.10.2 IDLE

IDLE is a standard feature of the C28x CPU. In this mode, the CPU clock is gated while all peripheral clocks are left running. IDLE can thus be used to conserve power while a CPU is waiting for peripheral events.

Any enabled interrupt will wake the CPU up from IDLE mode.

To enter IDLE mode, set LPMCR.LPM to 0x0 and execute the IDLE instruction.

The CPU will resume normal operations upon any enabled interrupt event.

3.10.3 STANDBY

STANDBY is a more aggressive low-power mode that gates both the CPU clock and any peripheral clocks derived from the CPU's SYSCLK. The watchdog however, is left active. STANDBY is best suited for an application where the wake-up signal will come from an external system (or CPU subsystem) rather than a peripheral input.

An NMI (or optionally) a watchdog interrupt or a configured GPIO can wake the CPU from STANDBY mode. Each GPIO from GPIO0-60 can be configured to wake the CPU when they are driven active low. Upon wakeup, the CPU receives the WAKEINT interrupt if configured.

To enter STANDBY mode:

1. Set LPMCR.LPM to 0x1.
2. Enable the WAKEINT interrupt in the PIE.
3. For watchdog interrupt wakeup, set LPMCR.WDINTE to 1 and configure the watchdog to generate interrupts.
4. For GPIO wakeup, set GPIOLPMSEL0 and GPIOLPMSEL1 to connect the chosen GPIOs to the LPM module, and set LPMCR.QUALSTDBY to select the number of OSCCLK cycles for input qualification.
5. Execute the IDLE instruction to enter STANDBY.

To wake up from STANDBY mode:

1. Configure the desired GPIO to trigger the wakeup.
2. Drive the selected GPIO signal low; it must remain low for the number of OSCCLK cycles specified in the QUALSTDBY bits in the LPMCR register. If the signal is sampled high during this period, the count restarts.

At the end of the qualification period, the PLL enables the CLKIN to the CPU and the WAKEINT interrupt is latched in the PIE block.

The CPU is now out of STANDBY mode and can resume normal execution.

3.10.4 HALT

HALT is a global low-power mode that gates almost all system clocks and allows for power-down of oscillators and analog blocks.

Unlike on other C2000™ devices, HALT mode will not automatically power down the XTAL upon HALT entry. Additionally, if the XTAL is not powered on, waking up from HALT mode will not automatically power on the XTAL. The XTALCR.OSCOFF bit has been added to power on and off the XTAL circuitry when not needed through application software.

For applications that require minimal power consumption during HALT mode, application software should power off the XTAL prior to entering HALT. If the OSCCLK source is configured to be XTAL, the application should first switch the OSSCLK source to INTOSC1 or INTOSC2 prior to setting XTALCR.OSCOFF.

GPIO0-60 can be configured to wake up the system from HALT. No other wakeup option is available. However, the watchdog timer may still be clocked, and can be configured to produce a watchdog reset if a timeout mechanism is needed. On wakeup, the CPU receives a WAKEINT interrupt.

To enter HALT mode:

1. Enable the WAKEINT interrupt in the PIE .
2. Set LPMCR.LPM to 0x2. Set GPIOLPMSEL0 and GPIOLPMSEL1 to connect the chosen GPIOs to the LPM module.
3. Set CLKSRCCTL1.WDHALTI to 1 to keep the watchdog timer active and INTOSC1 and INTOSC2 powered up in HALT.
4. Set CLKSRCCTL1.WDHALTI to 0 to disable the watchdog timer and power down INTOSC1 and INTOSC2 in HALT.
5. Execute the IDLE instruction to enter HALT.

If an interrupt or NMI is received while the IDLE instruction is in the pipeline, the system will begin executing the WAKEINT ISR. After HALT wakeup, ISR execution will resume where it left off.

Note

Before entering HALT mode, if the system PLL is locked (SYSPLL.LOCKS = 1), it must also be connected to the system clock (PLLCTL1.PLLCLKEN = 1). Otherwise, the device will never wake up.

To wake up from HALT mode:

1. Drive the selected GPIO low for a minimum 5us. This will activate the WAKEINT PIE interrupt.
2. Drive the wake-up GPIO high again to initiate the powering up of the SYSPLL
3. Wait 16us plus 1024 OSCLK cycles to allow the PLLs to lock and the WAKEINT ISR to be latched.
4. Execute the WAKEINT ISR.

The device is now out of HALT mode and can resume normal execution.

3.10.5 Flash Power-down Considerations

The Flash module on this device can be powered down at any time during an application. There are some considerations that must be made when powering down the Flash.

When the application software powers down the Flash, it must ensure that the function that puts the Flash to sleep is executed from RAM. Note that there should not be any access to Flash after the Flash is put to sleep to realize the power savings. If there is an access to Flash, Flash wakeup process (wakeup time depends on PSLEEP and RWAIT) gets initialized and the application will not realize Flash power savings. For example, if the application has to execute any code after putting the Flash to sleep and before putting the device in to low power mode, the application should execute that code from RAM and not from Flash.

PSLEEP and RWAIT can be optimized to reduce the Flash wakeup time for a given SYSCLK frequency. BootROM configures the best possible PSLEEP value for the 120 MHz operation. However, the application software can decrease the PSLEEP value to reduce the Flash wakeup time if the application SYSCLK is less than 120 MHz. This is applicable in the context of an application entering the Halt mode since PLL must be disabled before entering the Halt mode. In this case, the PSLEEP value can be decreased to get a faster Flash wakeup upon exit from LPM.

If the Wake ISR is in Flash, it is suggested to optimize the PSLEEP and RWAIT values before entering the LPM, and after the Flash is in sleep, since application does not get a chance to modify these before Flash is awake after exiting from LPM. However, after the LPM exit and once the Flash is awake, application should branch to RAM to restore RWAIT and PSLEEP (as per the application SYSCLK to which PLL will be locked for) and then proceed with Flash execution to lock the PLL.

If the Wake ISR is in RAM, application can optimize the PSLEEP and RWAIT values in the Wake ISR and then do a dummy Flash access to initiate the Flash wakeup process. While the Flash is waking up, application can initialize the PLL lock process. Once the Flash is awake, application can put the PLL in clock path. If the user does not want to lock the PLL from RAM, PLL can be locked from Flash (this means Flash wakeup and PLL lock are not done in parallel), but in any case make sure to restore the RWAIT and PSLEEP (as per the application SYSCLK to which PLL will be locked for) in Wake ISR before proceeding to Flash execution.

3.11 Memory Controller Module

On this device, the RAMs have different characteristics. These are:

- Dedicated to the CPU: M0, M1 RAMs
- Shared between the CPU and CLA: LSx RAMs
- Shared between the CPU, DMA and HIC: GSx RAMs
- Used to send and receive messages between processors: MSG RAMs

All these RAMs are highly configurable to achieve control for write access and fetch access from different masters. All the RAMs are enabled with the ECC feature (both data and address). Some of the memories are secure memory as well. Refer to the *Dual Code Security Module (DCSM)* chapter for more details. Each RAM has its own controller which takes care of the access protection/security related checks and ECC features for that RAM. Figure 3-13 shows the configuration of these RAMs.

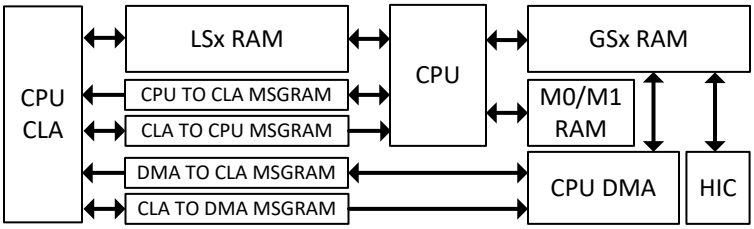


Figure 3-13. Memory Architecture

3.11.1 Dedicated RAM (Mx RAM)

This device has two dedicated RAM blocks: M0 and M1. M0 and M1 memories are small blocks of memory which are tightly coupled with the CPU. Only the CPU has access to these memories. No other masters (CLA, DMA or HIC) have access to these memories.

All dedicated RAMs have ECC and access protection (CPU write protection/CPU fetch protection) feature. Each type of access protection for each RAM block can be enabled/disabled by configuring the specific bit in the access protection register, allocated to each subsystem (DxACCPROT).

3.11.2 Local Shared RAM (LSx RAM)

Local shared RAMs (LSx RAMs) are secure memories and have ECC. These memories are shared between the CPU and CLA but are by default dedicated to the CPU only. CLA access can be enabled by configuring MSEL_LSx bit field in the LSxMSEL register.

Further, when these memories are shared between the CPU and CLA, the user could choose to use these memories as CLA program memory by configuring the CLAPGM_LSx bit field in the LSxCLAPGM registers. CPU access to all memory blocks, which are programmed as CLA program memory, are blocked.

All these RAMs have the access protection (CPU write and CPU fetch) feature. Each type of access protection for each RAM block can be enabled or disabled by configuring the specific bit in the local shared RAM access protection registers, mapped to each CPU subsystem. Table 3-11 shows the LSx RAM features.

Table 3-11. Local Shared RAM

MSEL_LSx	CLAPGM_LSx	CPUx Allowed Access	CPUx.CLA1 Allowed Access	Comment
00	X	All	-	LSx memory is configured as CPU dedicated RAM
01	0	All	Data Read Data Write	LSx memory is shared between CPU and CLA1
01	1	Emulation Read Emulation Write	Fetch Only Emulation Read Emulation Write	LSx memory is CLA1 program memory

3.11.3 Global Shared RAM (GSx RAM)

RAM blocks which are accessible from the CPU, DMA and HIC are called global shared RAMs (GSx RAMs). Table 3-12 shows the features of the GSx RAM.

Table 3-12. Global Shared RAM

CPU (Fetch)	CPU (Read)	CPU (Write)	CPU.DMA (Read)	CPU.DMA (Write)	HIC (Read)	HIC (Write)
Yes	Yes	Yes	Yes	Yes	Yes	Yes

The shared RAM has different levels of access protection that can be enabled or disabled by configuring specific bits in the GSxACCPROT registers.

Access protection configuration for the GSx RAM block can be locked by the user to prevent further updates to this bit field. The user can also choose to permanently lock the configuration to individual bit fields by setting the specific bit fields in the GSxCOMMIT register (refer to the register description for more details). Once a configuration is committed for a particular GSx RAM block, it can not be changed further until CPU.$\overline{\text{SYSRS}}$ is issued.

3.11.4 CLA-CPU Message RAM

These RAM blocks can be used to share data between the CPU and CLA. The CLA has read and write access to the "CLA to CPU MSGRAM." The CPU has read and write access to the "CPU to CLA MSGRAM." The CPU and CLA both have read access to both MSGRAMs.

3.11.5 CLA-DMA Message RAM

These RAMs blocks can be used to share data between CLA and DMA. The CLA has read and write access to the "CLA to DMA MSGRAM." The DMA has read and write access to the "DMA to CLA MSGRAM." The CLA and DMA both have read access to both MSGRAMs.

3.11.6 Access Arbitration

For a shared RAM, multiple accesses can happen at any given time. The maximum number of accesses to any shared RAM at any given time depends on the type of shared RAM. On this device, a combination of a fixed and round-robin scheme is followed to arbitrate multiple access at any given time.

The following is the order of fixed priority for CPU accesses:

1. Data Write/Program Write
2. Data Read
3. Program Read/Program Fetch

The following is the order of fixed priority for CLA accesses:

1. Data Write
2. Data Read/Program Fetch

Figure 3-14 represents the arbitration scheme on global shared memories.

Figure 3-15 represents the arbitration scheme on local shared memories.

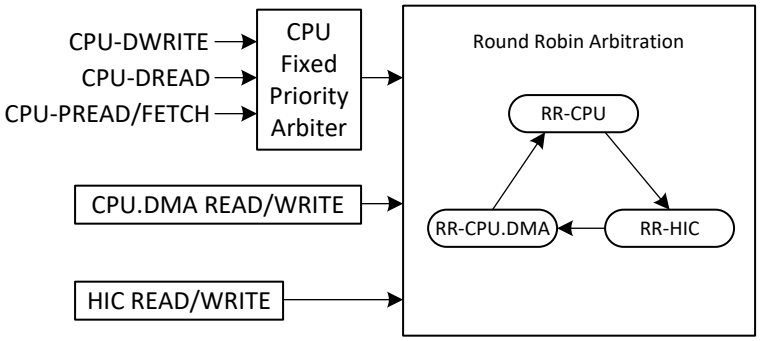


Figure 3-14. Arbitration Scheme on Global Shared Memories

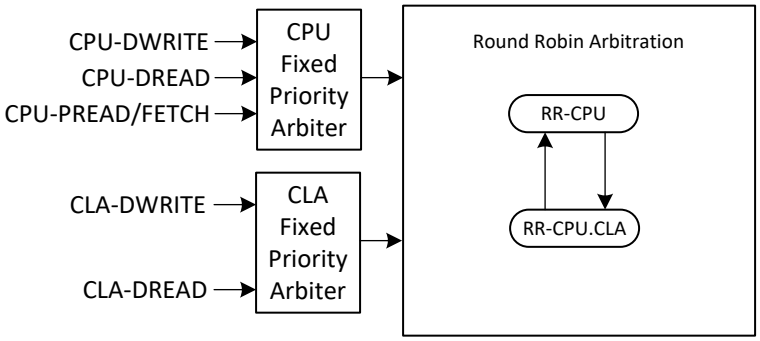


Figure 3-15. Arbitration Scheme on Local Shared Memories

3.11.7 Access Protection

The RAM blocks have different levels of protection. This feature allows the user to enable or disable specific access to individual RAM blocks from individual masters. There is no protection for read accesses, hence reads are always allowed from all the masters which have access to that RAM block.

The following sections describe the different kinds of protection available for RAM blocks on this device.

Note

For debug accesses, all the protections are disabled.

Note 1: All access protections are ignored during debug accesses. Write access to a protected memory will go through when it is done via the debugger, irrespective of the write protection configuration for that memory.

3.11.7.1 CPU Fetch Protection

Fetch accesses from the CPU can be protected by setting the FETCHPROTx bit of the specific register to '1.' If fetch access is done by the CPU to a memory where CPU fetch protection is enabled, a fetch protection violation occurs.

If a fetch protection violation occurs, it results in an ITRAP for CPU. A flag gets set in the appropriate access violation flag register, and the memory address for which the access violation occurred, gets latched into the appropriate CPU fetch access violation address register.

3.11.7.2 CPU Write Protection

Write accesses from the CPU can be protected by setting the CPUWRPROTx bit of the specific register to '1.' If write access is done by a CPU to memory where it is protected, a write protection violation occurs.

If a write protection violation occurs, the write gets ignored, a flag gets set in the appropriate access violation flag register, and the memory address for which the access violation occurred, gets latched into the appropriate CPU write access violation address register. Also, an access violation interrupt is generated if enabled in the interrupt enable register.

3.11.7.3 CPU Read Protection

For local shared RAM, if memory is shared between the CPU and its CLA, the CPU will only have access if the memory is configured as data RAM for the CLA. If it is programmed as program RAM, all the access from the CPU, including a read, will be blocked and the violation will be considered as a non-master access violation.

If a read protection violation occurs, a flag gets set in the appropriate access violation flag register, and the memory address for which the access violation occurred, gets latched in the appropriate CPU read access violation address register. Also, an access violation interrupt is generated, if enabled in the interrupt enable register.

3.11.7.4 CLA Fetch Protection

If local shared RAM is configured as dedicated RAM for the CPU, or if it is configured as data RAM for the CLA, any fetch access from the CLA to that particular LSx RAM results in a CLA fetch protection violation, which is a non-master access violation.

If a CLA fetch protection violation occurs, it results in a MSTOP, a flag gets set into the appropriate access violation flag register, and the memory address for which the access violation occurred, gets latched into the appropriate CLA fetch access violation address register. Also, an access violation interrupt is generated to the master CPU if enabled in the interrupt enable register.

3.11.7.5 CLA Read Protection

If local shared RAM is configured as dedicated RAM for the CPU, or if it is configured as program RAM for the CLA, any data read access from the CLA to that particular LSx RAM results in a CLA read protection violation, which is a non-master access violation.

If a CLA read protection violation occurs, a flag gets set into the appropriate access violation flag register, and the memory address for which the access violation occurred, gets latched into the appropriate CLA read access violation address register. Also, an access violation interrupt is generated to the master CPU if enabled in the interrupt enable register.

3.11.7.6 CLA Write Protection

If local shared RAM is configured as dedicated RAM for the CPU, or if it is configured as program RAM for the CLA, any data write access from the CLA to that particular LSx RAM results in a CLA write protection violation, which is a non-master access violation. Similarly any data write access from CLA to CPUTOCLA or DMATOCLA MSGRAM will result in a CLA write protection violation, which is a non-master access violation.

If a CLA write protection violation occurs, write gets ignored, a flag gets set into the appropriate access violation flag register, and the memory address for which the access violation occurred, gets latched into the appropriate CLA write access violation address register. Also, an access violation interrupt is generated to the master CPU if enabled in the interrupt enable register.

3.11.7.7 HIC Write Protection

Write accesses from the HIC can be protected by setting the HICWRPROTx bit of a specific register to ‘1.’ If a write access is done by the HIC to protected memory, a write protection violation occurs.

If a write access is made to GSx memory by a non-master HIC, it is called a non-master write protection violation. If a write access is made to a dedicated or shared memory by a master HIC, and HICWRPROTx is set to ‘1’ for that memory, it is called a master HIC write protection violation.

A flag gets set in the HIC access violation flag register, and the memory address where the violation happened gets latched in the HIC fetch access violation address register. These are dedicated registers for each subsystem.

3.11.7.8 DMA Write Protection

Write accesses from the DMA can be protected by setting the DMAWRPROTx bit of a specific register to ‘1.’ If a write access is done by the DMA to protected memory, a write protection violation occurs.

If a write access is made to GSx memory by a non-master DMA, it is called a non-master write protection violation. If a write access is made to a dedicated or shared memory by a master DMA, and DMAWRPROTx is set to ‘1’ for that memory, it is called a master DMA write protection violation.

A flag gets set in the DMA access violation flag register, and the memory address where the violation happened gets latched in the DMA fetch access violation address register. These are dedicated registers for each subsystem.

3.11.8 Memory Error Detection and Correction, and Error Handling

These devices have memory error detection and correction features to satisfy safety standards requirements. These requirements warrant the addition of detection mechanisms for finite dangerous failures.

In this device, all RAMs support error correction code (ECC) protection. The ECC scheme used is Single Error Correction Double Error Detection (SECDED). ECC will cover the data bits stored in memory as well as address.

ECC calculation is done inside the memory controller module and written into the memory along with the data. ECC is computed for 16-bit data; hence, for each 32-bit of data, there will be three 7-bit ECC codes, two of which are for data and a third one for the address.

3.11.8.1 Error Detection and Correction

Error detection is done while reading the data from memory. The error detection is performed for data as well as address. For ECC memory, along with a single-bit error, a double-bit error also gets detected. These errors are called correctable and uncorrectable errors. The following are characteristics of these errors:

- Single-bit ECC errors are correctable errors
- Double-bit ECC errors are uncorrectable errors

- Address ECC errors are also uncorrectable errors

Correctable errors get corrected by the memory controller module and then correct data is given back as read data to the master. It is also written back into the memory to prevent a double-bit error due to another single-bit error at the same memory address.

3.11.8.2 Error Handling

For each correctable error, the count in the correctable error count register will increment by one. When the value in this count register becomes equal to the value configured in the correctable error threshold register, an interrupt is generated to the CPU, if the interrupt is enabled in the correctable interrupt enable register. The user needs to configure the correctable error threshold register based on the system requirements. Also, the address for which the error occurred, gets latched into a register and a flag also gets set in a status register.

If there are uncorrectable errors, an NMI gets generated for the CPU. In this case also, the address for which the error occurred gets latched into a register, and a flag gets set in a status register.

Table 3-13 summarizes different error situations that can arise. These need to be handled appropriately in the software, using the status and interrupt indications provided.

Table 3-13. Error Handling in Different Scenarios

Access Type	Error Found In	Error Type	Status Indication	Error Notification
Reads	Data read from memory	Uncorrectable Error (Double bit Error for ECC RAMs)	Yes -CPU/CPU.DMA/ CPU.CLA1/DMA/CLA Read Error Address Register Data returned to CPU/ CPU.DMA/CPU.CLA1 is incorrect	NMI for CPU access NMI for CPU.DMA access NMI to CPU for CPU.CLA1 access
Reads	Data read from memory	Single-bit error for ECC RAMs	Yes - CPU/CPU.DMA CPU/DMA Read Error Address Register Increment single error counter	Interrupt when error counter reaches the user programmable threshold for single errors
Reads	Address	Address error	Yes - CPU/CPU.DMA/ CPU.CLA1/DMA/CLA Read Address Error Register Data returned to CPU/ CPU.DMA/CPU.CLA1 is incorrect	NMI to CPU for CPU access NMI to CPU for CPU.DMA access NMI to CPU for CPU.CLA1 access

Note

In the case of an uncorrectable error during fetch on the CPU, there is the possibility of getting an ITRAP before an NMI exception, since garbage instructions enter into the CPU pipeline before the NMI gets generated.

During debug accesses, correctable as well as uncorrectable errors are masked.

3.11.9 Application Test Hooks for Error Detection and Correction

Since error detection and correction logic is part of safety critical logic, safety applications may need to ensure that the logic is always working fine (during run time also). To enable this, a test mode is provided, in which a user can modify the data bits (without modifying the ECC bits) or ECC bits directly. Using this feature, an ECC error could be injected into data.

Note

The memory map for ECC bits and data bits are the same. The user must choose a different test mode to access ECC bits. In test mode, all access to memories (data as well as ECC) should be done as 32-bit access only.

Table 3-14 shows the bit mapping for the ECC bits when they are read in RAMTEST mode using their respective addresses.

Table 3-14. Mapping of ECC Bits in Read Data from ECC Address Map

Data Bits Location in Read Data	Content (ECC Memory)
6:0	ECC Code for lower 16 bits of data
7	Not Used
14:8	ECC Code for upper 16 bits of data
15	Not Used
22:16	ECC Code for address
31:23	Not Used

3.11.10 RAM Initialization

To ensure that read/fetch from uninitialized RAM locations do not cause ECC errors, the RAM_INIT feature is provided for each memory block. Using this feature, any RAM block can be initialized with 0x0 data and respective ECC bits accordingly. This can be initiated by setting the INIT bit to ‘1’ for the specific RAM block in INIT registers. To check the status of RAM initialization, SW should poll for the INITDONE bit for that RAM block in the INITDONE register to be set. Unless this bit gets set, no access should be made to that RAM memory block.

Note

None of the masters should access the memory while initialization is taking place. If memory is accessed before RAMINITDONE is set, the memory read/write as well as initialization will not happen correctly.

3.12 JTAG

Gel files perform certain initialization tasks. This helps the users in a debug environment. However, when executed standalone (without the emulator connected) the application may not work as expected, since there is no gel file to perform those initializations. For example, gel file disables watchdog. If user code does not service the watchdog in the application (or fails to disable it), there will be a difference in how the application behaves with the debugger and without.

Common tasks performed by the gel files (but not boot-ROM).

On Reset:

- Disable Flash ECC on some devices.
  - Disabling ECC only when using Flash API functions, see the Flash API User Guide for details. Otherwise, TI suggests to always program ECC and enable ECC-check.
- Disable Watchdog
- Enable CLA clock
- Select real-time mode or C28x mode

On Restart:

- Select real-time mode or C28x mode
- Clear IER and IFR

On Target Connect:

- Select real-time mode or C28x mode

3.13 Live Firmware Update

This device includes hardware hooks to streamline firmware updates. These hardware hooks enable seamless switching from the old firmware to the new firmware without resetting the application.

This section discusses the Live Firmware Update (LFU) and the hardware features present on the device to support LFU.

3.13.1 LFU Background

End equipment like Server Power Supply (PSU) are high availability systems that need to have minimum downtime, even during firmware upgrades. Firmware upgrades are essential to add additional functionality, enhance performance and fix software bugs/vulnerabilities. LFU helps update firmware while the application is running, thus eliminating downtime (with respect to critical real-time interrupts) and also providing a more cost-effective alternative compared to manually updating firmware.

LFU has traditionally been implemented in the C2000 family of MCUs using software-only techniques. This impacts LFU switchover time, which is the time to switchover to new firmware once the transition has begun. User application code initiates this transition, typically by jumping to an entry point in the new firmware. There, a compiler provided initialization routine specific to LFU is called. This initializes user-specified data variables. When execution arrives in main() of the new application, user application code performs minimal initialization to get the new application running.

3.13.2 LFU Switchover Steps

A simplified representation of the LFU switchover is shown in Figure 3-16, and is described in the following steps:

1. In typical systems, a host – typically a PC or another MCU, will initiate LFU (depicted as LFU Request) on the application MCU (in this case, the C2000 MCU) that is executing the real-time control application. This initiates the Flash Program sequence in the application MCU. This runs as a background process even as the application MCU continues executing firmware (depicted as Firmware - 1).
2. Since the compiler may move existing PIE vectors and function pointers to new locations between firmware versions, or PIE vectors or function pointers could get added or removed between firmware versions, user application code needs to manage these properly and efficiently during LFU. In the absence of Flash

remapping (where different Flash memory banks can be mapped to the same address), PIE vector table remapping, that is “swapping” and RAM memory block swapping are features supported on the device. Without swapping, user application code would need to individually update each PIE vector and each function pointer, adding valuable cycles to the LFU switchover time. With swapping, prior to LFU switchover, user application code can populate a different PIE vector table (depicted as PIE Swap Memory Update) and a different LS RAM region (depicted as LSx Swap Memory Update).

3. When complete, at a suitable time (depicted as LFU Switchover – waiting for appropriate time), user application code initiates the transition to new firmware. Once the compiler LFU initialization routine completes and transfers execution to the new application (depicted as Firmware – 2), user application code needs to perform necessary initialization before the new application can begin running. Since PIE vectors and function pointers have already been populated in the “swap” locations, all that is required is a PIE vector table swap and LSx RAM Memory Swap (depicted as PIE Vector Swap, LSx Memory Swap).

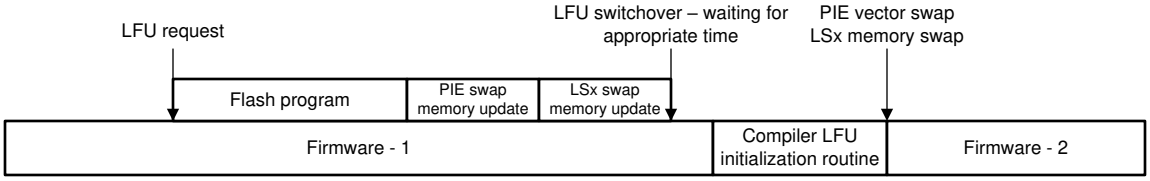


Figure 3-16. Simplified LFU Representation

3.13.3 Device Features Supporting LFU

The new hardware capabilities implemented in the device to support LFU are:

1. Multi-Bank Flash
2. PIE Vector Table Swap
3. LS0/LS1 RAM Memory Swap

3.13.3.1 Multi-Bank Flash

The device has up to three Flash banks, each of size 128KB. With multiple banks, it is possible to Program/Erase a bank while other banks are in read mode.

3.13.3.2 PIE Vector Table Swap

The device contains an additional PIE vector table, in addition to the typical PIE vector table that is present. This allows PIE vector addresses for the new firmware to be populated prior to the LFU switchover. During LFU switchover, a simple swap operation which activates the PIE vector swap table and deactivates the previously active PIE vector table is initiated by user application code, and this operation takes just 1 CPU clock cycle. To initiate the swap, user application code sets LFUConfig.PieVectorSwap = 1. The PIE vector table swap features are also implemented on a redundant PIE vector table implemented for safety. Therefore, to implement PIE vector table swap, the sizes of PIE vector memory and redundant PIE vector memory are both doubled.

The changes are summarized in Figure 3-17. In previous devices, PIE vector RAM and redundant PIE vector RAM are present. In this device, these are duplicated. There are now four physical PIE vector RAM memories – Block A, Block B, Block C, and Block D. By default, Block A and Block B are *active*, and are mapped to addresses 0x0000_0D00-0x0000_0EFF and 0x0100_0D00-0x0100_0EFF respectively. Block C and Block D are *inactive*, and are mapped to addresses 0x0100_0900-0x0100_0AFF and 0x0100_0B00-0x0100_0CFF respectively.

When user application code initiates a PIE vector table swap by setting LFUConfig.PieVectorSwap = 1, Block C and Block D *become active*, and are mapped to addresses 0x0000_0D00-0x0000_0EFF and 0x0100_0D00-0x0100_0EFF respectively. Block A and Block B *become inactive*, and are mapped to addresses 0x0100_0900-0x0100_0AFF and 0x0100_0B00-0x0100_0CFF respectively.

Thus, note that the active addresses are always 0x0000_0D00-0x0000_0EFF, and 0x0100_0D00-0x0100_0EFF (for redundancy). The inactive addresses are always 0x0100_0900-0x0100_0AFF, and 0x0100_0B00-0x0100_0CFF (for redundancy). As mentioned above, prior to the LFU switchover, user application code will need to write to the inactive addresses with the PIE vector locations corresponding to the new firmware.

The register bit LFUStatus.PieVectorSwap provides the status of Pie Vector Swap.

Writes to addresses 0x0000_0D00-0x0000_0EFF will update both the currently active block and its redundant counterpart. Writes to addresses 0x0100_0900-0x0100_0AFF will update both the currently inactive block and its redundant counterpart.

Reads from addresses 0x0000_0D00-0x0000_0EFF will issue reads from both addresses 0x0000_0D00-0x0000_0EFF and the redundant counterpart 0x0100_0D00-0x0100_0EFF. The read values will be compared, and any data mismatches will generate the same error response as that of the existing PIE vector fetch mismatch (refer to PIEVERRADDR). On the other hand, a read from or write to the redundant PIE vector RAM (0x0100_0D00-0x0100_0EFF or 0x0100_0B00-0x0100_0CFF) will impact only the redundant PIE vector RAM.

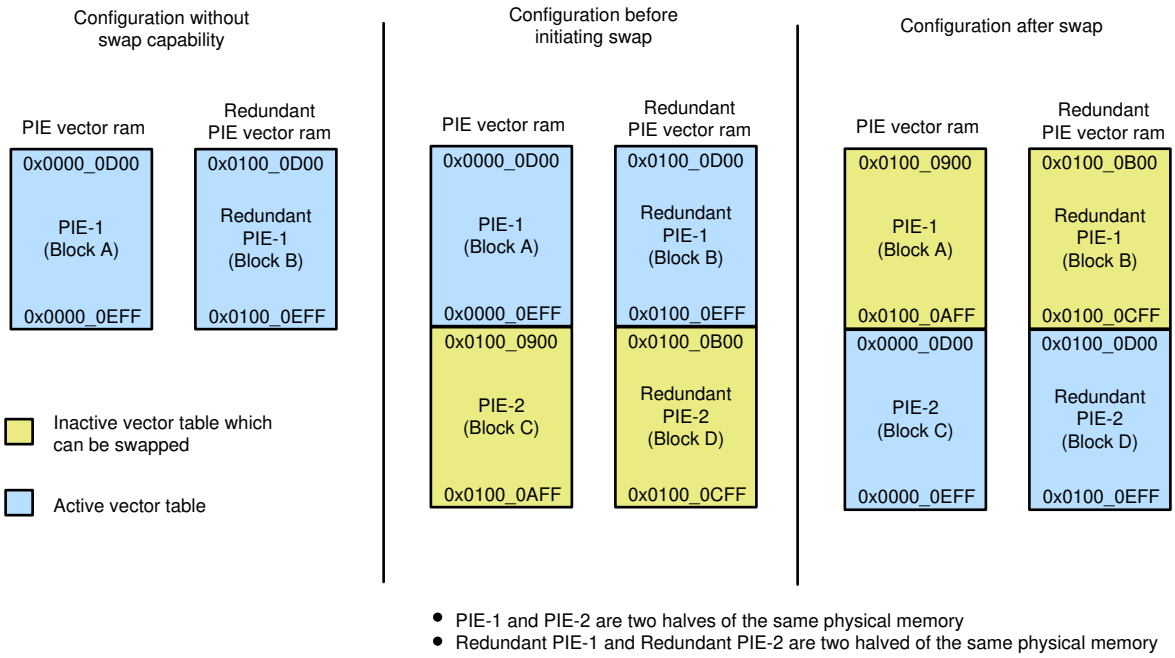


Figure 3-17. PIE Vector Table Swap

3.13.3.3 LS0/LS1 RAM Memory Swap

Similar to PIE Vector Table Swap, LS0 and LS1 physical RAM memory blocks can also be swapped. The memory architecture is similar to PIE vector table swap, and is shown in Figure 3-18. By default, physical Block 1 is assigned to addresses 0x8000-0x87FF (that is, the address range for LS0), and physical Block 2 is assigned to addresses 0x8800-0x8FFF (that is, the address range for LS1). By configuring LFUConfig.LS01Swap = 1, user application code can execute a swap, where physical *Block 2* is now assigned to addresses 0x8000-0x87FF (that is, the address range for LS0), and physical *Block 1* is now assigned to addresses 0x8800-0x8FFF (that is, the address range for LS1).

If physical memory Block 1 contains function pointers for the current firmware, the same relative locations in physical memory Block 2 can be populated with function pointers for the new firmware prior to LFU switchover. During LFU switchover, a simple swap operation is initiated by user application code, and this operation takes

just 1 CPU clock cycle. This allows user application code to always have function pointers in LS0, yet have two different physical blocks that can map to the LS0 address range.

For example, if current firmware contains 10 function pointers present at the start of Block 1 (LS0 address space). If the new firmware contains the same 10 function pointers that now need to be updated, user application code would place these at the start of Block 2 (LS1 address space) prior to LFU switchover. During LFU switchover, user application code would execute a LS0/LS1 RAM memory swap, where the physical RAM block previously mapped to the LS1 address space would *now* be mapped to the LS0 address space, and hence can be used seamlessly for function pointer addressing for the new firmware.

The register bit LFUStatus.LS01Swap provides the status of LS0/LS1 RAM memory swap.

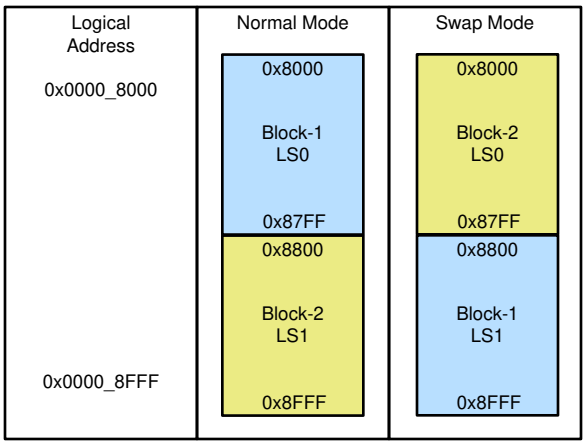


Figure 3-18. LS0/LS1 RAM Memory Swap

Additional points pertaining to LS0/LS1 RAM memory swap are described below:

1. LFU registers can be accessed from both CPU and CLA.
2. Only LS0 and LS1 blocks can be swapped. LS2 to LS7 blocks cannot be swapped.
3. LS0 and LS1 blocks have ECC protection. Address ECC is computed based on the physical address and hence it will not change based on the memory swap.
4. A number of LSx RAM registers are available to the user application code to configure options such as master select (LSxMSEL.MSEL_LS0, LSxMSEL.MSEL_LS1), fetch protect (LSxACCPROT0.FETCHPROT_LS0, LSxACCPROT0.FETCHPROT_LS1), write protect (LSxACCPROT0.CPUWRPROT_LS0, LSxACCPROT0.CPUWRPROT_LS1), CLA program memory LSxCLAPGM.CLAPGM_LS0, LSxCLAPGM.CLAPGM_LS1). These register bits indicate the status of the memory block that is deemed as LS0 (CPU address 0x8000 to 0x87FF) and LS1 (CPU address 0x8800 to 0x8FFF) at any point of time. When a LS0/LS1 RAM memory swap occurs, the corresponding control/status bits will also automatically swap.
5. It is recommended to service all pending errors (access violation, ECC, parity) associated with memory before initiating a LS0/LS1 RAM memory swap.
6. LS0/LS1 RAM memory swap shall be initiated only after completion of RAM initialization for both LS0 and LS1 memories (LSxINITDONE.INITDONE_LS0 = 1 and LSxINITDONE.INITDONE_LS1 = 1).
7. LS0/LS1 RAM memory swap shall not be initiated when RAM-test (LSxTEST.TEST_LS0 = 1 or LSxTEST.TEST_LS1 = 1) is in progress for LS0 or LS1 blocks.
8. With DCSM security on the device, in general, LS0 and LS1 RAM blocks can be assigned to different security zones. However, with LS0/LS1 RAM memory swaps, different physical RAM blocks can get mapped to the same address space. Application software shall therefore ensure that both LS0 and LS1 have the same security settings (for example, zone, EXE protection), if there is a plan to implement LS0/LS1 RAM memory swap. Hardware logic is implemented on the device to prevent swap of LS0 and LS1 if the blocks have different security configurations.
9. In order to prevent security vulnerabilities, LS0/LS1 RAM memory swap will not be allowed if it is initiated by code from a different zone. For example, (i) If LS0 and LS1 are part of Zone1, swap will not be allowed if code that initiates the swap resides in Zone2 or unsecure zone; (ii) If LS0 and LS1 are part of Zone2, swap

will not be allowed if code that initiates the swap resides in Zone1 or unsecure zone; (iii) If LS0 and LS1 are part of the same zone which is unsecure, swap will be allowed in all cases irrespective of where the code that initiates the swap resides; (iv) if LS0 and LS1 are part of the same zone and it is unlocked, the swap can be initiated from code residing anywhere (including from the debugger).

10. Once swap is initiated, it will happen in the next cycle itself, subject to it meeting the security requirements mentioned above. After initiation of a swap, application software shall check if the swap was correctly configured by checking the LFUStatus.LS01Swap status register. Consistency between LFUStatus.LS01Swap and LFUConfig.LS01Swap helps determine if the swap was correctly configured. If LFUStatus.LS01Swap does not match LFUConfig.LS01Swap, LFUConfig.LS01Swap needs to be cleared by user application code.
11. Since the logical address accessed by BGCRC will change with LS0/LS1 RAM memory swap, the computed CRC values for these memories need to be updated after the LS0/LS1 RAM memory swap.

3.13.3.3.1 Applicability to CLA LFU

The device does not support a swap table for the CLA task vectors (MVECTs). CLA LFU is implemented typically on the CPU side, where the MVECTs are updated sequentially at an appropriate time. The techniques for when to update the MVECTs will be described in the LFU system reference design guide, but it should be noted here that the approach is different from the CPU PIE vector table case, where a simple single cycle swap achieves the switch to the new PIE vector table.

In order for the LS0/LS1 RAM memory swap feature to be useful for CLA LFU switchover, two conditions need to be satisfied:

- CLA code has to fit into a single LSx block. The MVECT table contains CLA task vectors, whose addresses correspond to locations in the LSx block. For example, if the current firmware CLA code is present in LS0, MVECTs will point to various locations in LS0. If the new firmware CLA code is present in LS1, MVECTs will point to various locations in LS1.
- When switching over from current to new firmware, the MVECTs will need to be updated, unless they reside at the same relative location in both LS0 and LS1. If that is the case, then simply swapping LS0/LS1 RAM memory blocks will effectively update the MVECT table, without the need to sequentially update the MVECTs.

3.13.4 LFU Switchover

After the new firmware has been programmed to Flash, user application code will need to determine when it is appropriate to switchover to the new firmware. The techniques to determine this differ between real-time critical firmware running on the CPU and CLA, and these techniques are beyond the scope of this document. They will be described in detail in the LFU system reference design guide.

The device supports two register bits that can be set or reset to indicate that LFU switchover is in progress on the CPU (LFUConfig.LFU_CPU) and CLA (LFUConfig.LFU_CLA1). These bits do not impact any hardware logic on the device. For example, LFUConfig.LFU_CPU can be set by user application code at the start of switchover, and then tested in the initialization code in main(). This can enable only LFU switchover specific initialization to be performed (for example, PIE vector table swap, LS0/LS1 RAM memory swap), while bypassing all other initialization that typically happens after a device reset.

3.13.5 LFU Resources

The following are additional LFU resources available.

- *Live Firmware Update Without Device Reset on C2000™ MCUs User's Guide*
- *Live Firmware Update With Device Reset on C2000™ MCUs User's Guide*
- *Live Firmware Update Reference Design with C2000™ Real-Time MCUs*

3.14 Software

3.14.1 INTERRUPT Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/f28003x/examples/interrupt

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

3.14.1.1 External Interrupts (ExternalInterrupt)

FILE: interrupt_ex1_external.c

This program sets up GPIO0 as XINT1 and GPIO1 as XINT2. Two other GPIO signals are used to trigger the interrupt (GPIO10 triggers XINT1 and GPIO11 triggers XINT2). The user is required to externally connect these signals for the program to work properly.

XINT1 input is synced to SYSCLKOUT.

XINT2 has a long qualification - 6 samples at 510*SYSCLKOUT each.

GPIO16 will go high outside of the interrupts and low within the interrupts. This signal can be monitored on a scope.

Each interrupt is fired in sequence - XINT1 first and then XINT2

External Connections

- Connect GPIO10 to GPIO0. GPIO0 will be assigned to XINT1
- Connect GPIO11 to GPIO1. GPIO1 will be assigned to XINT2

Monitor GPIO16 with an oscilloscope. GPIO16 will be high outside of the ISRs and low within each ISR.

Watch Variables

- xint1Count for the number of times through XINT1 interrupt
- xint2Count for the number of times through XINT2 interrupt
- loopCount for the number of times through the idle loop

3.14.1.2 Multiple interrupt handling of I2C, SCI & SPI Digital Loopback

FILE: interrupt_ex2_with_i2c_sci_spi_loopback.c

This program is used to demonstrate how to handle multiple interrupts when using multiple communication peripherals like I2C, SCI & SPI Digital Loopback all in a single example. The data transfers would be done with FIFO Interrupts.

It uses the internal loopback test mode of these modules. Both the TX and RX FIFOs and their interrupts are used. Other than boot mode pin configuration, no other hardware configuration is required.

A stream of data is sent and then compared to the received stream. The sent data looks like this for I2C and SCI:
0000 0001
0001 0002
0002 0003
....
00FE 00FF
00FF 0000
etc..
The sent data looks like this for SPI:
0000 0001
0001 0002
0002 0003
....
FFFE FFFF

FFFF 0000
etc..
This pattern is repeated forever.

External Connections

- None

Watch Variables

- *sDatai2cA* - Data to send through I2C
- *rDatai2cA* - Received I2C data
- *rDataPoint* - Used to keep track of the last position in the receive I2C stream for error checking
- *sDataspiA* - Data to send through SPI
- *rDataspiA* - Received SPI data
- *rDataPointspiA* - Used to keep track of the last position in the receive SPI stream for error checking
- *sDatasciA* - SCI Data being sent
- *rDatasciA* - SCI Data received
- *rDataPointA* - Keep track of where we are in the SCI data stream. This is used to check the incoming data

3.14.1.3 CPU Timer Interrupt Software Prioritization

FILE: interrupt_ex3_sw_prioritization.c

This examples demonstrates the software prioritization of interrupts through CPU Timer Interrupts. Software prioritization of interrupts is achieved by enabling interrupt nesting.

In this device, hardware priorities for CPU Timer 0, 1 and 2 are set as timer 0 being highest priority and timer 2 being lowest priority. This example configures CPU Timer0, 1, and 2 priority in software with timer 2 priority being highest and timer 0 being lowest in software and prints a trace for the order of execution.

For most applications, the hardware prioritizing of the interrupts is sufficient. For applications that need custom prioritizing, this example illustrates how this can be done through software.User specific priorities can be configured in sw_prioritized_isr_level.h header file.

To enable interrupt nesting, following sequence needs to followed in ISRs. *Step 1:* Set the global priority: Modify the IER register to allow CPU interrupts with a higher user priority to be serviced. Note: at this time IER has already been saved on the stack. *Step 2:* Set the group priority: (optional) Modify the appropriate PIEIERx register to allow group interrupts with a higher user set priority to be serviced. Do NOT clear PIEIER register bits from another group other than that being serviced by this ISR. Doing so can cause erroneous interrupts to occur. *Step 3:* Enable interrupts: There are three steps to do this: a. Clear the PIEACK bits b. Wait at least one cycle c. Clear the INTM bit. *Step 4:* Run the main part of the ISR *Step 5:* Set INTM to disable interrupts. *Step 6:* Restore PIEIERx (optional depending on step 2) *Step 7:* Return from ISR

Refer to below link on more details on Interrupt nesting in C28x devices:
<C2000Ware>\docs\c28x_interrupt_nesting\html\index.html

External Connections

- None

Watch Variables

- traceISR - shows the order in which ISRs are executed.

3.14.1.4 EPWM Real-Time Interrupt

FILE: interrupt_ex4_epwm_realtime_interrupt.c

This example configures the ePWM1 Timer and increments a counter each time the ISR is executed. ePWM interrupt can be configured as time critical to demonstrate real-time mode functionality and real-time interrupt capability.

The example uses 2 LEDs - LED1 is toggled in the main loop and LED2 is toggled in the EPWM Timer Interrupt. FREE_SOFT bits and DBGIER.INT3 bit must be set to enable ePWM1 interrupt to be time critical and operational in real time mode after halt command

How to run the example?

- Add the watch variables as mentioned below and enable Continuous Refresh.
- Enable real-time mode (Run->Advanced->Enable Silicon Real-time Mode)
- Initially, the DBGIER register is set to 0 and the EPWM emulation mode is set to EPWM_EMULATION_STOP_AFTER_NEXT_TB (FREE_SOFT = 0)
- When the application is running, you will find both LEDs toggling and the watch variables EPwm1TimerIntCount, EPwm1Regs.TBCTR getting updated.
- When the application is halted, both LEDs stop toggling and the watch variables remain constant. EPWM counter is stopped on debugger halt.
- To enable EPWM counter run during debugger halt, set emulation mode as EPWM_EMULATION_FREE_RUN (FREE_SOFT = 2). You will find EPwm1Regs.TBCTR is running, but EPwm1TimerIntCount remains constant. This means, the EPWM counter is running, but the ISRs are not getting serviced.
- To enable real-time interrupts, set DBGIER.INT3 = 1 (EPWM1 interrupt is part of PIE Group 3). You will find that the EPwm1TimerIntCount is incrementing and the LED starts toggling. The EPWM ISR is getting serviced even during a debugger halt.

For more details, watch this video : https://training.ti.com/c2000-real-time-features

External Connections

- None

Watch Variables

- EPwm1TimerIntCount - EPWM1 ISR counter
- EPwm1Regs.TBCTR.TBCTR - EPWM1 Time Base counter
- EPwm1Regs.TBCTL.FREE_SOFT - Set this to 2 to enable free run
- DBGIER.INT3 - Set to 1 to enable real time interrupt

3.14.2 SYSCTL Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/sysctl

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

3.14.2.1 Missing Clock Detection (MCD)

FILE: sysctl_ex1_missing_clock_detection.c

This example demonstrates the missing clock detection functionality and the way to handle it. Once the MCD is simulated by disconnecting the OSCCLK to the MCD module an NMI would be generated. This NMI determines that an MCD was generated due to a clock failure which is handled in the ISR.

Before an MCD the clock frequency would be as per device initialization (120 MHz). Post MCD the frequency would move to 10 MHz or INTOSC1.

The example also shows how we can lock the PLL after missing clock, detection, by first explicitly switching the clock source to INTOSC1, resetting the missing clock detect circuit and then re-locking the PLL. Post a re-lock the clock frequency would be 120 MHz but using the INTOSC1 as clock source.

External Connections

- None.

Watch Variables

- *fail* - Indicates that a missing clock was either not detected or was not handled correctly.
- *mcd_clkfail_isr* - Indicates that the missing clock failure caused an NMI to be triggered and called an the ISR to handle it.
- *mcd_detect* - Indicates that a missing clock was detected.
- *result* - Status of a successful handling of missing clock detection

3.14.2.2 XCLKOUT (External Clock Output) Configuration

FILE: sysctl_ex2_xclkout_config.c

This example demonstrates how to configure the XCLKOUT pin for observing internal clocks through an external pin, for debugging and testing purposes.

In this example, we are using INTOSC1 as the XCLKOUT clock source and configuring the divider as 8. Expected frequency of XCLKOUT = (INTOSC1 freq)/8 = 10/8 = 1.25 MHz

View the XCLKOUT on GPIO16 using an oscilloscope.

3.14.3 TIMER Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/timer

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

3.14.3.1 CPU Timers

FILE: timer_ex1_cputimers.c

This example configures CPU Timer0, 1, and 2 and increments a counter each time the timer asserts an interrupt.

External Connections
- None

Watch Variables
- cpuTimer0IntCount
- cpuTimer1IntCount
- cpuTimer2IntCount

3.14.4 LPM Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/lpm

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

3.14.4.1 Low Power Modes: Device Idle Mode and Wakeup using GPIO

FILE: lpm_ex1_idlewake_gpio.c

This example puts the device into IDLE mode and then wakes up the device from IDLE using XINT1 which triggers on a falling edge of GPIO0.

The GPIO0 pin must be pulled from high to low by an external agent for wakeup. GPIO0 is configured as an XINT1 pin to trigger an XINT1 interrupt upon detection of a falling edge.

Initially, pull GPIO0 high externally. To wake device from IDLE mode by triggering an XINT1 interrupt, pull GPIO0 low (falling edge). The wakeup process begins as soon as GPIO0 is held low for the time indicated in the *TMS320F28003x Real-Time Microcontrollers Data Sheet*.

GPIO1 is pulled high before entering the IDLE mode and is pulled low when in the external interrupt ISR.

External Connections
- GPIO0 needs to be pulled low to wake up the device.
- On device wakeup, the GPIO1 will be low and LED1 will start blinking.

3.14.4.2 Low Power Modes: Device Idle Mode and Wakeup using Watchdog

FILE: lpm_ex2_idlewake_watchdog.c

This example puts the device into IDLE mode and then wakes up the device from IDLE using watchdog timer.

The device wakes up from the IDLE mode when the watchdog timer overflows, triggering an interrupt. A pre scalar is set for the watchdog timer to change the counter overflow time.

GPIO1 is pulled high before entering the IDLE mode and is pulled low when in the wakeup ISR.

External Connections

- On device wakeup, the GPIO1 will be low and LED1 will start blinking

3.14.4.3 Low Power Modes: Device Standby Mode and Wakeup using GPIO

FILE: lpm_ex3_standbywake_gpio.c

This example puts the device into STANDBY mode. If the lowest possible current consumption in STANDBY mode is desired, the JTAG connector must be removed from the device board while the device is in STANDBY mode.

This example puts the device into STANDBY mode and then wakes up the device from STANDBY using an LPM wakeup pin.

The pin GPIO0 is configured as the LPM wakeup pin to trigger a WAKEINT interrupt upon detection of a low pulse. Initially, pull GPIO0 high externally. To wake device from STANDBY mode, pull GPIO0 low for at least (2+QUALSTDBY), OSCLKS, then pull it high again.

The example then wakes up the device from STANDBY using GPIO0. GPIO0 wakes the device from STANDBY mode when a low pulse (signal goes high->low->high)is detected on the pin. This pin must be pulsed by an external agent for wakeup.

GPIO1 is pulled high before entering the STANDBY mode and is pulled low when in the wakeup ISR.

External Connections

- GPIO0 needs to be pulled low to wake up the device.
- On device wakeup, the GPIO1 will be low and LED1 will start blinking

3.14.4.4 Low Power Modes: Device Standby Mode and Wakeup using Watchdog

FILE: lpm_ex4_standbywake_watchdog.c

This example puts the device into STANDBY mode. If the lowest possible current consumption in STANDBY mode is desired, the JTAG connector must be removed from the device board while the device is in STANDBY mode.

This example puts the device into STANDBY mode then wakes up the device from STANDBY using watchdog timer.

The device wakes up from the STANDBY mode when the watchdog timer overflows triggering an interrupt. In the ISR, the GPIO1 is pulled low. the GPIO1 is toggled to indicate the device is out of STANDBY mode. A pre scalar is set for the watchdog timer to change the counter overflow time.

GPIO1 is pulled high before entering the STANDBY mode and is pulled low when in the wakeup ISR.

External Connections

- On device wakeup, the GPIO1 will be low and LED1 will start blinking

3.14.4.5 Low Power Modes: Halt Mode and Wakeup using GPIO

FILE: lpm_ex5_haltwake_gpio.c

This example puts the device into HALT mode. If the lowest possible current consumption in HALT mode is desired, the JTAG connector must be removed from the device board while the device is in HALT mode.

For applications that require minimal power consumption during HALT mode, application software should power off the XTAL prior to entering HALT by setting the XTALCR.OSCOFF bit or by using the driverlib function SysCtl_turnOffOsc(SYSCTL_OSCSRC_XTAL);. If the OSCCLK source is configured to be XTAL, the application should first switch the OSSCLK source to INTOSC1 or INTOSC2 prior to setting XTALCR.OSCOFF.

This example puts the device into HALT mode and then wakes up the device from HALT using an LPM wakeup pin.

The pin GPIO0 is configured as the LPM wakeup pin to trigger a WAKEINT interrupt upon detection of a low pulse.The GPIO0 pin must be pulled from high to low by an external agent for wakeup.

GPIO1 is pulled high before entering the STANDBY mode and is pulled low when in the wakeup ISR.

External Connections

- On device wakeup, the GPIO1 will be low and LED1 will start blinking

3.14.4.6 Low Power Modes: Halt Mode and Wakeup

FILE: lpm_ex6_haltwake_gpio_watchdog.c

This example puts the device into HALT mode. If the lowest possible current consumption in HALT mode is desired, the JTAG connector must be removed from the device board while the device is in HALT mode.

For applications that require minimal power consumption during HALT mode, application software should power off the XTAL prior to entering HALT by setting the XTALCR.OSCOFF bit or by using the driverlib function SysCtl_turnOffOsc(SYSCTL_OSCSRC_XTAL);. If the OSCCLK source is configured to be XTAL, the application should first switch the OSSCLK source to INTOSC1 or INTOSC2 prior to setting XTALCR.OSCOFF.

This example puts the device into HALT mode and then wakes up the device from HALT using an LPM wakeup pin.

The pin GPIO0 is configured as the LPM wakeup pin to trigger a WAKEINT interrupt upon detection of a low pulse.The GPIO0 pin must be pulled from high to low by an external agent for wakeup.

In this example, the watchdog timer is clocked, and is configured to produce watchdog reset as a timeout mechanism.

GPIO1 is pulled high before entering the STANDBY mode and is pulled low when in the wakeup ISR.

External Connections

- On device wakeup, the GPIO1 will be low and LED1 will start blinking

3.14.5 MEMCFG Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/memcfg

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

3.14.5.1 Correctable and Uncorrectable Memory Error Handling

FILE: memcfg_ex1_error_handling.c

This example demonstrates error handling in case of various erroneous memory read/write operations. Error handling in case of CPU read/write violations, correctable and uncorrectable memory errors has been demonstrated.

Test functions used in this example:

- generateMasterCPUWrViolation -
  - This test configures Memconfig to block CPU writes to GS0 RAM. A write attempt to this memory location by CPU causes RAM_ACC_VIOL Interrupt
- generateECCMemCorrError
  - This test induces single bit ECC error in LS6 RAM. A read from the corrupted memory location causes INT_RAM_CORR_ERR Interrupt
- generateECCMemUncorrError
  - This test induces double bit ECC error in LS7 RAM. A read from the corrupted memory location causes NMI

- forceNonMasterDMAReadViolation
  - This forces a DMA access violation using MemCfg_forceViolationInterrupt API. This casuses RAM_ACC_VIOL Interrupt

External Connections

- None

Watch Variables

- *testStatusGlobal* - Equivalent to *TEST_PASS* if test finished correctly, else the value is set to *TEST_FAIL*
- *errCountGlobal* - Error counter

3.14.6 WATCHDOG Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/watchdog

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

3.14.6.1 Watchdog

FILE: watchdog_ex1_service.c

This example shows how to service the watchdog or generate a wakeup interrupt using the watchdog. By default the example will generate a Wake interrupt. To service the watchdog and not generate the interrupt, uncomment the SysCtl_serviceWatchdog() line in the main for loop.

External Connections

- None.

Watch Variables

- wakeCount - The number of times entered into the watchdog ISR
- loopCount - The number of loops performed while not in ISR

3.15 System Control Registers

3.15.1 SYSCTRL Base Address Table

Table 3-15. SYSCTRL Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
CpuTimer0Regs	CPUTIMER_REGS	CPUTIMER0_BASE	0x0000_0C00	YES	-	-	-	-
CpuTimer1Regs	CPUTIMER_REGS	CPUTIMER1_BASE	0x0000_0C08	YES	-	-	-	-
CpuTimer2Regs	CPUTIMER_REGS	CPUTIMER2_BASE	0x0000_0C10	YES	-	-	-	-
PieCtrlRegs	PIE_CTRL_REGS	PIECTRL_BASE	0x0000_0CE0	YES	-	-	-	-
PieVectTable	PIE_VECT_TABLE	PIEVECTTABLE_BASE	0x0000_0D00	YES	-	-	-	-
WdRegs	WD_REGS	WD_BASE	0x0000_7000	YES	-	-	-	YES
NmiIntruptRegs	NMI_INTRUPT_REGS	NMI_BASE	0x0000_7060	YES	-	-	-	YES
XintRegs	XINT_REGS	XINT_BASE	0x0000_7070	YES	-	-	-	YES
SyncSocRegs	SYNC_SOC_REGS	SYNCSOC_BASE	0x0000_7940	YES	-	-	-	YES
DmaClaSrcSelRegs	DMA_CLA_SRC_SEL_REGS	DMACLASRCSEL_BASE	0x0000_7980	YES	-	-	-	YES
LfuRegs	LFU_REGS	LFU_BASE	0x0000_7FE0	YES	-	-	YES	YES
DevCfgRegs	DEV_CFG_REGS	DEVCFG_BASE	0x0005_D000	YES	-	-	-	YES
ClkCfgRegs	CLK_CFG_REGS	CLKCFG_BASE	0x0005_D200	YES	-	-	-	YES
CpuSysRegs	CPU_SYS_REGS	CPUSYS_BASE	0x0005_D300	YES	-	-	-	YES
SysStatusRegs	SYS_STATUS_REGS	SYSSTAT_BASE	0x0005_D400	YES	-	-	-	YES
PeriphAcRegs	PERIPH_AC_REGS	PERIPHAC_BASE	0x0005_D500	YES	-	-	-	YES
MemCfgRegs	MEM_CFG_REGS	MEMCFG_BASE	0x0005_F400	YES	-	-	-	YES
AccessProtectionRegs	ACCESS_PROTECTION_REGS	ACCESSPROTECTION_BASE	0x0005_F500	YES	-	-	-	YES
MemoryErrorRegs	MEMORY_ERROR_REGS	MEMORYERROR_BASE	0x0005_F540	YES	-	-	-	YES
TestErrorRegs	TEST_ERROR_REGS	TESTERROR_BASE	0x0005_F590	YES	-	-	-	YES
UidRegs	UID_REGS	UID_BASE	0x0007_0200	YES	-	-	-	-

3.15.2 ACCESS_PROTECTION_REGS Registers

Table 3-16 lists the memory-mapped registers for the ACCESS_PROTECTION_REGS registers. All register offset addresses not listed in Table 3-16 should be considered as reserved locations and the register contents should not be modified.

Table 3-16. ACCESS_PROTECTION_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	NMAVFLG	Non-Master Access Violation Flag Register		Go
2h	NMAVSET	Non-Master Access Violation Flag Set Register	EALLOW	Go
4h	NMAVCLR	Non-Master Access Violation Flag Clear Register	EALLOW	Go
6h	NMAVINTEN	Non-Master Access Violation Interrupt Enable Register	EALLOW	Go
8h	NMCPURDAVADDR	Non-Master CPU Read Access Violation Address		Go
Ah	NMCPUWRAVADDR	Non-Master CPU Write Access Violation Address		Go
Ch	NMCPUFAVADDR	Non-Master CPU Fetch Access Violation Address		Go
Eh	NMDMAWRAVADDR	Non-Master DMA Write Access Violation Address		Go
10h	NMCLA1RDAVADDR	Non-Master CLA1 Read Access Violation Address		Go
12h	NMCLA1WRAVADDR	Non-Master CLA1 Write Access Violation Address		Go
14h	NMCLA1FAVADDR	Non-Master CLA1 Fetch Access Violation Address		Go
1Ch	NMDMARDAVADDR	Non-Master DMA Read Access Violation Address		Go
20h	MAVFLG	Master Access Violation Flag Register		Go
22h	MAVSET	Master Access Violation Flag Set Register	EALLOW	Go
24h	MAVCLR	Master Access Violation Flag Clear Register	EALLOW	Go
26h	MAVINTEN	Master Access Violation Interrupt Enable Register	EALLOW	Go
28h	MCPUFAVADDR	Master CPU Fetch Access Violation Address		Go
2Ah	MCPUWRAVADDR	Master CPU Write Access Violation Address		Go
2Ch	MDMAWRAVADDR	Master DMA Write Access Violation Address		Go
2Eh + formula	MHICWRAVADDR_y	Master HIC Write Access Violation Address		Go
3Ch	NMHICRDAVADDR	Non-Master HIC Read Access Violation Address		Go
3Eh	NMHICWRAVADDR	Non-Master HIC Write Access Violation Address		Go

Complex bit access types are encoded to fit into small table cells. Table 3-17 shows the codes that are used for access types in this section.

Table 3-17. ACCESS_PROTECTION_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value

Table 3-17. ACCESS_PROTECTION_REGS Access Type Codes (continued)

Access Type	Code	Description
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.15.2.1 NMAVFLG Register (Offset = 0h) [Reset = 0h]

NMAVFLG is shown in Figure 3-19 and described in Table 3-18.

Return to the Summary Table.

Non-Master Access Violation Flag Register

Figure 3-19. NMAVFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			HICWRITE	HICREAD	DMAREAD	RESERVED	RESERVED
R-0h			R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	CLA1FETCH	CLA1WRITE	CLA1READ	DMAWRITE	CPUFETCH	CPUWRITE	CPUREAD
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 3-18. NMAVFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-13	RESERVED	R	0h	Reserved
12	HICWRITE	R	0h	Non Master HIC Write Access Violation Flag: 0: No violation. 1: Access violation occured. Reset type: SYSRSn
11	HICREAD	R	0h	Non Master HIC Read Access Violation Flag: 0: No violation. 1: Access violation occured. Reset type: SYSRSn
10	DMAREAD	R	0h	Non Master DMA Read Access Violation Flag: 0: No violation. 1: Access violation occured. Reset type: SYSRSn
9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	RESERVED	R	0h	Reserved
6	CLA1FETCH	R	0h	Non Master CLA1 Fetch Access Violation Flag: 0: No violation. 1: Access violation occured. Reset type: SYSRSn
5	CLA1WRITE	R	0h	Non Master CLA1 Write Access Violation Flag: 0: No violation. 1: Access violation occured. Reset type: SYSRSn
4	CLA1READ	R	0h	Non Master CLA1 Read Access Violation Flag: 0: No violation. 1: Access violation occured. Reset type: SYSRSn

Table 3-18. NMAVFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	DMAWRITE	R	0h	Non Master DMA Write Access Violation Flag: 0: No violation. 1: Access violation occured. Reset type: SYSRSn
2	CPUFETCH	R	0h	Non Master CPU Fetch Access Violation Flag: 0: No violation. 1: Access violation occured. Reset type: SYSRSn
1	CPUWRITE	R	0h	Non Master CPU Write Access Violation Flag: 0: No violation. 1: Access violation occured. Reset type: SYSRSn
0	CPUREAD	R	0h	Non Master CPU Read Access Violation Flag: 0: No violation. 1: Access violation occured. Reset type: SYSRSn

3.15.2.2 NMAVSET Register (Offset = 2h) [Reset = 0h]

NMAVSET is shown in Figure 3-20 and described in Table 3-19.

Return to the Summary Table.

Non-Master Access Violation Flag Set Register

Figure 3-20. NMAVSET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			HICWRITE	HICREAD	DMAREAD	RESERVED	RESERVED
R-0h			R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
RESERVED	CLA1FETCH	CLA1WRITE	CLA1READ	DMAWRITE	CPUFETCH	CPUWRITE	CPUREAD
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-19. NMAVSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-13	RESERVED	R	0h	Reserved
12	HICWRITE	R-0/W1S	0h	0: No action. 1: HIC Write Access Violation Flag in NMAVFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn
11	HICREAD	R-0/W1S	0h	0: No action. 1: HIC Read Access Violation Flag in NMAVFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn
10	DMAREAD	R-0/W1S	0h	0: No action. 1: DMA Read Access Violation Flag in NMAVFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn
9	RESERVED	R-0/W1S	0h	Reserved
8	RESERVED	R-0/W1S	0h	Reserved
7	RESERVED	R-0/W1S	0h	Reserved
6	CLA1FETCH	R-0/W1S	0h	0: No action. 1: CLA1 Fetch Access Violation Flag in NMAVFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn
5	CLA1WRITE	R-0/W1S	0h	0: No action. 1: CLA1 Write Access Violation Flag in NMAVFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn
4	CLA1READ	R-0/W1S	0h	0: No action. 1: CLA1 Read Access Violation Flag in NMAVFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn

Table 3-19. NMAVSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	DMAWRITE	R-0/W1S	0h	0: No action. 1: DMA Write Access Violation Flag in NMAVFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn
2	CPUFETCH	R-0/W1S	0h	0: No action. 1: CPU Fetch Access Violation Flag in NMAVFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn
1	CPUWRITE	R-0/W1S	0h	0: No action. 1: CPU Write Access Violation Flag in NMAVFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn
0	CPUREAD	R-0/W1S	0h	0: No action. 1: CPU Read Access Violation Flag in NMAVFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn

3.15.2.3 NMAVCLR Register (Offset = 4h) [Reset = 0h]

NMAVCLR is shown in Figure 3-21 and described in Table 3-20.

Return to the Summary Table.

Non-Master Access Violation Flag Clear Register

Figure 3-21. NMAVCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			HICWRITE	HICREAD	DMAREAD	RESERVED	RESERVED
R-0h			R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
RESERVED	CLA1FETCH	CLA1WRITE	CLA1READ	DMAWRITE	CPUFETCH	CPUWRITE	CPUREAD
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-20. NMAVCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-13	RESERVED	R	0h	Reserved
12	HICWRITE	R-0/W1S	0h	0: No action. 1: HIC Write Access Violation Flag in NMAVFLG register will be cleared. Reset type: SYSRSn
11	HICREAD	R-0/W1S	0h	0: No action. 1: HIC Read Access Violation Flag in NMAVFLG register will be cleared. Reset type: SYSRSn
10	DMAREAD	R-0/W1S	0h	0: No action. 1: DMA Read Access Violation Flag in NMAVFLG register will be cleared. Reset type: SYSRSn
9	RESERVED	R-0/W1S	0h	Reserved
8	RESERVED	R-0/W1S	0h	Reserved
7	RESERVED	R-0/W1S	0h	Reserved
6	CLA1FETCH	R-0/W1S	0h	0: No action. 1: CLA1 Fetch Access Violation Flag in NMAVFLG register will be cleared. Reset type: SYSRSn
5	CLA1WRITE	R-0/W1S	0h	0: No action. 1: CLA1 Write Access Violation Flag in NMAVFLG register will be cleared. Reset type: SYSRSn
4	CLA1READ	R-0/W1S	0h	0: No action. 1: CLA1 Read Access Violation Flag in NMAVFLG register will be cleared. Reset type: SYSRSn

Table 3-20. NMAVCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	DMAWRITE	R-0/W1S	0h	0: No action. 1: DMA Write Access Violation Flag in NMAVFLG register will be cleared. Reset type: SYSRSn
2	CPUFETCH	R-0/W1S	0h	0: No action. 1: CPU Fetch Access Violation Flag in NMAVFLG register will be cleared. Reset type: SYSRSn
1	CPUWRITE	R-0/W1S	0h	0: No action. 1: CPU Write Access Violation Flag in NMAVFLG register will be cleared. Reset type: SYSRSn
0	CPUREAD	R-0/W1S	0h	0: No action. 1: CPU Read Access Violation Flag in NMAVFLG register will be cleared. Reset type: SYSRSn

3.15.2.4 NMAVINTEN Register (Offset = 6h) [Reset = 0h]

NMAVINTEN is shown in Figure 3-22 and described in Table 3-21.

Return to the Summary Table.

Non-Master Access Violation Interrupt Enable Register

Figure 3-22. NMAVINTEN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			HICWRITE	HICREAD	DMAREAD	RESERVED	RESERVED
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	CLA1FETCH	CLA1WRITE	CLA1READ	DMAWRITE	CPUFETCH	CPUWRITE	CPUREAD
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-21. NMAVINTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-13	RESERVED	R	0h	Reserved
12	HICWRITE	R/W	0h	0: HIC Non Master Write Access Violation Interrupt is disabled. 1: HIC Non Master Write Access Violation Interrupt is enabled. Reset type: SYSRSn
11	HICREAD	R/W	0h	0: HIC Non Master Read Access Violation Interrupt is disabled. 1: HIC Non Master Read Access Violation Interrupt is enabled. Reset type: SYSRSn
10	DMAREAD	R/W	0h	0: DMA Non Master Read Access Violation Interrupt is disabled. 1: DMA Non Master Read Access Violation Interrupt is enabled. Reset type: SYSRSn
9	RESERVED	R/W	0h	Reserved
8	RESERVED	R/W	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	CLA1FETCH	R/W	0h	0: CLA1 Non Master Fetch Access Violation Interrupt is disabled. 1: CLA1 Non Master Fetch Access Violation Interrupt is enabled. Reset type: SYSRSn
5	CLA1WRITE	R/W	0h	0: CLA1 Non Master Write Access Violation Interrupt is disabled. 1: CLA1 Non Master Write Access Violation Interrupt is enabled. Reset type: SYSRSn
4	CLA1READ	R/W	0h	0: CLA1 Non Master Read Access Violation Interrupt is disabled. 1: CLA1 Non Master Read Access Violation Interrupt is enabled. Reset type: SYSRSn
3	DMAWRITE	R/W	0h	0: DMA Non Master Write Access Violation Interrupt is disabled. 1: DMA Non Master Write Access Violation Interrupt is enabled. Reset type: SYSRSn
2	CPUFETCH	R/W	0h	0: CPU Non Master Fetch Access Violation Interrupt is disabled. 1: CPU Non Master Fetch Access Violation Interrupt is enabled. Reset type: SYSRSn

Table 3-21. NMAVINTEN Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	CPUWRITE	R/W	0h	0: CPU Non Master Write Access Violation Interrupt is disabled. 1: CPU Non Master Write Access Violation Interrupt is enabled. Reset type: SYSRSn
0	CPUREAD	R/W	0h	0: CPU Non Master Read Access Violation Interrupt is disabled. 1: CPU Non Master Read Access Violation Interrupt is enabled. Reset type: SYSRSn

3.15.2.5 NMCPURDAVADDR Register (Offset = 8h) [Reset = 0h]

NMCPURDAVADDR is shown in Figure 3-23 and described in Table 3-22.

Return to the Summary Table.

Non-Master CPU Read Access Violation Address

Figure 3-23. NMCPURDAVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMCPURDAVADDR																															
R-0h																															

Table 3-22. NMCPURDAVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	NMCPURDAVADDR	R	0h	This register captures the address location for which non master CPU read access violation occurred. Reset type: SYSRSn

3.15.2.6 NMCPUWRAVADDR Register (Offset = Ah) [Reset = 0h]

NMCPUWRAVADDR is shown in Figure 3-24 and described in Table 3-23.

Return to the Summary Table.

Non-Master CPU Write Access Violation Address

Figure 3-24. NMCPUWRAVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMCPUWRAVADDR																															
R-0h																															

Table 3-23. NMCPUWRAVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	NMCPUWRAVADDR	R	0h	This register captures the address location for which non master CPU write access violation occurred. Reset type: SYSRSn

3.15.2.7 NMCPUFAVADDR Register (Offset = Ch) [Reset = 0h]

NMCPUFAVADDR is shown in Figure 3-25 and described in Table 3-24.

Return to the Summary Table.

Non-Master CPU Fetch Access Violation Address

Figure 3-25. NMCPUFAVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMCPUFAVADDR																															
R-0h																															

Table 3-24. NMCPUFAVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	NMCPUFAVADDR	R	0h	This register captures the address location for which non master CPU fetch access violation occurred. Reset type: SYSRSn

3.15.2.8 NMDMAWRAVADDR Register (Offset = Eh) [Reset = 0h]

NMDMAWRAVADDR is shown in Figure 3-26 and described in Table 3-25.

Return to the Summary Table.

Non-Master DMA Write Access Violation Address

Figure 3-26. NMDMAWRAVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMDMAWRAVADDR																															
R-0h																															

Table 3-25. NMDMAWRAVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	NMDMAWRAVADDR	R	0h	This register captures the address location for which non master DMA write access violation occurred. Reset type: SYSRSn

3.15.2.9 NMCLA1RDAVADDR Register (Offset = 10h) [Reset = 0h]

NMCLA1RDAVADDR is shown in Figure 3-27 and described in Table 3-26.

Return to the Summary Table.

Non-Master CLA1 Read Access Violation Address

Figure 3-27. NMCLA1RDAVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMCLA1RDAVADDR																															
R-0h																															

Table 3-26. NMCLA1RDAVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	NMCLA1RDAVADDR	R	0h	This register captures the address location for which non master CLA1 read access violation occurred. Reset type: SYSRSn

3.15.2.10 NMCLA1WRAVADDR Register (Offset = 12h) [Reset = 0h]

NMCLA1WRAVADDR is shown in Figure 3-28 and described in Table 3-27.

Return to the Summary Table.

Non-Master CLA1 Write Access Violation Address

Figure 3-28. NMCLA1WRAVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMCLA1WRAVADDR																															
R-0h																															

Table 3-27. NMCLA1WRAVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	NMCLA1WRAVADDR	R	0h	This register captures the address location for which non master CLA1 write access violation occurred. Reset type: SYSRSn

3.15.2.11 NMCLA1FAVADDR Register (Offset = 14h) [Reset = 0h]

NMCLA1FAVADDR is shown in Figure 3-29 and described in Table 3-28.

Return to the Summary Table.

Non-Master CLA1 Fetch Access Violation Address

Figure 3-29. NMCLA1FAVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMCLA1FAVADDR																															
R-0h																															

Table 3-28. NMCLA1FAVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	NMCLA1FAVADDR	R	0h	This register captures the address location for which non master CLA1 fetch access violation occurred. Reset type: SYSRSn

3.15.2.12 NMDMARDAVADDR Register (Offset = 1Ch) [Reset = 0h]

NMDMARDAVADDR is shown in Figure 3-30 and described in Table 3-29.

Return to the Summary Table.

Non-Master DMA Read Access Violation Address

Figure 3-30. NMDMARDAVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMDMARDAVADDR																															
R-0h																															

Table 3-29. NMDMARDAVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	NMDMARDAVADDR	R	0h	This register captures the address location for which non master DMA read access violation occurred. Reset type: SYSRSn

3.15.2.13 MAVFLG Register (Offset = 20h) [Reset = 0h]

MAVFLG is shown in Figure 3-31 and described in Table 3-30.

Return to the Summary Table.

Master Access Violation Flag Register

Figure 3-31. MAVFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				HICAWRITE	DMAWRITE	CPUWRITE	CPUFETCH
R-0h				R-0h	R-0h	R-0h	R-0h

Table 3-30. MAVFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	HICAWRITE	R	0h	Master HICA Write Access Violation Flag: 0: No violation. 1: Access violation occured. Reset type: SYSRSn
2	DMAWRITE	R	0h	Master DMA Write Access Violation Flag: 0: No violation. 1: Access violation occured. Reset type: SYSRSn
1	CPUWRITE	R	0h	Master CPU Write Access Violation Flag: 0: No violation. 1: Access violation occured. Reset type: SYSRSn
0	CPUFETCH	R	0h	Master CPU Fetch Access Violation Flag: 0: No violation. 1: Access violation occured. Reset type: SYSRSn

3.15.2.14 MAVSET Register (Offset = 22h) [Reset = 0h]

MAVSET is shown in Figure 3-32 and described in Table 3-31.

Return to the Summary Table.

Master Access Violation Flag Set Register

Figure 3-32. MAVSET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				HICAWRITE	DMAWRITE	CPUWRITE	CPUFETCH
R-0h				R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-31. MAVSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	HICAWRITE	R-0/W1S	0h	0: No action. 1: HICA Write Access Violation Flag in MAVFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn
2	DMAWRITE	R-0/W1S	0h	0: No action. 1: DMA Write Access Violation Flag in MAVFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn
1	CPUWRITE	R-0/W1S	0h	0: No action. 1: CPU Write Access Violation Flag in MAVFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn
0	CPUFETCH	R-0/W1S	0h	0: No action. 1: CPU Fetch Access Violation Flag in MAVFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn

3.15.2.15 MAVCLR Register (Offset = 24h) [Reset = 0h]

MAVCLR is shown in Figure 3-33 and described in Table 3-32.

Return to the Summary Table.

Master Access Violation Flag Clear Register

Figure 3-33. MAVCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				HICAWRITE	DMAWRITE	CPUWRITE	CPUFETCH
R-0h				R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-32. MAVCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	HICAWRITE	R-0/W1S	0h	0: No action. 1: HICA Write Access Violation Flag in MAVFLG register will be cleared. Reset type: SYSRSn
2	DMAWRITE	R-0/W1S	0h	0: No action. 1: DMA Write Access Violation Flag in MAVFLG register will be cleared. Reset type: SYSRSn
1	CPUWRITE	R-0/W1S	0h	0: No action. 1: CPU Write Access Violation Flag in MAVFLG register will be cleared . Reset type: SYSRSn
0	CPUFETCH	R-0/W1S	0h	0: No action. 1: CPU Fetch Access Violation Flag in MAVFLG register will be cleared. Reset type: SYSRSn

3.15.2.16 MAVINTEN Register (Offset = 26h) [Reset = 0h]

MAVINTEN is shown in Figure 3-34 and described in Table 3-33.

Return to the Summary Table.

Master Access Violation Interrupt Enable Register

Figure 3-34. MAVINTEN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				HICAWRITE	DMAWRITE	CPUWRITE	CPUFETCH
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-33. MAVINTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	HICAWRITE	R/W	0h	0: HICA Write Access Violation Interrupt is disabled. 1: HICA Write Access Violation Interrupt is enabled. Reset type: SYSRSn
2	DMAWRITE	R/W	0h	0: DMA Write Access Violation Interrupt is disabled. 1: DMA Write Access Violation Interrupt is enabled. Reset type: SYSRSn
1	CPUWRITE	R/W	0h	0: CPU Write Access Violation Interrupt is disabled. 1: CPU Write Access Violation Interrupt is enabled. Reset type: SYSRSn
0	CPUFETCH	R/W	0h	0: CPU Fetch Access Violation Interrupt is disabled. 1: CPU Fetch Access Violation Interrupt is enabled. Reset type: SYSRSn

3.15.2.17 MCPUFAVADDR Register (Offset = 28h) [Reset = 0h]

MCPUFAVADDR is shown in Figure 3-35 and described in Table 3-34.

Return to the Summary Table.

Master CPU Fetch Access Violation Address

Figure 3-35. MCPUFAVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCPUFAVADDR																															
R-0h																															

Table 3-34. MCPUFAVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MCPUFAVADDR	R	0h	This register captures the address location for which master CPU fetch access violation occurred. Reset type: SYSRSn

3.15.2.18 MCPUWRAVADDR Register (Offset = 2Ah) [Reset = 0h]

MCPUWRAVADDR is shown in Figure 3-36 and described in Table 3-35.

Return to the Summary Table.

Master CPU Write Access Violation Address

Figure 3-36. MCPUWRAVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MCPUWRAVADDR																															
R-0h																															

Table 3-35. MCPUWRAVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MCPUWRAVADDR	R	0h	This register captures the address location for which master CPU write access violation occurred. Reset type: SYSRSn

3.15.2.19 MDMAWRAVADDR Register (Offset = 2Ch) [Reset = 0h]

MDMAWRAVADDR is shown in Figure 3-37 and described in Table 3-36.

Return to the Summary Table.

Master DMA Write Access Violation Address

Figure 3-37. MDMAWRAVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MDMAWRAVADDR																															
R-0h																															

Table 3-36. MDMAWRAVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MDMAWRAVADDR	R	0h	This register captures the address location for which master DMA write access violation occurred. Reset type: SYSRSn

3.15.2.20 MHICWRAVADDR_y Register (Offset = 2Eh + formula) [Reset = 0h]

MHICWRAVADDR_y is shown in Figure 3-38 and described in Table 3-37.

Return to the Summary Table.

Master HIC Write Access Violation Address

Offset = 2Eh + (y * 2h); where y = 0h to 1h

Figure 3-38. MHICWRAVADDR_y Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MHICWRAVADDR																															
R-0h																															

Table 3-37. MHICWRAVADDR_y Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MHICWRAVADDR	R	0h	This register captures the address location for which master HICA write access violation occurred. Reset type: SYSRSn

3.15.2.21 NMHICRDAVADDR Register (Offset = 3Ch) [Reset = 0h]

NMHICRDAVADDR is shown in Figure 3-39 and described in Table 3-38.

Return to the Summary Table.

Non-Master HIC Read Access Violation Address

Figure 3-39. NMHICRDAVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMHICRDAVADDR																															
R-0h																															

Table 3-38. NMHICRDAVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	NMHICRDAVADDR	R	0h	This register captures the address location for which non master HIC read access violation occurred. Reset type: SYSRSn

3.15.2.22 NMHICWRAVADDR Register (Offset = 3Eh) [Reset = 0h]

NMHICWRAVADDR is shown in Figure 3-40 and described in Table 3-39.

Return to the Summary Table.

Non-Master HIC Write Access Violation Address

Figure 3-40. NMHICWRAVADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMHICWRAVADDR																															
R-0h																															

Table 3-39. NMHICWRAVADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	NMHICWRAVADDR	R	0h	This register captures the address location for which non master HIC write access violation occurred. Reset type: SYSRSn

3.15.3 CLK_CFG_REGS Registers

Table 3-40 lists the memory-mapped registers for the CLK_CFG_REGS registers. All register offset addresses not listed in Table 3-40 should be considered as reserved locations and the register contents should not be modified.

Table 3-40. CLK_CFG_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
2h	CLKCFGLOCK1	Lock bit for CLKCFG registers	EALLOW	Go
8h	CLKSRCCTL1	Clock Source Control register-1	EALLOW	Go
Ah	CLKSRCCTL2	Clock Source Control register-2	EALLOW	Go
Ch	CLKSRCCTL3	Clock Source Control register-3	EALLOW	Go
Eh	SYSPLLCTL1	SYSPLL Control register-1	EALLOW	Go
14h	SYSPLLMULT	SYSPLL Multiplier register	EALLOW	Go
16h	SYSPLLSTS	SYSPLL Status register		Go
22h	SYSCLKDIVSEL	System Clock Divider Select register	EALLOW	Go
24h	AUXCLKDIVSEL	Auxillary Clock Divider Select register	EALLOW	Go
28h	XCLKOUTDIVSEL	XCLKOUT Divider Select register	EALLOW	Go
2Ch	LOSPCP	Low Speed Clock Source Prescalar	EALLOW	Go
2Eh	MCDCR	Missing Clock Detect Control Register	EALLOW	Go
30h	X1CNT	10-bit Counter on X1 Clock		Go
32h	XTALCR	XTAL Control Register	EALLOW	Go
3Ah	XTALCR2	XTAL Control Register for pad init	EALLOW	Go
3Ch	CLKFAILCFG	Clock Fail cause Configuration	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 3-41 shows the codes that are used for access types in this section.

Table 3-41. CLK_CFG_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 3-41. CLK_CFG_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.15.3.1 CLKCFGLOCK1 Register (Offset = 2h) [Reset = 0h]

CLKCFGLOCK1 is shown in Figure 3-41 and described in Table 3-42.

Return to the Summary Table.

Lock bit for CLKCFG registers
Notes:
[1] Any bit in this register, once set can only be cleared through a CPU1.SYSRSn. Write of 0 to any bit of this register has no effect
[2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed

Figure 3-41. CLKCFGLOCK1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							XTALCR
R-0h							R/WSonce-0h
15	14	13	12	11	10	9	8
LOSPCP	RESERVED	RESERVED	AUXCLKDIVSEL	SYSCLKDIVSEL	RESERVED	RESERVED	RESERVED
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	SYSPLLMULT	RESERVED	RESERVED	SYSPLLCTL1	CLKSRCCTL3	CLKSRCCTL2	CLKSRCCTL1
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 3-42. CLKCFGLOCK1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16	XTALCR	R/WSonce	0h	Common Lock bit for XTALCR & XTAL CR2 register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
15	LOSPCP	R/WSonce	0h	Lock bit for LOSPCP register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
14	RESERVED	R/WSonce	0h	Reserved
13	RESERVED	R/WSonce	0h	Reserved
12	AUXCLKDIVSEL	R/WSonce	0h	Lock bit for AUXCLKDIVSEL register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
11	SYSCLKDIVSEL	R/WSonce	0h	Lock bit for SYSCLKDIVSEL register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
10	RESERVED	R/WSonce	0h	Reserved
9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	RESERVED	R/WSonce	0h	Reserved

Table 3-42. CLKCFGLOCK1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	SYSPLLMULT	R/WSonce	0h	Lock bit for SYSPLLMULT register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
5	RESERVED	R/WSonce	0h	Reserved
4	RESERVED	R/WSonce	0h	Reserved
3	SYSPLLCTL1	R/WSonce	0h	Lock bit for SYSPLLCTL1 register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
2	CLKSRCCTL3	R/WSonce	0h	Lock bit for CLKSRCCTL3 register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
1	CLKSRCCTL2	R/WSonce	0h	Lock bit for CLKSRCCTL2 register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
0	CLKSRCCTL1	R/WSonce	0h	Lock bit for CLKSRCCTL1 register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn

3.15.3.2 CLKSRCCTL1 Register (Offset = 8h) [Reset = 0h]

CLKSRCCTL1 is shown in Figure 3-42 and described in Table 3-43.

Return to the Summary Table.

Clock Source Control register-1
This memory mapped register requires a delay of 45 SYSCLK cycles between subsequent writes to the register, otherwise a second write can be lost. This delay can be realized by adding 45 NOP instructions.

Figure 3-42. CLKSRCCTL1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		WDHALTI	RESERVED	INTOSC2OFF	RESERVED	OSCCLKSRCSEL	
R-0h		R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	

Table 3-43. CLKSRCCTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	WDHALTI	R/W	0h	Watchdog HALT Mode Ignore Bit: This bit determines if WD is functional in the HALT mode or not. 0 = WD is not functional in the HALT mode. Clock to WD is gated when system enters HALT mode. Additionally, INTOSC1 and INTOSC2 are powered-down when system enters HALT mode 1 = WD is functional in the HALT mode. Clock to WD is not gated and INTOSC1/2 are not powered-down when system enters HALT mode Reset type: XRSn
4	RESERVED	R/W	0h	Reserved
3	INTOSC2OFF	R/W	0h	Internal Oscillator 2 Off Bit: This bit turns oscillator 2 off: 0 = Internal Oscillator 2 On (default on reset) 1 = Internal Oscillator 2 Off This bit could be used by the user to turn off the internal oscillator 2 if it is not used. NOTE: Ensure no resources are using a clock source prior to disabling it. For example OSCCLKSRCSEL (SYSPLL), TMR2CLKSRCSEL (CPUTIMER2) and XCLOCKOUT (XCLKOUT). Reset type: XRSn
2	RESERVED	R	0h	Reserved

Table 3-43. CLKSRCCTL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	OSCCLKSRCSEL	R/W	0h	Oscillator Clock Source Select Bit: This bit selects the source for OSCCLK. 00 = INTOSC2 (default on reset) 01 = External Oscillator (XTAL) 10 = INTOSC1 11 = reserved (default to INTOSC1) At power-up or after an XRSn, INTOSC2 is selected by default. Whenever the user changes the clock source using these bits, the SYSPLLMULT[13:0] register will be forced to zero and the PLL will be bypassed and powered down. This prevents potential PLL overshoot. The user will then have to write to the SYSPLLMULT register to configure the appropriate multiplier. The user must wait 10 OSCCLK cycles before writing to SYSPLLMULT or disabling the previous clock source to allow the change to complete.. Notes: [1] INTOSC1 is recommended to be used only after missing clock detection. If user wants to re-lock the PLL with INTOSC1 (the back-up clock source) after missing clock is detected, he can do a MCLKCLR and lock the PLL. Reset type: XRSn

3.15.3.3 CLKSRCCTL2 Register (Offset = Ah) [Reset = 0h]

CLKSRCCTL2 is shown in Figure 3-43 and described in Table 3-44.

Return to the Summary Table.

Clock Source Control register-2
This memory mapped register requires a delay of 45 SYSCLK cycles between subsequent writes to the register, otherwise a second write can be lost. This delay can be realized by adding 45 NOP instructions.

Figure 3-43. CLKSRCCTL2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						RESERVED	
R-0h						R/W-0h	
15	14	13	12	11	10	9	8
RESERVED		RESERVED		MCANABCLKSEL		RESERVED	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		RESERVED		CANABCLKSEL		RESERVED	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 3-44. CLKSRCCTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reserved
17-16	RESERVED	R/W	0h	Reserved
15-14	RESERVED	R/W	0h	Reserved
13-12	RESERVED	R/W	0h	Reserved
11-10	MCANABCLKSEL	R/W	0h	MCAN Bit Clock Source Select Bit: 00 = CPU1SYSCLK 10 = AUXCLKIN 11 = PLLRAWCLK Missing clock detect circuit doesnt have any impact on these bits. Reset type: XRSn
9-8	RESERVED	R/W	0h	Reserved
7-6	RESERVED	R/W	0h	Reserved
5-4	RESERVED	R/W	0h	Reserved
3-2	CANABCLKSEL	R/W	0h	CANA Bit-Clock Source Select Bit: 00 = PERx.SYSCLK (default on reset) 01 = External Oscillator (XTAL) 10 = AUXCLKIN (from GPIO) 11 = Reserved Missing clock detect circuit doesnt have any impact on these bits. Reset type: XRSn
1-0	RESERVED	R/W	0h	Reserved

3.15.3.4 CLKSRCCTL3 Register (Offset = Ch) [Reset = 0h]

CLKSRCCTL3 is shown in Figure 3-44 and described in Table 3-45.

Return to the Summary Table.

Clock Source Control register-3
This memory mapped register requires a delay of 45 SYSCLK cycles between subsequent writes to the register, otherwise a second write can be lost. This delay can be realized by adding 45 NOP instructions.

Figure 3-44. CLKSRCCTL3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												XCLKOUTSEL			
R-0h												R/W-0h			

Table 3-45. CLKSRCCTL3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3-0	XCLKOUTSEL	R/W	0h	XCLKOUT Source Select Bit: This bit selects the source for XCLKOUT: 0000 = PLLSYSCLK (default on reset) 0001 = PLLCLK 0010 = SYSCLK 0101 = INTOSC1 0110 = INTOSC2 0111 = XTAL Rest = Reserved Reset type: SYSRSn

3.15.3.5 SYSPLLCTL1 Register (Offset = Eh) [Reset = 0h]

SYSPLLCTL1 is shown in Figure 3-45 and described in Table 3-46.

Return to the Summary Table.

SYSPLL Control register-1
This memory mapped register requires a delay of 45 SYSCLK cycles between subsequent writes to the register, otherwise a second write can be lost. This delay can be realized by adding 45 NOP instructions.

Figure 3-45. SYSPLLCTL1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RESERVED	RESERVED	RESERVED	RESERVED	PLLCLKEN	PLLEN
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-46. SYSPLLCTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	PLLCLKEN	R/W	0h	SYSPLL bypassed or included in the PLLSYSCLK path: This bit decides if the SYSPLL is bypassed when PLLSYSCLK is generated 1 = PLLSYSCLK is fed from the SYSPLL clock output. Users need to make sure that the PLL is locked before enabling this clock to the system. 0 = SYSPLL is bypassed. Clock to system is direct feed from OSCCLK Reset type: XRSn
0	PLLEN	R/W	0h	SYSPLL enabled or disabled: This bit decides if the SYSPLL is enabled or not 1 = SYSPLL is enabled 0 = SYSPLL is powered off. Clock to system is direct feed from OSCCLK Reset type: XRSn

3.15.3.6 SYSPLLMULT Register (Offset = 14h) [Reset = 0h]

SYSPLLMULT is shown in Figure 3-46 and described in Table 3-47.

Return to the Summary Table.

SYSPLL Multiplier register
NOTE: FMULT and IMULT fields must be written at the same time for correct PLL operation.
This memory mapped register requires a delay of 45 SYSCLK cycles between subsequent writes to the register, otherwise a second write can be lost. This delay can be realized by adding 45 NOP instructions.

Figure 3-46. SYSPLLMULT Register

31	30	29	28	27	26	25	24
RESERVED				REFDIV			
R-0h				R/W-0h			
23	22	21	20	19	18	17	16
RESERVED				ODIV			
R-0h				R/W-0h			
15	14	13	12	11	10	9	8
RESERVED		RESERVED		RESERVED		RESERVED	
R-0h		R/W-0h		R-0h		R/W-0h	
7	6	5	4	3	2	1	0
IMULT							
R/W-0h							

Table 3-47. SYSPLLMULT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reserved
28-24	REFDIV	R/W	0h	SYSPLL Reference Clock Divider PLL Reference Divider = REFDIV + 1 Reset type: XRSn
23-21	RESERVED	R	0h	Reserved
20-16	ODIV	R/W	0h	SYSPLL Output Clock Divider PLL Output Divider = ODIV + 1 Reset type: XRSn
15-14	RESERVED	R	0h	Reserved
13-12	RESERVED	R/W	0h	Reserved
11-10	RESERVED	R	0h	Reserved
9-8	RESERVED	R/W	0h	Reserved
7-0	IMULT	R/W	0h	SYSPLL Integer Multiplier: For 0000000 Fout = Fref (PLLBYPASS) Integer Multiplier = 1 0000001 Integer Multiplier = 1 0000010 Integer Multiplier = 2 0000011 Integer Multiplier = 3 ..... 1111111 Integer Multipler = 127 Note for APLL Multiplier values from 0-3 are invalid, internally those will be treated to 4. Reset type: XRSn

3.15.3.7 SYSPLLSTS Register (Offset = 16h) [Reset = 30h]

SYSPLLSTS is shown in Figure 3-47 and described in Table 3-48.

Return to the Summary Table.

SYSPLL Status register

Figure 3-47. SYSPLLSTS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RESERVED	RESERVED	REF_LOSTS	RESERVED	SLIPS	LOCKS
R-0h		R-1h	R-1h	W1C-0h	R-0h	R-0h	R-0h

Table 3-48. SYSPLLSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	RESERVED	R	1h	Reserved
4	RESERVED	R	1h	Reserved
3	REF_LOSTS	W1C	0h	SYSPLL "Reference Lost" Status Bit: This bit indicates whether the SYSPLL is out of lock range 0 = "Reference Lost" event has not occurred. 1 = "Reference Lost" event has occurred. Reset type: XRSn
2	RESERVED	R	0h	Reserved
1	SLIPS	R	0h	RESERVED: This bit is reserved and the value read should be ignored. TI recommends using DCC to evaluate SYSPLL Slip status. Refer to InitSysPll() or SysCtl_setClock() functions inside the latest example software from C2000Ware for checking SYSPLL Slip status using DCC. Reset type: XRSn
0	LOCKS	R	0h	SYSPLL Lock Status Bit: This bit indicates whether the SYSPLL is locked or not 0 = SYSPLL is not yet locked 1 = SYSPLL is locked Reset type: XRSn

3.15.3.8 SYSCLKDIVSEL Register (Offset = 22h) [Reset = 0h]

SYSCLKDIVSEL is shown in Figure 3-48 and described in Table 3-49.

Return to the Summary Table.

System Clock Divider Select register
This memory mapped register requires a delay of 45 SYSCLK cycles between subsequent writes to the register, otherwise a second write can be lost. This delay can be realized by adding 45 NOP instructions.

Figure 3-48. SYSCLKDIVSEL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							PLLSYSCLKDIV_LSB
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED		PLLSYSCLKDIV					
R-0h		R/W-0h					

Table 3-49. SYSCLKDIVSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	PLLSYSCLKDIV_LSB	R/W	0h	This bit is LSB of the Divider that when set allows the ODD divisions such that the divider value is {PLLSYSCLKDIV,PLLSYSCLKDIV_LSB}. E.g. if PLLSYSCLKDIV=0x1, and PLLSYSCLKDIV_LSB=0 then divider of 2 is used else in case PLLSYSCLKDIV_LSB=1 then divider value is 3. Reset type: XRSn
7-6	RESERVED	R	0h	Reserved
5-0	PLLSYSCLKDIV	R/W	0h	PLLSYSCLK Divide Select: This bit selects the divider setting for the PLLSYSCLK. 000000 = /1 000001 = /2 000010 = /4 (default on reset) 000011 = /6 000100 = /8 ..... 111111 = /126 The minimum value of this divider is 2 when PLL is enabled. Reset type: XRSn

3.15.3.9 AUXCLKDIVSEL Register (Offset = 24h) [Reset = 1301h]

AUXCLKDIVSEL is shown in Figure 3-49 and described in Table 3-50.

Return to the Summary Table.

Auxillary Clock Divider Select register

Figure 3-49. AUXCLKDIVSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED			MCANCLKDIV					RESERVED					RESERVED		
R-0h			R/W-13h					R-0h					R/W-1h		

Table 3-50. AUXCLKDIVSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12-8	MCANCLKDIV	R/W	13h	00000 = /1 00001 = /2 ... 10010 = /19 10011 = /20 101xx = Rsvd 11xxx = Rsvd Reset type: XRSn
7-3	RESERVED	R	0h	Reserved
2-0	RESERVED	R/W	1h	Reserved

3.15.3.10 XCLKOUTDIVSEL Register (Offset = 28h) [Reset = 3h]

XCLKOUTDIVSEL is shown in Figure 3-50 and described in Table 3-51.

Return to the Summary Table.

XCLKOUT Divider Select register
This memory mapped register requires a delay of 45 SYSCLK cycles between subsequent writes to the register, otherwise a second write can be lost. This delay can be realized by adding 45 NOP instructions.

Figure 3-50. XCLKOUTDIVSEL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						XCLKOUTDIV	
R-0h						R/W-3h	

Table 3-51. XCLKOUTDIVSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1-0	XCLKOUTDIV	R/W	3h	XCLKOUT Divide Select: This bit selects the divider setting for the XCLKOUT. 00 = /1 01 = /2 10 = /4 11 = /8 (default on reset) Reset type: SYSRSn

3.15.3.11 LOSPCP Register (Offset = 2Ch) [Reset = 2h]

LOSPCP is shown in Figure 3-51 and described in Table 3-52.

Return to the Summary Table.

Low Speed Clock Source Prescalar

Figure 3-51. LOSPCP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													LSPCLKDIV		
R-0h													R/W-2h		

Table 3-52. LOSPCP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reserved
2-0	LSPCLKDIV	R/W	2h	These bits configure the low-speed peripheral clock (LSPCLK) rate 000,LSPCLK = / 1 001,LSPCLK = / 2 010,LSPCLK = / 4 (default on reset) 011,LSPCLK = / 6 100,LSPCLK = / 8 101,LSPCLK = / 10 110,LSPCLK = / 12 111,LSPCLK = / 14 Note: [1] This clock is used as strobe for the SCI and SPI modules. Reset type: SYSRSn

3.15.3.12 MCDCR Register (Offset = 2Eh) [Reset = 0h]

MCDCR is shown in Figure 3-52 and described in Table 3-53.

Return to the Summary Table.

Missing Clock Detect Control Register

Figure 3-52. MCDCR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						RESERVED	RESERVED
R-0h						R/W-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
RESERVED	SYSREF_LOST_MCD_EN	SYSREF_LOSTSCLR	SYSREF_LOSTS	OSCOFF	MCLKOFF	MCLKCLR	MCLKSTS
R-0h	R/W-0h	R-0/W1S-0h	R-0h	R/W-0h	R/W-0h	R-0/W1S-0h	R-0h

Table 3-53. MCDCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9	RESERVED	R/W	0h	Reserved
8	RESERVED	R-0/W1S	0h	Reserved
7	RESERVED	R	0h	Reserved
6	SYSREF_LOST_MCD_EN	R/W	0h	Control to add "PLL reference clock lost" as cause for MCD 0 = "PLL reference clock Lost" does not affect MCD. 1 = Upon "PLL reference clock Lost" MCD is asserted. Reset type: XRSn
5	SYSREF_LOSTSCLR	R-0/W1S	0h	Clears the REF_LOST_STS from PLLSTS which is root for MCD trigger. 0 = No effect on present state of the REF_LOST_STS 1 = Clears the REF_LOST_STS bit to '0'. Bit clears itself after clear pulse to REF_LOST_STS. Read always gives '0'. Reset type: XRSn
4	SYSREF_LOSTS	R	0h	SYSPLL "Reference Lost" Status Bit: This bit indicates whether the SYSPLL is out of lock range 0 = "Reference Lost" event has not occurred. 1 = "Reference Lost" event has occurred. Reset type: XRSn
3	OSCOFF	R/W	0h	Oscillator Clock Disconnect from MCD Bit: 0 = OSCCLK Connected to OSCCLK Counter in MCD module 1 = OSCCLK Disconnected to OSCCLK Counter in MCD module Reset type: XRSn
2	MCLKOFF	R/W	0h	Missing Clock Detect Off Bit: 0 = Missing Clock Detect Circuit Enabled 1 = Missing Clock Detect Circuit Disabled Reset type: XRSn
1	MCLKCLR	R-0/W1S	0h	Missing Clock Clear Bit: Write 1" to this bit to clear MCLKSTS bit and reset the missing clock detect circuit." Reset type: XRSn

Table 3-53. MCDCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	MCLKSTS	R	0h	Missing Clock Status Bit: 0 = OSCCLK Is OK 1 = OSCCLK Detected Missing, CLOCKFAILn Generated Reset type: XRSn

3.15.3.13 X1CNT Register (Offset = 30h) [Reset = 0h]

X1CNT is shown in Figure 3-53 and described in Table 3-54.

Return to the Summary Table.

10-bit Counter on X1 Clock

Figure 3-53. X1CNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															CLR
R-0h															R/W-0h
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED					X1CNT										
R-0h					R-0h										

Table 3-54. X1CNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16	CLR	R/W	0h	X1 Counter clear: A write of '1' to this bit field clears the X1CNT and makes it count from 0x0 again (provided X1 clock is ticking). Writes of '0' are ignore to this bit field Reset type: XRSn
15-11	RESERVED	R	0h	Reserved
10-0	X1CNT	R	0h	X1 Counter: - This counter increments on every X1 CLOCKs positive-edge. - Once it reaches the values of 0x7ff, it freezes - Before switching from INTOSC2 to X1, application must check this counter and make sure that it has saturated. This will ensure that the Crystal connected to X1/X2 is oscillating. Reset type: XRSn

3.15.3.14 XTALCR Register (Offset = 32h) [Reset = 5h]

XTALCR is shown in Figure 3-54 and described in Table 3-55.

Return to the Summary Table.

XTAL Control Register
This memory mapped register requires a delay of 45 SYSCLK cycles between subsequent writes to the register, otherwise a second write can be lost. This delay can be realized by adding 45 NOP instructions.

Figure 3-54. XTALCR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					RESERVED	SE	OSCOFF
R-0h					R/W-1h	R/W-0h	R/W-1h

Table 3-55. XTALCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reserved
2	RESERVED	R/W	1h	Reserved
1	SE	R/W	0h	Configures XTAL oscillator in single-ended or Crystal mode when XTAL oscillator is powered up(i.e. OSCOFF = 0) 0 XTAL oscillator in Crystal mode 1 XTAL oscilator in single-ended mode (through X1) Reset type: XRSn
0	OSCOFF	R/W	1h	This bit if '1', powers-down the XTAL oscillator macro and hence doesn't let X2 to be driven by the XTAL oscillator. If a crystal is connected to X1/X2, user needs to first clear this bit, wait for the oscillator to power up (using X1CNT) and then only switch the clock source to X1/X2 Reset type: XRSn

3.15.3.15 XTALCR2 Register (Offset = 3Ah) [Reset = 3h]

XTALCR2 is shown in Figure 3-55 and described in Table 3-56.

Return to the Summary Table.

XTAL Control Register for pad init

Figure 3-55. XTALCR2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													FEN	XOF	XIF
R-0h													R/W-0h	R/W-1h	R/W-1h

Table 3-56. XTALCR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R/W	0h	Reserved
15-3	RESERVED	R	0h	Reserved
2	FEN	R/W	0h	Configures XTAL oscillator pad initilisation. 0 : XOSC pads are not driven through GPIO connection. 1 : XOSC pads are driven through connected GPIO as per XIF & XOF values. This register has effect only when XOSC is OFF (no SE , no XTAL mode). If this register is set during XOSC off state (XOSCOFF=1 & SE=0) then upon change of these controls this bit gets reset and rearmed. Reset type: XRSn
1	XOF	R/W	1h	Polarity selection to initialise XO /X2 pad of the XOSC before start-up This value shall be deposited on the pad before XOSC started (XOSCOFF=1) If FEN=0 or XOSC is in XTAL or SE mode then this value will not be applied to the pad. Reset type: XRSn
0	XIF	R/W	1h	Polarity selection to initialise XI /X1 pad of the XOSC before start-up This value shall be deposited on the pad before XOSC started (XOSCOFF=1) If FEN=0 or XOSC is in XTAL or SE mode then this value will not be applied to the pad. Reset type: XRSn

3.15.3.16 CLKFAILCFG Register (Offset = 3Ch) [Reset = 0h]

CLKFAILCFG is shown in Figure 3-56 and described in Table 3-57.

Return to the Summary Table.

Clock Fail cause Configuration

Figure 3-56. CLKFAILCFG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						DCC1_ERROR_EN	DCC0_ERROR_EN
R-0h						R/W-0h	R/W-0h

Table 3-57. CLKFAILCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	DCC1_ERROR_EN	R/W	0h	This field enables DCC1 Error to cause the clock-fail NMI to get asserted. 0 : DCC1 Error does not affect Clock fail NMI 1: Occurrence of DCC1 Error triggers Clock fail NMI assertion and ERROR pin assertion. Reset type: XRSn
0	DCC0_ERROR_EN	R/W	0h	This field enables DCC0 Error to cause the clock-fail NMI to get asserted. 0 : DCC0 Error does not affect Clock fail NMI 1: Occurrence of DCC0 Error triggers Clock fail NMI assertion and ERROR pin assertion. Reset type: XRSn

3.15.4 CPU_SYS_REGS Registers

Table 3-58 lists the memory-mapped registers for the CPU_SYS_REGS registers. All register offset addresses not listed in Table 3-58 should be considered as reserved locations and the register contents should not be modified.

Table 3-58. CPU_SYS_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CPUSYSLOCK1	Lock bit for CPUSYS registers	EALLOW	Go
2h	CPUSYSLOCK2	Lock bit for CPUSYS registers	EALLOW	Go
Ah	PIEVERRADDR	PIE Vector Fetch Error Address register	EALLOW	Go
22h	PCLKCR0	Peripheral Clock Gating Registers	EALLOW	Go
26h	PCLKCR2	Peripheral Clock Gating Register - ETPWM	EALLOW	Go
28h	PCLKCR3	Peripheral Clock Gating Register - ECAP	EALLOW	Go
2Ah	PCLKCR4	Peripheral Clock Gating Register - EQEP	EALLOW	Go
2Eh	PCLKCR6	Peripheral Clock Gating Register - SDFM	EALLOW	Go
30h	PCLKCR7	Peripheral Clock Gating Register - SCI	EALLOW	Go
32h	PCLKCR8	Peripheral Clock Gating Register - SPI	EALLOW	Go
34h	PCLKCR9	Peripheral Clock Gating Register - I2C	EALLOW	Go
36h	PCLKCR10	Peripheral Clock Gating Register - CAN	EALLOW	Go
3Ch	PCLKCR13	Peripheral Clock Gating Register - ADC	EALLOW	Go
3Eh	PCLKCR14	Peripheral Clock Gating Register - CMPSS	EALLOW	Go
42h	PCLKCR16	Peripheral Clock Gating Register Buf_DAC	EALLOW	Go
44h	PCLKCR17	Peripheral Clock Gating Register - CLB	EALLOW	Go
46h	PCLKCR18	Peripheral Clock Gating Register - FSI	EALLOW	Go
48h	PCLKCR19	Peripheral Clock Gating Register - LIN	EALLOW	Go
4Ah	PCLKCR20	Peripheral Clock Gating Register - PMBUS	EALLOW	Go
4Ch	PCLKCR21	Peripheral Clock Gating Register - DCC	EALLOW	Go
54h	PCLKCR25	Peripheral Clock Gating Register - HIC	EALLOW	Go
56h	PCLKCR26	Peripheral Clock Gating Register - AES	EALLOW	Go
58h	PCLKCR27	Peripheral Clock Gating Register - EPG	EALLOW	Go
70h	SIMRESET	Simulated Reset Register		Go
76h	LPMCR	LPM Control Register	EALLOW	Go
78h	GPIOLPMSEL0	GPIO LPM Wakeup select registers	EALLOW	Go
7Ah	GPIOLPMSEL1	GPIO LPM Wakeup select registers	EALLOW	Go
7Ch	TMR2CLKCTL	Timer2 Clock Measurement functionality control register	EALLOW	Go
7Eh	RESCCLR	Reset Cause Clear Register		Go
80h	RESC	Reset Cause register		Go
98h	MCANWAKESTATUS	MCAN Wake Status Register		Go
9Ah	MCANWAKESTATUSCLR	MCAN Wake Status Clear Register		Go
9Ch	CLKSTOPREQ	Peripheral Clock Stop Request Register		Go
9Eh	CLKSTOPACK	Peripheral Clock Stop Ackonwledge Register		Go

Complex bit access types are encoded to fit into small table cells. Table 3-59 shows the codes that are used for access types in this section.

Table 3-59. CPU_SYS_REGS Access Type Codes

Access Type	Code	Description
Read Type		

Table 3-59. CPU_SYS_REGS Access Type Codes (continued)

Access Type	Code	Description
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.15.4.1 CPUSYSLOCK1 Register (Offset = 0h) [Reset = 0h]

CPUSYSLOCK1 is shown in Figure 3-57 and described in Table 3-60.

Return to the Summary Table.

Lock bit for CPUSYS registers
Notes:
[1] Any bit in this register, once set can only be cleared through a CPU1.SYSRSn. Write of 0 to any bit of this register has no effect
[2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed

Figure 3-57. CPUSYSLOCK1 Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	PCLKCR22	PCLKCR21	PCLKCR20	PCLKCR19	PCLKCR18	PCLKCR17
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h
23	22	21	20	19	18	17	16
GPIOLPMSEL1	GPIOLPMSEL0	LPMCR	RESERVED	PCLKCR16	RESERVED	PCLKCR14	PCLKCR13
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	PCLKCR10	PCLKCR9	PCLKCR8	PCLKCR7	PCLKCR6	RESERVED
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h
7	6	5	4	3	2	1	0
PCLKCR4	PCLKCR3	PCLKCR2	RESERVED	PCLKCR0	PIEVERRADDR	RESERVED	RESERVED
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 3-60. CPUSYSLOCK1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/WSonce	0h	Reserved
30	RESERVED	R/WSonce	0h	Reserved
29	PCLKCR22	R/WSonce	0h	Lock bit for PCLKCR22 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
28	PCLKCR21	R/WSonce	0h	Lock bit for PCLKCR21 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
27	PCLKCR20	R/WSonce	0h	Lock bit for PCLKCR20 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
26	PCLKCR19	R/WSonce	0h	Lock bit for PCLKCR19 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
25	PCLKCR18	R/WSonce	0h	Lock bit for PCLKCR18 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
24	PCLKCR17	R/WSonce	0h	Lock bit for PCLKCR17 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn

Table 3-60. CPUSYSLOCK1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23	GPIOLPMSEL1	R/WSonce	0h	Lock bit for GPIOLPMSEL1 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
22	GPIOLPMSEL0	R/WSonce	0h	Lock bit for GPIOLPMSEL0 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
21	LPMCR	R/WSonce	0h	Lock bit for LPMCR Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
20	RESERVED	R/WSonce	0h	Reserved
19	PCLKCR16	R/WSonce	0h	Lock bit for PCLKCR16 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
18	RESERVED	R/WSonce	0h	Reserved
17	PCLKCR14	R/WSonce	0h	Lock bit for PCLKCR14 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
16	PCLKCR13	R/WSonce	0h	Lock bit for PCLKCR13 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
15	RESERVED	R/WSonce	0h	Reserved
14	RESERVED	R/WSonce	0h	Reserved
13	PCLKCR10	R/WSonce	0h	Lock bit for PCLKCR10 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
12	PCLKCR9	R/WSonce	0h	Lock bit for PCLKCR9 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
11	PCLKCR8	R/WSonce	0h	Lock bit for PCLKCR8 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
10	PCLKCR7	R/WSonce	0h	Lock bit for PCLKCR7 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
9	PCLKCR6	R/WSonce	0h	Lock bit for PCLKCR6 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
8	RESERVED	R/WSonce	0h	Reserved
7	PCLKCR4	R/WSonce	0h	Lock bit for PCLKCR4 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
6	PCLKCR3	R/WSonce	0h	Lock bit for PCLKCR3 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn

Table 3-60. CPUSYSLOCK1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	PCLKCR2	R/WSonce	0h	Lock bit for PCLKCR2 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
4	RESERVED	R/WSonce	0h	Reserved
3	PCLKCR0	R/WSonce	0h	Lock bit for PCLKCR0 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
2	PIEVERRADDR	R/WSonce	0h	Lock bit for PIEVERRADDR Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
1	RESERVED	R/WSonce	0h	Reserved
0	RESERVED	R/WSonce	0h	Reserved

3.15.4.2 CPUSYSLOCK2 Register (Offset = 2h) [Reset = 0h]

CPUSYSLOCK2 is shown in Figure 3-58 and described in Table 3-61.

Return to the Summary Table.

Lock bit for CPUSYS registers
Notes:
[1] Any bit in this register, once set can only be cleared through a CPU1.SYSRSn. Write of 0 to any bit of this register has no effect
[2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed

Figure 3-58. CPUSYSLOCK2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				PCLKCR27	PCLKCR26	PCLKCR25	RESERVED
R-0h				R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 3-61. CPUSYSLOCK2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	PCLKCR27	R/WSonce	0h	Lock bit for PCLKCR27 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
2	PCLKCR26	R/WSonce	0h	Lock bit for PCLKCR26 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
1	PCLKCR25	R/WSonce	0h	Lock bit for PCLKCR25 Register: 0: Respective register is not locked 1: Respective register is locked. Reset type: SYSRSn
0	RESERVED	R/WSonce	0h	Reserved

3.15.4.3 PIEVERRADDR Register (Offset = Ah) [Reset = 003FFFFFh]

PIEVERRADDR is shown in Figure 3-59 and described in Table 3-62.

Return to the Summary Table.

PIE Vector Fetch Error Address register

Figure 3-59. PIEVERRADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED										ADDR																					
R-0h										R/W-003FFFFFh																					

Table 3-62. PIEVERRADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R	0h	Reserved
21-0	ADDR	R/W	003FFFFFh	This register defines the address of the PIE Vector Fetch Error handler routine. Its the responsibility of user to initialize this register. If this register is not initialized, a default error handler at address 0x3fffbe will get executed. Refer to the Boot ROM section for more details on this register. Reset type: XRSn

3.15.4.4 PCLKCR0 Register (Offset = 22h) [Reset = 38h]

PCLKCR0 is shown in Figure 3-60 and described in Table 3-63.

Return to the Summary Table.

Peripheral Clock Gating Registers

Figure 3-60. PCLKCR0 Register

31	30	29	28	27	26	25	24
RESERVED							ERAD
R-0h							R/W-0h
23	22	21	20	19	18	17	16
RESERVED				RESERVED	TBCLKSYNC	RESERVED	HRCAL
R-0h				R/W-0h	R/W-0h	R-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED	CLA1BGCRC	CPUBGCRC	RESERVED				
R-0h	R/W-0h	R/W-0h	R-0h				
7	6	5	4	3	2	1	0
RESERVED		CPUTIMER2	CPUTIMER1	CPUTIMER0	DMA	RESERVED	CLA1
R-0h		R/W-1h	R/W-1h	R/W-1h	R/W-0h	R/W-0h	R/W-0h

Table 3-63. PCLKCR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24	ERAD	R/W	0h	ERAD Clock Enable Bit: When set, this enables the clock to the ERAD module 1: ERAD clock is enabled 0: ERAD clock is disabled Reset type: SYSRSn
23-20	RESERVED	R	0h	Reserved
19	RESERVED	R/W	0h	Reserved
18	TBCLKSYNC	R/W	0h	EPWM Time Base Clock sync: When set PWM time bases of all the PWM modules belonging to the same CPU-Subsystem (as partitioned using their CPUSEL bits) start counting Reset type: SYSRSn
17	RESERVED	R	0h	Reserved
16	HRCAL	R/W	0h	HRCAL Clock Enable Bit: When set, this enables the clock to the HRCAL module 1: HRCAL clock is enabled 0: HRCAL clock is disabled Reset type: SYSRSn
15	RESERVED	R	0h	Reserved
14	CLA1BGCRC	R/W	0h	CLA1BGCRC Clock Enable Bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
13	CPUBGCRC	R/W	0h	CPUBGCRC Clock Enable Bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
12-6	RESERVED	R	0h	Reserved
5	CPUTIMER2	R/W	1h	CPUTIMER2 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

Table 3-63. PCLKCR0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	CPUTIMER1	R/W	1h	CPUTIMER1 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
3	CPUTIMER0	R/W	1h	CPUTIMER0 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
2	DMA	R/W	0h	DMA Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
1	RESERVED	R/W	0h	Reserved
0	CLA1	R/W	0h	CLA1 Clock Enable Bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.15.4.5 PCLKCR2 Register (Offset = 26h) [Reset = 0h]

PCLKCR2 is shown in Figure 3-61 and described in Table 3-64.

Return to the Summary Table.

Peripheral Clock Gating Register - ETPWM

Figure 3-61. PCLKCR2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
EPWM8	EPWM7	EPWM6	EPWM5	EPWM4	EPWM3	EPWM2	EPWM1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-64. PCLKCR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	RESERVED	R/W	0h	Reserved
14	RESERVED	R/W	0h	Reserved
13	RESERVED	R/W	0h	Reserved
12	RESERVED	R/W	0h	Reserved
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	RESERVED	R/W	0h	Reserved
8	RESERVED	R/W	0h	Reserved
7	EPWM8	R/W	0h	EPWM8 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
6	EPWM7	R/W	0h	EPWM7 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
5	EPWM6	R/W	0h	EPWM6 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
4	EPWM5	R/W	0h	EPWM5 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
3	EPWM4	R/W	0h	EPWM4 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

Table 3-64. PCLKCR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	EPWM3	R/W	0h	EPWM3 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
1	EPWM2	R/W	0h	EPWM2 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
0	EPWM1	R/W	0h	EPWM1 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.15.4.6 PCLKCR3 Register (Offset = 28h) [Reset = 0h]

PCLKCR3 is shown in Figure 3-62 and described in Table 3-65.

Return to the Summary Table.

Peripheral Clock Gating Register - ECAP

Figure 3-62. PCLKCR3 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	ECAP3	ECAP2	ECAP1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-65. PCLKCR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	RESERVED	R/W	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	ECAP3	R/W	0h	ECAP3 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
1	ECAP2	R/W	0h	ECAP2 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
0	ECAP1	R/W	0h	ECAP1 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.15.4.7 PCLKCR4 Register (Offset = 2Ah) [Reset = 0h]

PCLKCR4 is shown in Figure 3-63 and described in Table 3-66.

Return to the Summary Table.

Peripheral Clock Gating Register - EQEP

Figure 3-63. PCLKCR4 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	RESERVED	EQEP2	EQEP1
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-66. PCLKCR4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	EQEP2	R/W	0h	EQEP2 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
0	EQEP1	R/W	0h	EQEP1 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.15.4.8 PCLKCR6 Register (Offset = 2Eh) [Reset = 0h]

PCLKCR6 is shown in Figure 3-64 and described in Table 3-67.

Return to the Summary Table.

Peripheral Clock Gating Register - SDFM

Figure 3-64. PCLKCR6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	SD2	SD1
R-0h								R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-67. PCLKCR6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	RESERVED	R/W	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	SD2	R/W	0h	SD2 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
0	SD1	R/W	0h	SD1 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.15.4.9 PCLKCR7 Register (Offset = 30h) [Reset = 0h]

PCLKCR7 is shown in Figure 3-65 and described in Table 3-68.

Return to the Summary Table.

Peripheral Clock Gating Register - SCI

Figure 3-65. PCLKCR7 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	RESERVED	SCI_B	SCI_A
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-68. PCLKCR7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	SCI_B	R/W	0h	SCI_B Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
0	SCI_A	R/W	0h	SCI_A Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.15.4.10 PCLKCR8 Register (Offset = 32h) [Reset = 0h]

PCLKCR8 is shown in Figure 3-66 and described in Table 3-69.

Return to the Summary Table.

Peripheral Clock Gating Register - SPI

Figure 3-66. PCLKCR8 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						RESERVED	RESERVED
R-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	RESERVED	SPI_B	SPI_A
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-69. PCLKCR8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	SPI_B	R/W	0h	SPI_B Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
0	SPI_A	R/W	0h	SPI_A Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.15.4.11 PCLKCR9 Register (Offset = 34h) [Reset = 0h]

PCLKCR9 is shown in Figure 3-67 and described in Table 3-70.

Return to the Summary Table.

Peripheral Clock Gating Register - I2C

Figure 3-67. PCLKCR9 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						I2C_B	I2C_A
R-0h						R/W-0h	R/W-0h

Table 3-70. PCLKCR9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	I2C_B	R/W	0h	I2C_B Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
0	I2C_A	R/W	0h	I2C_A Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.15.4.12 PCLKCR10 Register (Offset = 36h) [Reset = 0h]

PCLKCR10 is shown in Figure 3-68 and described in Table 3-71.

Return to the Summary Table.

Peripheral Clock Gating Register - CAN

Figure 3-68. PCLKCR10 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	MCAN_A	RESERVED	RESERVED	RESERVED	CAN_A
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-71. PCLKCR10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	RESERVED	R/W	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	MCAN_A	R/W	0h	MCAN_A Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	RESERVED	R/W	0h	Reserved
0	CAN_A	R/W	0h	CAN_A Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.15.4.13 PCLKCR13 Register (Offset = 3Ch) [Reset = 0h]

PCLKCR13 is shown in Figure 3-69 and described in Table 3-72.

Return to the Summary Table.

Peripheral Clock Gating Register - ADC

Figure 3-69. PCLKCR13 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	ADC_C	ADC_B	ADC_A
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-72. PCLKCR13 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	ADC_C	R/W	0h	ADC_C Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
1	ADC_B	R/W	0h	ADC_B Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
0	ADC_A	R/W	0h	ADC_A Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.15.4.14 PCLKCR14 Register (Offset = 3Eh) [Reset = 0h]

PCLKCR14 is shown in Figure 3-70 and described in Table 3-73.

Return to the Summary Table.

Peripheral Clock Gating Register - CMPSS

Figure 3-70. PCLKCR14 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	CMPSS4	CMPSS3	CMPSS2	CMPSS1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-73. PCLKCR14 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	RESERVED	R/W	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	CMPSS4	R/W	0h	CMPSS4 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
2	CMPSS3	R/W	0h	CMPSS3 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
1	CMPSS2	R/W	0h	CMPSS2 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
0	CMPSS1	R/W	0h	CMPSS1 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.15.4.15 PCLKCR16 Register (Offset = 42h) [Reset = 0h]

PCLKCR16 is shown in Figure 3-71 and described in Table 3-74.

Return to the Summary Table.

Peripheral Clock Gating Register Buf_DAC

Figure 3-71. PCLKCR16 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				RESERVED	RESERVED	DAC_B	DAC_A
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	RESERVED	RESERVED	RESERVED
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-74. PCLKCR16 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19	RESERVED	R/W	0h	Reserved
18	RESERVED	R/W	0h	Reserved
17	DAC_B	R/W	0h	Buffered_DAC_B Clock Enable Bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
16	DAC_A	R/W	0h	Buffered_DAC_A Clock Enable Bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
15-4	RESERVED	R	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	RESERVED	R/W	0h	Reserved
0	RESERVED	R/W	0h	Reserved

3.15.4.16 PCLKCR17 Register (Offset = 44h) [Reset = 0h]

PCLKCR17 is shown in Figure 3-72 and described in Table 3-75.

Return to the Summary Table.

Peripheral Clock Gating Register - CLB

Figure 3-72. PCLKCR17 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				CLB4	CLB3	CLB2	CLB1
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-75. PCLKCR17 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	CLB4	R/W	0h	CLB4 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
2	CLB3	R/W	0h	CLB3 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
1	CLB2	R/W	0h	CLB2 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
0	CLB1	R/W	0h	CLB1 Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.15.4.17 PCLKCR18 Register (Offset = 46h) [Reset = 0h]

PCLKCR18 is shown in Figure 3-73 and described in Table 3-76.

Return to the Summary Table.

Peripheral Clock Gating Register - FSI

Figure 3-73. PCLKCR18 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	RESERVED	FSIRX_A	FSITX_A
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-76. PCLKCR18 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	FSIRX_A	R/W	0h	FSIRX_A Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
0	FSITX_A	R/W	0h	FSITX_A Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.15.4.18 PCLKCR19 Register (Offset = 48h) [Reset = 0h]

PCLKCR19 is shown in Figure 3-74 and described in Table 3-77.

Return to the Summary Table.

Peripheral Clock Gating Register - LIN

Figure 3-74. PCLKCR19 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	RESERVED	LIN_B	LIN_A
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-77. PCLKCR19 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	LIN_B	R/W	0h	LIN_B Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
0	LIN_A	R/W	0h	LIN_A Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.15.4.19 PCLKCR20 Register (Offset = 4Ah) [Reset = 0h]

PCLKCR20 is shown in Figure 3-75 and described in Table 3-78.

Return to the Summary Table.

Peripheral Clock Gating Register - PMBUS

Figure 3-75. PCLKCR20 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						RESERVED	PMBUS_A
R-0h						R/W-0h	R/W-0h

Table 3-78. PCLKCR20 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	RESERVED	R/W	0h	Reserved
0	PMBUS_A	R/W	0h	PMBUS_A Clock Enable bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.15.4.20 PCLKCR21 Register (Offset = 4Ch) [Reset = 0h]

PCLKCR21 is shown in Figure 3-76 and described in Table 3-79.

Return to the Summary Table.

Peripheral Clock Gating Register - DCC

Figure 3-76. PCLKCR21 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						DCC1	DCC0
R-0h						R/W-0h	R/W-0h

Table 3-79. PCLKCR21 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	DCC1	R/W	0h	DCC Clock Enable Bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn
0	DCC0	R/W	0h	DCC Clock Enable Bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.15.4.21 PCLKCR25 Register (Offset = 54h) [Reset = 0h]

PCLKCR25 is shown in Figure 3-77 and described in Table 3-80.

Return to the Summary Table.

Peripheral Clock Gating Register - HIC

Figure 3-77. PCLKCR25 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							HICA
R-0h							R/W-0h

Table 3-80. PCLKCR25 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	HICA	R/W	0h	HICA Clock Enable Bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.15.4.22 PCLKCR26 Register (Offset = 56h) [Reset = 0h]

PCLKCR26 is shown in Figure 3-78 and described in Table 3-81.

Return to the Summary Table.

Peripheral Clock Gating Register - AES

Figure 3-78. PCLKCR26 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							AESA
R-0h							R/W-0h

Table 3-81. PCLKCR26 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	AESA	R/W	0h	AESA Clock Enable Bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.15.4.23 PCLKCR27 Register (Offset = 58h) [Reset = 0h]

PCLKCR27 is shown in Figure 3-79 and described in Table 3-82.

Return to the Summary Table.

Peripheral Clock Gating Register - EPG

Figure 3-79. PCLKCR27 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EPG1
R-0h							R/W-0h

Table 3-82. PCLKCR27 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	EPG1	R/W	0h	EPG1 Clock Enable Bit: 0: Module clock is gated-off 1: Module clock is turned-on Reset type: SYSRSn

3.15.4.24 SIMRESET Register (Offset = 70h) [Reset = 0h]

SIMRESET is shown in Figure 3-80 and described in Table 3-83.

Return to the Summary Table.

Simulated Reset Register
Note: This register exists only on CPU1

Figure 3-80. SIMRESET Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						XRSn	CPU1RSn
R-0h						R-0/W1S-0h	R-0/W1S-0h

Table 3-83. SIMRESET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Write to this register succeeds only if this field is written with a value of 0xa5a5 Note: [1] Due to this KEY, only 32-bit writes will succeed (provided the KEY matches). 16-bit writes to the upper or lower half of this register will be ignored Reset type: XRSn
15-2	RESERVED	R	0h	Reserved
1	XRSn	R-0/W1S	0h	Writing a 1 to this field generates a XRSn like reset. Writing a 0 has no effect. Note: Writing to this pin will pull the XRSn pin low for 512 INTOSC1 clock cycles. Reset type: XRSn
0	CPU1RSn	R-0/W1S	0h	Writing a 1 to this field generates a reset to to CPU1. Writing a 0 has no effect. Reset type: XRSn

3.15.4.25 LPMCR Register (Offset = 76h) [Reset = FCh]

LPMCR is shown in Figure 3-81 and described in Table 3-84.

Return to the Summary Table.

LPM Control Register

Figure 3-81. LPMCR Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED						
R/W1S-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED						RESERVED	
R-0h						R/W-0h	
15	14	13	12	11	10	9	8
WDINTE	RESERVED						
R/W-0h	R-0h						
7	6	5	4	3	2	1	0
QUALSTDBY						LPM	
R/W-3Fh						R/W-0h	

Table 3-84. LPMCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W1S	0h	Reserved
30-18	RESERVED	R	0h	Reserved
17-16	RESERVED	R/W	0h	Reserved
15	WDINTE	R/W	0h	When this bit is set to 1, it enables the watchdog interrupt signal to wake the device from STANDBY mode. Note: [1] To use this signal, the user must also enable the WDINTn signal using the WDENINT bit in the SCSR register. This signal will not wake the device from HALT mode because the clock to watchdog module is turned off Reset type: SYSRSn
14-8	RESERVED	R	0h	Reserved
7-2	QUALSTDBY	R/W	3Fh	Select number of OSCCLK clock cycles to qualify the selected inputs when waking the from STANDBY mode: 000000 = 2 OSCCLKs 000001 = 3 OSCCLKs ..... 111111 = 65 OSCCLKs Note: The LPMCR.QUALSTDBY register should be set to a value greater than the ratio of INTOSC1/PLLSYSCLK to ensure proper wake up. Reset type: SYSRSn
1-0	LPM	R/W	0h	These bits set the low power mode for the device. Takes effect when CPU executes the IDLE instruction (when IDLE instruction is out of EXE Phase of the Pipeline) 00: IDLE Mode 01: STANDBY Mode 1x: HALT Mode Reset type: SYSRSn

3.15.4.26 GPIOLPMSEL0 Register (Offset = 78h) [Reset = 0h]

GPIOLPMSEL0 is shown in Figure 3-82 and described in Table 3-85.

Return to the Summary Table.

GPIO LPM Wakeup select registers
Connects the selected pin to the LPM circuit. Refer to LPM section of the TRM for the wakeup capabilities of the selected pin.

Figure 3-82. GPIOLPMSEL0 Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-85. GPIOLPMSEL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
30	GPIO30	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
29	GPIO29	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
28	GPIO28	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
27	GPIO27	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
26	GPIO26	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
25	GPIO25	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
24	GPIO24	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
23	GPIO23	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
22	GPIO22	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn

Table 3-85. GPIOLPMSEL0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	GPIO21	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
20	GPIO20	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
19	GPIO19	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
18	GPIO18	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
17	GPIO17	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
16	GPIO16	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
15	GPIO15	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
14	GPIO14	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
13	GPIO13	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
12	GPIO12	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
11	GPIO11	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
10	GPIO10	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
9	GPIO9	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
8	GPIO8	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
7	GPIO7	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
6	GPIO6	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
5	GPIO5	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
4	GPIO4	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
3	GPIO3	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn

Table 3-85. GPIOLPMSEL0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	GPIO2	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
1	GPIO1	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
0	GPIO0	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn

3.15.4.27 GPIOLPMSEL1 Register (Offset = 7Ah) [Reset = 0h]

GPIOLPMSEL1 is shown in Figure 3-83 and described in Table 3-86.

Return to the Summary Table.

GPIO LPM Wakeup select registers
Connects the selected pin to the LPM circuit. Refer to LPM section of the TRM for the wakeup capabilities of the selected pin.

Figure 3-83. GPIOLPMSEL1 Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	RESERVED	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO39	RESERVED	GPIO37	RESERVED	GPIO35	GPIO34	GPIO33	GPIO32
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-86. GPIOLPMSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30	RESERVED	R/W	0h	Reserved
29	RESERVED	R/W	0h	Reserved
28	GPIO60	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
27	GPIO59	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
26	GPIO58	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
25	GPIO57	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
24	GPIO56	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
23	GPIO55	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
22	GPIO54	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
21	GPIO53	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn

Table 3-86. GPIOLPMSEL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	GPIO52	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
19	GPIO51	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
18	GPIO50	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
17	GPIO49	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
16	GPIO48	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
15	GPIO47	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
14	GPIO46	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
13	GPIO45	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
12	GPIO44	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
11	GPIO43	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
10	GPIO42	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
9	GPIO41	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
8	GPIO40	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
7	GPIO39	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
6	RESERVED	R/W	0h	Reserved
5	GPIO37	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
4	RESERVED	R/W	0h	Reserved
3	GPIO35	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
2	GPIO34	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn
1	GPIO33	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn

Table 3-86. GPIOLPMSEL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	GPIO32	R/W	0h	0 pin is dis-connected from LPM circuit 1 pin is connected to LPM circuit Reset type: SYSRSn

3.15.4.28 TMR2CLKCTL Register (Offset = 7Ch) [Reset = 0h]

TMR2CLKCTL is shown in Figure 3-84 and described in Table 3-87.

Return to the Summary Table.

Timer2 Clock Measurement functionality control register

Figure 3-84. TMR2CLKCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		TMR2CLKPRESCALE			TMR2CLKSRCSEL		
R-0h		R/W-0h			R/W-0h		

Table 3-87. TMR2CLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5-3	TMR2CLKPRESCALE	R/W	0h	CPU Timer 2 Clock Pre-Scale Value: These bits select the pre-scale value for the selected clock source for CPU Timer 2: 000: /1 (default on reset) 001: /2, 010: /4 011: /8 100: /16 101: spare (defaults to /16) 110: spare (defaults to /16) 111: spare (defaults to /16) Note: [1] The CPU Timer2s Clock sync logic detects an input clock edge when configured for any clock source other than SYSCLK and generates an appropriate clock pulse to the CPU timer2. If SYSCLK is approximately the same or less then the input clock source, then the user would need to configure the pre-scale value such that SYSCLK is at least twice as fast as the pre-scaled value. Reset type: SYSRSn
2-0	TMR2CLKSRCSEL	R/W	0h	CPU Timer 2 Clock Source Select Bit: This bit selects the source for CPU Timer 2: 000 =SYSCLK Selected (default on reset, pre-scale is bypassed) 001 = INTOSC1 010 = INTOSC2 011 = XTAL Other values = reserved Reset type: SYSRSn

3.15.4.29 RESCCLR Register (Offset = 7Eh) [Reset = 0h]

RESCCLR is shown in Figure 3-85 and described in Table 3-88.

Return to the Summary Table.

Reset Cause Clear Register

Figure 3-85. RESCCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				SIMRESET_XRSn	SIMRESET_CPU1RSn	RESERVED	SCCRESETn
R-0h				W1C-0h	W1C-0h	R-0h	W1S-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	HWBISTn	RESERVED	NMIWDRSn	WDRSn	XRSn	POR
R-0h	W1S-0h	W1S-0h	R-0h	W1S-0h	W1S-0h	W1S-0h	W1S-0h

Table 3-88. RESCCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved
11	SIMRESET_XRSn	W1C	0h	Clear bit for corresponding status bit in RESC. Read of RESCCLR always gives 0. Writing a 1 to this bit clears the status bit in RESC to 0 Writing 0 has no effect. Reset type: SYSRSn
10	SIMRESET_CPU1RSn	W1C	0h	Clear bit for corresponding status bit in RESC. Read of RESCCLR always gives 0. Writing a 1 to this bit clears the status bit in RESC to 0 Writing 0 has no effect. Reset type: SYSRSn
9	RESERVED	R	0h	Reserved
8	SCCRESETn	W1S	0h	Clear bit for corresponding status bit in RESC. Read of RESCCLR always gives 0. Writing a 1 to this bit clears the status bit in RESC to 0 Writing 0 has no effect. Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6	RESERVED	W1S	0h	Reserved
5	HWBISTn	W1S	0h	Clear bit for corresponding status bit in RESC. Read of RESCCLR always gives 0. Writing a 1 to this bit clears the status bit in RESC to 0 Writing 0 has no effect. Reset type: SYSRSn
4	RESERVED	R	0h	Reserved
3	NMIWDRSn	W1S	0h	Clear bit for corresponding status bit in RESC. Read of RESCCLR always gives 0. Writing a 1 to this bit clears the status bit in RESC to 0 Writing 0 has no effect. Reset type: SYSRSn

Table 3-88. RESCCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	WDRSn	W1S	0h	Clear bit for corresponding status bit in RESC. Read of RESCCLR always gives 0. Writing a 1 to this bit clears the status bit in RESC to 0 Writing 0 has no effect. Reset type: SYSRSn
1	XRSn	W1S	0h	Clear bit for corresponding status bit in RESC. Read of RESCCLR always gives 0. Writing a 1 to this bit clears the status bit in RESC to 0 Writing 0 has no effect. Reset type: SYSRSn
0	POR	W1S	0h	Clear bit for corresponding status bit in RESC. Read of RESCCLR always gives 0. Writing a 1 to this bit clears the status bit in RESC to 0 Writing 0 has no effect. Reset type: SYSRSn

3.15.4.30 RESC Register (Offset = 80h) [Reset = X]

RESC is shown in Figure 3-86 and described in Table 3-89.

Return to the Summary Table.

Reset Cause register

Figure 3-86. RESC Register

31	30	29	28	27	26	25	24
DCON	XRSn_pin_status	RESERVED					
R-0h	R-X	R-0h					
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				SIMRESET_XRSn	SIMRESET_CPU1RSn	RESERVED	SCCRESETn
R-0h				R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	HWBISTn	RESERVED	NMIWDRSn	WDRSn	XRSn	POR
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-1h	R-1h

Table 3-89. RESC Register Field Descriptions

Bit	Field	Type	Reset	Description
31	DCON	R	0h	Reading this bit provides the status of debugger connection to the C28x CPU. 0 : Debugger is not connected to the C28x CPU 1 : Debugger is connected to the C28x CPU Notes: [1] This bit is connected to the DCON o/p signal of the C28x CPU Reset type: N/A
30	XRSn_pin_status	R	X	Reading this bit provides the current status of the XRSn pin. Reset value is reflective of the pin status. Reset type: N/A
29-16	RESERVED	R	0h	Reserved
15-12	RESERVED	R	0h	Reserved
11	SIMRESET_XRSn	R	0h	If this bit is set, indicates that the device was reset by SIMRESET_XRSn Reset type: PORESETn
10	SIMRESET_CPU1RSn	R	0h	If this bit is set, indicates that the device was reset by SIMRESET_CPU1RSn Reset type: PORESETn
9	RESERVED	R	0h	Reserved
8	SCCRESETn	R	0h	If this bit is set, indicates that the device was reset by SCCRESETn (fired by DCSM). Reset type: PORESETn
7	RESERVED	R	0h	Reserved
6	RESERVED	R	0h	Reserved
5	HWBISTn	R	0h	If this bit is set, indicates that the device was reset by HWBIST. Reset type: PORESETn
4	RESERVED	R	0h	Reserved

Table 3-89. RESC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	NMIWDRSn	R	0h	If this bit is set, indicates that the device was reset by NMIWDRSn. Note: To know the exact cause of NMI after the reset, software needs to read NMISHDFLG registers Reset type: PORESETn
2	WDRSn	R	0h	If this bit is set, indicates that the device was reset by WDRSn. Note: [1] A bit inside WD module also provides the same information. This bit is present to keep things consistent. This register is a one-stop shop for the software to know the reset cause for the C28x core. Reset type: PORESETn
1	XRSn	R	1h	If this bit is set, indicates that the device was reset by XRSn. Reset type: PORESETn
0	POR	R	1h	If this bit is set, indicates that the device was reset by PORn/BORn. Reset type: PORESETn

3.15.4.31 MCANWAKESTATUS Register (Offset = 98h) [Reset = 0h]

MCANWAKESTATUS is shown in Figure 3-87 and described in Table 3-90.

Return to the Summary Table.

MCAN Wake Status Register

Figure 3-87. MCANWAKESTATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							WAKE
R-0h							R-0h

Table 3-90. MCANWAKESTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	WAKE	R	0h	0 : wakeup event has not occured. 1 : wakeup event has occured. Reset type: SYSRSn

3.15.4.32 MCANWAKESTATUSCLR Register (Offset = 9Ah) [Reset = 0h]

MCANWAKESTATUSCLR is shown in Figure 3-88 and described in Table 3-91.

Return to the Summary Table.

MCAN Wake Status Clear Register

Figure 3-88. MCANWAKESTATUSCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							WAKE
R-0h							R-0/W1S-0h

Table 3-91. MCANWAKESTATUSCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	WAKE	R-0/W1S	0h	0 : No effect. 1 : Clears WAKE bit of MCANWAKESTATUS register Reset type: SYSRSn

3.15.4.33 CLKSTOPREQ Register (Offset = 9Ch) [Reset = 0h]

CLKSTOPREQ is shown in Figure 3-89 and described in Table 3-92.

Return to the Summary Table.

Peripheral Clock Stop Request Register
Note: This register exists only on CPU1

Figure 3-89. CLKSTOPREQ Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED				RESERVED			MCAN_A
R-0h				R-0h			R/W-0h
7	6	5	4	3	2	1	0
RESERVED		RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R-0h		R/W-0h	R/W-0h	R-0h	R/W-0h	R-0h	R/W-0h

Table 3-92. CLKSTOPREQ Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Write to any of the bits in this register will succeed only if a value of 0x5634 is written to the KEY field. Reset type: SYSRSn
15-12	RESERVED	R	0h	Reserved
11-9	RESERVED	R	0h	Reserved
8	MCAN_A	R/W	0h	MCAN_A Clock Stop Request Bit 0: If clock to MCAN_A is turned off, it will be turned on, else no effect. 1: Clock stop request toMCAN_A Note: Once set, this bit is cleared when clock to MCAN_A is turned on as a result of a wakeup event in hardware Reset type: SYSRSn
7-6	RESERVED	R	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	RESERVED	R	0h	Reserved
0	RESERVED	R/W	0h	Reserved

3.15.4.34 CLKSTOPACK Register (Offset = 9Eh) [Reset = 0h]

CLKSTOPACK is shown in Figure 3-90 and described in Table 3-93.

Return to the Summary Table.

Peripheral Clock Stop Ackonwledge Register
Note: This register exists only on CPU1

Figure 3-90. CLKSTOPACK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							MCAN_A
R-0h							R-0h
7	6	5	4	3	2	1	0
RESERVED		RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R-0h		R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 3-93. CLKSTOPACK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	MCAN_A	R	0h	MCAN_A Clock Stop Acknowledge Bit 0: Clock stop request not acknowledged 1: Clock stop acknowledged Reset type: SYSRSn
7-6	RESERVED	R	0h	Reserved
5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3	RESERVED	R	0h	Reserved
2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

3.15.5 CPUTIMER_REGS Registers

Table 3-94 lists the memory-mapped registers for the CPUTIMER_REGS registers. All register offset addresses not listed in Table 3-94 should be considered as reserved locations and the register contents should not be modified.

Table 3-94. CPUTIMER_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	TIM	CPU-Timer, Counter Register		Go
2h	PRD	CPU-Timer, Period Register		Go
4h	TCR	CPU-Timer, Control Register		Go
6h	TPR	CPU-Timer, Prescale Register		Go
7h	TPRH	CPU-Timer, Prescale Register High		Go

Complex bit access types are encoded to fit into small table cells. Table 3-95 shows the codes that are used for access types in this section.

Table 3-95. CPUTIMER_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.15.5.1 TIM Register (Offset = 0h) [Reset = FFFFh]

TIM is shown in Figure 3-91 and described in Table 3-96.

Return to the Summary Table.

CPU-Timer, Counter Register

Figure 3-91. TIM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSW																LSW															
R/W-0h																R/W-FFFFh															

Table 3-96. TIM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	MSW	R/W	0h	CPU-Timer Counter Registers The TIMH register holds the high 16 bits of the current 32-bit count of the timer. The TIMH:TIM decrements by one every (TDDRH:TDDR+1) clock cycles, where TDDRH:TDDR is the timer prescale dividedown value. When the TIMH:TIM decrements to zero, the TIMH:TIM register is reloaded with the period value contained in the PRDH:PRD registers. The timer interrupt (TINT) signal is generated. Reset type: SYSRSn
15-0	LSW	R/W	FFFFh	CPU-Timer Counter Registers The TIM register holds the low 16 bits of the current 32-bit count of the timer. The TIMH:TIM decrements by one every (TDDRH:TDDR+1) clock cycles, where TDDRH:TDDR is the timer prescale dividedown value. When the TIMH:TIM decrements to zero, the TIMH:TIM register is reloaded with the period value contained in the PRDH:PRD registers. The timer interrupt (TINT) signal is generated. Reset type: SYSRSn

3.15.5.2 PRD Register (Offset = 2h) [Reset = FFFFh]

PRD is shown in Figure 3-92 and described in Table 3-97.

Return to the Summary Table.

CPU-Timer, Period Register

Figure 3-92. PRD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MSW																LSW															
R/W-0h																R/W-FFFFh															

Table 3-97. PRD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	MSW	R/W	0h	CPU-Timer Period Registers The PRDH register holds the high 16 bits of the 32-bit period. When the TIMH:TIM decrements to zero, the TIMH:TIM register is reloaded with the period value contained in the PRDH:PRD registers, at the start of the next timer input clock cycle (the output of the prescaler). The PRDH:PRD contents are also loaded into the TIMH:TIM when you set the timer reload bit (TRB) in the Timer Control Register (TCR). Reset type: SYSRSn
15-0	LSW	R/W	FFFFh	CPU-Timer Period Registers The PRD register holds the low 16 bits of the 32-bit period. When the TIMH:TIM decrements to zero, the TIMH:TIM register is reloaded with the period value contained in the PRDH:PRD registers, at the start of the next timer input clock cycle (the output of the prescaler). The PRDH:PRD contents are also loaded into the TIMH:TIM when you set the timer reload bit (TRB) in the Timer Control Register (TCR). Reset type: SYSRSn

3.15.5.3 TCR Register (Offset = 4h) [Reset = 1h]

TCR is shown in Figure 3-93 and described in Table 3-98.

Return to the Summary Table.

CPU-Timer, Control Register

Figure 3-93. TCR Register

15	14	13	12	11	10	9	8
TIF	TIE	RESERVED		FREE	SOFT	RESERVED	
R/W1C-0h	R/W-0h	R-0h		R/W-0h	R/W-0h	R-0h	
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
RESERVED		TRB	TSS	RESERVED			
R-0h		R/W-0h	R/W-0h	R-0h			

Table 3-98. TCR Register Field Descriptions

Bit	Field	Type	Reset	Description
15	TIF	R/W1C	0h	CPU-Timer Overflow Flag. TIF indicates whether a timer overflow has happened since TIF was last cleared. TIF is not cleared automatically and does not need to be cleared to enable the next timer interrupt. Reset type: SYSRSn 0h (R/W) = The CPU-Timer has not decremented to zero. Writes of 0 are ignored. 1h (R/W) = This flag gets set when the CPU-timer decrements to zero. Writing a 1 to this bit clears the flag.
14	TIE	R/W	0h	CPU-Timer Interrupt Enable. Reset type: SYSRSn 0h (R/W) = The CPU-Timer interrupt is disabled. 1h (R/W) = The CPU-Timer interrupt is enabled. If the timer decrements to zero, and TIE is set, the timer asserts its interrupt request.
13-12	RESERVED	R	0h	Reserved
11	FREE	R/W	0h	If the FREE bit is set to 1, then, upon a software breakpoint, the timer continues to run. If FREE is 0, then the SOFT bit controls the emulation behavior. Reset type: SYSRSn 0h (R/W) = Stop after the next decrement of the TIMH:TIM (hard stop) (SOFT bit controls the emulation behavior) 1h (R/W) = Free Run (SOFT bit is don't care, counter is free running)
10	SOFT	R/W	0h	If the FREE bit is set to 1, then, upon a software breakpoint, the timer continues to run (that is, free runs). In this case, SOFT is a don't care. But if FREE is 0, then SOFT takes effect. Reset type: SYSRSn 0h (R/W) = Stop after the next decrement of the TIMH:TIM (hard stop). (ONLY if FREE=0, if FREE=1 this bit is don't care) 1h (R/W) = Stop after the TIMH:TIM decrements to 0 (soft stop) In the SOFT STOP mode, the timer generates an interrupt before shutting down (since reaching 0 is the interrupt causing condition). (ONLY if FREE=0, if FREE=1 this bit is don't care)
9-6	RESERVED	R	0h	Reserved

Table 3-98. TCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	TRB	R/W	0h	Timer reload Reset type: SYSRSn 0h (R/W) = The TRB bit is always read as zero. Writes of 0 are ignored. 1h (R/W) = When you write a 1 to TRB, the TIMH:TIM is loaded with the value in the PRDH:PRD, and the prescaler counter (PSCH:PSC) is loaded with the value in the timer dividedown register (TDDRH:TDDR).
4	TSS	R/W	0h	CPU-Timer stop status bit. TSS is a 1-bit flag that stops or starts the CPU-timer. Reset type: SYSRSn 0h (R/W) = Reads of 0 indicate the CPU-timer is running. To start or restart the CPU-timer, set TSS to 0. At reset, TSS is cleared to 0 and the CPU-timer immediately starts. 1h (R/W) = Reads of 1 indicate that the CPU-timer is stopped. To stop the CPU-timer, set TSS to 1.
3-0	RESERVED	R	0h	Reserved

3.15.5.4 TPR Register (Offset = 6h) [Reset = 0h]

TPR is shown in Figure 3-94 and described in Table 3-99.

Return to the Summary Table.

CPU-Timer, Prescale Register

Figure 3-94. TPR Register

15	14	13	12	11	10	9	8
PSC							
R-0h							
7	6	5	4	3	2	1	0
TDDR							
R/W-0h							

Table 3-99. TPR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	PSC	R	0h	CPU-Timer Prescale Counter. These bits hold the current prescale count for the timer. For every timer clock source cycle that the PSCH:PSC value is greater than 0, the PSCH:PSC decrements by one. One timer clock (output of the timer prescaler) cycle after the PSCH:PSC reaches 0, the PSCH:PSC is loaded with the contents of the TDDRH:TDDR, and the timer counter register (TIMH:TIM) decrements by one. The PSCH:PSC is also reloaded whenever the timer reload bit (TRB) is set by software. The PSCH:PSC can be checked by reading the register, but it cannot be set directly. It must get its value from the timer divide-down register (TDDRH:TDDR). At reset, the PSCH:PSC is set to 0. Reset type: SYSRSn
7-0	TDDR	R/W	0h	CPU-Timer Divide-Down. Every (TDDRH:TDDR + 1) timer clock source cycles, the timer counter register (TIMH:TIM) decrements by one. At reset, the TDDRH:TDDR bits are cleared to 0. To increase the overall timer count by an integer factor, write this factor minus one to the TDDRH:TDDR bits. When the prescaler counter (PSCH:PSC) value is 0, one timer clock source cycle later, the contents of the TDDRH:TDDR reload the PSCH:PSC, and the TIMH:TIM decrements by one. TDDRH:TDDR also reloads the PSCH:PSC whenever the timer reload bit (TRB) is set by software. Reset type: SYSRSn

3.15.5.5 TPRH Register (Offset = 7h) [Reset = 0h]

TPRH is shown in Figure 3-95 and described in Table 3-100.

Return to the Summary Table.

CPU-Timer, Prescale Register High

Figure 3-95. TPRH Register

15	14	13	12	11	10	9	8
PSCH							
R-0h							
7	6	5	4	3	2	1	0
TDDRH							
R/W-0h							

Table 3-100. TPRH Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	PSCH	R	0h	See description of TIMERxTPR. Reset type: SYSRSn
7-0	TDDRH	R/W	0h	See description of TIMERxTPR. Reset type: SYSRSn

3.15.6 DEV_CFG_REGS Registers

Table 3-101 lists the memory-mapped registers for the DEV_CFG_REGS registers. All register offset addresses not listed in Table 3-101 should be considered as reserved locations and the register contents should not be modified.

Table 3-101. DEV_CFG_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
8h	PARTIDL	Lower 32-bit of Device PART Identification Number		Go
Ah	PARTIDH	Upper 32-bit of Device PART Identification Number		Go
Ch	REVID	Device Revision Number		Go
74h	FUSEERR	e-Fuse error Status register		Go
82h	SOFTPRES0	Processing Block Software Reset register	EALLOW	Go
86h	SOFTPRES2	EPWM Software Reset register	EALLOW	Go
88h	SOFTPRES3	ECAP Software Reset register	EALLOW	Go
8Ah	SOFTPRES4	EQEP Software Reset register	EALLOW	Go
8Eh	SOFTPRES6	Sigma Delta Software Reset register	EALLOW	Go
90h	SOFTPRES7	SCI Software Reset register	EALLOW	Go
92h	SOFTPRES8	SPI Software Reset register	EALLOW	Go
94h	SOFTPRES9	I2C Software Reset register	EALLOW	Go
96h	SOFTPRES10	CAN Software Reset register	EALLOW	Go
9Ch	SOFTPRES13	ADC Software Reset register	EALLOW	Go
9Eh	SOFTPRES14	CMPSS Software Reset register	EALLOW	Go
A2h	SOFTPRES16	DAC Software Reset register	EALLOW	Go
A4h	SOFTPRES17	CLB Software Reset register	EALLOW	Go
A6h	SOFTPRES18	FSI Software Reset register	EALLOW	Go
A8h	SOFTPRES19	LIN Software Reset register	EALLOW	Go
AAh	SOFTPRES20	PMBUS Software Reset register	EALLOW	Go
ACh	SOFTPRES21	DCC Software Reset register	EALLOW	Go
B4h	SOFTPRES25	HIC Software Reset register	EALLOW	Go
B6h	SOFTPRES26	AES Software Reset register	EALLOW	Go
B8h	SOFTPRES27	EPG Software Reset register	EALLOW	Go
130h	TAP_STATUS	Status of JTAG State machine & Debugger Connect		Go
19Bh	ECAPTYPE	Configures ECAP Type for the device	EALLOW	Go
19Ch	SDFMTYPE	Configures SDFM Type for the device	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 3-102 shows the codes that are used for access types in this section.

Table 3-102. DEV_CFG_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write

Table 3-102. DEV_CFG_REGS Access Type Codes (continued)

Access Type	Code	Description
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
WOnce	W Once	Write Write once
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.15.6.1 PARTIDL Register (Offset = 8h) [Reset = X]

PARTIDL is shown in Figure 3-96 and described in Table 3-103.

Return to the Summary Table.

Lower 32-bit of Device PART Identification Number

Figure 3-96. PARTIDL Register

31	30	29	28	27	26	25	24
RESERVED				RESERVED			
R-0h				R-0h			
23	22	21	20	19	18	17	16
FLASH_SIZE							
R-X							
15	14	13	12	11	10	9	8
RESERVED	INSTASPIN		RESERVED	RESERVED	PIN_COUNT		
R-0h	R-X		R-0h	R-X	R-X		
7	6	5	4	3	2	1	0
QUAL		RESERVED	RESERVED		RESERVED		
R-X		R-0h	R-0h		R-0h		

Table 3-103. PARTIDL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-24	RESERVED	R	0h	Reserved
23-16	FLASH_SIZE	R	X	0x7 - 384KB 0x6 - 256KB 0x5 - 128KB Reset type: PORESETn
15	RESERVED	R	0h	Reserved
14-13	INSTASPIN	R	X	1 = InstaSPIN-FOC 2 = NONE 3 = NONE Reset type: PORESETn
12	RESERVED	R	0h	Reserved
11	RESERVED	R	X	Reserved
10-8	PIN_COUNT	R	X	1 = 64 pin (QFP - Q100) 2 = 64 pin (QFP) 3 = 80 pin (QFP) 4 = 48 pin (QFP) 5 = 100 pin (QFP) Reset type: PORESETn
7-6	QUAL	R	X	0 = Engineering sample (TMX) 1 = Pilot production (TMP) 2 = Fully qualified (TMS) Reset type: PORESETn
5	RESERVED	R	0h	Reserved
4-3	RESERVED	R	0h	Reserved
2-0	RESERVED	R	0h	Reserved

3.15.6.2 PARTIDH Register (Offset = Ah) [Reset = X]

PARTIDH is shown in Figure 3-97 and described in Table 3-104.

Return to the Summary Table.

Upper 32-bit of Device PART Identification Number

Figure 3-97. PARTIDH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DEVICE_CLASS_ID								PARTNO							
R-5h								R-X							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FAMILY								RESERVED				RESERVED			
R-5h								R-0h				R-0h			

Table 3-104. PARTIDH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	DEVICE_CLASS_ID	R	5h	Device class ID Reset type: PORESETn
23-16	PARTNO	R	X	Part Number Designator 0xFF - F280039x 0xFE - F280038x 0xFD - F280037x 0xFC - F280036x 0xFB - F280035x 0xFA - F280034x Reset type: PORESETn
15-8	FAMILY	R	5h	Device Family Reset type: PORESETn
7-4	RESERVED	R	0h	Reserved
3-0	RESERVED	R	0h	Reserved

3.15.6.3 REVID Register (Offset = Ch) [Reset = 0h]

REVID is shown in Figure 3-98 and described in Table 3-105.

Return to the Summary Table.

Device Revision Number

Figure 3-98. REVID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																REVID															
R-0h																R-0h															

Table 3-105. REVID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	REVID	R	0h	Device Revision Reset type: N/A

3.15.6.4 FUSEERR Register (Offset = 74h) [Reset = 0h]

FUSEERR is shown in Figure 3-99 and described in Table 3-106.

Return to the Summary Table.

e-Fuse error Status register

Figure 3-99. FUSEERR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED										ERR	ALERR				
R-0h										R-0h	R-0h				

Table 3-106. FUSEERR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	ERR	R	0h	Efuse Self Test Error Status set by hardware after fuse self test completes, in case of self test error 0: No error during fuse self test 1: Fuse self test error Reset type: XRSn
4-0	ALERR	R	0h	Efuse Autoload Error Status set by hardware after fuse auto load completes 00000: No error in auto load Other: Non zero value indicates error in autoload Note: [1] 10101 means a single-bit error during autoload. Since this gets corrected by the ECC mechanism, this value shouldn't be treated as an error condition. Reset type: XRSn

3.15.6.5 SOFTPRES0 Register (Offset = 82h) [Reset = 0h]

SOFTPRES0 is shown in Figure 3-100 and described in Table 3-107.

Return to the Summary Table.

Processing Block Software Reset register
When bits in this register are set, the respective module is in reset. All design data is lost and the module registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-100. SOFTPRES0 Register

31	30	29	28	27	26	25	24
RESERVED							CPU1_ERAD
R-0h							R/W-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	CPU1_CLA1BGCRC	CPU1_CPUBGCRC	RESERVED				
R-0h	R/W-0h	R/W-0h	R-0h				
7	6	5	4	3	2	1	0
RESERVED				RESERVED	RESERVED	RESERVED	CPU1_CLA1
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-107. SOFTPRES0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24	CPU1_ERAD	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
23-16	RESERVED	R	0h	Reserved
15	RESERVED	R	0h	Reserved
14	CPU1_CLA1BGCRC	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
13	CPU1_CPUBGCRC	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
12-4	RESERVED	R	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	RESERVED	R/W	0h	Reserved
0	CPU1_CLA1	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn

3.15.6.6 SOFTPRES2 Register (Offset = 86h) [Reset = 0h]

SOFTPRES2 is shown in Figure 3-101 and described in Table 3-108.

Return to the Summary Table.

Peripheral Software Reset register
When bits in this register are set, the respective peripheral is in reset. All data is lost and the peripheral registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-101. SOFTPRES2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
EPWM8	EPWM7	EPWM6	EPWM5	EPWM4	EPWM3	EPWM2	EPWM1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-108. SOFTPRES2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	RESERVED	R/W	0h	Reserved
14	RESERVED	R/W	0h	Reserved
13	RESERVED	R/W	0h	Reserved
12	RESERVED	R/W	0h	Reserved
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	RESERVED	R/W	0h	Reserved
8	RESERVED	R/W	0h	Reserved
7	EPWM8	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
6	EPWM7	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
5	EPWM6	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
4	EPWM5	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
3	EPWM4	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
2	EPWM3	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn

Table 3-108. SOFTPRES2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	EPWM2	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
0	EPWM1	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn

3.15.6.7 SOFTPRES3 Register (Offset = 88h) [Reset = 0h]

SOFTPRES3 is shown in Figure 3-102 and described in Table 3-109.

Return to the Summary Table.

Peripheral Software Reset register
When bits in this register are set, the respective peripheral is in reset. All data is lost and the peripheral registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-102. SOFTPRES3 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	ECAP3	ECAP2	ECAP1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-109. SOFTPRES3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	RESERVED	R/W	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	ECAP3	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
1	ECAP2	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
0	ECAP1	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn

3.15.6.8 SOFTPRES4 Register (Offset = 8Ah) [Reset = 0h]

SOFTPRES4 is shown in Figure 3-103 and described in Table 3-110.

Return to the Summary Table.

Peripheral Software Reset register
When bits in this register are set, the respective peripheral is in reset. All data is lost and the peripheral registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-103. SOFTPRES4 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	RESERVED	EQEP2	EQEP1
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-110. SOFTPRES4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	EQEP2	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
0	EQEP1	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn

3.15.6.9 SOFTPRES6 Register (Offset = 8Eh) [Reset = 0h]

SOFTPRES6 is shown in Figure 3-104 and described in Table 3-111.

Return to the Summary Table.

Peripheral Software Reset register
When bits in this register are set, the respective peripheral is in reset. All data is lost and the peripheral registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-104. SOFTPRES6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	SD2	SD1
R-0h								R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-111. SOFTPRES6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	RESERVED	R/W	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	SD2	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
0	SD1	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn

3.15.6.10 SOFTPRES7 Register (Offset = 90h) [Reset = 0h]

SOFTPRES7 is shown in Figure 3-105 and described in Table 3-112.

Return to the Summary Table.

Peripheral Software Reset register
When bits in this register are set, the respective peripheral is in reset. All data is lost and the peripheral registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-105. SOFTPRES7 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	RESERVED	SCI_B	SCI_A
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-112. SOFTPRES7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	SCI_B	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
0	SCI_A	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn

3.15.6.11 SOFTPRES8 Register (Offset = 92h) [Reset = 0h]

SOFTPRES8 is shown in Figure 3-106 and described in Table 3-113.

Return to the Summary Table.

Peripheral Software Reset register
When bits in this register are set, the respective peripheral is in reset. All data is lost and the peripheral registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-106. SOFTPRES8 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						RESERVED	RESERVED
R-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	RESERVED	SPI_B	SPI_A
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-113. SOFTPRES8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	SPI_B	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
0	SPI_A	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn

3.15.6.12 SOFTPRES9 Register (Offset = 94h) [Reset = 0h]

SOFTPRES9 is shown in Figure 3-107 and described in Table 3-114.

Return to the Summary Table.

Peripheral Software Reset register
When bits in this register are set, the respective peripheral is in reset. All data is lost and the peripheral registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-107. SOFTPRES9 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						I2C_B	I2C_A
R-0h						R/W-0h	R/W-0h

Table 3-114. SOFTPRES9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	I2C_B	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
0	I2C_A	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn

3.15.6.13 SOFTPRES10 Register (Offset = 96h) [Reset = X]

SOFTPRES10 is shown in Figure 3-108 and described in Table 3-115.

Return to the Summary Table.

Peripheral Software Reset register
When bits in this register are set, the respective peripheral is in reset. All data is lost and the peripheral registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-108. SOFTPRES10 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	MCAN_A	RESERVED	RESERVED	RESERVED	CAN_A
R-X	R-X	R-X	R-X	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-115. SOFTPRES10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	RESERVED	R	X	Reserved
6	RESERVED	R	X	Reserved
5	RESERVED	R	X	Reserved
4	MCAN_A	R	X	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	RESERVED	R/W	0h	Reserved
0	CAN_A	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn

3.15.6.14 SOFTPRES13 Register (Offset = 9Ch) [Reset = 0h]

SOFTPRES13 is shown in Figure 3-109 and described in Table 3-116.

Return to the Summary Table.

Peripheral Software Reset register
When bits in this register are set, the respective peripheral is in reset. All data is lost and the peripheral registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-109. SOFTPRES13 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	ADC_C	ADC_B	ADC_A
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-116. SOFTPRES13 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	ADC_C	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
1	ADC_B	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
0	ADC_A	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn

3.15.6.15 SOFTPRES14 Register (Offset = 9Eh) [Reset = 0h]

SOFTPRES14 is shown in Figure 3-110 and described in Table 3-117.

Return to the Summary Table.

Peripheral Software Reset register
When bits in this register are set, the respective peripheral is in reset. All data is lost and the peripheral registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-110. SOFTPRES14 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	CMPSS4	CMPSS3	CMPSS2	CMPSS1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-117. SOFTPRES14 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	RESERVED	R/W	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	CMPSS4	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
2	CMPSS3	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
1	CMPSS2	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
0	CMPSS1	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn

3.15.6.16 SOFTPRES16 Register (Offset = A2h) [Reset = 0h]

SOFTPRES16 is shown in Figure 3-111 and described in Table 3-118.

Return to the Summary Table.

Peripheral Software Reset register
When bits in this register are set, the respective peripheral is in reset. All data is lost and the peripheral registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-111. SOFTPRES16 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				RESERVED	RESERVED	DAC_B	DAC_A
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	RESERVED	RESERVED	RESERVED
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-118. SOFTPRES16 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19	RESERVED	R/W	0h	Reserved
18	RESERVED	R/W	0h	Reserved
17	DAC_B	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
16	DAC_A	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
15-4	RESERVED	R	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	RESERVED	R/W	0h	Reserved
0	RESERVED	R/W	0h	Reserved

3.15.6.17 SOFTPRES17 Register (Offset = A4h) [Reset = X]

SOFTPRES17 is shown in Figure 3-112 and described in Table 3-119.

Return to the Summary Table.

Peripheral Software Reset register
When bits in this register are set, the respective peripheral is in reset. All data is lost and the peripheral registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-112. SOFTPRES17 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				CLB4	CLB3	CLB2	CLB1
R-0h				R/W-0h	R/W-0h	R-X	R-X

Table 3-119. SOFTPRES17 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	CLB4	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
2	CLB3	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
1	CLB2	R	X	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
0	CLB1	R	X	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn

3.15.6.18 SOFTPRES18 Register (Offset = A6h) [Reset = X]

SOFTPRES18 is shown in Figure 3-113 and described in Table 3-120.

Return to the Summary Table.

Peripheral Software Reset register
When bits in this register are set, the respective peripheral is in reset. All data is lost and the peripheral registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-113. SOFTPRES18 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	RESERVED	FSIRX_A	FSITX_A
R-0h				R/W-0h	R/W-0h	R-X	R-X

Table 3-120. SOFTPRES18 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	FSIRX_A	R	X	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
0	FSITX_A	R	X	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn

3.15.6.19 SOFTPRES19 Register (Offset = A8h) [Reset = 0h]

SOFTPRES19 is shown in Figure 3-114 and described in Table 3-121.

Return to the Summary Table.

Peripheral Software Reset register
When bits in this register are set, the respective peripheral is in reset. All data is lost and the peripheral registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-114. SOFTPRES19 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	RESERVED	LIN_B	LIN_A
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-121. SOFTPRES19 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	LIN_B	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
0	LIN_A	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn

3.15.6.20 SOFTPRES20 Register (Offset = AAh) [Reset = X]

SOFTPRES20 is shown in Figure 3-115 and described in Table 3-122.

Return to the Summary Table.

Peripheral Software Reset register
When bits in this register are set, the respective peripheral is in reset. All data is lost and the peripheral registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-115. SOFTPRES20 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						RESERVED	PMBUS_A
R-0h						R-X	R-X

Table 3-122. SOFTPRES20 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	RESERVED	R	X	Reserved
0	PMBUS_A	R	X	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn

3.15.6.21 SOFTPRES21 Register (Offset = ACh) [Reset = 0h]

SOFTPRES21 is shown in Figure 3-116 and described in Table 3-123.

Return to the Summary Table.

Peripheral Software Reset register
When bits in this register are set, the respective peripheral is in reset. All data is lost and the peripheral registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-116. SOFTPRES21 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						DCC1	DCC0
R-0h						R/W-0h	R/W-0h

Table 3-123. SOFTPRES21 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	DCC1	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn
0	DCC0	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn

3.15.6.22 SOFTPRES25 Register (Offset = B4h) [Reset = 0h]

SOFTPRES25 is shown in Figure 3-117 and described in Table 3-124.

Return to the Summary Table.

Peripheral Software Reset register
When bits in this register are set, the respective peripheral is in reset. All data is lost and the peripheral registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-117. SOFTPRES25 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							HIC_A
R-0h							R/W-0h

Table 3-124. SOFTPRES25 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	HIC_A	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn

3.15.6.23 SOFTPRES26 Register (Offset = B6h) [Reset = 0h]

SOFTPRES26 is shown in Figure 3-118 and described in Table 3-125.

Return to the Summary Table.

Peripheral Software Reset register
When bits in this register are set, the respective peripheral is in reset. All data is lost and the peripheral registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-118. SOFTPRES26 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							AESA
R-0h							R/W-0h

Table 3-125. SOFTPRES26 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	AESA	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn

3.15.6.24 SOFTPRES27 Register (Offset = B8h) [Reset = 0h]

SOFTPRES27 is shown in Figure 3-119 and described in Table 3-126.

Return to the Summary Table.

Peripheral Software Reset register
When bits in this register are set, the respective peripheral is in reset. All data is lost and the peripheral registers are returned to their reset states. Bits must be manually cleared after being set.

Figure 3-119. SOFTPRES27 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EPG1
R-0h							R/W-0h

Table 3-126. SOFTPRES27 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	EPG1	R/W	0h	1: Module is under reset 0: Module reset is determined by the normal device reset structure Reset type: SYSRSn

3.15.6.25 TAP_STATUS Register (Offset = 130h) [Reset = 0h]

TAP_STATUS is shown in Figure 3-120 and described in Table 3-127.

Return to the Summary Table.

Status of JTAG State machine & Debugger Connect

Figure 3-120. TAP_STATUS Register

31	30	29	28	27	26	25	24
DCON	RESERVED						
R-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
TAP_STATE							
R-0h							
7	6	5	4	3	2	1	0
TAP_STATE							
R-0h							

Table 3-127. TAP_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31	DCON	R	0h	DebugConnect indication from IcePick. Reset type: PORESETn
30-16	RESERVED	R	0h	Reserved
15-0	TAP_STATE	R	0h	TAP State Vector. With bits representing, Connect coresponding POTAP* output to the 0:TLR, 1:IDLE, 2:SELECTDR, 3:CAPDR, 4:SHIFTDR, 5:EXIT1DR, 6:PAUSEDR, 7:EXIT2DR, 8:UPDTDR, 9:SLECTIR, 10:CAPIR, 11:SHIFTIR, 12:EXIT1IR, 13:PAUSEIR, 14:EXIT2IR, 15:UPDTIR, Reset type: PORESETn

3.15.6.26 ECAPTYPE Register (Offset = 19Bh) [Reset = 0h]

ECAPTYPE is shown in Figure 3-121 and described in Table 3-128.

Return to the Summary Table.

Based on the configuration enables disables features associated with the ECAP type.

Figure 3-121. ECAPTYPE Register

15	14	13	12	11	10	9	8
LOCK	RESERVED						
R/WSonce-0h	R-0h						
7	6	5	4	3	2	1	0
RESERVED						TYPE	
R-0h						R/W-0h	

Table 3-128. ECAPTYPE Register Field Descriptions

Bit	Field	Type	Reset	Description
15	LOCK	R/WSonce	0h	1: Write to this register is not allowed. 0: Write to this register is allowed. Reset type: SYSRSn
14-2	RESERVED	R	0h	Reserved
1-0	TYPE	R/W	0h	"00,10,11" : 1. No EALLOW protection to ECAP registers. "01" : 1. ECAP registers are EALLOW protected. Reset type: SYSRSn

3.15.6.27 SDFMTYPE Register (Offset = 19Ch) [Reset = 0h]

SDFMTYPE is shown in Figure 3-122 and described in Table 3-129.

Return to the Summary Table.

Based on the configuration enables disables features associated with the SDFM type.

Figure 3-122. SDFMTYPE Register

15	14	13	12	11	10	9	8
LOCK	RESERVED						
R/WSonce-0h	R-0h						
7	6	5	4	3	2	1	0
RESERVED						TYPE	
R-0h						R/W-0h	

Table 3-129. SDFMTYPE Register Field Descriptions

Bit	Field	Type	Reset	Description
15	LOCK	R/WSonce	0h	1: Write to this register is not allowed. 0: Write to this register is allowed. Reset type: SYSRSn
14-2	RESERVED	R	0h	Reserved
1-0	TYPE	R/W	0h	"00,10,11" : 1. Data Ready conditions combined with the fault conditions on the SDFM interrupt line. 2. Data ready interrupts from individual filters are not generated. "01" : 1. Data Ready conditions do not generate the SDFMINT. 2. Each filter generates a separate data ready interrupts. Reset type: SYSRSn

3.15.7 DMA_CLA_SRC_SEL_REGS Registers

Table 3-130 lists the memory-mapped registers for the DMA_CLA_SRC_SEL_REGS registers. All register offset addresses not listed in Table 3-130 should be considered as reserved locations and the register contents should not be modified.

Table 3-130. DMA_CLA_SRC_SEL_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CLA1TASKSRCSELLOCK	CLA1 Task Trigger Source Select Lock Register	EALLOW	Go
4h	DMACHSRCSELLOCK	DMA Channel Triger Source Select Lock Register	EALLOW	Go
6h	CLA1TASKSRCSEL1	CLA1 Task Trigger Source Select Register-1	EALLOW	Go
8h	CLA1TASKSRCSEL2	CLA1 Task Trigger Source Select Register-2	EALLOW	Go
16h	DMACHSRCSEL1	DMA Channel Trigger Source Select Register-1	EALLOW	Go
18h	DMACHSRCSEL2	DMA Channel Trigger Source Select Register-2	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 3-131 shows the codes that are used for access types in this section.

Table 3-131. DMA_CLA_SRC_SEL_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.15.7.1 CLA1TASKSRCSELLOCK Register (Offset = 0h) [Reset = 0h]

CLA1TASKSRCSELLOCK is shown in Figure 3-123 and described in Table 3-132.

Return to the Summary Table.

CLA1 Task Trigger Source Select Lock Register

Figure 3-123. CLA1TASKSRCSELLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						CLA1TASKSRCSEL2	CLA1TASKSRCSEL1
R-0h						R/WSonce-0h	R/WSonce-0h

Table 3-132. CLA1TASKSRCSELLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	CLA1TASKSRCSEL2	R/WSonce	0h	CLA1TASKSRCSEL2 Register Lock bit: 0: Respective register is not locked 1: Respective register is locked. Notes: [1] Any SOnce bit in this register, once set can only be cleared through a SYSRSn. Write of 0 to any bit of this register has no effect [2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed Reset type: SYSRSn
0	CLA1TASKSRCSEL1	R/WSonce	0h	CLA1TASKSRCSEL1 Register Lock bit: 0: Respective register is not locked 1: Respective register is locked. Notes: [1] Any SOnce bit in this register, once set can only be cleared through a SYSRSn. Write of 0 to any bit of this register has no effect [2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed Reset type: SYSRSn

3.15.7.2 DMACHSRCSELLOCK Register (Offset = 4h) [Reset = 0h]

DMACHSRCSELLOCK is shown in Figure 3-124 and described in Table 3-133.

Return to the Summary Table.

DMA Channel Triger Source Select Lock Register

Figure 3-124. DMACHSRCSELLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						DMACHSRCSEL2	DMACHSRCSEL1
R-0h						R/WSonce-0h	R/WSonce-0h

Table 3-133. DMACHSRCSELLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	DMACHSRCSEL2	R/WSonce	0h	DMACHSRCSEL2 Register Lock bit: 0: Respective register is not locked 1: Respective register is locked. Notes: [1] Any SOnce bit in this register, once set can only be cleared through a SYSRSn. Write of 0 to any bit of this register has no effect [2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed Reset type: SYSRSn
0	DMACHSRCSEL1	R/WSonce	0h	DMACHSRCSEL1 Register Lock bit: 0: Respective register is not locked 1: Respective register is locked. Notes: [1] Any SOnce bit in this register, once set can only be cleared through a SYSRSn. Write of 0 to any bit of this register has no effect [2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed Reset type: SYSRSn

3.15.7.3 CLA1TASKSRCSEL1 Register (Offset = 6h) [Reset = 0h]

CLA1TASKSRCSEL1 is shown in Figure 3-125 and described in Table 3-134.

Return to the Summary Table.

CLA1 Task Trigger Source Select Register-1

Figure 3-125. CLA1TASKSRCSEL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TASK4								TASK3								TASK2								TASK1							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 3-134. CLA1TASKSRCSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	TASK4	R/W	0h	Selects the Trigger Source for TASK4 of CLA1 Reset type: SYSRSn
23-16	TASK3	R/W	0h	Selects the Trigger Source for TASK3 of CLA1 Reset type: SYSRSn
15-8	TASK2	R/W	0h	Selects the Trigger Source for TASK2 of CLA1 Reset type: SYSRSn
7-0	TASK1	R/W	0h	Selects the Trigger Source for TASK1 of CLA1 Reset type: SYSRSn

3.15.7.4 CLA1TASKSRCSEL2 Register (Offset = 8h) [Reset = 0h]

CLA1TASKSRCSEL2 is shown in Figure 3-126 and described in Table 3-135.

Return to the Summary Table.

CLA1 Task Trigger Source Select Register-2

Figure 3-126. CLA1TASKSRCSEL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TASK8								TASK7								TASK6								TASK5							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 3-135. CLA1TASKSRCSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	TASK8	R/W	0h	Selects the Trigger Source for TASK8 of CLA1 Reset type: SYSRSn
23-16	TASK7	R/W	0h	Selects the Trigger Source for TASK7 of CLA1 Reset type: SYSRSn
15-8	TASK6	R/W	0h	Selects the Trigger Source for TASK6 of CLA1 Reset type: SYSRSn
7-0	TASK5	R/W	0h	Selects the Trigger Source for TASK5 of CLA1 Reset type: SYSRSn

3.15.7.5 DMACHSRCSEL1 Register (Offset = 16h) [Reset = 0h]

DMACHSRCSEL1 is shown in Figure 3-127 and described in Table 3-136.

Return to the Summary Table.

DMA Channel Trigger Source Select Register-1

Figure 3-127. DMACHSRCSEL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CH4								CH3								CH2								CH1							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 3-136. DMACHSRCSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	CH4	R/W	0h	Selects the Trigger and Sync Source CH4 of DMA Reset type: SYSRSn
23-16	CH3	R/W	0h	Selects the Trigger and Sync Source CH3 of DMA Reset type: SYSRSn
15-8	CH2	R/W	0h	Selects the Trigger and Sync Source CH2 of DMA Reset type: SYSRSn
7-0	CH1	R/W	0h	Selects the Trigger and Sync Source CH1 of DMA Reset type: SYSRSn

3.15.7.6 DMACHSRCSEL2 Register (Offset = 18h) [Reset = 0h]

DMACHSRCSEL2 is shown in Figure 3-128 and described in Table 3-137.

Return to the Summary Table.

DMA Channel Trigger Source Select Register-2

Figure 3-128. DMACHSRCSEL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CH6								CH5							
R-0h																R/W-0h								R/W-0h							

Table 3-137. DMACHSRCSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	CH6	R/W	0h	Selects the Trigger and Sync Source CH6 of DMA Reset type: SYSRSn
7-0	CH5	R/W	0h	Selects the Trigger and Sync Source CH5 of DMA Reset type: SYSRSn

3.15.8 MEM_CFG_REGS Registers

Table 3-138 lists the memory-mapped registers for the MEM_CFG_REGS registers. All register offset addresses not listed in Table 3-138 should be considered as reserved locations and the register contents should not be modified.

Table 3-138. MEM_CFG_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	DxLOCK	Dedicated RAM Config Lock Register	EALLOW	Go
2h	DxCOMMIT	Dedicated RAM Config Lock Commit Register	EALLOW	Go
8h	DxACCPROT0	Dedicated RAM Config Register	EALLOW	Go
10h	DxTEST	Dedicated RAM TEST Register		Go
12h	DxINIT	Dedicated RAM Init Register	EALLOW	Go
14h	DxINITDONE	Dedicated RAM InitDone Status Register		Go
16h	DxRAMTEST_LOCK	Lock register to Dx RAM TEST registers		Go
20h	LSxLOCK	Local Shared RAM Config Lock Register	EALLOW	Go
22h	LSxCOMMIT	Local Shared RAM Config Lock Commit Register	EALLOW	Go
24h	LSxMSEL	Local Shared RAM Master Sel Register	EALLOW	Go
26h	LSxCLAPGM	Local Shared RAM Prog/Exe control Register	EALLOW	Go
28h	LSxACCPROT0	Local Shared RAM Config Register 0	EALLOW	Go
2Ah	LSxACCPROT1	Local Shared RAM Config Register 1	EALLOW	Go
30h	LSxTEST	Local Shared RAM TEST Register		Go
32h	LSxINIT	Local Shared RAM Init Register	EALLOW	Go
34h	LSxINITDONE	Local Shared RAM InitDone Status Register		Go
36h	LSxRAMTEST_LOCK	Lock register to LSx RAM TEST registers		Go
40h	GSxLOCK	Global Shared RAM Config Lock Register	EALLOW	Go
42h	GSxCOMMIT	Global Shared RAM Config Lock Commit Register	EALLOW	Go
48h	GSxACCPROT0	Global Shared RAM Config Register 0	EALLOW	Go
50h	GSxTEST	Global Shared RAM TEST Register		Go
52h	GSxINIT	Global Shared RAM Init Register	EALLOW	Go
54h	GSxINITDONE	Global Shared RAM InitDone Status Register		Go
56h	GSxRAMTEST_LOCK	Lock register to GSx RAM TEST registers		Go
60h	MSGxLOCK	Message RAM Config Lock Register	EALLOW	Go
62h	MSGxCOMMIT	Message RAM Config Lock Commit Register	EALLOW	Go
70h	MSGxTEST	Message RAM TEST Register		Go
72h	MSGxINIT	Message RAM Init Register	EALLOW	Go
74h	MSGxINITDONE	Message RAM InitDone Status Register		Go
76h	MSGxRAMTEST_LOCK	Lock register for MSGx RAM TEST Register		Go
A0h	ROM_LOCK	ROM Config Lock Register		Go
A2h	ROM_TEST	ROM TEST Register		Go
A4h	ROM_FORCE_ERROR	ROM Force Error register		Go

Complex bit access types are encoded to fit into small table cells. Table 3-139 shows the codes that are used for access types in this section.

Table 3-139. MEM_CFG_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read

Table 3-139. MEM_CFG_REGS Access Type Codes (continued)

Access Type	Code	Description
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.15.8.1 DxLOCK Register (Offset = 0h) [Reset = 0h]

DxLOCK is shown in Figure 3-129 and described in Table 3-140.

Return to the Summary Table.

Dedicated RAM Config Lock Register

Figure 3-129. DxLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	RESERVED	LOCK_M1	LOCK_M0
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-140. DxLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	LOCK_M1	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for M1 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed. 1: Write to ACCPROT, TEST, INIT and MSEL fields are blocked. Reset type: SYSRSn
0	LOCK_M0	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for M0 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed. 1: Write to ACCPROT, TEST, INIT and MSEL fields are blocked. Reset type: SYSRSn

3.15.8.2 DxCOMMIT Register (Offset = 2h) [Reset = 0h]

DxCOMMIT is shown in Figure 3-130 and described in Table 3-141.

Return to the Summary Table.

Dedicated RAM Config Lock Commit Register

Figure 3-130. DxCOMMIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	RESERVED	COMMIT_M1	COMMIT_M0
R-0h				R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 3-141. DxCOMMIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	RESERVED	R/WSonce	0h	Reserved
2	RESERVED	R/WSonce	0h	Reserved
1	COMMIT_M1	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for M1 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed based on value of lock field in DxLOCK register. 1: Write to ACCPROT, TEST, INIT and MSEL fields are permanently blocked. Reset type: SYSRSn
0	COMMIT_M0	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for M0 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed based on value of lock field in DxLOCK register. 1: Write to ACCPROT, TEST, INIT and MSEL fields are permanently blocked. Reset type: SYSRSn

3.15.8.3 DxACCPROT0 Register (Offset = 8h) [Reset = 0h]

DxACCPROT0 is shown in Figure 3-131 and described in Table 3-142.

Return to the Summary Table.

Dedicated RAM Config Register

Figure 3-131. DxACCPROT0 Register

31	30	29	28	27	26	25	24
RESERVED						RESERVED	RESERVED
R-0h						R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED						RESERVED	RESERVED
R-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED						CPUWRPROT_M1	FETCHPROT_M1
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED						CPUWRPROT_M0	FETCHPROT_M0
R-0h						R/W-0h	R/W-0h

Table 3-142. DxACCPROT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved
25	RESERVED	R/W	0h	Reserved
24	RESERVED	R/W	0h	Reserved
23-18	RESERVED	R	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15-10	RESERVED	R	0h	Reserved
9	CPUWRPROT_M1	R/W	0h	CPU WR Protection For M1 RAM: 0: CPU Writes are allowed. 1: CPU Writes are block. Reset type: SYSRSn
8	FETCHPROT_M1	R/W	0h	Fetch Protection For M1 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
7-2	RESERVED	R	0h	Reserved
1	CPUWRPROT_M0	R/W	0h	CPU WR Protection For M0 RAM: 0: CPU Writes are allowed. 1: CPU Writes are block. Reset type: SYSRSn
0	FETCHPROT_M0	R/W	0h	Fetch Protection For M0 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn

3.15.8.4 DxTEST Register (Offset = 10h) [Reset = 0h]

DxTEST is shown in Figure 3-132 and described in Table 3-143.

Return to the Summary Table.

Dedicated RAM TEST Register

Figure 3-132. DxTEST Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RESERVED		TEST_M1		TEST_M0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 3-143. DxTEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7-6	RESERVED	R/W	0h	Reserved
5-4	RESERVED	R/W	0h	Reserved
3-2	TEST_M1	R/W	0h	Selects the defferent modes for M1 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ECC bits. 10: Writes are allowed to ECC bits only. No write to data bits. 11: Same as functional mode, but interrupt/NMI is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
1-0	TEST_M0	R/W	0h	Selects the defferent modes for M0 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ECC bits. 10: Writes are allowed to ECC bits only. No write to data bits. 11: Same as functional mode, but interrupt/NMI is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn

3.15.8.5 DxINIT Register (Offset = 12h) [Reset = 0h]

DxINIT is shown in Figure 3-133 and described in Table 3-144.

Return to the Summary Table.

Dedicated RAM Init Register

Figure 3-133. DxINIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	RESERVED	INIT_M1	INIT_M0
R-0h				R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-144. DxINIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	RESERVED	R-0/W1S	0h	Reserved
2	RESERVED	R-0/W1S	0h	Reserved
1	INIT_M1	R-0/W1S	0h	RAM Initialization control for M1 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
0	INIT_M0	R-0/W1S	0h	RAM Initialization control for M0 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn

3.15.8.6 DxINITDONE Register (Offset = 14h) [Reset = 0h]

DxINITDONE is shown in Figure 3-134 and described in Table 3-145.

Return to the Summary Table.

Dedicated RAM InitDone Status Register

Figure 3-134. DxINITDONE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	RESERVED	INITDONE_M1	INITDONE_M0
R-0h				R-0h	R-0h	R-0h	R-0h

Table 3-145. DxINITDONE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	RESERVED	R	0h	Reserved
2	RESERVED	R	0h	Reserved
1	INITDONE_M1	R	0h	RAM Initialization status for M1 RAM: 0: RAM Initialization is not done. 1: RAM Initialization has completed. Reset type: SYSRSn
0	INITDONE_M0	R	0h	RAM Initialization status for M0 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn

3.15.8.7 DxRAMTEST_LOCK Register (Offset = 16h) [Reset = 0h]

DxRAMTEST_LOCK is shown in Figure 3-135 and described in Table 3-146.

Return to the Summary Table.

Lock register to Dx RAM TEST registers

Figure 3-135. DxRAMTEST_LOCK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY															
R-0/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												RESERVED	RESERVED	M1	M0
R-0h												R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-146. DxRAMTEST_LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	A value of 0xa5a5 to this field is simultaneously required for the writes to the rest of the fields of this register to succeed, Reset type: SYSRSn
15-4	RESERVED	R	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	M1	R/W	0h	0: Allows writes to DxTEST.TEST_M1 field. 1: Blocks writes to DxTEST.TEST_M1 field Reset type: SYSRSn
0	M0	R/W	0h	0: Allows writes to DxTEST.TEST_M0 field. 1: Blocks writes to DxTEST.TEST_M0 field Reset type: SYSRSn

3.15.8.8 LSxLOCK Register (Offset = 20h) [Reset = 0h]

LSxLOCK is shown in Figure 3-136 and described in Table 3-147.

Return to the Summary Table.

Local Shared RAM Config Lock Register

Figure 3-136. LSxLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
LOCK_LS7	LOCK_LS6	LOCK_LS5	LOCK_LS4	LOCK_LS3	LOCK_LS2	LOCK_LS1	LOCK_LS0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-147. LSxLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7	LOCK_LS7	R/W	0h	Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS7 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed. 1: Write to ACCPROT, TEST, INIT and MSEL fields are blocked. Reset type: SYSRSn
6	LOCK_LS6	R/W	0h	Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS6 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed. 1: Write to ACCPROT, TEST, INIT and MSEL fields are blocked. Reset type: SYSRSn
5	LOCK_LS5	R/W	0h	Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS5 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed. 1: Write to ACCPROT, TEST, INIT and MSEL fields are blocked. Reset type: SYSRSn
4	LOCK_LS4	R/W	0h	Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS4 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed. 1: Write to ACCPROT, TEST, INIT and MSEL fields are blocked. Reset type: SYSRSn
3	LOCK_LS3	R/W	0h	Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS3 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed. 1: Write to ACCPROT, TEST, INIT and MSEL fields are blocked. Reset type: SYSRSn

Table 3-147. LSxLOCK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	LOCK_LS2	R/W	0h	Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS2 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed. 1: Write to ACCPROT, TEST, INIT and MSEL fields are blocked. Reset type: SYSRSn
1	LOCK_LS1	R/W	0h	Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS1 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed. 1: Write to ACCPROT, TEST, INIT and MSEL fields are blocked. Reset type: SYSRSn
0	LOCK_LS0	R/W	0h	Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS0 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed. 1: Write to ACCPROT, TEST, INIT and MSEL fields are blocked. Reset type: SYSRSn

3.15.8.9 LSxCOMMIT Register (Offset = 22h) [Reset = 0h]

LSxCOMMIT is shown in Figure 3-137 and described in Table 3-148.

Return to the Summary Table.

Local Shared RAM Config Lock Commit Register

Figure 3-137. LSxCOMMIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
COMMIT_LS7	COMMIT_LS6	COMMIT_LS5	COMMIT_LS4	COMMIT_LS3	COMMIT_LS2	COMMIT_LS1	COMMIT_LS0
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 3-148. LSxCOMMIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7	COMMIT_LS7	R/WSonce	0h	Permanently Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS7 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed based on value of lock field in LSxLOCK register. 1: Write to ACCPROT, TEST, INIT and MSEL fields are permanently blocked. Reset type: SYSRSn
6	COMMIT_LS6	R/WSonce	0h	Permanently Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS6 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed based on value of lock field in LSxLOCK register. 1: Write to ACCPROT, TEST, INIT and MSEL fields are permanently blocked. Reset type: SYSRSn
5	COMMIT_LS5	R/WSonce	0h	Permanently Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS5 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed based on value of lock field in LSxLOCK register. 1: Write to ACCPROT, TEST, INIT and MSEL fields are permanently blocked. Reset type: SYSRSn
4	COMMIT_LS4	R/WSonce	0h	Permanently Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS4 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed based on value of lock field in LSxLOCK register. 1: Write to ACCPROT, TEST, INIT and MSEL fields are permanently blocked. Reset type: SYSRSn

Table 3-148. LSxCOMMIT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	COMMIT_LS3	R/WSonce	0h	Permanently Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS3 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed based on value of lock field in LSxLOCK register. 1: Write to ACCPROT, TEST, INIT and MSEL fields are permanently blocked. Reset type: SYSRSn
2	COMMIT_LS2	R/WSonce	0h	Permanently Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS2 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed based on value of lock field in LSxLOCK register. 1: Write to ACCPROT, TEST, INIT and MSEL fields are permanently blocked. Reset type: SYSRSn
1	COMMIT_LS1	R/WSonce	0h	Permanently Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS1 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed based on value of lock field in LSxLOCK register. 1: Write to ACCPROT, TEST, INIT and MSEL fields are permanently blocked. Reset type: SYSRSn
0	COMMIT_LS0	R/WSonce	0h	Permanently Locks the write to access protection, master select, program or data memory select, initialization control and test register fields for LS0 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed based on value of lock field in LSxLOCK register. 1: Write to ACCPROT, TEST, INIT and MSEL fields are permanently blocked. Reset type: SYSRSn

3.15.8.10 LSxMSEL Register (Offset = 24h) [Reset = 0h]

LSxMSEL is shown in Figure 3-138 and described in Table 3-149.

Return to the Summary Table.

Local Shared RAM Master Sel Register

Figure 3-138. LSxMSEL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
MSEL_LS7		MSEL_LS6		MSEL_LS5		MSEL_LS4	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
MSEL_LS3		MSEL_LS2		MSEL_LS1		MSEL_LS0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 3-149. LSxMSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-14	MSEL_LS7	R/W	0h	Master Select for LS7 RAM: 00: Memory is dedicated to CPU. 01: Memory is shared between CPU and CLA1 if CLAPGM_LSx bit in LSxCLAPGM register is programmed as '0'. 10: Reserved. 11: Reserved. Reset type: SYSRSn
13-12	MSEL_LS6	R/W	0h	Master Select for LS6 RAM: 00: Memory is dedicated to CPU. 01: Memory is shared between CPU and CLA1 if CLAPGM_LSx bit in LSxCLAPGM register is programmed as '0'. 10: Reserved. 11: Reserved. Reset type: SYSRSn
11-10	MSEL_LS5	R/W	0h	Master Select for LS5 RAM: 00: Memory is dedicated to CPU. 01: Memory is shared between CPU and CLA1 if CLAPGM_LSx bit in LSxCLAPGM register is programmed as '0'. 10: Reserved. 11: Reserved. Reset type: SYSRSn
9-8	MSEL_LS4	R/W	0h	Master Select for LS4 RAM: 00: Memory is dedicated to CPU. 01: Memory is shared between CPU and CLA1 if CLAPGM_LSx bit in LSxCLAPGM register is programmed as '0'. 10: Reserved. 11: Reserved. Reset type: SYSRSn

Table 3-149. LSxMSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MSEL_LS3	R/W	0h	Master Select for LS3 RAM: 00: Memory is dedicated to CPU. 01: Memory is shared between CPU and CLA1 if CLAPGM_LSx bit in LSxCLAPGM register is programmed as '0'. 10: Reserved. 11: Reserved. Reset type: SYSRSn
5-4	MSEL_LS2	R/W	0h	Master Select for LS2 RAM: 00: Memory is dedicated to CPU. 01: Memory is shared between CPU and CLA1 if CLAPGM_LSx bit in LSxCLAPGM register is programmed as '0'. 10: Reserved. 11: Reserved. Reset type: SYSRSn
3-2	MSEL_LS1	R/W	0h	Master Select for LS1 RAM: 00: Memory is dedicated to CPU. 01: Memory is shared between CPU and CLA1 if CLAPGM_LSx bit in LSxCLAPGM register is programmed as '0'. 10: Reserved. 11: Reserved. Reset type: SYSRSn
1-0	MSEL_LS0	R/W	0h	Master Select for LS0 RAM: 00: Memory is dedicated to CPU. 01: Memory is shared between CPU and CLA1 if CLAPGM_LSx bit in LSxCLAPGM register is programmed as '0'. 10: Reserved. 11: Reserved. Reset type: SYSRSn

3.15.8.11 LSxCLAPGM Register (Offset = 26h) [Reset = 0h]

LSxCLAPGM is shown in Figure 3-139 and described in Table 3-150.

Return to the Summary Table.

Local Shared RAM Prog/Exe control Register

Figure 3-139. LSxCLAPGM Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
CLAPGM_LS7	CLAPGM_LS6	CLAPGM_LS5	CLAPGM_LS4	CLAPGM_LS3	CLAPGM_LS2	CLAPGM_LS1	CLAPGM_LS0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-150. LSxCLAPGM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7	CLAPGM_LS7	R/W	0h	Selects LS7 RAM as program vs data memory for CLA: 0: CLA Data memory. 1: CLA Program memory. Reset type: SYSRSn
6	CLAPGM_LS6	R/W	0h	Selects LS6 RAM as program vs data memory for CLA: 0: CLA Data memory. 1: CLA Program memory. Reset type: SYSRSn
5	CLAPGM_LS5	R/W	0h	Selects LS5 RAM as program vs data memory for CLA: 0: CLA Data memory. 1: CLA Program memory. Reset type: SYSRSn
4	CLAPGM_LS4	R/W	0h	Selects LS4 RAM as program vs data memory for CLA: 0: CLA Data memory. 1: CLA Program memory. Reset type: SYSRSn
3	CLAPGM_LS3	R/W	0h	Selects LS3 RAM as program vs data memory for CLA: 0: CLA Data memory. 1: CLA Program memory. Reset type: SYSRSn
2	CLAPGM_LS2	R/W	0h	Selects LS2 RAM as program vs data memory for CLA: 0: CLA Data memory. 1: CLA Program memory. Reset type: SYSRSn
1	CLAPGM_LS1	R/W	0h	Selects LS1 RAM as program vs data memory for CLA: 0: CLA Data memory. 1: CLA Program memory. Reset type: SYSRSn

Table 3-150. LSxCLAPGM Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	CLAPGM_LS0	R/W	0h	Selects LS0 RAM as program vs data memory for CLA: 0: CLA Data memory. 1: CLA Program memory. Reset type: SYSRSn

3.15.8.12 LSxACCPROT0 Register (Offset = 28h) [Reset = 0h]

LSxACCPROT0 is shown in Figure 3-140 and described in Table 3-151.

Return to the Summary Table.

Local Shared RAM Config Register 0

Figure 3-140. LSxACCPROT0 Register

31	30	29	28	27	26	25	24
RESERVED						CPUWRPROT_LS3	FETCHPROT_LS3
R-0h						R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED						CPUWRPROT_LS2	FETCHPROT_LS2
R-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED						CPUWRPROT_LS1	FETCHPROT_LS1
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED						CPUWRPROT_LS0	FETCHPROT_LS0
R-0h						R/W-0h	R/W-0h

Table 3-151. LSxACCPROT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved
25	CPUWRPROT_LS3	R/W	0h	CPU WR Protection For LS3 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
24	FETCHPROT_LS3	R/W	0h	Fetch Protection For LS3 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
23-18	RESERVED	R	0h	Reserved
17	CPUWRPROT_LS2	R/W	0h	CPU WR Protection For LS2 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
16	FETCHPROT_LS2	R/W	0h	Fetch Protection For LS2 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
15-10	RESERVED	R	0h	Reserved
9	CPUWRPROT_LS1	R/W	0h	CPU WR Protection For LS1 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
8	FETCHPROT_LS1	R/W	0h	Fetch Protection For LS1 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
7-2	RESERVED	R	0h	Reserved

Table 3-151. LSxACCPROT0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	CPUWRPROT_LS0	R/W	0h	CPU WR Protection For LS0 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
0	FETCHPROT_LS0	R/W	0h	Fetch Protection For LS0 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn

3.15.8.13 LSxACCPROT1 Register (Offset = 2Ah) [Reset = 0h]

LSxACCPROT1 is shown in Figure 3-141 and described in Table 3-152.

Return to the Summary Table.

Local Shared RAM Config Register 1

Figure 3-141. LSxACCPROT1 Register

31	30	29	28	27	26	25	24
RESERVED						CPUWRPROT_LS7	FETCHPROT_LS7
R-0h						R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED						CPUWRPROT_LS6	FETCHPROT_LS6
R-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED						CPUWRPROT_LS5	FETCHPROT_LS5
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED						CPUWRPROT_LS4	FETCHPROT_LS4
R-0h						R/W-0h	R/W-0h

Table 3-152. LSxACCPROT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved
25	CPUWRPROT_LS7	R/W	0h	CPU WR Protection For LS7 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
24	FETCHPROT_LS7	R/W	0h	Fetch Protection For LS7 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
23-18	RESERVED	R	0h	Reserved
17	CPUWRPROT_LS6	R/W	0h	CPU WR Protection For LS6 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
16	FETCHPROT_LS6	R/W	0h	Fetch Protection For LS6 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
15-10	RESERVED	R	0h	Reserved
9	CPUWRPROT_LS5	R/W	0h	CPU WR Protection For LS5 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
8	FETCHPROT_LS5	R/W	0h	Fetch Protection For LS5 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
7-2	RESERVED	R	0h	Reserved

Table 3-152. LSxACCPROT1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	CPUWRPROT_LS4	R/W	0h	CPU WR Protection For LS4 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
0	FETCHPROT_LS4	R/W	0h	Fetch Protection For LS4 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn

3.15.8.14 LSxTEST Register (Offset = 30h) [Reset = 0h]

LSxTEST is shown in Figure 3-142 and described in Table 3-153.

Return to the Summary Table.

Local Shared RAM TEST Register

Figure 3-142. LSxTEST Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
TEST_LS7		TEST_LS6		TEST_LS5		TEST_LS4	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
TEST_LS3		TEST_LS2		TEST_LS1		TEST_LS0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 3-153. LSxTEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-14	TEST_LS7	R/W	0h	Selects the defferent modes for LS7 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ECC bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/NMI is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
13-12	TEST_LS6	R/W	0h	Selects the defferent modes for LS6 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ECC bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/NMI is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
11-10	TEST_LS5	R/W	0h	Selects the defferent modes for LS5 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ECC bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/NMI is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn

Table 3-153. LSxTEST Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9-8	TEST_LS4	R/W	0h	Selects the defferent modes for LS4 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ECC bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/NMI is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
7-6	TEST_LS3	R/W	0h	Selects the defferent modes for LS3 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ECC bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/NMI is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
5-4	TEST_LS2	R/W	0h	Selects the defferent modes for LS2 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ECC bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/NMI is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
3-2	TEST_LS1	R/W	0h	Selects the defferent modes for LS1 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ECC bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/NMI is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
1-0	TEST_LS0	R/W	0h	Selects the defferent modes for LS0 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ECC bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/NMI is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn

3.15.8.15 LSxINIT Register (Offset = 32h) [Reset = 0h]

LSxINIT is shown in Figure 3-143 and described in Table 3-154.

Return to the Summary Table.

Local Shared RAM Init Register

Figure 3-143. LSxINIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
INIT_LS7	INIT_LS6	INIT_LS5	INIT_LS4	INIT_LS3	INIT_LS2	INIT_LS1	INIT_LS0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-154. LSxINIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7	INIT_LS7	R-0/W1S	0h	RAM Initialization control for LS7 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
6	INIT_LS6	R-0/W1S	0h	RAM Initialization control for LS6 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
5	INIT_LS5	R-0/W1S	0h	RAM Initialization control for LS5 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
4	INIT_LS4	R-0/W1S	0h	RAM Initialization control for LS4 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
3	INIT_LS3	R-0/W1S	0h	RAM Initialization control for LS3 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
2	INIT_LS2	R-0/W1S	0h	RAM Initialization control for LS2 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
1	INIT_LS1	R-0/W1S	0h	RAM Initialization control for LS1 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn

Table 3-154. LSxINIT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INIT_LS0	R-0/W1S	0h	RAM Initialization control for LS0 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn

3.15.8.16 LSxINITDONE Register (Offset = 34h) [Reset = 0h]

LSxINITDONE is shown in Figure 3-144 and described in Table 3-155.

Return to the Summary Table.

Local Shared RAM InitDone Status Register

Figure 3-144. LSxINITDONE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
INITDONE_LS7	INITDONE_LS6	INITDONE_LS5	INITDONE_LS4	INITDONE_LS3	INITDONE_LS2	INITDONE_LS1	INITDONE_LS0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 3-155. LSxINITDONE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7	INITDONE_LS7	R	0h	RAM Initialization status for LS7 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
6	INITDONE_LS6	R	0h	RAM Initialization status for LS6 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
5	INITDONE_LS5	R	0h	RAM Initialization status for LS5 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
4	INITDONE_LS4	R	0h	RAM Initialization status for LS4 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
3	INITDONE_LS3	R	0h	RAM Initialization status for LS3 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
2	INITDONE_LS2	R	0h	RAM Initialization status for LS2 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
1	INITDONE_LS1	R	0h	RAM Initialization status for LS1 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn

Table 3-155. LSxINITDONE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INITDONE_LS0	R	0h	RAM Initialization status for LS0 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn

3.15.8.17 LSxRAMTEST_LOCK Register (Offset = 36h) [Reset = 0h]

LSxRAMTEST_LOCK is shown in Figure 3-145 and described in Table 3-156.

Return to the Summary Table.

Lock register to LSx RAM TEST registers

Figure 3-145. LSxRAMTEST_LOCK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY															
R-0/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								LS7	LS6	LS5	LS4	LS3	LS2	LS1	LS0
R-0h								R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-156. LSxRAMTEST_LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	A value of 0xa5a5 to this field is simultaneously required for the writes to the rest of the fields of this register to succeed, Reset type: SYSRSn
15-8	RESERVED	R	0h	Reserved
7	LS7	R/W	0h	0: Allows writes to LSxTEST.TEST_LS7 field. 1: Blocks writes to LSxTEST.TEST_LS7 field. Reset type: SYSRSn
6	LS6	R/W	0h	0: Allows writes to LSxTEST.TEST_LS6 field. 1: Blocks writes to LSxTEST.TEST_LS6 field. Reset type: SYSRSn
5	LS5	R/W	0h	0: Allows writes to LSxTEST.TEST_LS5 field. 1: Blocks writes to LSxTEST.TEST_LS5 field. Reset type: SYSRSn
4	LS4	R/W	0h	0: Allows writes to LSxTEST.TEST_LS4 field. 1: Blocks writes to LSxTEST.TEST_LS4 field. Reset type: SYSRSn
3	LS3	R/W	0h	0: Allows writes to LSxTEST.TEST_LS3 field. 1: Blocks writes to LSxTEST.TEST_LS3 field. Reset type: SYSRSn
2	LS2	R/W	0h	0: Allows writes to LSxTEST.TEST_LS2 field. 1: Blocks writes to LSxTEST.TEST_LS2 field. Reset type: SYSRSn
1	LS1	R/W	0h	0: Allows writes to LSxTEST.TEST_LS1 field. 1: Blocks writes to LSxTEST.TEST_LS1 field. Reset type: SYSRSn
0	LS0	R/W	0h	0: Allows writes to LSxTEST.TEST_LS0 field. 1: Blocks writes to LSxTEST.TEST_LS0 field. Reset type: SYSRSn

3.15.8.18 GSxLOCK Register (Offset = 40h) [Reset = 0h]

GSxLOCK is shown in Figure 3-146 and described in Table 3-157.

Return to the Summary Table.

Global Shared RAM Config Lock Register

Figure 3-146. GSxLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	LOCK_GS3	LOCK_GS2	LOCK_GS1	LOCK_GS0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-157. GSxLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	RESERVED	R/W	0h	Reserved
14	RESERVED	R/W	0h	Reserved
13	RESERVED	R/W	0h	Reserved
12	RESERVED	R/W	0h	Reserved
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	RESERVED	R/W	0h	Reserved
8	RESERVED	R/W	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	RESERVED	R/W	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	LOCK_GS3	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for GS3 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed. 1: Write to ACCPROT, TEST, INIT and MSEL fields are blocked. Reset type: SYSRSn
2	LOCK_GS2	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for GS2 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed. 1: Write to ACCPROT, TEST, INIT and MSEL fields are blocked. Reset type: SYSRSn
1	LOCK_GS1	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for GS1 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed. 1: Write to ACCPROT, TEST, INIT and MSEL fields are blocked. Reset type: SYSRSn

Table 3-157. GSxLOCK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	LOCK_GS0	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for GS0 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed. 1: Write to ACCPROT, TEST, INIT and MSEL fields are blocked. Reset type: SYSRSn

3.15.8.19 GSxCOMMIT Register (Offset = 42h) [Reset = 0h]

GSxCOMMIT is shown in Figure 3-147 and described in Table 3-158.

Return to the Summary Table.

Global Shared RAM Config Lock Commit Register

Figure 3-147. GSxCOMMIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	COMMIT_GS3	COMMIT_GS2	COMMIT_GS1	COMMIT_GS0
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 3-158. GSxCOMMIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	RESERVED	R/WSonce	0h	Reserved
14	RESERVED	R/WSonce	0h	Reserved
13	RESERVED	R/WSonce	0h	Reserved
12	RESERVED	R/WSonce	0h	Reserved
11	RESERVED	R/WSonce	0h	Reserved
10	RESERVED	R/WSonce	0h	Reserved
9	RESERVED	R/WSonce	0h	Reserved
8	RESERVED	R/WSonce	0h	Reserved
7	RESERVED	R/WSonce	0h	Reserved
6	RESERVED	R/WSonce	0h	Reserved
5	RESERVED	R/WSonce	0h	Reserved
4	RESERVED	R/WSonce	0h	Reserved
3	COMMIT_GS3	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for GS3 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed based on value of lock field in GSxLOCK register. 1: Write to ACCPROT, TEST, INIT and MSEL fields are permanently blocked. Reset type: SYSRSn
2	COMMIT_GS2	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for GS2 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed based on value of lock field in GSxLOCK register. 1: Write to ACCPROT, TEST, INIT and MSEL fields are permanently blocked. Reset type: SYSRSn

Table 3-158. GSxCOMMIT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	COMMIT_GS1	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for GS1 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed based on value of lock field in GSxLOCK register. 1: Write to ACCPROT, TEST, INIT and MSEL fields are permanently blocked. Reset type: SYSRSn
0	COMMIT_GS0	R/WSonce	0h	Permanently Locks the write to access protection, master select, initialization control and test register fields for GS0 RAM: 0: Write to ACCPROT, TEST, INIT and MSEL fields are allowed based on value of lock field in GSxLOCK register. 1: Write to ACCPROT, TEST, INIT and MSEL fields are permanently blocked. Reset type: SYSRSn

3.15.8.20 GSxACCPROT0 Register (Offset = 48h) [Reset = 0h]

GSxACCPROT0 is shown in Figure 3-148 and described in Table 3-159.

Return to the Summary Table.

Global Shared RAM Config Register 0

Figure 3-148. GSxACCPROT0 Register

31	30	29	28	27	26	25	24
RESERVED				HICWRPROT_GS3	DMAWRPROT_GS3	CPUWRPROT_GS3	FETCHPROT_GS3
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED				HICWRPROT_GS2	DMAWRPROT_GS2	CPUWRPROT_GS2	FETCHPROT_GS2
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED				HICWRPROT_GS1	DMAWRPROT_GS1	CPUWRPROT_GS1	FETCHPROT_GS1
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED				HICWRPROT_GS0	DMAWRPROT_GS0	CPUWRPROT_GS0	FETCHPROT_GS0
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-159. GSxACCPROT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27	HICWRPROT_GS3	R/W	0h	HICA WR Protection For GS0 RAM: 0: HICA Writes are allowed. 1: HICA Writes are blocked. Reset type: SYSRSn
26	DMAWRPROT_GS3	R/W	0h	DMA WR Protection For GS3 RAM: 0: DMA Writes are allowed. 1: DMA Writes are blocked. Reset type: SYSRSn
25	CPUWRPROT_GS3	R/W	0h	CPU WR Protection For GS3 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
24	FETCHPROT_GS3	R/W	0h	Fetch Protection For GS3 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
23-20	RESERVED	R	0h	Reserved
19	HICWRPROT_GS2	R/W	0h	HICA WR Protection For GS0 RAM: 0: HICA Writes are allowed. 1: HICA Writes are blocked. Reset type: SYSRSn
18	DMAWRPROT_GS2	R/W	0h	DMA WR Protection For GS2 RAM: 0: DMA Writes are allowed. 1: DMA Writes are blocked. Reset type: SYSRSn

Table 3-159. GSxACCPROT0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	CPUWRPROT_GS2	R/W	0h	CPU WR Protection For GS2 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
16	FETCHPROT_GS2	R/W	0h	Fetch Protection For GS2 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
15-12	RESERVED	R	0h	Reserved
11	HICWRPROT_GS1	R/W	0h	HICA WR Protection For GS0 RAM: 0: HICA Writes are allowed. 1: HICA Writes are blocked. Reset type: SYSRSn
10	DMAWRPROT_GS1	R/W	0h	DMA WR Protection For GS1 RAM: 0: DMA Writes are allowed. 1: DMA Writes are blocked. Reset type: SYSRSn
9	CPUWRPROT_GS1	R/W	0h	CPU WR Protection For GS1 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
8	FETCHPROT_GS1	R/W	0h	Fetch Protection For GS1 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn
7-4	RESERVED	R	0h	Reserved
3	HICWRPROT_GS0	R/W	0h	HICA WR Protection For GS0 RAM: 0: HICA Writes are allowed. 1: HICA Writes are blocked. Reset type: SYSRSn
2	DMAWRPROT_GS0	R/W	0h	DMA WR Protection For GS0 RAM: 0: DMA Writes are allowed. 1: DMA Writes are blocked. Reset type: SYSRSn
1	CPUWRPROT_GS0	R/W	0h	CPU WR Protection For GS0 RAM: 0: CPU Writes are allowed. 1: CPU Writes are blocked. Reset type: SYSRSn
0	FETCHPROT_GS0	R/W	0h	Fetch Protection For GS0 RAM: 0: CPU Fetch are allowed. 1: CPU Fetch are blocked. Reset type: SYSRSn

3.15.8.21 GSxTEST Register (Offset = 50h) [Reset = 0h]

GSxTEST is shown in Figure 3-149 and described in Table 3-160.

Return to the Summary Table.

Global Shared RAM TEST Register

Figure 3-149. GSxTEST Register

31	30	29	28	27	26	25	24
RESERVED		RESERVED		RESERVED		RESERVED	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
23	22	21	20	19	18	17	16
RESERVED		RESERVED		RESERVED		RESERVED	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
RESERVED		RESERVED		RESERVED		RESERVED	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
TEST_GS3		TEST_GS2		TEST_GS1		TEST_GS0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 3-160. GSxTEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R/W	0h	Reserved
29-28	RESERVED	R/W	0h	Reserved
27-26	RESERVED	R/W	0h	Reserved
25-24	RESERVED	R/W	0h	Reserved
23-22	RESERVED	R/W	0h	Reserved
21-20	RESERVED	R/W	0h	Reserved
19-18	RESERVED	R/W	0h	Reserved
17-16	RESERVED	R/W	0h	Reserved
15-14	RESERVED	R/W	0h	Reserved
13-12	RESERVED	R/W	0h	Reserved
11-10	RESERVED	R/W	0h	Reserved
9-8	RESERVED	R/W	0h	Reserved
7-6	TEST_GS3	R/W	0h	Selects the defferent modes for GS3 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ECC bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/NMI is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn

Table 3-160. GSxTEST Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-4	TEST_GS2	R/W	0h	Selects the defferent modes for GS2 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ECC bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/NMI is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
3-2	TEST_GS1	R/W	0h	Selects the defferent modes for GS1 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ECC bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/NMI is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
1-0	TEST_GS0	R/W	0h	Selects the defferent modes for GS0 RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ECC bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/NMI is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn

3.15.8.22 GSxINIT Register (Offset = 52h) [Reset = 0h]

GSxINIT is shown in Figure 3-150 and described in Table 3-161.

Return to the Summary Table.

Global Shared RAM Init Register

Figure 3-150. GSxINIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	INIT_GS3	INIT_GS2	INIT_GS1	INIT_GS0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-161. GSxINIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	RESERVED	R-0/W1S	0h	Reserved
14	RESERVED	R-0/W1S	0h	Reserved
13	RESERVED	R-0/W1S	0h	Reserved
12	RESERVED	R-0/W1S	0h	Reserved
11	RESERVED	R-0/W1S	0h	Reserved
10	RESERVED	R-0/W1S	0h	Reserved
9	RESERVED	R-0/W1S	0h	Reserved
8	RESERVED	R-0/W1S	0h	Reserved
7	RESERVED	R-0/W1S	0h	Reserved
6	RESERVED	R-0/W1S	0h	Reserved
5	RESERVED	R-0/W1S	0h	Reserved
4	RESERVED	R-0/W1S	0h	Reserved
3	INIT_GS3	R-0/W1S	0h	RAM Initialization control for GS3 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
2	INIT_GS2	R-0/W1S	0h	RAM Initialization control for GS2 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
1	INIT_GS1	R-0/W1S	0h	RAM Initialization control for GS1 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn

Table 3-161. GSxINIT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INIT_GS0	R-0/W1S	0h	RAM Initialization control for GS0 RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn

3.15.8.23 GSxINITDONE Register (Offset = 54h) [Reset = 0h]

GSxINITDONE is shown in Figure 3-151 and described in Table 3-162.

Return to the Summary Table.

Global Shared RAM InitDone Status Register

Figure 3-151. GSxINITDONE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	INITDONE_GS3	INITDONE_GS2	INITDONE_GS1	INITDONE_GS0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 3-162. GSxINITDONE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	RESERVED	R	0h	Reserved
14	RESERVED	R	0h	Reserved
13	RESERVED	R	0h	Reserved
12	RESERVED	R	0h	Reserved
11	RESERVED	R	0h	Reserved
10	RESERVED	R	0h	Reserved
9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	RESERVED	R	0h	Reserved
6	RESERVED	R	0h	Reserved
5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3	INITDONE_GS3	R	0h	RAM Initialization status for GS3 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
2	INITDONE_GS2	R	0h	RAM Initialization status for GS2 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
1	INITDONE_GS1	R	0h	RAM Initialization status for GS1 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn

Table 3-162. GSxINITDONE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INITDONE_GS0	R	0h	RAM Initialization status for GS0 RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn

3.15.8.24 GSxRAMTEST_LOCK Register (Offset = 56h) [Reset = 0h]

GSxRAMTEST_LOCK is shown in Figure 3-152 and described in Table 3-163.

Return to the Summary Table.

Lock register to GSx RAM TEST registers

Figure 3-152. GSxRAMTEST_LOCK Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	GS3	GS2	GS1	GS0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-163. GSxRAMTEST_LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	A value of 0xa5a5 to this field is simultaneously required for the writes to the rest of the fields of this register to succeed, Reset type: SYSRSn
15	RESERVED	R/W	0h	Reserved
14	RESERVED	R/W	0h	Reserved
13	RESERVED	R/W	0h	Reserved
12	RESERVED	R/W	0h	Reserved
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	RESERVED	R/W	0h	Reserved
8	RESERVED	R/W	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	RESERVED	R/W	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	GS3	R/W	0h	0: Allows writes to GSxTEST.TEST_GS3 field. 1: Blocks writes to GSxTEST.TEST_GS3 field. Reset type: SYSRSn
2	GS2	R/W	0h	0: Allows writes to GSxTEST.TEST_GS2 field. 1: Blocks writes to GSxTEST.TEST_GS2 field. Reset type: SYSRSn
1	GS1	R/W	0h	0: Allows writes to GSxTEST.TEST_GS1 field. 1: Blocks writes to GSxTEST.TEST_GS1 field. Reset type: SYSRSn
0	GS0	R/W	0h	0: Allows writes to GSxTEST.TEST_GS0 field. 1: Blocks writes to GSxTEST.TEST_GS0 field. Reset type: SYSRSn

3.15.8.25 MSGxLOCK Register (Offset = 60h) [Reset = 0h]

MSGxLOCK is shown in Figure 3-153 and described in Table 3-164.

Return to the Summary Table.

Message RAM Config Lock Register

Figure 3-153. MSGxLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				RESERVED	RESERVED	RESERVED	RESERVED
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	LOCK_DMATOCLA1	LOCK_CLA1TODMA	RESERVED	RESERVED	LOCK_CLA1TOCPU	LOCK_CPUTOCLA1	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-164. MSGxLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	RESERVED	R/W	0h	Reserved
8	RESERVED	R/W	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	LOCK_DMATOCLA1	R/W	0h	Locks the write to access protection, master select, initialization control and test for DMATOCLA1 RAM: 0: Write to TEST, INIT fields are allowed. 1: Write to TEST, INIT fields are blocked. Reset type: SYSRSn
5	LOCK_CLA1TODMA	R/W	0h	Locks the write to access protection, master select, initialization control and test for CLA1TODMA RAM: 0: Write to TEST, INIT fields are allowed. 1: Write to TEST, INIT fields are blocked. Reset type: SYSRSn
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	LOCK_CLA1TOCPU	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for CLA1TOCPU RAM: 0: Write to TEST, INIT fields are allowed. 1: Write to TEST, INIT fields are blocked. Reset type: SYSRSn
1	LOCK_CPUTOCLA1	R/W	0h	Locks the write to access protection, master select, initialization control and test register fields for CPUTOCLA1 RAM: 0: Write to TEST, INIT fields are allowed. 1: Write to TEST, INIT fields are blocked. Reset type: SYSRSn
0	RESERVED	R/W	0h	Reserved

3.15.8.26 MSGxCOMMIT Register (Offset = 62h) [Reset = 0h]

MSGxCOMMIT is shown in Figure 3-154 and described in Table 3-165.

Return to the Summary Table.

Message RAM Config Lock Commit Register

Figure 3-154. MSGxCOMMIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				RESERVED	RESERVED	RESERVED	RESERVED
R-0h				R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h
7	6	5	4	3	2	1	0
RESERVED	COMMIT_DMATOCLA1	COMMIT_CLA1TODMA	RESERVED	RESERVED	COMMIT_CLA1TOCPU	COMMIT_CPUTOCLA1	RESERVED
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 3-165. MSGxCOMMIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved
11	RESERVED	R/WSonce	0h	Reserved
10	RESERVED	R/WSonce	0h	Reserved
9	RESERVED	R/WSonce	0h	Reserved
8	RESERVED	R/WSonce	0h	Reserved
7	RESERVED	R/WSonce	0h	Reserved
6	COMMIT_DMATOCLA1	R/WSonce	0h	Locks the write to access protection, master select, initialization control and test register fields for DMATOCLA1 RAM: 0: Write to TEST, INIT fields are allowed based on value of corresponding lock field in MSGxLOCK register. 1: Write to TEST, INIT fields are permanently blocked. Reset type: SYSRSn
5	COMMIT_CLA1TODMA	R/WSonce	0h	Locks the write to access protection, master select, initialization control and test register fields for CLA1TODMA RAM: 0: Write to TEST, INIT fields are allowed based on value of corresponding lock field in MSGxLOCK register. 1: Write to TEST, INIT fields are permanently blocked. Reset type: SYSRSn
4	RESERVED	R/WSonce	0h	Reserved
3	RESERVED	R/WSonce	0h	Reserved
2	COMMIT_CLA1TOCPU	R/WSonce	0h	Locks the write to access protection, master select, initialization control and test register fields for CLA1TOCPU RAM: 0: Write to TEST, INIT fields are allowed based on value of corresponding lock field in MSGxLOCK register. 1: Write to TEST, INIT fields are permanently blocked. Reset type: SYSRSn

Table 3-165. MSGxCOMMIT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	COMMIT_CPUTOCLA1	R/WSonce	0h	Locks the write to access protection, master select, initialization control and test register fields for CPUTOCLA1 RAM: 0: Write to TEST, INIT fields are allowed based on value of corresponding lock field in MSGxLOCK register. 1: Write to TEST, INIT fields are permanently blocked. Reset type: SYSRSn
0	RESERVED	R/WSonce	0h	Reserved

3.15.8.27 MSGxTEST Register (Offset = 70h) [Reset = 0h]

MSGxTEST is shown in Figure 3-155 and described in Table 3-166.

Return to the Summary Table.

Message RAM TEST Register

Figure 3-155. MSGxTEST Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED		RESERVED		RESERVED		RESERVED	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
RESERVED		TEST_DMATOCLA1		TEST_CLA1TODMA		RESERVED	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		TEST_CLA1TOCPU		TEST_CPUTOCLA1		RESERVED	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 3-166. MSGxTEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-22	RESERVED	R/W	0h	Reserved
21-20	RESERVED	R/W	0h	Reserved
19-18	RESERVED	R/W	0h	Reserved
17-16	RESERVED	R/W	0h	Reserved
15-14	RESERVED	R/W	0h	Reserved
13-12	TEST_DMATOCLA1	R/W	0h	Selects the defferent modes for DMATOCLA1 MSG RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ECC bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/NMI is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
11-10	TEST_CLA1TODMA	R/W	0h	Selects the defferent modes for CLA1TODMA MSG RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ECC bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/NMI is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
9-8	RESERVED	R/W	0h	Reserved
7-6	RESERVED	R/W	0h	Reserved

Table 3-166. MSGxTEST Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-4	TEST_CLA1TOCPU	R/W	0h	Selects the defferent modes for CLA1TOCPU MSG RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ECC bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/NMI is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
3-2	TEST_CPUTOCLA1	R/W	0h	Selects the defferent modes for CPUTOCLA1 MSG RAM: 00: Functional Mode. 01: Writes are allowed to data bits only. No write to ECC bits. 10: Writes are allowed to parity bits only. No write to data bits. 11: Same as functional mode, but interrupt/NMI is not generated on error. Note: Any non zero value would enable CPU writes over-riding write access protection if any and will not generate a access protection violation. Reset type: SYSRSn
1-0	RESERVED	R/W	0h	Reserved

3.15.8.28 MSGxINIT Register (Offset = 72h) [Reset = 0h]

MSGxINIT is shown in Figure 3-156 and described in Table 3-167.

Return to the Summary Table.

Message RAM Init Register

Figure 3-156. MSGxINIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				RESERVED	RESERVED	RESERVED	RESERVED
R-0h				R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
RESERVED	INIT_DMATOCLA1	INIT_CLA1TODMA	RESERVED	RESERVED	INIT_CLA1TOCPU	INIT_CPUTOCLA1	RESERVED
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-167. MSGxINIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved
11	RESERVED	R-0/W1S	0h	Reserved
10	RESERVED	R-0/W1S	0h	Reserved
9	RESERVED	R-0/W1S	0h	Reserved
8	RESERVED	R-0/W1S	0h	Reserved
7	RESERVED	R-0/W1S	0h	Reserved
6	INIT_DMATOCLA1	R-0/W1S	0h	RAM Initialization control for DMATOCLA1 MSG RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
5	INIT_CLA1TODMA	R-0/W1S	0h	RAM Initialization control for CLA1TODMA MSG RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
4	RESERVED	R-0/W1S	0h	Reserved
3	RESERVED	R-0/W1S	0h	Reserved
2	INIT_CLA1TOCPU	R-0/W1S	0h	RAM Initialization control for CLA1TOCPU MSG RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
1	INIT_CPUTOCLA1	R-0/W1S	0h	RAM Initialization control for CPUTOCLA1 MSG RAM: 0: None. 1: Start RAM Initialization. Reset type: SYSRSn
0	RESERVED	R-0/W1S	0h	Reserved

3.15.8.29 MSGxINITDONE Register (Offset = 74h) [Reset = 0h]

MSGxINITDONE is shown in Figure 3-157 and described in Table 3-168.

Return to the Summary Table.

Message RAM InitDone Status Register

Figure 3-157. MSGxINITDONE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				RESERVED	RESERVED	RESERVED	RESERVED
R-0h				R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	INITDONE_DMATOCLA1	INITDONE_CLA1TODMA	RESERVED	RESERVED	INITDONE_CLA1TOCPU	INITDONE_CPUTOCLA1	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 3-168. MSGxINITDONE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved
11	RESERVED	R	0h	Reserved
10	RESERVED	R	0h	Reserved
9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	RESERVED	R	0h	Reserved
6	INITDONE_DMATOCLA1	R	0h	RAM Initialization status for DMATOCLA1 MSG RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
5	INITDONE_CLA1TODMA	R	0h	RAM Initialization status for CLA1TODMA MSG RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
4	RESERVED	R	0h	Reserved
3	RESERVED	R	0h	Reserved
2	INITDONE_CLA1TOCPU	R	0h	RAM Initialization status for CLA1TOCPU MSG RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
1	INITDONE_CPUTOCLA1	R	0h	RAM Initialization status for CPUTOCLA1 MSG RAM: 0: RAM Initialization is not done. 1: RAM Initialization is done. Reset type: SYSRSn
0	RESERVED	R	0h	Reserved

3.15.8.30 MSGxRAMTEST_LOCK Register (Offset = 76h) [Reset = 0h]

MSGxRAMTEST_LOCK is shown in Figure 3-158 and described in Table 3-169.

Return to the Summary Table.

Lock register for MSGx RAM TEST Register

Figure 3-158. MSGxRAMTEST_LOCK Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED				RESERVED	RESERVED	RESERVED	RESERVED
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	DMATOCLA1	CLA1TODMA	RESERVED	RESERVED	CLA1TOCPU	CPUTOCLA1	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-169. MSGxRAMTEST_LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	A value of 0xa5a5 to this field is simultaneously required for the writes to the rest of the fields of this register to succeed, Reset type: SYSRSn
15-12	RESERVED	R	0h	Reserved
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	RESERVED	R/W	0h	Reserved
8	RESERVED	R/W	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	DMATOCLA1	R/W	0h	0: Allows writes to MSGxTEST.TEST_DMATOCLA1 field 1: Blocks writes to MSGxTEST.TEST_DMATOCLA1 field Reset type: SYSRSn
5	CLA1TODMA	R/W	0h	0: Allows writes to MSGxTEST.TEST_CLA1TODMA field 1: Blocks writes to MSGxTEST.TEST_CLA1TODMA field Reset type: SYSRSn
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	CLA1TOCPU	R/W	0h	0: Allows writes to MSGxTEST.TEST_CLA1TOCPU field 1: Blocks writes to MSGxTEST.TEST_CLA1TOCPU field Reset type: SYSRSn
1	CPUTOCLA1	R/W	0h	0: Allows writes to MSGxTEST.TEST_CPUTOCLA1 field 1: Blocks writes to MSGxTEST.TEST_CPUTOCLA1 field Reset type: SYSRSn
0	RESERVED	R/W	0h	Reserved

3.15.8.31 ROM_LOCK Register (Offset = A0h) [Reset = 0h]

ROM_LOCK is shown in Figure 3-159 and described in Table 3-170.

Return to the Summary Table.

ROM Config Lock Register

Figure 3-159. ROM_LOCK Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				LOCK_CLAPROGROM	LOCK_CLADATAROM	LOCK_SECUREROM	LOCK_BOOTROM
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-170. ROM_LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	A value of 0xa5a5 to this field is simultaneously required for the writes to the rest of the fields of this register to succeed, Reset type: SYSRSn
15-4	RESERVED	R	0h	Reserved
3	LOCK_CLAPROGROM	R/W	0h	Locks write access to test control fields (TEST and FORCE_ERROR) of CLAPROGROM 0: Write access allowed 1: Write access blocked Reset type: SYSRSn
2	LOCK_CLADATAROM	R/W	0h	Locks write access to test control fields (TEST and FORCE_ERROR) of CLADATAROM 0: Write access allowed 1: Write access blocked Reset type: SYSRSn
1	LOCK_SECUREROM	R/W	0h	Locks write access to test control fields (TEST and FORCE_ERROR) of SECUREROM 0: Write access allowed 1: Write access blocked Reset type: SYSRSn
0	LOCK_BOOTROM	R/W	0h	Locks write access to test control fields (TEST and FORCE_ERROR) of BOOTROM 0: Write access allowed 1: Write access blocked Reset type: SYSRSn

3.15.8.32 ROM_TEST Register (Offset = A2h) [Reset = 0h]

ROM_TEST is shown in Figure 3-160 and described in Table 3-171.

Return to the Summary Table.

ROM TEST Register

Figure 3-160. ROM_TEST Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
TEST_CLAPROGROM		TEST_CLADATAROM		TEST_SECUREROM		TEST_BOOTROM	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 3-171. ROM_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	TEST_CLAPROGROM	R/W	0h	Selects the different modes for CLAPROGROM: 00: Functional Mode. 01: same as "00" but Parity check on data read is disabled (for debug) 10: Parity Bits are visible on memory map (for debug) 11: Same as "00" but NMI is not generated on errors, used for diagnostics. (for diagnostics) Reset type: SYSRSn
5-4	TEST_CLADATAROM	R/W	0h	Selects the different modes for CLADATAROM: 00: Functional Mode. 01: same as "00" but Parity check on data read is disabled (for debug) 10: Parity Bits are visible on memory map (for debug) 11: Same as "00" but NMI is not generated on errors, used for diagnostics. (for diagnostics) Reset type: SYSRSn
3-2	TEST_SECUREROM	R/W	0h	Selects the different modes for SECUREROM: 00: Functional Mode. 01: same as "00" but Parity check on data read is disabled (for debug) 10: Parity Bits are visible on memory map (for debug) 11: Same as "00" but NMI is not generated on errors, used for diagnostics. (for diagnostics) Reset type: SYSRSn
1-0	TEST_BOOTROM	R/W	0h	Selects the different modes for BOOTROM: 00: Functional Mode. 01: same as "00" but Parity check on data read is disabled (for debug) 10: Parity Bits are visible on memory map (for debug) 11: Same as "00" but NMI is not generated on errors, used for diagnostics. (for diagnostics) Reset type: SYSRSn

3.15.8.33 ROM_FORCE_ERROR Register (Offset = A4h) [Reset = 0h]

ROM_FORCE_ERROR is shown in Figure 3-161 and described in Table 3-172.

Return to the Summary Table.

ROM Force Error register

Figure 3-161. ROM_FORCE_ERROR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				FORCE_CLAPROGROM_ERROR	FORCE_CLADATAROM_ERROR	FORCE_SECUREROM_ERROR	FORCE_BOOTROM_ERROR
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-172. ROM_FORCE_ERROR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	FORCE_CLAPROGROM_ERROR	R/W	0h	Force parity error by feeding inverted Parity bit to Parity checking logic. Reset type: SYSRSn
2	FORCE_CLADATAROM_ERROR	R/W	0h	Force parity error by feeding inverted Parity bit to Parity checking logic. Reset type: SYSRSn
1	FORCE_SECUREROM_ERROR	R/W	0h	Force parity error by feeding inverted Parity bit to Parity checking logic. Reset type: SYSRSn
0	FORCE_BOOTROM_ERROR	R/W	0h	Force parity error by feeding inverted Parity bit to Parity checking logic. Reset type: SYSRSn

3.15.9 MEMORY_ERROR_REGS Registers

Table 3-173 lists the memory-mapped registers for the MEMORY_ERROR_REGS registers. All register offset addresses not listed in Table 3-173 should be considered as reserved locations and the register contents should not be modified.

Table 3-173. MEMORY_ERROR_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	UCERRFLG	Uncorrectable Error Flag Register		Go
2h	UCERRSET	Uncorrectable Error Flag Set Register	EALLOW	Go
4h	UCERRCLR	Uncorrectable Error Flag Clear Register	EALLOW	Go
6h	UCCPUREADDR	Uncorrectable CPU Read Error Address		Go
8h	UCDMAREADDR	Uncorrectable DMA Read Error Address		Go
Ah	UCCLA1READDR	Uncorrectable CLA1 Read Error Address		Go
10h	UCHICAREADDR	Uncorrectable HICA Read Error Address		Go
20h	CERRFLG	Correctable Error Flag Register		Go
22h	CERRSET	Correctable Error Flag Set Register	EALLOW	Go
24h	CERRCLR	Correctable Error Flag Clear Register	EALLOW	Go
26h	CCPUREADDR	Correctable CPU Read Error Address		Go
28h	CDMAREADDR	Correctable DMA Read Error Address		Go
2Ah	CCLA1READDR	Correctable CLA1 Read Error Address		Go
2Eh	CERRCNT	Correctable Error Count Register		Go
30h	CERRTHRES	Correctable Error Threshold Value Register	EALLOW	Go
32h	CEINTFLG	Correctable Error Interrupt Flag Status Register		Go
34h	CEINTCLR	Correctable Error Interrupt Flag Clear Register	EALLOW	Go
36h	CEINTSET	Correctable Error Interrupt Flag Set Register	EALLOW	Go
38h	CEINTEN	Correctable Error Interrupt Enable Register	EALLOW	Go
3Ah	CHICREADDR	Correctable HIC Read Error Address		Go

Complex bit access types are encoded to fit into small table cells. Table 3-174 shows the codes that are used for access types in this section.

Table 3-174. MEMORY_ERROR_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 3-174. MEMORY_ERROR_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.15.9.1 UCERRFLG Register (Offset = 0h) [Reset = 0h]

UCERRFLG is shown in Figure 3-162 and described in Table 3-175.

Return to the Summary Table.

Uncorrectable Error Flag Register

Figure 3-162. UCERRFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		HICARDERR	RESERVED	RESERVED	CLA1RDERR	DMARDERR	CPURDERR
R-0h		R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 3-175. UCERRFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-6	RESERVED	R	0h	Reserved
5	HICARDERR	R	0h	HICA Uncorrectable Read Error Flag 0: No Error. 1: Uncorrectable error occurred during HICA read. Reset type: SYSRSn
4	RESERVED	R	0h	Reserved
3	RESERVED	R	0h	Reserved
2	CLA1RDERR	R	0h	CLA1 Uncorrectable Read Error Flag 0: No Error. 1: Uncorrectable error occurred during CLA1 read. Reset type: SYSRSn
1	DMARDERR	R	0h	DMA Uncorrectable Read Error Flag 0: No Error. 1: Uncorrectable error occurred during DMA read. Reset type: SYSRSn
0	CPURDERR	R	0h	CPU Uncorrectable Read Error Flag 0: No Error. 1: Uncorrectable error occurred during CPU read. Reset type: SYSRSn

3.15.9.2 UCERRSET Register (Offset = 2h) [Reset = 0h]

UCERRSET is shown in Figure 3-163 and described in Table 3-176.

Return to the Summary Table.

Uncorrectable Error Flag Set Register

Figure 3-163. UCERRSET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		HICARDERR	RESERVED	RESERVED	CLA1RDERR	DMARDERR	CPURDERR
R-0h		R-0/W1S-0h	R-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-176. UCERRSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-6	RESERVED	R	0h	Reserved
5	HICARDERR	R-0/W1S	0h	0: No action. 1: HICA Read Error Flag in UCERRFLG register will be set and interrupt will be generated if enabled.. Reset type: SYSRSn
4	RESERVED	R	0h	Reserved
3	RESERVED	R-0/W1S	0h	Reserved
2	CLA1RDERR	R-0/W1S	0h	0: No action. 1: CLA1 Read Error Flag in UCERRFLG register will be set and interrupt will be generated if enabled.. Reset type: SYSRSn
1	DMARDERR	R-0/W1S	0h	0: No action. 1: DMA Read Error Flag in UCERRFLG register will be set and interrupt will be generated if enabled.. Reset type: SYSRSn
0	CPURDERR	R-0/W1S	0h	0: No action. 1: CPU Read Error Flag in UCERRFLG register will be set and interrupt will be generated if enabled.. Reset type: SYSRSn

3.15.9.3 UCERRCLR Register (Offset = 4h) [Reset = 0h]

UCERRCLR is shown in Figure 3-164 and described in Table 3-177.

Return to the Summary Table.

Uncorrectable Error Flag Clear Register

Figure 3-164. UCERRCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		HICARDERR	RESERVED	RESERVED	CLA1RDERR	DMARDERR	CPURDERR
R-0h		R-0/W1S-0h	R-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-177. UCERRCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-6	RESERVED	R	0h	Reserved
5	HICARDERR	R-0/W1S	0h	0: No action. 1: HICA Read Error Flag in UCERRFLG register will be cleared. Reset type: SYSRSn
4	RESERVED	R	0h	Reserved
3	RESERVED	R-0/W1S	0h	Reserved
2	CLA1RDERR	R-0/W1S	0h	0: No action. 1: CLA1 Read Error Flag in UCERRFLG register will be cleared. Reset type: SYSRSn
1	DMARDERR	R-0/W1S	0h	0: No action. 1: DMA Read Error Flag in UCERRFLG register will be cleared . Reset type: SYSRSn
0	CPURDERR	R-0/W1S	0h	0: No action. 1: CPU Read Error Flag in UCERRFLG register will be cleared. Reset type: SYSRSn

3.15.9.4 UCCPUREADDR Register (Offset = 6h) [Reset = 0h]

UCCPUREADDR is shown in Figure 3-165 and described in Table 3-178.

Return to the Summary Table.

Uncorrectable CPU Read Error Address

Figure 3-165. UCCPUREADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCCPUREADDR																															
R-0h																															

Table 3-178. UCCPUREADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	UCCPUREADDR	R	0h	This register captures the address location for which CPU read/fetch access resulted in uncorrectable ECC/Parity error. Reset type: SYSRSn

3.15.9.5 UCDMAREADDR Register (Offset = 8h) [Reset = 0h]

UCDMAREADDR is shown in Figure 3-166 and described in Table 3-179.

Return to the Summary Table.

Uncorrectable DMA Read Error Address

Figure 3-166. UCDMAREADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCDMAREADDR																															
R-0h																															

Table 3-179. UCDMAREADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	UCDMAREADDR	R	0h	This register captures the address location for which DMA read access resulted in uncorrectable Parity error. Reset type: SYSRSn

3.15.9.6 UCCLA1READDR Register (Offset = Ah) [Reset = 0h]

UCCLA1READDR is shown in Figure 3-167 and described in Table 3-180.

Return to the Summary Table.

Uncorrectable CLA1 Read Error Address

Figure 3-167. UCCLA1READDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCCLA1READDR																															
R-0h																															

Table 3-180. UCCLA1READDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	UCCLA1READDR	R	0h	This register captures the address location for which CLA1 read/ fetch access resulted in uncorrectable Parity error. Reset type: SYSRSn

3.15.9.7 UCHICAREADDR Register (Offset = 10h) [Reset = 0h]

UCHICAREADDR is shown in Figure 3-168 and described in Table 3-181.

Return to the Summary Table.

Uncorrectable HICA Read Error Address

Figure 3-168. UCHICAREADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UCHICAREADDR																															
R-0h																															

Table 3-181. UCHICAREADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	UCHICAREADDR	R	0h	This register captures the address location for which HICA read access resulted in uncorrectable ECC/Parity error. Reset type: SYSRSn

3.15.9.8 CERRFLG Register (Offset = 20h) [Reset = 0h]

CERRFLG is shown in Figure 3-169 and described in Table 3-182.

Return to the Summary Table.

Correctable Error Flag Register

Figure 3-169. CERRFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		HICRDERR	RESERVED	RESERVED	CLA1RDERR	DMARDERR	CPURDERR
R-0h		R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 3-182. CERRFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-6	RESERVED	R	0h	Reserved
5	HICRDERR	R	0h	HIC Correctable Read Error Flag 0: No Error. 1: Correctable error occurred during HIC read. Reset type: SYSRSn
4	RESERVED	R	0h	Reserved
3	RESERVED	R	0h	Reserved
2	CLA1RDERR	R	0h	CLA1 Correctable Read Error Flag 0: No Error. 1: Correctable error occurred during CLA1 read. Reset type: SYSRSn
1	DMARDERR	R	0h	DMA Correctable Read Error Flag 0: No Error. 1: Correctable error occurred during DMA read. Reset type: SYSRSn
0	CPURDERR	R	0h	CPU Correctable Read Error Flag 0: No Error. 1: Correctable error occurred during CPU read. Reset type: SYSRSn

3.15.9.9 CERRSET Register (Offset = 22h) [Reset = 0h]

CERRSET is shown in Figure 3-170 and described in Table 3-183.

Return to the Summary Table.

Correctable Error Flag Set Register

Figure 3-170. CERRSET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		HICRDERR	RESERVED	RESERVED	CLA1RDERR	DMARDERR	CPURDERR
R-0h		R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-183. CERRSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-6	RESERVED	R	0h	Reserved
5	HICRDERR	R-0/W1S	0h	0: No action. 1: HIC Read Error Flag in CERRFLG register will be set and interrupt will be generated if enabled.. Reset type: SYSRSn
4	RESERVED	R-0/W1S	0h	Reserved
3	RESERVED	R-0/W1S	0h	Reserved
2	CLA1RDERR	R-0/W1S	0h	0: No action. 1: CLA1 Read Error Flag in CERRFLG register will be set and interrupt will be generated if enabled.. Reset type: SYSRSn
1	DMARDERR	R-0/W1S	0h	0: No action. 1: DMA Read Error Flag in CERRFLG register will be set and interrupt will be generated if enabled.. Reset type: SYSRSn
0	CPURDERR	R-0/W1S	0h	0: No action. 1: CPU Read Error Flag in CERRFLG register will be set and interrupt will be generated if enabled.. Reset type: SYSRSn

3.15.9.10 CERRCLR Register (Offset = 24h) [Reset = 0h]

CERRCLR is shown in Figure 3-171 and described in Table 3-184.

Return to the Summary Table.

Correctable Error Flag Clear Register

Figure 3-171. CERRCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		HICRDERR	RESERVED	RESERVED	CLA1RDERR	DMARDERR	CPURDERR
R-0h		R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-184. CERRCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-6	RESERVED	R	0h	Reserved
5	HICRDERR	R-0/W1S	0h	0: No action. 1: HIC Read Error Flag in CERRFLG register will be cleared. Reset type: SYSRSn
4	RESERVED	R-0/W1S	0h	Reserved
3	RESERVED	R-0/W1S	0h	Reserved
2	CLA1RDERR	R-0/W1S	0h	0: No action. 1: CLA1 Read Error Flag in CERRFLG register will be cleared. Reset type: SYSRSn
1	DMARDERR	R-0/W1S	0h	0: No action. 1: DMA Read Error Flag in CERRFLG register will be cleared . Reset type: SYSRSn
0	CPURDERR	R-0/W1S	0h	0: No action. 1: CPU Read Error Flag in CERRFLG register will be cleared. Reset type: SYSRSn

3.15.9.11 CCPUREADDR Register (Offset = 26h) [Reset = 0h]

CCPUREADDR is shown in Figure 3-172 and described in Table 3-185.

Return to the Summary Table.

Correctable CPU Read Error Address

Figure 3-172. CCPUREADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCPUREADDR																															
R-0h																															

Table 3-185. CCPUREADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CCPUREADDR	R	0h	This register captures the address location for which CPU read/fetch access resulted in correctable ECC error. Reset type: SYSRSn

3.15.9.12 CDMAREADDR Register (Offset = 28h) [Reset = 0h]

CDMAREADDR is shown in Figure 3-173 and described in Table 3-186.

Return to the Summary Table.

Correctable DMA Read Error Address

Figure 3-173. CDMAREADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CDMAREADDR																															
R-0h																															

Table 3-186. CDMAREADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CDMAREADDR	R	0h	This register captures the address location for which DMA read access resulted in correctable ECC error. Reset type: SYSRSn

3.15.9.13 CCLA1READDR Register (Offset = 2Ah) [Reset = 0h]

CCLA1READDR is shown in Figure 3-174 and described in Table 3-187.

Return to the Summary Table.

Correctable CLA1 Read Error Address

Figure 3-174. CCLA1READDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CCLA1READDR																															
R-0h																															

Table 3-187. CCLA1READDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CCLA1READDR	R	0h	This register captures the address location for which CLA1 read/ fetch access resulted in correctable ECC error. Reset type: SYSRSn

3.15.9.14 CERRCNT Register (Offset = 2Eh) [Reset = 0h]

CERRCNT is shown in Figure 3-175 and described in Table 3-188.

Return to the Summary Table.

Correctable Error Count Register

Figure 3-175. CERRCNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CERRCNT																															
R-0h																															

Table 3-188. CERRCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CERRCNT	R	0h	This register holds the count of how many times correctable error occurred. Reset type: SYSRSn

3.15.9.15 CERRTHRES Register (Offset = 30h) [Reset = 0h]

CERRTHRES is shown in Figure 3-176 and described in Table 3-189.

Return to the Summary Table.

Correctable Error Threshold Value Register

Figure 3-176. CERRTHRES Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																CERRTHRES															
R-0h																R/W-0h															

Table 3-189. CERRTHRES Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	CERRTHRES	R/W	0h	When value in CERRCNT register is greater than value configured in this register, corretable interrupt gets generated, if enabled. Reset type: SYSRSn

3.15.9.16 CEINTFLG Register (Offset = 32h) [Reset = 0h]

CEINTFLG is shown in Figure 3-177 and described in Table 3-190.

Return to the Summary Table.

Correctable Error Interrupt Flag Status Register

Figure 3-177. CEINTFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CEINTFLAG
R-0h							R-0h

Table 3-190. CEINTFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	CEINTFLAG	R	0h	Total corrected error count exceeded threshold Flag 0: Total correctable errors < Threshold value configured in CERRTHRES register. 1: Total correctable errors >= Threshold value configured in CERRTHRES register. Reset type: SYSRSn

3.15.9.17 CEINTCLR Register (Offset = 34h) [Reset = 0h]

CEINTCLR is shown in Figure 3-178 and described in Table 3-191.

Return to the Summary Table.

Correctable Error Interrupt Flag Clear Register

Figure 3-178. CEINTCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CEINTCLR
R-0h							R-0/W1S-0h

Table 3-191. CEINTCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	CEINTCLR	R-0/W1S	0h	0: No action. 1: Total corrected error count exceeded flag in CEINTFLG register will be cleared. Reset type: SYSRSn

3.15.9.18 CEINTSET Register (Offset = 36h) [Reset = 0h]

CEINTSET is shown in Figure 3-179 and described in Table 3-192.

Return to the Summary Table.

Correctable Error Interrupt Flag Set Register

Figure 3-179. CEINTSET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CEINTSET
R-0h							R-0/W1S-0h

Table 3-192. CEINTSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	CEINTSET	R-0/W1S	0h	0: No action. 1: Total corrected error count exceeded flag in CEINTFLG register will be set and interrupt will be generated if enabled. Reset type: SYSRSn

3.15.9.19 CEINTEN Register (Offset = 38h) [Reset = 0h]

CEINTEN is shown in Figure 3-180 and described in Table 3-193.

Return to the Summary Table.

Correctable Error Interrupt Enable Register

Figure 3-180. CEINTEN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CEINTEN
R-0h							R/W-0h

Table 3-193. CEINTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	CEINTEN	R/W	0h	0: Correctable Error Interrupt is disabled. 1: Correctable Error Interrupt is enabled. Reset type: SYSRSn

3.15.9.20 CHICREADDR Register (Offset = 3Ah) [Reset = 0h]

CHICREADDR is shown in Figure 3-181 and described in Table 3-194.

Return to the Summary Table.

Correctable HIC Read Error Address

Figure 3-181. CHICREADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CHICREADDR																															
R-0h																															

Table 3-194. CHICREADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CHICREADDR	R	0h	This register captures the address location for which HIC read access resulted in correctable ECC error. Reset type: SYSRSn

3.15.10 NMI_INTRUPT_REGS Registers

Table 3-195 lists the memory-mapped registers for the NMI_INTRUPT_REGS registers. All register offset addresses not listed in Table 3-195 should be considered as reserved locations and the register contents should not be modified.

Table 3-195. NMI_INTRUPT_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	NMICFG	NMI Configuration Register	EALLOW	Go
1h	NMIFLG	NMI Flag Register (SYSRsn Clear)		Go
2h	NMIFLGCLR	NMI Flag Clear Register	EALLOW	Go
3h	NMIFLGFRC	NMI Flag Force Register	EALLOW	Go
4h	NMIWDCNT	NMI Watchdog Counter Register		Go
5h	NMIWDPRD	NMI Watchdog Period Register	EALLOW	Go
6h	NMISHDFLG	NMI Shadow Flag Register		Go
7h	ERRORSTS	Error pin status		Go
8h	ERRORSTSCLR	ERRORSTS clear register	EALLOW	Go
9h	ERRORSTSFRC	ERRORSTS force register	EALLOW	Go
Ah	ERRORCTL	Error pin control register	EALLOW	Go
Bh	ERRORLOCK	Lock register to Error pin registers.	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 3-196 shows the codes that are used for access types in this section.

Table 3-196. NMI_INTRUPT_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.15.10.1 NMICFG Register (Offset = 0h) [Reset = 0h]

NMICFG is shown in Figure 3-182 and described in Table 3-197.

Return to the Summary Table.

NMI Configuration Register

Figure 3-182. NMICFG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							NMIE
R-0h							R/W1S-0h

Table 3-197. NMICFG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R	0h	Reserved
0	NMIE	R/W1S	0h	When set to 1 any condition will generate an NMI interrupt to the C28 CPU and kick off the NMI watchdog counter. As part of boot sequence this bit should be set after the device security related initialization is complete. 0 NMI disabled 1 NMI enabled Reset type: SYSRSn

3.15.10.2 NMIFLG Register (Offset = 1h) [Reset = 0h]

NMIFLG is shown in Figure 3-183 and described in Table 3-198.

Return to the Summary Table.

NMI Flag Register (SYSRsn Clear)

Figure 3-183. NMIFLG Register

15	14	13	12	11	10	9	8
RESERVED	CRC_FAIL	SWERR	RESERVED	RESERVED	RESERVED	RESERVED	CLBNMI
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
SYSDBGNMI	RESERVED	RESERVED	CPU1HWBISTERR	FLUNCERR	RAMUNCERR	CLOCKFAIL	NMIINT
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 3-198. NMIFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	CRC_FAIL	R	0h	0 Background CRC check on memories has not failed. 1 Background CRC check on memories has failed. Reset type: SYSRSn
13	SWERR	R	0h	SW Error Force NMI Flag: This bit indicates if an NMI was forced through the NMIFLGFRC register. This bit can only be cleared by the user writing to the respective bit in the NMIFLGCLR register or by an SYSRSn reset: 0 No SW Error force Generated 1 SW Error NMI is forced by SW. No further NMI pulses are generated until this flag is cleared by the user. Reset type: SYSRSn
12	RESERVED	R	0h	Reserved
11	RESERVED	R	0h	Reserved
10	RESERVED	R	0h	Reserved
9	RESERVED	R	0h	Reserved
8	CLBNMI	R	0h	Reconfigurable Logic NMI Flag: This bit indicates if an NMI was generated by the Reconfigurable Logic. This bit can only be cleared by the user writing to the corresponding clear bit in the NMIFLGCLR register or by SYSRSn reset: 0,No Reconfigurable Logic NMI pending 1,Reconfigurable Logic NMI generated Reset type: SYSRSn
7	SYSDBGNMI	R	0h	System Debug Module NMI Flag: This bit indicates if an NMI was generated by the System Debug Module. This bit can only be cleared by the user writing to the corresponding clear bit in the NMIFLGCLR register or by SYSRSn reset: 0,No System Debug NMI pending 1,System Debug NMI generated Reset type: SYSRSn
6	RESERVED	R	0h	Reserved
5	RESERVED	R	0h	Reserved

Table 3-198. NMIFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	CPU1HWBISTERR	R	0h	HW BIST Error NMI Flag: This bit indicates if the time out error or a signature mismatch error condition during hardware BIST of C28 CPU1 occurred. This bit can only be cleared by the user writing to the corresponding clear bit in the NMIFLGCLR register or by SYSRSn reset: 0,No C28 HWBIST error condition pending 1,C28 BIST error condition generated Reset type: SYSRSn
3	FLUNCERR	R	0h	Flash Uncorrectable Error NMI Flag: This bit indicates if an uncorrectable error occurred on a C28 Flash access and that condition is latched. This bit can only be cleared by the user writing to the corresponding clear bit in the NMIFLGCLR register or by SYSRSn reset: 0,No C28 Flash uncorrectable error condition pending 1,C28 Flash uncorrectable error condition generated Reset type: SYSRSn
2	RAMUNCERR	R	0h	RAM Uncorrectable Error NMI Flag: This bit indicates if an uncorrectable error occurred on a RAM access (by any master) and that condition is latched. This bit can only be cleared by the user writing to the corresponding clear bit in the NMIFLGCLR register or by SYSRSn reset: 0,No RAM uncorrectable error condition pending 1,RAM uncorrectable error condition generated Reset type: SYSRSn
1	CLOCKFAIL	R	0h	Clock Fail Interrupt Flag: These bits indicates if the CLOCKFAIL condition is latched. These bits can only be cleared by the user writing to the respective bit in the NMIFLGCLR register or by SYSRSn reset: 0,No CLOCKFAIL Condition Pending 1,CLOCKFAIL Condition Generated Reset type: SYSRSn
0	NMIINT	R	0h	NMI Interrupt Flag: This bit indicates if an NMI interrupt was generated. This bit can only be cleared by the user writing to the respective bit in the NMIFLGCLR register or by SYSRSn reset: 0 No NMI Interrupt Generated 1 NMI Interrupt Generated No further NMI interrupts pulses are generated until this flag is cleared by the user. Reset type: SYSRSn

3.15.10.3 NMIFLGCLR Register (Offset = 2h) [Reset = 0h]

NMIFLGCLR is shown in Figure 3-184 and described in Table 3-199.

Return to the Summary Table.

NMI Flag Clear Register

Figure 3-184. NMIFLGCLR Register

15	14	13	12	11	10	9	8
RESERVED	CRC_FAIL	SWERR	RESERVED	RESERVED	RESERVED	RESERVED	RLNMI
R-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
SYSDBGNMI	RESERVED	RESERVED	CPU1HWBISTERR	FLUNCERR	RAMUNCERR	CLOCKFAIL	NMIINT
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-199. NMIFLGCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	CRC_FAIL	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: SYSRSn
13	SWERR	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: SYSRSn
12	RESERVED	R-0/W1S	0h	Reserved
11	RESERVED	R-0/W1S	0h	Reserved
10	RESERVED	R-0/W1S	0h	Reserved
9	RESERVED	R-0/W1S	0h	Reserved
8	RLNMI	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: SYSRSn
7	SYSDBGNMI	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: SYSRSn

Table 3-199. NMIFLGCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	RESERVED	R-0/W1S	0h	Reserved
5	RESERVED	R-0/W1S	0h	Reserved
4	CPU1HWBISTERR	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: SYSRSn
3	FLUNCERR	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: SYSRSn
2	RAMUNCERR	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: SYSRSn
1	CLOCKFAIL	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: SYSRSn
0	NMIINT	R-0/W1S	0h	Writing a 1 to the respective bit clears the corresponding flag bit in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. Notes: [1] If hardware is trying to set a bit to 1 while software is trying to clear a bit to 0 on the same cycle, hardware has priority. [2] Users should clear the pending FAIL flag first and then clear the NMIINT flag. Reset type: SYSRSn

3.15.10.4 NMIFLGFRC Register (Offset = 3h) [Reset = 0h]

NMIFLGFRC is shown in Figure 3-185 and described in Table 3-200.

Return to the Summary Table.

NMI Flag Force Register

Figure 3-185. NMIFLGFRC Register

15	14	13	12	11	10	9	8
RESERVED	CRC_FAIL	SWERR	RESERVED	RESERVED	RESERVED	RESERVED	RLNMI
R-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
SYSDBGNMI	RESERVED	RESERVED	CPU1HWBISTERR	FLUNCERR	RAMUNCERR	CLOCKFAIL	RESERVED
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0h

Table 3-200. NMIFLGFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	CRC_FAIL	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. Reset type: SYSRSn
13	SWERR	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. Reset type: SYSRSn
12	RESERVED	R-0/W1S	0h	Reserved
11	RESERVED	R-0/W1S	0h	Reserved
10	RESERVED	R-0/W1S	0h	Reserved
9	RESERVED	R-0/W1S	0h	Reserved
8	RLNMI	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. Reset type: SYSRSn
7	SYSDBGNMI	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. Reset type: SYSRSn
6	RESERVED	R-0/W1S	0h	Reserved
5	RESERVED	R-0/W1S	0h	Reserved
4	CPU1HWBISTERR	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. Reset type: SYSRSn
3	FLUNCERR	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. Reset type: SYSRSn

Table 3-200. NMIFLGFRC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	RAMUNCERR	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. Reset type: SYSRSn
1	CLOCKFAIL	R-0/W1S	0h	Writing a 1 to these bits will set the respective FAIL flag in the NMIFLG and NMISHDFLG registers. Writes of 0 are ignored. Always reads back 0. This can be used as a means to test the NMI mechanisms. Reset type: SYSRSn
0	RESERVED	R	0h	Reserved

3.15.10.5 NMIWDCNT Register (Offset = 4h) [Reset = 0h]

NMIWDCNT is shown in Figure 3-186 and described in Table 3-201.

Return to the Summary Table.

NMI Watchdog Counter Register

Figure 3-186. NMIWDCNT Register

15	14	13	12	11	10	9	8
NMIWDCNT							
R-0h							
7	6	5	4	3	2	1	0
NMIWDCNT							
R-0h							

Table 3-201. NMIWDCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	NMIWDCNT	R	0h	NMI Watchdog Counter: This 16-bit incremental counter will start incrementing whenever any one of the enabled FAIL flags are set. If the counter reaches the period value, an NMIRSn signal is fired which will then resets the system. The counter will reset to zero when it reaches the period value and will then restart counting if any of the enabled FAIL flags are set. If no enabled FAIL flag is set, then the counter will reset to zero and remain at zero until an enabled FAIL flag is set. Normally, the software would respond to the NMI interrupt generated and clear the offending FLAG(s) before the NMI watchdog triggers a reset. In some situations, the software may decide to allow the watchdog to reset the device anyway. The counter is clocked at the SYSCLKOUT rate. Reset type: SYSRSn

3.15.10.6 NMIWDPRD Register (Offset = 5h) [Reset = FFFFh]

NMIWDPRD is shown in Figure 3-187 and described in Table 3-202.

Return to the Summary Table.

NMI Watchdog Period Register

Figure 3-187. NMIWDPRD Register

15	14	13	12	11	10	9	8
NMIWDPRD							
R/W-FFFFh							
7	6	5	4	3	2	1	0
NMIWDPRD							
R/W-FFFFh							

Table 3-202. NMIWDPRD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	NMIWDPRD	R/W	FFFFh	NMI Watchdog Period: This 16-bit value contains the period value at which a reset is generated when the watchdog counter matches. At reset this value is set at the maximum. The software can decrease the period value at initialization time. Writing a PERIOD value that is smaller then the current counter value will automatically force an NMIRSn and hence reset the watchdog counter. Reset type: SYSRSn

3.15.10.7 NMISHDFLG Register (Offset = 6h) [Reset = 0h]

NMISHDFLG is shown in Figure 3-188 and described in Table 3-203.

Return to the Summary Table.

NMI Shadow Flag Register

Figure 3-188. NMISHDFLG Register

15	14	13	12	11	10	9	8
RESERVED	CRC_FAIL	SWERR	RESERVED	RESERVED	RESERVED	RESERVED	RLNMI
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
SYSDBGNMI	RESERVED	RESERVED	CPU1HWBISTERR	FLUNCERR	RAMUNCERR	CLOCKFAIL	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 3-203. NMISHDFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	CRC_FAIL	R	0h	Shadow NMI Flags: When an NMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that NMIFLGFRC and NMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is resetted only by PORn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. Reset type: SYSRSn
13	SWERR	R	0h	Shadow NMI Flags: When an NMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that NMIFLGFRC and NMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is resetted only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. Reset type: PORESETn
12	RESERVED	R	0h	Reserved
11	RESERVED	R	0h	Reserved
10	RESERVED	R	0h	Reserved
9	RESERVED	R	0h	Reserved
8	RLNMI	R	0h	Shadow NMI Flags: When an NMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that NMIFLGFRC and NMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is resetted only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. Reset type: PORESETn
7	SYSDBGNMI	R	0h	Shadow NMI Flags: When an NMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that NMIFLGFRC and NMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is resetted only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. Reset type: PORESETn

Table 3-203. NMISHDFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	RESERVED	R	0h	Reserved
5	RESERVED	R	0h	Reserved
4	CPU1HWBISTERR	R	0h	Shadow NMI Flags: When an NMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that NMIFLGFRC and NMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is resetted only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. Reset type: PORESETn
3	FLUNCERR	R	0h	Shadow NMI Flags: When an NMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that NMIFLGFRC and NMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is resetted only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. Reset type: PORESETn
2	RAMUNCERR	R	0h	Shadow NMI Flags: When an NMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that NMIFLGFRC and NMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is resetted only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. Reset type: PORESETn
1	CLOCKFAIL	R	0h	Shadow NMI Flags: When an NMIFLG bit is set due to any of the possible NMI source in the device, the corresponding bit in this register is also set. Note that NMIFLGFRC and NMIFLGCLR register also affects the bits of this register in the same way as they do for the NMIFLG register. This register is resetted only by PORESETn. Notes: [1] This register is added to keep the definition of System Control Reset Cause Register Clean. Reset type: PORESETn
0	RESERVED	R	0h	Reserved

3.15.10.8 ERRORSTS Register (Offset = 7h) [Reset = 0h]

ERRORSTS is shown in Figure 3-189 and described in Table 3-204.

Return to the Summary Table.

Error pin status

Figure 3-189. ERRORSTS Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						PINSTS	ERROR
R-0h						R-0h	R-0h

Table 3-204. ERRORSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-2	RESERVED	R	0h	Reserved
1	PINSTS	R	0h	0, Error Pin is 0 1, Error Pin is 1 Reset type: PORESETn
0	ERROR	R	0h	0,None of the error sources were triggered. 1, One or more of the error sources triggered, or ERRORSTS.ERROR was set by a write of 1 to ERRORSTSFRC.ERROR bit. Once set, the ERROR flag can be cleared by writing 1 to ERRORSTSCLR.ERROR bit. Following are the events/triggers which can set this bit: 1. nmi interrupt on C28x 2. Watchdog reset 3. Error on a Pie vector fetch 4. Efuse error On a read of this bit, the pin Error pin state will be returned. Reset type: PORESETn

3.15.10.9 ERRORSTSCLR Register (Offset = 8h) [Reset = 0h]

ERRORSTSCLR is shown in Figure 3-190 and described in Table 3-205.

Return to the Summary Table.

ERRORSTS clear register

Figure 3-190. ERRORSTSCLR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							ERROR
R-0h							R-0/W1S-0h

Table 3-205. ERRORSTSCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R	0h	Reserved
0	ERROR	R-0/W1S	0h	0,No effect 1, ERRORSTS.ERROR is cleared to 0 Reset type: PORESETn

3.15.10.10 ERRORSTSFRC Register (Offset = 9h) [Reset = 0h]

ERRORSTSFRC is shown in Figure 3-191 and described in Table 3-206.

Return to the Summary Table.

ERRORSTS force register

Figure 3-191. ERRORSTSFRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							ERROR
R-0h							R-0/W1S-0h

Table 3-206. ERRORSTSFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R	0h	Reserved
0	ERROR	R-0/W1S	0h	0,No effect 1, ERRORSTS.ERROR is set to 1 Reset type: PORESETn

3.15.10.11 ERRORCTL Register (Offset = Ah) [Reset = 0h]

ERRORCTL is shown in Figure 3-192 and described in Table 3-207.

Return to the Summary Table.

Error pin control register

Figure 3-192. ERRORCTL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							ERRORPOLSEL
R-0h							R/W-0h

Table 3-207. ERRORCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R	0h	Reserved
0	ERRORPOLSEL	R/W	0h	0, If ERRORSTS.ERROR is 1, Error pin will be driven with a value of 0, else 1. 1, If ERRORSTS.ERROR is 1, Error pin will be driven with a value of 1, else 0. Reset type: PORESETn

3.15.10.12 ERRORLOCK Register (Offset = Bh) [Reset = 0h]

ERRORLOCK is shown in Figure 3-193 and described in Table 3-208.

Return to the Summary Table.

Lock register to Error pin registers.

Figure 3-193. ERRORLOCK Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							ERRORCTL
R-0h							R/WSonce-0h

Table 3-208. ERRORLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R	0h	Reserved
0	ERRORCTL	R/WSonce	0h	0, Writes to ERRORCTL register allowed. 1, Writes to ERRORCTL register is blocked. Writes of 0 to this bit has no effect. Write of 1 will set this bit, cleared only on a SYSRSn. Reset type: SYSRSn

3.15.11 PERIPH_AC_REGS Registers

Table 3-209 lists the memory-mapped registers for the PERIPH_AC_REGS registers. All register offset addresses not listed in Table 3-209 should be considered as reserved locations and the register contents should not be modified.

Table 3-209. PERIPH_AC_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	ADCA_AC	ADCA Master Access Control Register	EALLOW	Go
2h	ADCB_AC	ADCB Master Access Control Register	EALLOW	Go
4h	ADCC_AC	ADCC Master Access Control Register	EALLOW	Go
10h	CMPSS1_AC	CMPSS1 Master Access Control Register	EALLOW	Go
12h	CMPSS2_AC	CMPSS2 Master Access Control Register	EALLOW	Go
14h	CMPSS3_AC	CMPSS3 Master Access Control Register	EALLOW	Go
16h	CMPSS4_AC	CMPSS4 Master Access Control Register	EALLOW	Go
28h	DACA_AC	DACA Master Access Control Register	EALLOW	Go
2Ah	DACB_AC	DACB Master Access Control Register	EALLOW	Go
48h	EPWM1_AC	EPWM1 Master Access Control Register	EALLOW	Go
4Ah	EPWM2_AC	EPWM2 Master Access Control Register	EALLOW	Go
4Ch	EPWM3_AC	EPWM3 Master Access Control Register	EALLOW	Go
4Eh	EPWM4_AC	EPWM4 Master Access Control Register	EALLOW	Go
50h	EPWM5_AC	EPWM5 Master Access Control Register	EALLOW	Go
52h	EPWM6_AC	EPWM6 Master Access Control Register	EALLOW	Go
54h	EPWM7_AC	EPWM7 Master Access Control Register	EALLOW	Go
56h	EPWM8_AC	EPWM8 Master Access Control Register	EALLOW	Go
70h	EQEP1_AC	EQEP1 Master Access Control Register	EALLOW	Go
72h	EQEP2_AC	EQEP2 Master Access Control Register	EALLOW	Go
80h	ECAP1_AC	ECAP1 Master Access Control Register	EALLOW	Go
82h	ECAP2_AC	ECAP2 Master Access Control Register	EALLOW	Go
84h	ECAP3_AC	ECAP3 Master Access Control Register	EALLOW	Go
A8h	SDFM1_AC	SDFM1 Master Access Control Register	EALLOW	Go
AAh	SDFM2_AC	SDFM2 Master Access Control Register	EALLOW	Go
B0h	CLB1_AC	CLB1 Master Access Control Register	EALLOW	Go
B2h	CLB2_AC	CLB2 Master Access Control Register	EALLOW	Go
B4h	CLB3_AC	CLB3 Master Access Control Register	EALLOW	Go
B6h	CLB4_AC	CLB4 Master Access Control Register	EALLOW	Go
100h	SCIA_AC	SCIA Master Access Control Register	EALLOW	Go
102h	SCIB_AC	SCIB Master Access Control Register	EALLOW	Go
110h	SPIA_AC	SPIA Master Access Control Register	EALLOW	Go
112h	SPIB_AC	SPIB Master Access Control Register	EALLOW	Go
120h	I2CA_AC	I2CA Master Access Control Register	EALLOW	Go
122h	I2CB_AC	I2CB Master Access Control Register	EALLOW	Go
130h	PMBUS_A_AC	PMBUSA Master Access Control Register	EALLOW	Go
138h	LIN_A_AC	LINA Master Access Control Register	EALLOW	Go
13Ah	LIN_B_AC	LINB Master Access Control Register	EALLOW	Go
140h	DCANA_AC	DCANA Master Access Control Register	EALLOW	Go
148h	MCANA_AC	MCANA Master Access Control Register	EALLOW	Go
158h	FSIATX_AC	FSIA Master Access Control Register	EALLOW	Go
15Ah	FSIARX_AC	FSIB Master Access Control Register	EALLOW	Go

Table 3-209. PERIPH_AC_REGS Registers (continued)

Offset	Acronym	Register Name	Write Protection	Section
1AAh	HRPWM_A_AC	HRPWM Master Access Control Register	EALLOW	Go
1ACh	HIC_A_AC	HIC Master Access Control Register	EALLOW	Go
1AEh	AESA_AC	AES Master Access Control Register	EALLOW	Go
1FEh	PERIPH_AC_LOCK	Lock Register to stop Write access to peripheral Access register.	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 3-210 shows the codes that are used for access types in this section.

Table 3-210. PERIPH_AC_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.15.11.1 ADCA_AC Register (Offset = 0h) [Reset = FFh]

ADCA_AC is shown in Figure 3-194 and described in Table 3-211.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-194. ADCA_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RESERVED		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-211. ADCA_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	RESERVED	R/W	3h	Reserved
5-4	RESERVED	R/W	3h	Reserved
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.2 ADCB_AC Register (Offset = 2h) [Reset = FFh]

ADCB_AC is shown in Figure 3-195 and described in Table 3-212.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-195. ADCB_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RESERVED		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-212. ADCB_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	RESERVED	R/W	3h	Reserved
5-4	RESERVED	R/W	3h	Reserved
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.3 ADCC_AC Register (Offset = 4h) [Reset = FFh]

ADCC_AC is shown in Figure 3-196 and described in Table 3-213.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-196. ADCC_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RESERVED		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-213. ADCC_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	RESERVED	R/W	3h	Reserved
5-4	RESERVED	R/W	3h	Reserved
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.4 CMPSS1_AC Register (Offset = 10h) [Reset = FFh]

CMPSS1_AC is shown in Figure 3-197 and described in Table 3-214.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-197. CMPSS1_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-214. CMPSS1_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.5 CMPSS2_AC Register (Offset = 12h) [Reset = FFh]

CMPSS2_AC is shown in Figure 3-198 and described in Table 3-215.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-198. CMPSS2_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-215. CMPSS2_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.6 CMPSS3_AC Register (Offset = 14h) [Reset = FFh]

CMPSS3_AC is shown in Figure 3-199 and described in Table 3-216.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-199. CMPSS3_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-216. CMPSS3_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.7 CMPSS4_AC Register (Offset = 16h) [Reset = FFh]

CMPSS4_AC is shown in Figure 3-200 and described in Table 3-217.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-200. CMPSS4_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-217. CMPSS4_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.8 DACA_AC Register (Offset = 28h) [Reset = FFh]

DACA_AC is shown in Figure 3-201 and described in Table 3-218.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-201. DACA_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-218. DACA_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.9 DACB_AC Register (Offset = 2Ah) [Reset = FFh]

DACB_AC is shown in Figure 3-202 and described in Table 3-219.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-202. DACB_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-219. DACB_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.10 EPWM1_AC Register (Offset = 48h) [Reset = FFh]

EPWM1_AC is shown in Figure 3-203 and described in Table 3-220.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-203. EPWM1_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-220. EPWM1_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.11 EPWM2_AC Register (Offset = 4Ah) [Reset = FFh]

EPWM2_AC is shown in Figure 3-204 and described in Table 3-221.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-204. EPWM2_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-221. EPWM2_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.12 EPWM3_AC Register (Offset = 4Ch) [Reset = FFh]

EPWM3_AC is shown in Figure 3-205 and described in Table 3-222.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-205. EPWM3_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-222. EPWM3_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.13 EPWM4_AC Register (Offset = 4Eh) [Reset = FFh]

EPWM4_AC is shown in Figure 3-206 and described in Table 3-223.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-206. EPWM4_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-223. EPWM4_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.14 EPWM5_AC Register (Offset = 50h) [Reset = FFh]

EPWM5_AC is shown in Figure 3-207 and described in Table 3-224.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-207. EPWM5_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-224. EPWM5_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.15 EPWM6_AC Register (Offset = 52h) [Reset = FFh]

EPWM6_AC is shown in Figure 3-208 and described in Table 3-225.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-208. EPWM6_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-225. EPWM6_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.16 EPWM7_AC Register (Offset = 54h) [Reset = FFh]

EPWM7_AC is shown in Figure 3-209 and described in Table 3-226.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-209. EPWM7_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-226. EPWM7_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.17 EPWM8_AC Register (Offset = 56h) [Reset = FFh]

EPWM8_AC is shown in Figure 3-210 and described in Table 3-227.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-210. EPWM8_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-227. EPWM8_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.18 EQEP1_AC Register (Offset = 70h) [Reset = FFh]

EQEP1_AC is shown in Figure 3-211 and described in Table 3-228.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-211. EQEP1_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-228. EQEP1_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.19 EQEP2_AC Register (Offset = 72h) [Reset = FFh]

EQEP2_AC is shown in Figure 3-212 and described in Table 3-229.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-212. EQEP2_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-229. EQEP2_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.20 ECAP1_AC Register (Offset = 80h) [Reset = FFh]

ECAP1_AC is shown in Figure 3-213 and described in Table 3-230.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-213. ECAP1_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-230. ECAP1_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.21 ECAP2_AC Register (Offset = 82h) [Reset = FFh]

ECAP2_AC is shown in Figure 3-214 and described in Table 3-231.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-214. ECAP2_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-231. ECAP2_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.22 ECAP3_AC Register (Offset = 84h) [Reset = FFh]

ECAP3_AC is shown in Figure 3-215 and described in Table 3-232.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-215. ECAP3_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-232. ECAP3_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.23 SDFM1_AC Register (Offset = A8h) [Reset = FFh]

SDFM1_AC is shown in Figure 3-216 and described in Table 3-233.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-216. SDFM1_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-233. SDFM1_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.24 SDFM2_AC Register (Offset = AAh) [Reset = FFh]

SDFM2_AC is shown in Figure 3-217 and described in Table 3-234.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-217. SDFM2_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-234. SDFM2_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.25 CLB1_AC Register (Offset = B0h) [Reset = FFh]

CLB1_AC is shown in Figure 3-218 and described in Table 3-235.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-218. CLB1_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		RESERVED		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-235. CLB1_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	RESERVED	R/W	3h	Reserved
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.26 CLB2_AC Register (Offset = B2h) [Reset = FFh]

CLB2_AC is shown in Figure 3-219 and described in Table 3-236.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-219. CLB2_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		RESERVED		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-236. CLB2_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	RESERVED	R/W	3h	Reserved
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.27 CLB3_AC Register (Offset = B4h) [Reset = FFh]

CLB3_AC is shown in Figure 3-220 and described in Table 3-237.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-220. CLB3_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		RESERVED		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-237. CLB3_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	RESERVED	R/W	3h	Reserved
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.28 CLB4_AC Register (Offset = B6h) [Reset = FFh]

CLB4_AC is shown in Figure 3-221 and described in Table 3-238.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-221. CLB4_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		RESERVED		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-238. CLB4_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	RESERVED	R/W	3h	Reserved
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.29 SCIA_AC Register (Offset = 100h) [Reset = CFh]

SCIA_AC is shown in Figure 3-222 and described in Table 3-239.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-222. SCIA_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		RESERVED		RESERVED		CPU1_ACC	
R/W-3h		R-0h		R/W-3h		R/W-3h	

Table 3-239. SCIA_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	RESERVED	R-0	0h	Reserved
3-2	RESERVED	R/W	3h	Reserved
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.30 SCIB_AC Register (Offset = 102h) [Reset = CFh]

SCIB_AC is shown in Figure 3-223 and described in Table 3-240.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-223. SCIB_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		RESERVED		RESERVED		CPU1_ACC	
R/W-3h		R-0h		R/W-3h		R/W-3h	

Table 3-240. SCIB_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	RESERVED	R-0	0h	Reserved
3-2	RESERVED	R/W	3h	Reserved
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.31 SPIA_AC Register (Offset = 110h) [Reset = FFh]

SPIA_AC is shown in Figure 3-224 and described in Table 3-241.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-224. SPIA_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-241. SPIA_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.32 SPIB_AC Register (Offset = 112h) [Reset = FFh]

SPIB_AC is shown in Figure 3-225 and described in Table 3-242.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-225. SPIB_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-242. SPIB_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.33 I2CA_AC Register (Offset = 120h) [Reset = CFh]

I2CA_AC is shown in Figure 3-226 and described in Table 3-243.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-226. I2CA_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		RESERVED		RESERVED		CPU1_ACC	
R/W-3h		R-0h		R/W-3h		R/W-3h	

Table 3-243. I2CA_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	RESERVED	R	0h	Reserved
3-2	RESERVED	R/W	3h	Reserved
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.34 I2CB_AC Register (Offset = 122h) [Reset = CFh]

I2CB_AC is shown in Figure 3-227 and described in Table 3-244.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-227. I2CB_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		RESERVED		RESERVED		CPU1_ACC	
R/W-3h		R-0h		R/W-3h		R/W-3h	

Table 3-244. I2CB_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	RESERVED	R	0h	Reserved
3-2	RESERVED	R/W	3h	Reserved
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.35 PMBUS_A_AC Register (Offset = 130h) [Reset = FFh]

PMBUS_A_AC is shown in Figure 3-228 and described in Table 3-245.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-228. PMBUS_A_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-245. PMBUS_A_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.36 LIN_A_AC Register (Offset = 138h) [Reset = FFh]

LIN_A_AC is shown in Figure 3-229 and described in Table 3-246.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-229. LIN_A_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-246. LIN_A_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.37 LIN_B_AC Register (Offset = 13Ah) [Reset = FFh]

LIN_B_AC is shown in Figure 3-230 and described in Table 3-247.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-230. LIN_B_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-247. LIN_B_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.38 DCANA_AC Register (Offset = 140h) [Reset = FFh]

DCANA_AC is shown in Figure 3-231 and described in Table 3-248.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-231. DCANA_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		RESERVED		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-248. DCANA_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	RESERVED	R/W	3h	Reserved
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.39 MCANA_AC Register (Offset = 148h) [Reset = FFh]

MCANA_AC is shown in Figure 3-232 and described in Table 3-249.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-232. MCANA_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		RESERVED		RESERVED		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-249. MCANA_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	RESERVED	R/W	3h	Reserved
3-2	RESERVED	R/W	3h	Reserved
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.40 FSIATX_AC Register (Offset = 158h) [Reset = FFh]

FSIATX_AC is shown in Figure 3-233 and described in Table 3-250.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-233. FSIATX_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-250. FSIATX_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.41 FSIARX_AC Register (Offset = 15Ah) [Reset = FFh]

FSIARX_AC is shown in Figure 3-234 and described in Table 3-251.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-234. FSIARX_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-251. FSIARX_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.42 HRPWM_A_AC Register (Offset = 1AAh) [Reset = FFh]

HRPWM_A_AC is shown in Figure 3-235 and described in Table 3-252.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-235. HRPWM_A_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HICA_ACC		DMA1_ACC		CLA1_ACC		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-252. HRPWM_A_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	HICA_ACC	R/W	3h	Defines Access control definition for the HICA as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	CLA1_ACC	R/W	3h	Defines Access control definition for the CLA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.43 HIC_A_AC Register (Offset = 1ACh) [Reset = FFh]

HIC_A_AC is shown in Figure 3-236 and described in Table 3-253.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-236. HIC_A_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		RESERVED		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-253. HIC_A_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	RESERVED	R/W	3h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	RESERVED	R/W	3h	Reserved
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.44 AESA_AC Register (Offset = 1AEh) [Reset = FFh]

AESA_AC is shown in Figure 3-237 and described in Table 3-254.

Return to the Summary Table.

Based on control settings allows Full, Protected Read, No Access to perpheral from corresponding connected master.

Figure 3-237. AESA_AC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		DMA1_ACC		RESERVED		CPU1_ACC	
R/W-3h		R/W-3h		R/W-3h		R/W-3h	

Table 3-254. AESA_AC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-6	RESERVED	R/W	3h	Reserved
5-4	DMA1_ACC	R/W	3h	Defines Access control definition for the DMA1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn
3-2	RESERVED	R/W	3h	Reserved
1-0	CPU1_ACC	R/W	3h	Defines Access control definition for the CPU1 as: 11: Full Access for both read and Write 10: Protected RD Access such that FIFOs, Clear on read registers are not changed + No Write Access 01: Reserved 00: No Read/Write Access to peripheral Reset type: XRSn

3.15.11.45 PERIPH_AC_LOCK Register (Offset = 1FEh) [Reset = 0h]

PERIPH_AC_LOCK is shown in Figure 3-238 and described in Table 3-255.

Return to the Summary Table.

Based on status bit control the Access registers are either RD/WR or RD only.

Figure 3-238. PERIPH_AC_LOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							LOCK_AC_WR
R-0h							R/WSonce-0h

Table 3-255. PERIPH_AC_LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	LOCK_AC_WR	R/WSonce	0h	Defines Access control definition for the CPU1 as: 1: Access Control registers are Read Only 0: Read/Write Access allowed to Access Control registers. Writing '1' sets the bit, writing '0' has no effect. Reset type: SYSRSn

3.15.12 PIE_CTRL_REGS Registers

Table 3-256 lists the memory-mapped registers for the PIE_CTRL_REGS registers. All register offset addresses not listed in Table 3-256 should be considered as reserved locations and the register contents should not be modified.

Table 3-256. PIE_CTRL_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	PIECTRL	ePIE Control Register		Go
1h	PIEACK	Interrupt Acknowledge Register		Go
2h	PIEIER1	Interrupt Group 1 Enable Register		Go
3h	PIEIFR1	Interrupt Group 1 Flag Register		Go
4h	PIEIER2	Interrupt Group 2 Enable Register		Go
5h	PIEIFR2	Interrupt Group 2 Flag Register		Go
6h	PIEIER3	Interrupt Group 3 Enable Register		Go
7h	PIEIFR3	Interrupt Group 3 Flag Register		Go
8h	PIEIER4	Interrupt Group 4 Enable Register		Go
9h	PIEIFR4	Interrupt Group 4 Flag Register		Go
Ah	PIEIER5	Interrupt Group 5 Enable Register		Go
Bh	PIEIFR5	Interrupt Group 5 Flag Register		Go
Ch	PIEIER6	Interrupt Group 6 Enable Register		Go
Dh	PIEIFR6	Interrupt Group 6 Flag Register		Go
Eh	PIEIER7	Interrupt Group 7 Enable Register		Go
Fh	PIEIFR7	Interrupt Group 7 Flag Register		Go
10h	PIEIER8	Interrupt Group 8 Enable Register		Go
11h	PIEIFR8	Interrupt Group 8 Flag Register		Go
12h	PIEIER9	Interrupt Group 9 Enable Register		Go
13h	PIEIFR9	Interrupt Group 9 Flag Register		Go
14h	PIEIER10	Interrupt Group 10 Enable Register		Go
15h	PIEIFR10	Interrupt Group 10 Flag Register		Go
16h	PIEIER11	Interrupt Group 11 Enable Register		Go
17h	PIEIFR11	Interrupt Group 11 Flag Register		Go
18h	PIEIER12	Interrupt Group 12 Enable Register		Go
19h	PIEIFR12	Interrupt Group 12 Flag Register		Go

Complex bit access types are encoded to fit into small table cells. Table 3-257 shows the codes that are used for access types in this section.

Table 3-257. PIE_CTRL_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		

Table 3-257. PIE_CTRL_REGS Access Type Codes (continued)

Access Type	Code	Description
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.15.12.1 PIECTRL Register (Offset = 0h) [Reset = 0h]

PIECTRL is shown in Figure 3-239 and described in Table 3-258.

Return to the Summary Table.

ePIE Control Register

Figure 3-239. PIECTRL Register

15	14	13	12	11	10	9	8
PIEVECT							
R-0h							
7	6	5	4	3	2	1	0
PIEVECT							ENPIE
R-0h							R/W-0h

Table 3-258. PIECTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	PIEVECT	R	0h	These bits indicate the vector address of the vector fetched from the ePIE vector table. The least significant bit of the address is ignored and only bits 1 to 15 of the address are shown. The vector value can be read by the user to determine which interrupt generated the vector fetch. Note: When a NMI is serviced, the PIEVECT bit-field does not reflect the vector as it does for other interrupts. Reset type: SYSRSn
0	ENPIE	R/W	0h	Enable vector fetching from ePIE block. This bit must be set to 1 for peripheral interrupts to work. All ePIE registers (PIEACK, PIEIFR, PIEIER) can be accessed even when the ePIE block is disabled. Reset type: SYSRSn

3.15.12.2 PIEACK Register (Offset = 1h) [Reset = 0h]

PIEACK is shown in Figure 3-240 and described in Table 3-259.

Return to the Summary Table.

Acknowledge Register
When an interrupt propagates from the ePIE to a CPU interrupt line, the interrupt group's PIEACK bit is set. This prevents other interrupts in that group from propagating to the CPU while the first interrupt is handled. Writing a 1 to a PIEACK bit clears it and allows another interrupt from the corresponding group to propagate. ISRs for PIE interrupts should clear the group's PIEACK bit before returning from the interrupt.
Writes of 0 are ignored.

Figure 3-240. PIEACK Register

15	14	13	12	11	10	9	8
RESERVED				ACK12	ACK11	ACK10	ACK9
R-0h				R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
7	6	5	4	3	2	1	0
ACK8	ACK7	ACK6	ACK5	ACK4	ACK3	ACK2	ACK1
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

Table 3-259. PIEACK Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11	ACK12	R/W1S	0h	Acknowledge PIE Interrupt Group 12 Reset type: SYSRSn
10	ACK11	R/W1S	0h	Acknowledge PIE Interrupt Group 11 Reset type: SYSRSn
9	ACK10	R/W1S	0h	Acknowledge PIE Interrupt Group 10 Reset type: SYSRSn
8	ACK9	R/W1S	0h	Acknowledge PIE Interrupt Group 9 Reset type: SYSRSn
7	ACK8	R/W1S	0h	Acknowledge PIE Interrupt Group 8 Reset type: SYSRSn
6	ACK7	R/W1S	0h	Acknowledge PIE Interrupt Group 7 Reset type: SYSRSn
5	ACK6	R/W1S	0h	Acknowledge PIE Interrupt Group 6 Reset type: SYSRSn
4	ACK5	R/W1S	0h	Acknowledge PIE Interrupt Group 5 Reset type: SYSRSn
3	ACK4	R/W1S	0h	Acknowledge PIE Interrupt Group 4 Reset type: SYSRSn
2	ACK3	R/W1S	0h	Acknowledge PIE Interrupt Group 3 Reset type: SYSRSn
1	ACK2	R/W1S	0h	Acknowledge PIE Interrupt Group 2 Reset type: SYSRSn
0	ACK1	R/W1S	0h	Acknowledge PIE Interrupt Group 1 Reset type: SYSRSn

3.15.12.3 PIEIER1 Register (Offset = 2h) [Reset = 0h]

PIEIER1 is shown in Figure 3-241 and described in Table 3-260.

Return to the Summary Table.

Interrupt Group 1 Enable Register
These register bits individually enable an interrupt within a group. They behave very much like the bits in the CPU interrupt enable register (IER).
Setting a bit to 1 allows the corresponding interrupt to propagate to the CPU.
Setting a bit to 0 prevents the corresponding interrupt from propagating. Note that a peripheral interrupt signal can still set the PIEIFR bit for the disabled interrupt.

Figure 3-241. PIEIER1 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-260. PIEIER1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Enable for Interrupt 1.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Enable for Interrupt 1.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Enable for Interrupt 1.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Enable for Interrupt 1.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Enable for Interrupt 1.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Enable for Interrupt 1.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Enable for Interrupt 1.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Enable for Interrupt 1.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Enable for Interrupt 1.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Enable for Interrupt 1.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Enable for Interrupt 1.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Enable for Interrupt 1.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Enable for Interrupt 1.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Enable for Interrupt 1.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Enable for Interrupt 1.2 Reset type: SYSRSn
0	INTx1	R/W	0h	Enable for Interrupt 1.1 Reset type: SYSRSn

3.15.12.4 PIEIFR1 Register (Offset = 3h) [Reset = 0h]

PIEIFR1 is shown in Figure 3-242 and described in Table 3-261.

Return to the Summary Table.

Interrupt Group 1 Flag Register
These register bits indicate whether each interrupt in the group is currently pending. They behave very much like the bits in the CPU interrupt flag register (IFR).
When a peripheral sends an interrupt, the corresponding bit is set. This bit is cleared when the interrupt propagates to the CPU, at which point PIEACK is set.
NOTE: PIE IFR flags can be written to create software interrupts.
The IFR flag will be cleared on a write of zero. Hence, when the intent is to fire an interrupt it may cause inadvertent cancellation of other interrupts. It is recommended to use this only for testing or with extreme caution in the application code. Reading the PIE IFR registers is safe.

Figure 3-242. PIEIFR1 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-261. PIEIFR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Flag for Interrupt 1.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Flag for Interrupt 1.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Flag for Interrupt 1.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Flag for Interrupt 1.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Flag for Interrupt 1.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Flag for Interrupt 1.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Flag for Interrupt 1.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Flag for Interrupt 1.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Flag for Interrupt 1.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Flag for Interrupt 1.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Flag for Interrupt 1.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Flag for Interrupt 1.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Flag for Interrupt 1.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Flag for Interrupt 1.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Flag for Interrupt 1.2 Reset type: SYSRSn

Table 3-261. PIEIFR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INTx1	R/W	0h	Flag for Interrupt 1.1 Reset type: SYSRSn

3.15.12.5 PIEIER2 Register (Offset = 4h) [Reset = 0h]

PIEIER2 is shown in Figure 3-243 and described in Table 3-262.

Return to the Summary Table.

Interrupt Group 2 Enable Register
These register bits individually enable an interrupt within a group. They behave very much like the bits in the CPU interrupt enable register (IER).
Setting a bit to 1 allows the corresponding interrupt to propagate to the CPU.
Setting a bit to 0 prevents the corresponding interrupt from propagating. Note that a peripheral interrupt signal can still set the PIEIFR bit for the disabled interrupt.

Figure 3-243. PIEIER2 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-262. PIEIER2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Enable for Interrupt 2.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Enable for Interrupt 2.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Enable for Interrupt 2.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Enable for Interrupt 2.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Enable for Interrupt 2.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Enable for Interrupt 2.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Enable for Interrupt 2.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Enable for Interrupt 2.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Enable for Interrupt 2.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Enable for Interrupt 2.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Enable for Interrupt 2.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Enable for Interrupt 2.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Enable for Interrupt 2.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Enable for Interrupt 2.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Enable for Interrupt 2.2 Reset type: SYSRSn
0	INTx1	R/W	0h	Enable for Interrupt 2.1 Reset type: SYSRSn

3.15.12.6 PIEIFR2 Register (Offset = 5h) [Reset = 0h]

PIEIFR2 is shown in Figure 3-244 and described in Table 3-263.

Return to the Summary Table.

Interrupt Group 2 Flag Register
These register bits indicate whether each interrupt in the group is currently pending. They behave very much like the bits in the CPU interrupt flag register (IFR).
When a peripheral sends an interrupt, the corresponding bit is set. This bit is cleared when the interrupt propagates to the CPU, at which point PIEACK is set.
NOTE: PIE IFR flags can be written to create software interrupts.
The IFR flag will be cleared on a write of zero. Hence, when the intent is to fire an interrupt it may cause inadvertent cancellation of other interrupts. It is recommended to use this only for testing or with extreme caution in the application code. Reading the PIE IFR registers is safe.

Figure 3-244. PIEIFR2 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-263. PIEIFR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Flag for Interrupt 2.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Flag for Interrupt 2.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Flag for Interrupt 2.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Flag for Interrupt 2.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Flag for Interrupt 2.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Flag for Interrupt 2.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Flag for Interrupt 2.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Flag for Interrupt 2.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Flag for Interrupt 2.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Flag for Interrupt 2.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Flag for Interrupt 2.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Flag for Interrupt 2.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Flag for Interrupt 2.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Flag for Interrupt 2.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Flag for Interrupt 2.2 Reset type: SYSRSn

Table 3-263. PIEIFR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INTx1	R/W	0h	Flag for Interrupt 2.1 Reset type: SYSRSn

3.15.12.7 PIEIER3 Register (Offset = 6h) [Reset = 0h]

PIEIER3 is shown in Figure 3-245 and described in Table 3-264.

Return to the Summary Table.

Interrupt Group 3 Enable Register
These register bits individually enable an interrupt within a group. They behave very much like the bits in the CPU interrupt enable register (IER).
Setting a bit to 1 allows the corresponding interrupt to propagate to the CPU.
Setting a bit to 0 prevents the corresponding interrupt from propagating. Note that a peripheral interrupt signal can still set the PIEIFR bit for the disabled interrupt.

Figure 3-245. PIEIER3 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-264. PIEIER3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Enable for Interrupt 3.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Enable for Interrupt 3.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Enable for Interrupt 3.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Enable for Interrupt 3.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Enable for Interrupt 3.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Enable for Interrupt 3.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Enable for Interrupt 3.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Enable for Interrupt 3.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Enable for Interrupt 3.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Enable for Interrupt 3.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Enable for Interrupt 3.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Enable for Interrupt 3.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Enable for Interrupt 3.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Enable for Interrupt 3.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Enable for Interrupt 3.2 Reset type: SYSRSn
0	INTx1	R/W	0h	Enable for Interrupt 3.1 Reset type: SYSRSn

3.15.12.8 PIEIFR3 Register (Offset = 7h) [Reset = 0h]

PIEIFR3 is shown in Figure 3-246 and described in Table 3-265.

Return to the Summary Table.

Interrupt Group 3 Flag Register
These register bits indicate whether each interrupt in the group is currently pending. They behave very much like the bits in the CPU interrupt flag register (IFR).
When a peripheral sends an interrupt, the corresponding bit is set. This bit is cleared when the interrupt propagates to the CPU, at which point PIEACK is set.
NOTE: PIE IFR flags can be written to create software interrupts.
The IFR flag will be cleared on a write of zero. Hence, when the intent is to fire an interrupt it may cause inadvertent cancellation of other interrupts. It is recommended to use this only for testing or with extreme caution in the application code. Reading the PIE IFR registers is safe.

Figure 3-246. PIEIFR3 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-265. PIEIFR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Flag for Interrupt 3.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Flag for Interrupt 3.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Flag for Interrupt 3.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Flag for Interrupt 3.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Flag for Interrupt 3.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Flag for Interrupt 3.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Flag for Interrupt 3.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Flag for Interrupt 3.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Flag for Interrupt 3.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Flag for Interrupt 3.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Flag for Interrupt 3.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Flag for Interrupt 3.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Flag for Interrupt 3.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Flag for Interrupt 3.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Flag for Interrupt 3.2 Reset type: SYSRSn

Table 3-265. PIEIFR3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INTx1	R/W	0h	Flag for Interrupt 3.1 Reset type: SYSRSn

3.15.12.9 PIEIER4 Register (Offset = 8h) [Reset = 0h]

PIEIER4 is shown in Figure 3-247 and described in Table 3-266.

Return to the Summary Table.

Interrupt Group 4 Enable Register
These register bits individually enable an interrupt within a group. They behave very much like the bits in the CPU interrupt enable register (IER).
Setting a bit to 1 allows the corresponding interrupt to propagate to the CPU.
Setting a bit to 0 prevents the corresponding interrupt from propagating. Note that a peripheral interrupt signal can still set the PIEIFR bit for the disabled interrupt.

Figure 3-247. PIEIER4 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-266. PIEIER4 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Enable for Interrupt 4.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Enable for Interrupt 4.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Enable for Interrupt 4.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Enable for Interrupt 4.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Enable for Interrupt 4.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Enable for Interrupt 4.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Enable for Interrupt 4.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Enable for Interrupt 4.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Enable for Interrupt 4.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Enable for Interrupt 4.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Enable for Interrupt 4.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Enable for Interrupt 4.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Enable for Interrupt 4.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Enable for Interrupt 4.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Enable for Interrupt 4.2 Reset type: SYSRSn
0	INTx1	R/W	0h	Enable for Interrupt 4.1 Reset type: SYSRSn

3.15.12.10 PIEIFR4 Register (Offset = 9h) [Reset = 0h]

PIEIFR4 is shown in Figure 3-248 and described in Table 3-267.

Return to the Summary Table.

Interrupt Group 4 Flag Register
These register bits indicate whether each interrupt in the group is currently pending. They behave very much like the bits in the CPU interrupt flag register (IFR).
When a peripheral sends an interrupt, the corresponding bit is set. This bit is cleared when the interrupt propagates to the CPU, at which point PIEACK is set.
NOTE: PIE IFR flags can be written to create software interrupts.
The IFR flag will be cleared on a write of zero. Hence, when the intent is to fire an interrupt it may cause inadvertent cancellation of other interrupts. It is recommended to use this only for testing or with extreme caution in the application code. Reading the PIE IFR registers is safe.

Figure 3-248. PIEIFR4 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-267. PIEIFR4 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Flag for Interrupt 4.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Flag for Interrupt 4.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Flag for Interrupt 4.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Flag for Interrupt 4.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Flag for Interrupt 4.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Flag for Interrupt 4.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Flag for Interrupt 4.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Flag for Interrupt 4.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Flag for Interrupt 4.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Flag for Interrupt 4.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Flag for Interrupt 4.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Flag for Interrupt 4.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Flag for Interrupt 4.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Flag for Interrupt 4.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Flag for Interrupt 4.2 Reset type: SYSRSn

Table 3-267. PIEIFR4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INTx1	R/W	0h	Flag for Interrupt 4.1 Reset type: SYSRSn

3.15.12.11 PIEIER5 Register (Offset = Ah) [Reset = 0h]

PIEIER5 is shown in Figure 3-249 and described in Table 3-268.

Return to the Summary Table.

Interrupt Group 5 Enable Register
These register bits individually enable an interrupt within a group. They behave very much like the bits in the CPU interrupt enable register (IER).
Setting a bit to 1 allows the corresponding interrupt to propagate to the CPU.
Setting a bit to 0 prevents the corresponding interrupt from propagating. Note that a peripheral interrupt signal can still set the PIEIFR bit for the disabled interrupt.

Figure 3-249. PIEIER5 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-268. PIEIER5 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Enable for Interrupt 5.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Enable for Interrupt 5.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Enable for Interrupt 5.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Enable for Interrupt 5.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Enable for Interrupt 5.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Enable for Interrupt 5.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Enable for Interrupt 5.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Enable for Interrupt 5.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Enable for Interrupt 5.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Enable for Interrupt 5.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Enable for Interrupt 5.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Enable for Interrupt 5.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Enable for Interrupt 5.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Enable for Interrupt 5.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Enable for Interrupt 5.2 Reset type: SYSRSn
0	INTx1	R/W	0h	Enable for Interrupt 5.1 Reset type: SYSRSn

3.15.12.12 PIEIFR5 Register (Offset = Bh) [Reset = 0h]

PIEIFR5 is shown in Figure 3-250 and described in Table 3-269.

Return to the Summary Table.

Interrupt Group 5 Flag Register
These register bits indicate whether each interrupt in the group is currently pending. They behave very much like the bits in the CPU interrupt flag register (IFR).
When a peripheral sends an interrupt, the corresponding bit is set. This bit is cleared when the interrupt propagates to the CPU, at which point PIEACK is set.
NOTE: PIE IFR flags can be written to create software interrupts.
The IFR flag will be cleared on a write of zero. Hence, when the intent is to fire an interrupt it may cause inadvertent cancellation of other interrupts. It is recommended to use this only for testing or with extreme caution in the application code. Reading the PIE IFR registers is safe.

Figure 3-250. PIEIFR5 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-269. PIEIFR5 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Flag for Interrupt 5.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Flag for Interrupt 5.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Flag for Interrupt 5.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Flag for Interrupt 5.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Flag for Interrupt 5.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Flag for Interrupt 5.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Flag for Interrupt 5.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Flag for Interrupt 5.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Flag for Interrupt 5.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Flag for Interrupt 5.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Flag for Interrupt 5.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Flag for Interrupt 5.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Flag for Interrupt 5.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Flag for Interrupt 5.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Flag for Interrupt 5.2 Reset type: SYSRSn

Table 3-269. PIEIFR5 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INTx1	R/W	0h	Flag for Interrupt 5.1 Reset type: SYSRSn

3.15.12.13 PIEIER6 Register (Offset = Ch) [Reset = 0h]

PIEIER6 is shown in Figure 3-251 and described in Table 3-270.

Return to the Summary Table.

Interrupt Group 6 Enable Register
These register bits individually enable an interrupt within a group. They behave very much like the bits in the CPU interrupt enable register (IER).
Setting a bit to 1 allows the corresponding interrupt to propagate to the CPU.
Setting a bit to 0 prevents the corresponding interrupt from propagating. Note that a peripheral interrupt signal can still set the PIEIFR bit for the disabled interrupt.

Figure 3-251. PIEIER6 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-270. PIEIER6 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Enable for Interrupt 6.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Enable for Interrupt 6.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Enable for Interrupt 6.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Enable for Interrupt 6.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Enable for Interrupt 6.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Enable for Interrupt 6.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Enable for Interrupt 6.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Enable for Interrupt 6.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Enable for Interrupt 6.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Enable for Interrupt 6.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Enable for Interrupt 6.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Enable for Interrupt 6.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Enable for Interrupt 6.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Enable for Interrupt 6.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Enable for Interrupt 6.2 Reset type: SYSRSn
0	INTx1	R/W	0h	Enable for Interrupt 6.1 Reset type: SYSRSn

3.15.12.14 PIEIFR6 Register (Offset = Dh) [Reset = 0h]

PIEIFR6 is shown in Figure 3-252 and described in Table 3-271.

Return to the Summary Table.

Interrupt Group 6 Flag Register
These register bits indicate whether each interrupt in the group is currently pending. They behave very much like the bits in the CPU interrupt flag register (IFR).
When a peripheral sends an interrupt, the corresponding bit is set. This bit is cleared when the interrupt propagates to the CPU, at which point PIEACK is set.
NOTE: PIE IFR flags can be written to create software interrupts.
The IFR flag will be cleared on a write of zero. Hence, when the intent is to fire an interrupt it may cause inadvertent cancellation of other interrupts. It is recommended to use this only for testing or with extreme caution in the application code. Reading the PIE IFR registers is safe.

Figure 3-252. PIEIFR6 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-271. PIEIFR6 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Flag for Interrupt 6.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Flag for Interrupt 6.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Flag for Interrupt 6.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Flag for Interrupt 6.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Flag for Interrupt 6.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Flag for Interrupt 6.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Flag for Interrupt 6.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Flag for Interrupt 6.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Flag for Interrupt 6.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Flag for Interrupt 6.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Flag for Interrupt 6.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Flag for Interrupt 6.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Flag for Interrupt 6.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Flag for Interrupt 6.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Flag for Interrupt 6.2 Reset type: SYSRSn

Table 3-271. PIEIFR6 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INTx1	R/W	0h	Flag for Interrupt 6.1 Reset type: SYSRSn

3.15.12.15 PIEIER7 Register (Offset = Eh) [Reset = 0h]

PIEIER7 is shown in Figure 3-253 and described in Table 3-272.

Return to the Summary Table.

Interrupt Group 7 Enable Register
These register bits individually enable an interrupt within a group. They behave very much like the bits in the CPU interrupt enable register (IER).
Setting a bit to 1 allows the corresponding interrupt to propagate to the CPU.
Setting a bit to 0 prevents the corresponding interrupt from propagating. Note that a peripheral interrupt signal can still set the PIEIFR bit for the disabled interrupt.

Figure 3-253. PIEIER7 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-272. PIEIER7 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Enable for Interrupt 7.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Enable for Interrupt 7.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Enable for Interrupt 7.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Enable for Interrupt 7.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Enable for Interrupt 7.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Enable for Interrupt 7.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Enable for Interrupt 7.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Enable for Interrupt 7.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Enable for Interrupt 7.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Enable for Interrupt 7.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Enable for Interrupt 7.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Enable for Interrupt 7.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Enable for Interrupt 7.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Enable for Interrupt 7.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Enable for Interrupt 7.2 Reset type: SYSRSn
0	INTx1	R/W	0h	Enable for Interrupt 7.1 Reset type: SYSRSn

3.15.12.16 PIEIFR7 Register (Offset = Fh) [Reset = 0h]

PIEIFR7 is shown in Figure 3-254 and described in Table 3-273.

Return to the Summary Table.

Interrupt Group 7 Flag Register
These register bits indicate whether each interrupt in the group is currently pending. They behave very much like the bits in the CPU interrupt flag register (IFR).
When a peripheral sends an interrupt, the corresponding bit is set. This bit is cleared when the interrupt propagates to the CPU, at which point PIEACK is set.
NOTE: PIE IFR flags can be written to create software interrupts.
The IFR flag will be cleared on a write of zero. Hence, when the intent is to fire an interrupt it may cause inadvertent cancellation of other interrupts. It is recommended to use this only for testing or with extreme caution in the application code. Reading the PIE IFR registers is safe.

Figure 3-254. PIEIFR7 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-273. PIEIFR7 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Flag for Interrupt 7.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Flag for Interrupt 7.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Flag for Interrupt 7.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Flag for Interrupt 7.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Flag for Interrupt 7.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Flag for Interrupt 7.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Flag for Interrupt 7.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Flag for Interrupt 7.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Flag for Interrupt 7.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Flag for Interrupt 7.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Flag for Interrupt 7.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Flag for Interrupt 7.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Flag for Interrupt 7.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Flag for Interrupt 7.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Flag for Interrupt 7.2 Reset type: SYSRSn

Table 3-273. PIEIFR7 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INTx1	R/W	0h	Flag for Interrupt 7.1 Reset type: SYSRSn

3.15.12.17 PIEIER8 Register (Offset = 10h) [Reset = 0h]

PIEIER8 is shown in Figure 3-255 and described in Table 3-274.

Return to the Summary Table.

Interrupt Group 8 Enable Register
These register bits individually enable an interrupt within a group. They behave very much like the bits in the CPU interrupt enable register (IER).
Setting a bit to 1 allows the corresponding interrupt to propagate to the CPU.
Setting a bit to 0 prevents the corresponding interrupt from propagating. Note that a peripheral interrupt signal can still set the PIEIFR bit for the disabled interrupt.

Figure 3-255. PIEIER8 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-274. PIEIER8 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Enable for Interrupt 8.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Enable for Interrupt 8.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Enable for Interrupt 8.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Enable for Interrupt 8.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Enable for Interrupt 8.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Enable for Interrupt 8.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Enable for Interrupt 8.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Enable for Interrupt 8.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Enable for Interrupt 8.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Enable for Interrupt 8.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Enable for Interrupt 8.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Enable for Interrupt 8.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Enable for Interrupt 8.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Enable for Interrupt 8.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Enable for Interrupt 8.2 Reset type: SYSRSn
0	INTx1	R/W	0h	Enable for Interrupt 8.1 Reset type: SYSRSn

3.15.12.18 PIEIFR8 Register (Offset = 11h) [Reset = 0h]

PIEIFR8 is shown in Figure 3-256 and described in Table 3-275.

Return to the Summary Table.

Interrupt Group 8 Flag Register
These register bits indicate whether each interrupt in the group is currently pending. They behave very much like the bits in the CPU interrupt flag register (IFR).
When a peripheral sends an interrupt, the corresponding bit is set. This bit is cleared when the interrupt propagates to the CPU, at which point PIEACK is set.
NOTE: PIE IFR flags can be written to create software interrupts.
The IFR flag will be cleared on a write of zero. Hence, when the intent is to fire an interrupt it may cause inadvertent cancellation of other interrupts. It is recommended to use this only for testing or with extreme caution in the application code. Reading the PIE IFR registers is safe.

Figure 3-256. PIEIFR8 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-275. PIEIFR8 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Flag for Interrupt 8.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Flag for Interrupt 8.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Flag for Interrupt 8.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Flag for Interrupt 8.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Flag for Interrupt 8.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Flag for Interrupt 8.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Flag for Interrupt 8.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Flag for Interrupt 8.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Flag for Interrupt 8.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Flag for Interrupt 8.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Flag for Interrupt 8.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Flag for Interrupt 8.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Flag for Interrupt 8.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Flag for Interrupt 8.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Flag for Interrupt 8.2 Reset type: SYSRSn

Table 3-275. PIEIFR8 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INTx1	R/W	0h	Flag for Interrupt 8.1 Reset type: SYSRSn

3.15.12.19 PIEIER9 Register (Offset = 12h) [Reset = 0h]

PIEIER9 is shown in Figure 3-257 and described in Table 3-276.

Return to the Summary Table.

Interrupt Group 9 Enable Register
These register bits individually enable an interrupt within a group. They behave very much like the bits in the CPU interrupt enable register (IER).
Setting a bit to 1 allows the corresponding interrupt to propagate to the CPU.
Setting a bit to 0 prevents the corresponding interrupt from propagating. Note that a peripheral interrupt signal can still set the PIEIFR bit for the disabled interrupt.

Figure 3-257. PIEIER9 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-276. PIEIER9 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Enable for Interrupt 9.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Enable for Interrupt 9.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Enable for Interrupt 9.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Enable for Interrupt 9.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Enable for Interrupt 9.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Enable for Interrupt 9.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Enable for Interrupt 9.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Enable for Interrupt 9.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Enable for Interrupt 9.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Enable for Interrupt 9.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Enable for Interrupt 9.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Enable for Interrupt 9.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Enable for Interrupt 9.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Enable for Interrupt 9.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Enable for Interrupt 9.2 Reset type: SYSRSn
0	INTx1	R/W	0h	Enable for Interrupt 9.1 Reset type: SYSRSn

3.15.12.20 PIEIFR9 Register (Offset = 13h) [Reset = 0h]

PIEIFR9 is shown in Figure 3-258 and described in Table 3-277.

Return to the Summary Table.

Interrupt Group 9 Flag Register
These register bits indicate whether each interrupt in the group is currently pending. They behave very much like the bits in the CPU interrupt flag register (IFR).
When a peripheral sends an interrupt, the corresponding bit is set. This bit is cleared when the interrupt propagates to the CPU, at which point PIEACK is set.
NOTE: PIE IFR flags can be written to create software interrupts.
The IFR flag will be cleared on a write of zero. Hence, when the intent is to fire an interrupt it may cause inadvertent cancellation of other interrupts. It is recommended to use this only for testing or with extreme caution in the application code. Reading the PIE IFR registers is safe.

Figure 3-258. PIEIFR9 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-277. PIEIFR9 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Flag for Interrupt 9.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Flag for Interrupt 9.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Flag for Interrupt 9.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Flag for Interrupt 9.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Flag for Interrupt 9.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Flag for Interrupt 9.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Flag for Interrupt 9.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Flag for Interrupt 9.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Flag for Interrupt 9.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Flag for Interrupt 9.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Flag for Interrupt 9.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Flag for Interrupt 9.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Flag for Interrupt 9.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Flag for Interrupt 9.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Flag for Interrupt 9.2 Reset type: SYSRSn

Table 3-277. PIEIFR9 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INTx1	R/W	0h	Flag for Interrupt 9.1 Reset type: SYSRSn

3.15.12.21 PIEIER10 Register (Offset = 14h) [Reset = 0h]

PIEIER10 is shown in Figure 3-259 and described in Table 3-278.

Return to the Summary Table.

Interrupt Group 10 Enable Register
These register bits individually enable an interrupt within a group. They behave very much like the bits in the CPU interrupt enable register (IER).
Setting a bit to 1 allows the corresponding interrupt to propagate to the CPU.
Setting a bit to 0 prevents the corresponding interrupt from propagating. Note that a peripheral interrupt signal can still set the PIEIFR bit for the disabled interrupt.

Figure 3-259. PIEIER10 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-278. PIEIER10 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Enable for Interrupt 10.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Enable for Interrupt 10.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Enable for Interrupt 10.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Enable for Interrupt 10.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Enable for Interrupt 10.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Enable for Interrupt 10.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Enable for Interrupt 10.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Enable for Interrupt 10.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Enable for Interrupt 10.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Enable for Interrupt 10.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Enable for Interrupt 10.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Enable for Interrupt 10.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Enable for Interrupt 10.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Enable for Interrupt 10.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Enable for Interrupt 10.2 Reset type: SYSRSn
0	INTx1	R/W	0h	Enable for Interrupt 10.1 Reset type: SYSRSn

3.15.12.22 PIEIFR10 Register (Offset = 15h) [Reset = 0h]

PIEIFR10 is shown in Figure 3-260 and described in Table 3-279.

Return to the Summary Table.

Interrupt Group 10 Flag Register
These register bits indicate whether each interrupt in the group is currently pending. They behave very much like the bits in the CPU interrupt flag register (IFR).
When a peripheral sends an interrupt, the corresponding bit is set. This bit is cleared when the interrupt propagates to the CPU, at which point PIEACK is set.
NOTE: PIE IFR flags can be written to create software interrupts.
The IFR flag will be cleared on a write of zero. Hence, when the intent is to fire an interrupt it may cause inadvertent cancellation of other interrupts. It is recommended to use this only for testing or with extreme caution in the application code. Reading the PIE IFR registers is safe.

Figure 3-260. PIEIFR10 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-279. PIEIFR10 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Flag for Interrupt 10.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Flag for Interrupt 10.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Flag for Interrupt 10.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Flag for Interrupt 10.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Flag for Interrupt 10.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Flag for Interrupt 10.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Flag for Interrupt 10.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Flag for Interrupt 10.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Flag for Interrupt 10.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Flag for Interrupt 10.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Flag for Interrupt 10.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Flag for Interrupt 10.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Flag for Interrupt 10.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Flag for Interrupt 10.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Flag for Interrupt 10.2 Reset type: SYSRSn

Table 3-279. PIEIFR10 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INTx1	R/W	0h	Flag for Interrupt 10.1 Reset type: SYSRSn

3.15.12.23 PIEIER11 Register (Offset = 16h) [Reset = 0h]

PIEIER11 is shown in Figure 3-261 and described in Table 3-280.

Return to the Summary Table.

Interrupt Group 11 Enable Register
These register bits individually enable an interrupt within a group. They behave very much like the bits in the CPU interrupt enable register (IER).
Setting a bit to 1 allows the corresponding interrupt to propagate to the CPU.
Setting a bit to 0 prevents the corresponding interrupt from propagating. Note that a peripheral interrupt signal can still set the PIEIFR bit for the disabled interrupt.

Figure 3-261. PIEIER11 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-280. PIEIER11 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Enable for Interrupt 11.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Enable for Interrupt 11.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Enable for Interrupt 11.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Enable for Interrupt 11.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Enable for Interrupt 11.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Enable for Interrupt 11.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Enable for Interrupt 11.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Enable for Interrupt 11.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Enable for Interrupt 11.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Enable for Interrupt 11.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Enable for Interrupt 11.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Enable for Interrupt 11.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Enable for Interrupt 11.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Enable for Interrupt 11.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Enable for Interrupt 11.2 Reset type: SYSRSn
0	INTx1	R/W	0h	Enable for Interrupt 11.1 Reset type: SYSRSn

3.15.12.24 PIEIFR11 Register (Offset = 17h) [Reset = 0h]

PIEIFR11 is shown in Figure 3-262 and described in Table 3-281.

Return to the Summary Table.

Interrupt Group 11 Flag Register
These register bits indicate whether each interrupt in the group is currently pending. They behave very much like the bits in the CPU interrupt flag register (IFR).
When a peripheral sends an interrupt, the corresponding bit is set. This bit is cleared when the interrupt propagates to the CPU, at which point PIEACK is set.
NOTE: PIE IFR flags can be written to create software interrupts.
The IFR flag will be cleared on a write of zero. Hence, when the intent is to fire an interrupt it may cause inadvertent cancellation of other interrupts. It is recommended to use this only for testing or with extreme caution in the application code. Reading the PIE IFR registers is safe.

Figure 3-262. PIEIFR11 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-281. PIEIFR11 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Flag for Interrupt 11.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Flag for Interrupt 11.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Flag for Interrupt 11.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Flag for Interrupt 11.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Flag for Interrupt 11.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Flag for Interrupt 11.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Flag for Interrupt 11.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Flag for Interrupt 11.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Flag for Interrupt 11.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Flag for Interrupt 11.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Flag for Interrupt 11.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Flag for Interrupt 11.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Flag for Interrupt 11.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Flag for Interrupt 11.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Flag for Interrupt 11.2 Reset type: SYSRSn

Table 3-281. PIEIFR11 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INTx1	R/W	0h	Flag for Interrupt 11.1 Reset type: SYSRSn

3.15.12.25 PIEIER12 Register (Offset = 18h) [Reset = 0h]

PIEIER12 is shown in Figure 3-263 and described in Table 3-282.

Return to the Summary Table.

Interrupt Group 12 Enable Register
These register bits individually enable an interrupt within a group. They behave very much like the bits in the CPU interrupt enable register (IER).
Setting a bit to 1 allows the corresponding interrupt to propagate to the CPU.
Setting a bit to 0 prevents the corresponding interrupt from propagating. Note that a peripheral interrupt signal can still set the PIEIFR bit for the disabled interrupt.

Figure 3-263. PIEIER12 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-282. PIEIER12 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Enable for Interrupt 12.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Enable for Interrupt 12.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Enable for Interrupt 12.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Enable for Interrupt 12.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Enable for Interrupt 12.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Enable for Interrupt 12.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Enable for Interrupt 12.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Enable for Interrupt 12.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Enable for Interrupt 12.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Enable for Interrupt 12.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Enable for Interrupt 12.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Enable for Interrupt 12.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Enable for Interrupt 12.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Enable for Interrupt 12.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Enable for Interrupt 12.2 Reset type: SYSRSn
0	INTx1	R/W	0h	Enable for Interrupt 12.1 Reset type: SYSRSn

3.15.12.26 PIEIFR12 Register (Offset = 19h) [Reset = 0h]

PIEIFR12 is shown in Figure 3-264 and described in Table 3-283.

Return to the Summary Table.

Interrupt Group 12 Flag Register
These register bits indicate whether each interrupt in the group is currently pending. They behave very much like the bits in the CPU interrupt flag register (IFR).
When a peripheral sends an interrupt, the corresponding bit is set. This bit is cleared when the interrupt propagates to the CPU, at which point PIEACK is set.
NOTE: PIE IFR flags can be written to create software interrupts.
The IFR flag will be cleared on a write of zero. Hence, when the intent is to fire an interrupt it may cause inadvertent cancellation of other interrupts. It is recommended to use this only for testing or with extreme caution in the application code. Reading the PIE IFR registers is safe.

Figure 3-264. PIEIFR12 Register

15	14	13	12	11	10	9	8
INTx16	INTx15	INTx14	INTx13	INTx12	INTx11	INTx10	INTx9
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INTx8	INTx7	INTx6	INTx5	INTx4	INTx3	INTx2	INTx1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-283. PIEIFR12 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	INTx16	R/W	0h	Flag for Interrupt 12.16 Reset type: SYSRSn
14	INTx15	R/W	0h	Flag for Interrupt 12.15 Reset type: SYSRSn
13	INTx14	R/W	0h	Flag for Interrupt 12.14 Reset type: SYSRSn
12	INTx13	R/W	0h	Flag for Interrupt 12.13 Reset type: SYSRSn
11	INTx12	R/W	0h	Flag for Interrupt 12.12 Reset type: SYSRSn
10	INTx11	R/W	0h	Flag for Interrupt 12.11 Reset type: SYSRSn
9	INTx10	R/W	0h	Flag for Interrupt 12.10 Reset type: SYSRSn
8	INTx9	R/W	0h	Flag for Interrupt 12.9 Reset type: SYSRSn
7	INTx8	R/W	0h	Flag for Interrupt 12.8 Reset type: SYSRSn
6	INTx7	R/W	0h	Flag for Interrupt 12.7 Reset type: SYSRSn
5	INTx6	R/W	0h	Flag for Interrupt 12.6 Reset type: SYSRSn
4	INTx5	R/W	0h	Flag for Interrupt 12.5 Reset type: SYSRSn
3	INTx4	R/W	0h	Flag for Interrupt 12.4 Reset type: SYSRSn
2	INTx3	R/W	0h	Flag for Interrupt 12.3 Reset type: SYSRSn
1	INTx2	R/W	0h	Flag for Interrupt 12.2 Reset type: SYSRSn

Table 3-283. PIEIFR12 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INTx1	R/W	0h	Flag for Interrupt 12.1 Reset type: SYSRSn

3.15.13 SYNC_SOC_REGS Registers

Table 3-284 lists the memory-mapped registers for the SYNC_SOC_REGS registers. All register offset addresses not listed in Table 3-284 should be considered as reserved locations and the register contents should not be modified.

Table 3-284. SYNC_SOC_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	SYNCSELECT	Sync Input and Output Select Register	EALLOW	Go
2h	ADCSOCOUTSELECT	External ADCSOC Select Register	EALLOW	Go
4h	SYNCSOCLOCK	SYNCSEL and EXTADCSOC Select Lock register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 3-285 shows the codes that are used for access types in this section.

Table 3-285. SYNC_SOC_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.15.13.1 SYNCSELECT Register (Offset = 0h) [Reset = E003FFFFh]

SYNCSELECT is shown in Figure 3-265 and described in Table 3-286.

Return to the Summary Table.

Sync Input and Output Select Register

Figure 3-265. SYNCSELECT Register

31	30	29	28	27	26	25	24
RESERVED			SYNCOUT				
R/W-7h			R/W-0h				
23	22	21	20	19	18	17	16
RESERVED						RESERVED	
R-0h						R/W-7h	
15	14	13	12	11	10	9	8
RESERVED	RESERVED			RESERVED			RESERVED
R/W-7h	R/W-7h			R/W-7h			R/W-7h
7	6	5	4	3	2	1	0
RESERVED		RESERVED			RESERVED		
R/W-7h		R/W-7h			R/W-7h		

Table 3-286. SYNCSELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R/W	7h	Reserved

Table 3-286. SYNCSELECT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
28-24	SYNCOUT	R/W	0h	Select Syncout Source: 00000: EPWM1SYNCOUT selected to drive the SYNCOUT pin. 00001: EPWM2SYNCOUT selected to drive the SYNCOUT pin. 00010: EPWM3SYNCOUT selected to drive the SYNCOUT pin. 00011: EPWM4SYNCOUT selected to drive the SYNCOUT pin. 00100: EPWM5SYNCOUT selected to drive the SYNCOUT pin. 00101: EPWM6SYNCOUT selected to drive the SYNCOUT pin. 00110: EPWM7SYNCOUT selected to drive the SYNCOUT pin. 00111: EPWM8SYNCOUT selected to drive the SYNCOUT pin. 01000: Reserved 01001: Reserved 01010: Reserved 01011: Reserved 01100: Reserved 01101: Reserved 01110: Reserved 01111: Reserved 10000: Reserved 10001: Reserved 10010: Reserved 10011: Reserved 10100: Reserved 10101: Reserved 10110: Reserved 10111: Reserved 11000: ECAP1SYNCOUT selected to drive the SYNCOUT pin. 11001: ECAP2SYNCOUT selected to drive the SYNCOUT pin. 11010: ECAP3SYNCOUT selected to drive the SYNCOUT pin. 11011: Reserved 11100: Reserved 11101: Reserved 11110: Reserved 11111: Reserved Notes: [1] Reserved position defaults to 00 selection Reset type: SYSRSn
23-18	RESERVED	R	0h	Reserved
17-15	RESERVED	R/W	7h	Reserved
14-12	RESERVED	R/W	7h	Reserved
11-9	RESERVED	R/W	7h	Reserved
8-6	RESERVED	R/W	7h	Reserved
5-3	RESERVED	R/W	7h	Reserved
2-0	RESERVED	R/W	7h	Reserved

3.15.13.2 ADCSOCOUTSELECT Register (Offset = 2h) [Reset = 0h]

ADCSOCOUTSELECT is shown in Figure 3-266 and described in Table 3-287.

Return to the Summary Table.

External ADCSOC Select Register

Figure 3-266. ADCSOCOUTSELECT Register

31	30	29	28	27	26	25	24
RESERVED				RESERVED	RESERVED	RESERVED	RESERVED
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
PWM8SOCBEN	PWM7SOCBEN	PWM6SOCBEN	PWM5SOCBEN	PWM4SOCBEN	PWM3SOCBEN	PWM2SOCBEN	PWM1SOCBEN
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED				RESERVED	RESERVED	RESERVED	RESERVED
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
PWM8SOCAEN	PWM7SOCAEN	PWM6SOCAEN	PWM5SOCAEN	PWM4SOCAEN	PWM3SOCAEN	PWM2SOCAEN	PWM1SOCAEN
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 3-287. ADCSOCOUTSELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27	RESERVED	R/W	0h	Reserved
26	RESERVED	R/W	0h	Reserved
25	RESERVED	R/W	0h	Reserved
24	RESERVED	R/W	0h	Reserved
23	PWM8SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: SYSRSn
22	PWM7SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: SYSRSn
21	PWM6SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: SYSRSn
20	PWM5SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: SYSRSn
19	PWM4SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: SYSRSn
18	PWM3SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: SYSRSn

Table 3-287. ADCSOCOUTSELECT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	PWM2SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: SYSRSn
16	PWM1SOCBEN	R/W	0h	ADCSOCBOn source select: 0: Respective EPWM SOCB output is not selected 1: Respective EPWM SOCB output is selected Reset type: SYSRSn
15-12	RESERVED	R	0h	Reserved
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	RESERVED	R/W	0h	Reserved
8	RESERVED	R/W	0h	Reserved
7	PWM8SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: SYSRSn
6	PWM7SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: SYSRSn
5	PWM6SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: SYSRSn
4	PWM5SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: SYSRSn
3	PWM4SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: SYSRSn
2	PWM3SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: SYSRSn
1	PWM2SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: SYSRSn
0	PWM1SOCAEN	R/W	0h	ADCSOCAOn source select: 0: Respective EPWM SOCA output is not selected 1: Respective EPWM SOCA output is selected Reset type: SYSRSn

3.15.13.3 SYNCSOCLOCK Register (Offset = 4h) [Reset = 0h]

SYNCSOCLOCK is shown in Figure 3-267 and described in Table 3-288.

Return to the Summary Table.

SYNCSEL and EXTADCSOC Select Lock register

Figure 3-267. SYNCSOCLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						ADCSOCOUTSELECT	SYNCSELECT
R-0h						R/WSonce-0h	R/WSonce-0h

Table 3-288. SYNCSOCLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	ADCSOCOUTSELECT	R/WSonce	0h	ADCSOCOUTSELECT Register Lock bit: 0: Respective register is not locked 1: Respective register is locked. Notes: [1] Any bit in this register, once set can only be creaed through a SYSRSn. Write of 0 to any bit of this regtister has no effect [2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed Reset type: SYSRSn
0	SYNCSELECT	R/WSonce	0h	SYNCSELECT Register Lock bit: 0: Respective register is not locked 1: Respective register is locked. Notes: [1] Any bit in this register, once set can only be creaed through a SYSRSn. Write of 0 to any bit of this regtister has no effect [2] The locking mechanism applies to only writes. Reads to the registers which have LOCK protection are always allowed Reset type: SYSRSn

3.15.14 SYS_STATUS_REGS Registers

Table 3-289 lists the memory-mapped registers for the SYS_STATUS_REGS registers. All register offset addresses not listed in Table 3-289 should be considered as reserved locations and the register contents should not be modified.

Table 3-289. SYS_STATUS_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
10h	SYS_ERR_INT_FLG	Status of interrupts due to multiple different errors in the system.		Go
12h	SYS_ERR_INT_CLR	SYS_ERR_INT_FLG clear register		Go
14h	SYS_ERR_INT_SET	SYS_ERR_INT_FLG set register	EALLOW	Go
16h	SYS_ERR_MASK	SYS_ERR_MASK register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 3-290 shows the codes that are used for access types in this section.

Table 3-290. SYS_STATUS_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.15.14.1 SYS_ERR_INT_FLG Register (Offset = 10h) [Reset = 0h]

SYS_ERR_INT_FLG is shown in Figure 3-268 and described in Table 3-291.

Return to the Summary Table.

Status of interrupts due to multiple different errors in the system.

Figure 3-268. SYS_ERR_INT_FLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	EPG1_INT	AES_BUS_ERROR	RESERVED	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	GINT
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 3-291. SYS_ERR_INT_FLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Reserved
14	RESERVED	R	0h	Reserved
13	RESERVED	R	0h	Reserved
12	RESERVED	R	0h	Reserved
11	EPG1_INT	R	0h	0: EPG1_INT has not fired an interrupt. 1: EPG1_INT has fired an interrupt Reset type: SYSRSn
10	AES_BUS_ERROR	R	0h	0: AES_BUS_ERROR has not fired an interrupt. 1: AES_BUS_ERROR has fired an interrupt Reset type: SYSRSn
9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	RESERVED	R	0h	Reserved
6	RESERVED	R	0h	Reserved
5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3	RESERVED	R	0h	Reserved
2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved
0	GINT	R	0h	Global Interrupt flag: 0: On any of the flags of SYS_ERR_INT_FLG register being set, SYS_ERR_INT is pulsed and GINT flag would be set 1: No further interrupts would be fired until GINT flag is cleared Reset type: SYSRSn

3.15.14.2 SYS_ERR_INT_CLR Register (Offset = 12h) [Reset = 0h]

SYS_ERR_INT_CLR is shown in Figure 3-269 and described in Table 3-292.

Return to the Summary Table.

SYS_ERR_INT_FLG clear register

Figure 3-269. SYS_ERR_INT_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	EPG1_INT	AES_BUS_ERROR	RESERVED	RESERVED
R-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	GINT
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 3-292. SYS_ERR_INT_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Reserved
14	RESERVED	R-0/W1S	0h	Reserved
13	RESERVED	R-0/W1S	0h	Reserved
12	RESERVED	R-0/W1S	0h	Reserved
11	EPG1_INT	R-0/W1S	0h	0: No effect 1: EPG1_INT flag of SYS_ERR_INT_FLG reister will be cleared. Reset type: SYSRSn
10	AES_BUS_ERROR	R-0/W1S	0h	0: No effect 1: AES_BUS_ERROR flag of SYS_ERR_INT_FLG reister will be cleared. Reset type: SYSRSn
9	RESERVED	R-0/W1S	0h	Reserved
8	RESERVED	R-0/W1S	0h	Reserved
7	RESERVED	R-0/W1S	0h	Reserved
6	RESERVED	R-0/W1S	0h	Reserved
5	RESERVED	R-0/W1S	0h	Reserved
4	RESERVED	R-0/W1S	0h	Reserved
3	RESERVED	R-0/W1S	0h	Reserved
2	RESERVED	R-0/W1S	0h	Reserved
1	RESERVED	R-0/W1S	0h	Reserved
0	GINT	R-0/W1S	0h	0: No effect 1: GINT flag of SYS_ERR_INT_FLG reister will be cleared. Reset type: SYSRSn

3.15.14.3 SYS_ERR_INT_SET Register (Offset = 14h) [Reset = 0h]

SYS_ERR_INT_SET is shown in Figure 3-270 and described in Table 3-293.

Return to the Summary Table.

SYS_ERR_INT_FLG set register

Figure 3-270. SYS_ERR_INT_SET Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	EPG1_INT	AES_BUS_ERROR	RESERVED	RESERVED
R-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0h

Table 3-293. SYS_ERR_INT_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	R-0/W	0h	A value of 0xa5 to this field would enable write to the other bit fields of this register. Any other value written to KEY field would block the write to the other fields of this register. Note: Only a 32 bit write to this register will succeed in updating the fields of this rigister, provided the correct value written to the KEY field simultaneously Reset type: SYSRSn
23-16	RESERVED	R	0h	Reserved
15	RESERVED	R	0h	Reserved
14	RESERVED	R-0/W1S	0h	Reserved
13	RESERVED	R-0/W1S	0h	Reserved
12	RESERVED	R-0/W1S	0h	Reserved
11	EPG1_INT	R-0/W1S	0h	0: No effect 1: EPG1_INT flag of SYS_ERR_INT_FLG reister will be set. Reset type: SYSRSn
10	AES_BUS_ERROR	R-0/W1S	0h	0: No effect 1: AES_BUS_ERROR flag of SYS_ERR_INT_FLG reister will be set. Reset type: SYSRSn
9	RESERVED	R-0/W1S	0h	Reserved
8	RESERVED	R-0/W1S	0h	Reserved
7	RESERVED	R-0/W1S	0h	Reserved
6	RESERVED	R-0/W1S	0h	Reserved
5	RESERVED	R-0/W1S	0h	Reserved
4	RESERVED	R-0/W1S	0h	Reserved
3	RESERVED	R-0/W1S	0h	Reserved
2	RESERVED	R-0/W1S	0h	Reserved
1	RESERVED	R-0/W1S	0h	Reserved

Table 3-293. SYS_ERR_INT_SET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	RESERVED	R	0h	Reserved

3.15.14.4 SYS_ERR_MASK Register (Offset = 16h) [Reset = 0h]

SYS_ERR_MASK is shown in Figure 3-271 and described in Table 3-294.

Return to the Summary Table.

SYS_ERR_MASK register

Figure 3-271. SYS_ERR_MASK Register

31	30	29	28	27	26	25	24
KEY							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	EPG1_INT	AES_BUS_ERROR	RESERVED	RESERVED
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h

Table 3-294. SYS_ERR_MASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	KEY	R/W	0h	A value of 0xa5 to this field would enable write to the other bit fields of this register. Any other value written to KEY field would block the write to the other fields of this register. Note: Only a 32 bit write to this register will succeed in updating the fields of this rigister, provided the correct value written to the KEY field simultaneously Reset type: SYSRSn
23-16	RESERVED	R	0h	Reserved
15	RESERVED	R	0h	Reserved
14	RESERVED	R/W	0h	Reserved
13	RESERVED	R/W	0h	Reserved
12	RESERVED	R/W	0h	Reserved
11	EPG1_INT	R/W	0h	0: EPG1_INT flag of SYS_ERR_INT_FLG reister will be set on a hardware event. 1: EPG1_INT flag of SYS_ERR_INT_FLG reister will not be set on a hardware event. Reset type: SYSRSn
10	AES_BUS_ERROR	R/W	0h	0: AES_BUS_ERROR flag of SYS_ERR_INT_FLG reister will be set on a hardware event. 1: AES_BUS_ERROR flag of SYS_ERR_INT_FLG reister will not be set on a hardware event. Reset type: SYSRSn
9	RESERVED	R/W	0h	Reserved
8	RESERVED	R/W	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	RESERVED	R/W	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R/W	0h	Reserved

Table 3-294. SYS_ERR_MASK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	RESERVED	R/W	0h	Reserved
1	RESERVED	R/W	0h	Reserved
0	RESERVED	R	0h	Reserved

3.15.15 TEST_ERROR_REGS Registers

Table 3-295 lists the memory-mapped registers for the TEST_ERROR_REGS registers. All register offset addresses not listed in Table 3-295 should be considered as reserved locations and the register contents should not be modified.

Table 3-295. TEST_ERROR_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CPU_RAM_TEST_ERROR_STS	Ram Test: Error Status Register		Go
2h	CPU_RAM_TEST_ERROR_STS_CLR	Ram Test: Error Status Clear Register		Go
4h	CPU_RAM_TEST_ERROR_ADDR	Ram Test: Error address register		Go

Complex bit access types are encoded to fit into small table cells. Table 3-296 shows the codes that are used for access types in this section.

Table 3-296. TEST_ERROR_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.15.15.1 CPU_RAM_TEST_ERROR_STS Register (Offset = 0h) [Reset = 0h]

CPU_RAM_TEST_ERROR_STS is shown in Figure 3-272 and described in Table 3-297.

Return to the Summary Table.

Ram Test: Error Status Register

Figure 3-272. CPU_RAM_TEST_ERROR_STS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						UNC_ERROR	COR_ERROR
R-0h						R-0h	R-0h

Table 3-297. CPU_RAM_TEST_ERROR_STS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	UNC_ERROR	R	0h	0: Indicates that there were no "un-correctable errors" generated in the RAM/ROM test mode. 1: Indicates that "un-correctable errors" wer generated in the RAM/ROM test mode. Reset type: SYSRSn
0	COR_ERROR	R	0h	0: Indicates that there were no "correctable errors" generated in the RAM/ROM test mode. 1: Indicates that "correctable errors" wer generated in the RAM/ROM test mode. Reset type: SYSRSn

3.15.15.2 CPU_RAM_TEST_ERROR_STS_CLR Register (Offset = 2h) [Reset = 0h]

CPU_RAM_TEST_ERROR_STS_CLR is shown in Figure 3-273 and described in Table 3-298.

Return to the Summary Table.

Ram Test: Error Status Clear Register

Figure 3-273. CPU_RAM_TEST_ERROR_STS_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						UNC_ERROR	COR_ERROR
R-0h						R-0/W1S-0h	R-0/W1S-0h

Table 3-298. CPU_RAM_TEST_ERROR_STS_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	UNC_ERROR	R-0/W1S	0h	0: No effect. 1: Clears the corresponding bit in CPU_RAM_TEST_ERROR_STS register. Reset type: SYSRSn
0	COR_ERROR	R-0/W1S	0h	0: No effect. 1: Clears the corresponding bit in CPU_RAM_TEST_ERROR_STS register. Reset type: SYSRSn

3.15.15.3 CPU_RAM_TEST_ERROR_ADDR Register (Offset = 4h) [Reset = 0h]

CPU_RAM_TEST_ERROR_ADDR is shown in Figure 3-274 and described in Table 3-299.

Return to the Summary Table.

Ram Test: Error address register

Figure 3-274. CPU_RAM_TEST_ERROR_ADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															
R-0h																															

Table 3-299. CPU_RAM_TEST_ERROR_ADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR	R	0h	Address of the location where error was detected in RAM/ROM test modes. Reset type: SYSRSn

3.15.16 UID_REGS Registers

Table 3-300 lists the memory-mapped registers for the UID_REGS registers. All register offset addresses not listed in Table 3-300 should be considered as reserved locations and the register contents should not be modified.

Table 3-300. UID_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	UID_PSRAND0	UID Psuedo-random 192 bit number		Go
2h	UID_PSRAND1	UID Psuedo-random 192 bit number		Go
4h	UID_PSRAND2	UID Psuedo-random 192 bit number		Go
6h	UID_PSRAND3	UID Psuedo-random 192 bit number		Go
8h	UID_PSRAND4	UID Psuedo-random 192 bit number		Go
Ah	UID_PSRAND5	UID Psuedo-random 192 bit number		Go
Ch	UID_UNIQUE	UID Unique 32 bit number		Go
Eh	UID_CHECKSUM	UID Checksum		Go

Complex bit access types are encoded to fit into small table cells. Table 3-301 shows the codes that are used for access types in this section.

Table 3-301. UID_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.15.16.1 UID_PSRAND0 Register (Offset = 0h) [Reset = X]

UID_PSRAND0 is shown in Figure 3-275 and described in Table 3-302.

Return to the Summary Table.

UID Psuedo-random 192 bit number

Figure 3-275. UID_PSRAND0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RandomID																															
R-X																															

Table 3-302. UID_PSRAND0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RandomID	R	X	Psuedorandom portion of the UID Reset type: N/A

3.15.16.2 UID_PSRAND1 Register (Offset = 2h) [Reset = X]

UID_PSRAND1 is shown in Figure 3-276 and described in Table 3-303.

Return to the Summary Table.

UID Psuedo-random 192 bit number

Figure 3-276. UID_PSRAND1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RandomID																															
R-X																															

Table 3-303. UID_PSRAND1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RandomID	R	X	Psuedorandom portion of the UID Reset type: N/A

3.15.16.3 UID_PSRAND2 Register (Offset = 4h) [Reset = X]

UID_PSRAND2 is shown in Figure 3-277 and described in Table 3-304.

Return to the Summary Table.

UID Psuedo-random 192 bit number

Figure 3-277. UID_PSRAND2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RandomID																															
R-X																															

Table 3-304. UID_PSRAND2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RandomID	R	X	Psuedorandom portion of the UID Reset type: N/A

3.15.16.4 UID_PSRAND3 Register (Offset = 6h) [Reset = X]

UID_PSRAND3 is shown in Figure 3-278 and described in Table 3-305.

Return to the Summary Table.

UID Psuedo-random 192 bit number

Figure 3-278. UID_PSRAND3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RandomID																															
R-X																															

Table 3-305. UID_PSRAND3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RandomID	R	X	Psuedorandom portion of the UID Reset type: N/A

3.15.16.5 UID_PSRAND4 Register (Offset = 8h) [Reset = X]

UID_PSRAND4 is shown in Figure 3-279 and described in Table 3-306.

Return to the Summary Table.

UID Psuedo-random 192 bit number

Figure 3-279. UID_PSRAND4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RandomID																															
R-X																															

Table 3-306. UID_PSRAND4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RandomID	R	X	Psuedorandom portion of the UID Reset type: N/A

3.15.16.6 UID_PSRAND5 Register (Offset = Ah) [Reset = X]

UID_PSRAND5 is shown in Figure 3-280 and described in Table 3-307.

Return to the Summary Table.

UID Psuedo-random 192 bit number

Figure 3-280. UID_PSRAND5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RandomID																															
R-X																															

Table 3-307. UID_PSRAND5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RandomID	R	X	Psuedorandom portion of the UID Reset type: N/A

3.15.16.7 UID_UNIQUE Register (Offset = Ch) [Reset = X]

UID_UNIQUE is shown in Figure 3-281 and described in Table 3-308.

Return to the Summary Table.

UID Unique 32 bit number

Figure 3-281. UID_UNIQUE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UniqueID																															
R-X																															

Table 3-308. UID_UNIQUE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	UniqueID	R	X	Unique portion of the UID. This identifier will be unique across all devices with the same PARTIDH. Reset type: N/A

3.15.16.8 UID_CHECKSUM Register (Offset = Eh) [Reset = X]

UID_CHECKSUM is shown in Figure 3-282 and described in Table 3-309.

Return to the Summary Table.

Fletcher checksum of UID_PSRAND and UID_UNIQUE registers

Figure 3-282. UID_CHECKSUM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Checksum																															
R-X																															

Table 3-309. UID_CHECKSUM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Checksum	R	X	Fletcher checksum of UID_PSRANDx and UID_UINIQUE Reset type: N/A

3.15.17 WD_REGS Registers

Table 3-310 lists the memory-mapped registers for the WD_REGS registers. All register offset addresses not listed in Table 3-310 should be considered as reserved locations and the register contents should not be modified.

Table 3-310. WD_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
22h	SCSR	System Control & Status Register	EALLOW	Go
23h	WDCNTR	Watchdog Counter Register	EALLOW	Go
25h	WDKEY	Watchdog Reset Key Register	EALLOW	Go
29h	WDCR	Watchdog Control Register	EALLOW	Go
2Ah	WDWCR	Watchdog Windowed Control Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 3-311 shows the codes that are used for access types in this section.

Table 3-311. WD_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.15.17.1 SCSR Register (Offset = 22h) [Reset = 5h]

SCSR is shown in Figure 3-283 and described in Table 3-312.

Return to the Summary Table.

System Control & Status Register
It is recommended to only use 16 bit accesses to write to this register. Use a read-modify-write instruction may inadvertently clear other bits.

Figure 3-283. SCSR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					WDINTS	WDENINT	WDOVERRIDE
R-0h					R-1h	R/W-0h	R/W1C-1h

Table 3-312. SCSR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-3	RESERVED	R	0h	Reserved
2	WDINTS	R	1h	Watchdog Interrupt Status This bit indicates the state of the active-low watchdog interrupt signal (synchronized to SYSCLK). If the watchdog interrupt is used to wake the system from a low-power mode, then that mode should only be entered while this bit is high. Likewise, this bit must go high before the watchdog can be safely disabled and re-enabled. Reset type: SYSRSn 0h (R/W) = The watchdog interrupt signal is active. 1h (R/W) = The watchdog interrupt signal is inactive.
1	WDENINT	R/W	0h	Watchdog Interrupt Enable/Reset Disable This bit determines whether the watchdog triggers an interrupt (WAKE/WDOG) or a reset (WDRS) when the counter expires. Reset type: SYSRSn 0h (R/W) = Counter expiration triggers a reset. This is the default state on power-up and after any system reset. 1h (R/W) = Counter expiration triggers an interrupt.
0	WDOVERRIDE	R/W1C	1h	If this bit is set to 1, the user is allowed to change the state of the Watchdog disable (WDDIS) bit in the Watchdog Control (WDCR) register. If the WDOVERRIDE bit is cleared, by writing a 1 the WDDIS bit cannot be modified by the user. Writing a 0 will have no effect. If this bit is cleared, then it will remain in this state until a reset occurs. The current state of this bit is readable by the user. Reset type: SYSRSn

3.15.17.2 WDCNTR Register (Offset = 23h) [Reset = 0h]

WDCNTR is shown in Figure 3-284 and described in Table 3-313.

Return to the Summary Table.

Watchdog Counter Register

Figure 3-284. WDCNTR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
WDCNTR							
R-0h							

Table 3-313. WDCNTR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-0	WDCNTR	R	0h	Watchdog Counter These bits contain the current value of the watchdog counter. This counter increments with each WDCLK (INTOSC1) cycle. If the counter overflows, either an interrupt or a reset is generated based on the value of the WDINTEN bit in the SCSR register. If the correct value is written to the WDKEY register, this counter is reset to zero. Reset type: IORSn

3.15.17.3 WDKEY Register (Offset = 25h) [Reset = 0h]

WDKEY is shown in Figure 3-285 and described in Table 3-314.

Return to the Summary Table.

Watchdog Reset Key Register

Figure 3-285. WDKEY Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
WDKEY							
R/W-0h							

Table 3-314. WDKEY Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-0	WDKEY	R/W	0h	Watchdog Counter Reset Writing 0x55 followed by 0xAA will cause the watchdog counter to reset to zero, preventing an overflow. Writing other values has no effect. Reads of this register return the value of the WDCR register. Reset type: IORSn

3.15.17.4 WDCR Register (Offset = 29h) [Reset = 0h]

WDCR is shown in Figure 3-286 and described in Table 3-315.

Return to the Summary Table.

Watchdog Control Register

Figure 3-286. WDCR Register

15	14	13	12	11	10	9	8
RESERVED				WDPRECLKDIV			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
WDFLG	WDDIS	WDCHK			WDPS		
R/W1S-0h	R/W-0h	R-0/W-0h			R/W-0h		

Table 3-315. WDCR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-8	WDPRECLKDIV	R/W	0h	Watchdog Clock Pre-divider These bits determine the watchdog clock pre-divider, which is the first of the two dividers between INTOSC1 and the watchdog counter clock (WDCLK). The frequency of WDCLK is given by the formulas: PREDIVCLK = INTOSC1 / Pre-divider WDCLK = PREDIVCLK / Prescaler Reset type: IORSn 0h (R/W) = PREDIVCLK = INTOSC1 / 512 1h (R/W) = PREDIVCLK = INTOSC1 / 1024 2h (R/W) = PREDIVCLK = INTOSC1 / 2048 3h (R/W) = PREDIVCLK = INTOSC1 / 4096 4h (R/W) = Reserved 5h (R/W) = Reserved 6h (R/W) = Reserved 7h (R/W) = Reserved 8h (R/W) = PREDIVCLK = INTOSC1 / 2 9h (R/W) = PREDIVCLK = INTOSC1 / 4 Ah (R/W) = PREDIVCLK = INTOSC1 / 8 Bh (R/W) = PREDIVCLK = INTOSC1 / 16 Ch (R/W) = PREDIVCLK = INTOSC1 / 32 Dh (R/W) = PREDIVCLK = INTOSC1 / 64 Eh (R/W) = PREDIVCLK = INTOSC1 / 128 Fh (R/W) = PREDIVCLK = INTOSC1 / 256
7	WDFLG	R/W1S	0h	Watchdog reset status flag bit. This bit, if set, indicates a watchdog reset (WDRSTn) generated the reset condition. If 0, then it was en external device or power-up reset condition. This bit remains latched until the user writes a 1 to clear the condition. Writes of 0 will be ignored. Reset type: IORSn
6	WDDIS	R/W	0h	Watchdog Disable Setting this bit disables the watchdog module. Clearing this bit enables the watchdog module. This bit can be locked by the WDOVERRIDE bit in the SCSR register. The watchdog is enabled on reset. Reset type: IORSn
5-3	WDCHK	R-0/W	0h	Watchdog Check Bits During any write to this register, these bits must be written with the value 101 (binary). Writing any other value will immediately trigger the watchdog reset or interrupt. Reset type: IORSn

Table 3-315. WDCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-0	WDPS	R/W	0h	Watchdog Clock Prescaler These bits determine the watchdog clock prescaler, which is the second of the two dividers between INTOSC1 and the watchdog counter clock (WDCLK). The frequency of WDCLK is given by the formulas: PREDIVCLK = INTOSC1 / Pre-divider WDCLK = PREDIVCLK / Prescaler The watchdog reset or interrupt pulse is 512 INTOSC1 cycles long, so the counter period must be longer. To guarantee this, the product of the prescaler and pre-divider must be greater than or equal to four. The default prescaler value is 1. Reset type: IORSn 0h (R/W) = WDCLK = PREDIVCLK / 1 1h (R/W) = WDCLK = PREDIVCLK / 1 2h (R/W) = WDCLK = PREDIVCLK / 2 3h (R/W) = WDCLK = PREDIVCLK / 4 4h (R/W) = WDCLK = PREDIVCLK / 8 5h (R/W) = WDCLK = PREDIVCLK / 16 6h (R/W) = WDCLK = PREDIVCLK / 32 7h (R/W) = WDCLK = PREDIVCLK / 64

3.15.17.5 WDWCR Register (Offset = 2Ah) [Reset = 0h]

WDWCR is shown in Figure 3-287 and described in Table 3-316.

Return to the Summary Table.

Watchdog Windowed Control Register

Figure 3-287. WDWCR Register

15	14	13	12	11	10	9	8
RESERVED							FIRSTKEY
R-0h							R-0h
7	6	5	4	3	2	1	0
MIN							
R/W-0h							

Table 3-316. WDWCR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-9	RESERVED	R	0h	Reserved
8	FIRSTKEY	R	0h	This bit indicates if the 1st valid WDKEY (0x55 + 0xAA) got detected after MIN was configured to a non-zero value 0: First Valid Key after non-zero MIN configuration has not happened yet 1: First Valid key after non-zero MIN configuration got detected Notes: [1] If MIN = 0, this bit is never set [2] If MIN is changed back to 0x0 from a non-zero value, this bit is auto-cleared [3] This bit is added for debug purposes only Reset type: IORSn
7-0	MIN	R/W	0h	Watchdog Window Threshold These bits specify the lower limit of the watchdog counter reset window. If the counter is reset via the WDKEY register before the counter value reaches the value in this register, the watchdog immediately triggers a reset or interrupt. Reset type: IORSn

3.15.18 XINT_REGS Registers

Table 3-317 lists the memory-mapped registers for the XINT_REGS registers. All register offset addresses not listed in Table 3-317 should be considered as reserved locations and the register contents should not be modified.

Table 3-317. XINT_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	XINT1CR	XINT1 configuration register		Go
1h	XINT2CR	XINT2 configuration register		Go
2h	XINT3CR	XINT3 configuration register		Go
3h	XINT4CR	XINT4 configuration register		Go
4h	XINT5CR	XINT5 configuration register		Go
8h	XINT1CTR	XINT1 counter register		Go
9h	XINT2CTR	XINT2 counter register		Go
Ah	XINT3CTR	XINT3 counter register		Go

Complex bit access types are encoded to fit into small table cells. Table 3-318 shows the codes that are used for access types in this section.

Table 3-318. XINT_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.15.18.1 XINT1CR Register (Offset = 0h) [Reset = 0h]

XINT1CR is shown in Figure 3-288 and described in Table 3-319.

Return to the Summary Table.

XINT1 configuration register

Figure 3-288. XINT1CR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				POLARITY		RESERVED	ENABLE
R-0h				R/W-0h		R-0h	R/W-0h

Table 3-319. XINT1CR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3-2	POLARITY	R/W	0h	00: Interrupt is selected as negative edge triggered 01: Interrupt is selected as positive edge triggered 10: Interrupt is selected as negative edge triggered 11: Interrupt is selected as positive or negative edge triggered Reset type: SYSRSn
1	RESERVED	R	0h	Reserved
0	ENABLE	R/W	0h	0: Interrupt Disabled 1: Interrupt Enabled Reset type: SYSRSn

3.15.18.2 XINT2CR Register (Offset = 1h) [Reset = 0h]

XINT2CR is shown in Figure 3-289 and described in Table 3-320.

Return to the Summary Table.

XINT2 configuration register

Figure 3-289. XINT2CR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				POLARITY		RESERVED	ENABLE
R-0h				R/W-0h		R-0h	R/W-0h

Table 3-320. XINT2CR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3-2	POLARITY	R/W	0h	00: Interrupt is selected as negative edge triggered 01: Interrupt is selected as positive edge triggered 10: Interrupt is selected as negative edge triggered 11: Interrupt is selected as positive or negative edge triggered Reset type: SYSRSn
1	RESERVED	R	0h	Reserved
0	ENABLE	R/W	0h	0: Interrupt Disabled 1: Interrupt Enabled Reset type: SYSRSn

3.15.18.3 XINT3CR Register (Offset = 2h) [Reset = 0h]

XINT3CR is shown in Figure 3-290 and described in Table 3-321.

Return to the Summary Table.

XINT3 configuration register

Figure 3-290. XINT3CR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				POLARITY		RESERVED	ENABLE
R-0h				R/W-0h		R-0h	R/W-0h

Table 3-321. XINT3CR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3-2	POLARITY	R/W	0h	00: Interrupt is selected as negative edge triggered 01: Interrupt is selected as positive edge triggered 10: Interrupt is selected as negative edge triggered 11: Interrupt is selected as positive or negative edge triggered Reset type: SYSRSn
1	RESERVED	R	0h	Reserved
0	ENABLE	R/W	0h	0: Interrupt Disabled 1: Interrupt Enabled Reset type: SYSRSn

3.15.18.4 XINT4CR Register (Offset = 3h) [Reset = 0h]

XINT4CR is shown in Figure 3-291 and described in Table 3-322.

Return to the Summary Table.

XINT4 configuration register

Figure 3-291. XINT4CR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				POLARITY		RESERVED	ENABLE
R-0h				R/W-0h		R-0h	R/W-0h

Table 3-322. XINT4CR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3-2	POLARITY	R/W	0h	00: Interrupt is selected as negative edge triggered 01: Interrupt is selected as positive edge triggered 10: Interrupt is selected as negative edge triggered 11: Interrupt is selected as positive or negative edge triggered Reset type: SYSRSn
1	RESERVED	R	0h	Reserved
0	ENABLE	R/W	0h	0: Interrupt Disabled 1: Interrupt Enabled Reset type: SYSRSn

3.15.18.5 XINT5CR Register (Offset = 4h) [Reset = 0h]

XINT5CR is shown in Figure 3-292 and described in Table 3-323.

Return to the Summary Table.

XINT5 configuration register

Figure 3-292. XINT5CR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				POLARITY		RESERVED	ENABLE
R-0h				R/W-0h		R-0h	R/W-0h

Table 3-323. XINT5CR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3-2	POLARITY	R/W	0h	00: Interrupt is selected as negative edge triggered 01: Interrupt is selected as positive edge triggered 10: Interrupt is selected as negative edge triggered 11: Interrupt is selected as positive or negative edge triggered Reset type: SYSRSn
1	RESERVED	R	0h	Reserved
0	ENABLE	R/W	0h	0: Interrupt Disabled 1: Interrupt Enabled Reset type: SYSRSn

3.15.18.6 XINT1CTR Register (Offset = 8h) [Reset = 0h]

XINT1CTR is shown in Figure 3-293 and described in Table 3-324.

Return to the Summary Table.

XINT1 counter register

Figure 3-293. XINT1CTR Register

15	14	13	12	11	10	9	8
INTCTR							
R-0h							
7	6	5	4	3	2	1	0
INTCTR							
R-0h							

Table 3-324. XINT1CTR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	INTCTR	R	0h	This is a free running 16-bit up-counter that is clocked at the SYSCLKOUT rate. The counter value is reset to 0x0000 when a valid interrupt edge is detected and then continues counting until the next valid interrupt edge is detected. The counter must only be reset by the selected POLARITY edge as selected in the respective interrupt control register. When the interrupt is disabled, the counter will stop. The counter is a free-running counter and will wrap around to zero when the max value is reached. The counter is a read only register and can only be reset to zero by a valid interrupt edge or by reset. Reset type: SYSRSn

3.15.18.7 XINT2CTR Register (Offset = 9h) [Reset = 0h]

XINT2CTR is shown in Figure 3-294 and described in Table 3-325.

Return to the Summary Table.

XINT2 counter register

Figure 3-294. XINT2CTR Register

15	14	13	12	11	10	9	8
INTCTR							
R-0h							
7	6	5	4	3	2	1	0
INTCTR							
R-0h							

Table 3-325. XINT2CTR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	INTCTR	R	0h	This is a free running 16-bit up-counter that is clocked at the SYSCLKOUT rate. The counter value is reset to 0x0000 when a valid interrupt edge is detected and then continues counting until the next valid interrupt edge is detected. The counter must only be reset by the selected POLARITY edge as selected in the respective interrupt control register. When the interrupt is disabled, the counter will stop. The counter is a free-running counter and will wrap around to zero when the max value is reached. The counter is a read only register and can only be reset to zero by a valid interrupt edge or by reset. Reset type: SYSRSn

3.15.18.8 XINT3CTR Register (Offset = Ah) [Reset = 0h]

XINT3CTR is shown in Figure 3-295 and described in Table 3-326.

Return to the Summary Table.

XINT3 counter register

Figure 3-295. XINT3CTR Register

15	14	13	12	11	10	9	8
INTCTR							
R-0h							
7	6	5	4	3	2	1	0
INTCTR							
R-0h							

Table 3-326. XINT3CTR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	INTCTR	R	0h	This is a free running 16-bit up-counter that is clocked at the SYSCLKOUT rate. The counter value is reset to 0x0000 when a valid interrupt edge is detected and then continues counting until the next valid interrupt edge is detected. The counter must only be reset by the selected POLARITY edge as selected in the respective interrupt control register. When the interrupt is disabled, the counter will stop. The counter is a free-running counter and will wrap around to zero when the max value is reached. The counter is a read only register and can only be reset to zero by a valid interrupt edge or by reset. Reset type: SYSRSn

3.15.19 LFU_REGS Registers

Table 3-327 lists the memory-mapped registers for the LFU_REGS registers. All register offset addresses not listed in Table 3-327 should be considered as reserved locations and the register contents should not be modified.

Table 3-327. LFU_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	LFUConfig	LFU configuration Register	EALLOW	Go
2h	LFUStatus	LFU Configuration Status Register		Go
10h	SWConfig1_SYSRSn	Spare registers reset by SYSRSn	EALLOW	Go
12h	SWConfig2_SYSRSn	Spare registers reset by SYSRSn	EALLOW	Go
14h	SWConfig1_XRSn	Spare registers reset by XRSn	EALLOW	Go
16h	SWConfig2_XRSn	Spare registers reset by XRSn	EALLOW	Go
18h	SWConfig1_PORESETn	Spare registers reset by PORESETn	EALLOW	Go
1Ah	SWConfig2_PORESETn	Spare registers reset by PORESETn	EALLOW	Go
1Ch	LFU_LOCK	LFU Lock Configuration		Go
1Eh	LFU_COMMIT	LFU Commit Configuration		Go

Complex bit access types are encoded to fit into small table cells. Table 3-328 shows the codes that are used for access types in this section.

Table 3-328. LFU_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

3.15.19.1 LFUConfig Register (Offset = 0h) [Reset = 0h]

LFUConfig is shown in Figure 3-296 and described in Table 3-329.

Return to the Summary Table.

LFU configuration Register

Figure 3-296. LFUConfig Register

31	30	29	28	27	26	25	24
RESERVED							
R/W-0h							
23	22	21	20	19	18	17	16
RESERVED							LS01Swap
R/W-0h							R/W-0h
15	14	13	12	11	10	9	8
RESERVED			PieVectorSwap	RESERVED			RESERVED
R/W-0h			R/W-0h	R/W-0h			R/W-0h
7	6	5	4	3	2	1	0
RESERVED			LFU_CLA1	RESERVED			LFU_CPU
R/W-0h			R/W-0h	R/W-0h			R/W-0h

Table 3-329. LFUConfig Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R/W	0h	Reserved
16	LS01Swap	R/W	0h	0: LS0 and LS1 mapped to the original location 1: Location of LS0 and LS1 is swapped. Reset type: SYSRSn
15-13	RESERVED	R/W	0h	Reserved
12	PieVectorSwap	R/W	0h	0: PIE vector table is mapped to the original location 1: PIE Vector Table is swapped to alternate location Reset type: SYSRSn
11-9	RESERVED	R/W	0h	Reserved
8	RESERVED	R/W	0h	Reserved
7-5	RESERVED	R/W	0h	Reserved
4	LFU_CLA1	R/W	0h	0: No pending LFU Requests 1: LFU Request in progress This bit is used by compiler/application code for implementing CLA1 LFU Reset type: SYSRSn
3-1	RESERVED	R/W	0h	Reserved
0	LFU_CPU	R/W	0h	0: No pending LFU Requests 1: LFU Request in progress This bit is used by compiler/application code for implementing CPU LFU Reset type: SYSRSn

3.15.19.2 LFUStatus Register (Offset = 2h) [Reset = 0h]

LFUStatus is shown in Figure 3-297 and described in Table 3-330.

Return to the Summary Table.

LFU Configuration Status Register

Figure 3-297. LFUStatus Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							LS01Swap
R-0h							R-0h
15	14	13	12	11	10	9	8
RESERVED			PieVectorSwap	RESERVED			
R-0h			R-0h	R-0h			
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

Table 3-330. LFUStatus Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16	LS01Swap	R	0h	0: LS0 and LS1 mapped to the original location 1: Location of LS0 and LS1 is swapped. Note: An initiated LSx swap will become unsucessful if the LS0 and LS1 memories have different security configurations Reset type: SYSRSn
15-13	RESERVED	R	0h	Reserved
12	PieVectorSwap	R	0h	0: PIE vector table is mapped to the original location 1: PIE Vector Table is swapped to alternate location Reset type: SYSRSn
11-0	RESERVED	R	0h	Reserved

3.15.19.3 SWConfig1_SYSRSn Register (Offset = 10h) [Reset = 0h]

SWConfig1_SYSRSn is shown in Figure 3-298 and described in Table 3-331.

Return to the Summary Table.

Spare registers reset by SYSRSn

Figure 3-298. SWConfig1_SYSRSn Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BITS																															
R/W-0h																															

Table 3-331. SWConfig1_SYSRSn Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BITS	R/W	0h	R/W bits reset by SYSRSn to be used by the application software Reset type: SYSRSn

3.15.19.4 SWConfig2_SYSRSn Register (Offset = 12h) [Reset = 0h]

SWConfig2_SYSRSn is shown in Figure 3-299 and described in Table 3-332.

Return to the Summary Table.

Spare registers reset by SYSRSn

Figure 3-299. SWConfig2_SYSRSn Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BITS																															
R/W-0h																															

Table 3-332. SWConfig2_SYSRSn Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BITS	R/W	0h	R/W bits reset by SYSRSn to be used by the application software Reset type: SYSRSn

3.15.19.5 SWConfig1_XRSn Register (Offset = 14h) [Reset = 0h]

SWConfig1_XRSn is shown in Figure 3-300 and described in Table 3-333.

Return to the Summary Table.

Spare registers reset by XRSn

Figure 3-300. SWConfig1_XRSn Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BITS																															
R/W-0h																															

Table 3-333. SWConfig1_XRSn Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BITS	R/W	0h	R/W bits reset by XRSn to be used by the application software Reset type: XRSn

3.15.19.6 SWConfig2_XRSn Register (Offset = 16h) [Reset = 0h]

SWConfig2_XRSn is shown in Figure 3-301 and described in Table 3-334.

Return to the Summary Table.

Spare registers reset by XRSn

Figure 3-301. SWConfig2_XRSn Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BITS																															
R/W-0h																															

Table 3-334. SWConfig2_XRSn Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BITS	R/W	0h	R/W bits reset by XRSn to be used by the application software Reset type: XRSn

3.15.19.7 SWConfig1_PORESETn Register (Offset = 18h) [Reset = 0h]

SWConfig1_PORESETn is shown in Figure 3-302 and described in Table 3-335.

Return to the Summary Table.

Spare registers reset by PORESETn

Figure 3-302. SWConfig1_PORESETn Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BITS																															
R/W-0h																															

Table 3-335. SWConfig1_PORESETn Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BITS	R/W	0h	R/W bits reset by PORESETn to be used by the application software Reset type: PORESETn

3.15.19.8 SWConfig2_PORESETn Register (Offset = 1Ah) [Reset = 0h]

SWConfig2_PORESETn is shown in Figure 3-303 and described in Table 3-336.

Return to the Summary Table.

Spare registers reset by PORESETn

Figure 3-303. SWConfig2_PORESETn Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BITS																															
R/W-0h																															

Table 3-336. SWConfig2_PORESETn Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BITS	R/W	0h	R/W bits reset by PORESETn to be used by the application software Reset type: PORESETn

3.15.19.9 LFU_LOCK Register (Offset = 1Ch) [Reset = 0h]

LFU_LOCK is shown in Figure 3-304 and described in Table 3-337.

Return to the Summary Table.

LFU Lock Configuration

Figure 3-304. LFU_LOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED		SWConfig2_PORESETn	SWConfig1_PORESETn	SWConfig2_XRSn	SWConfig1_XRSn	SWConfig2_SYSRSn	SWConfig1_SYSRSn
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED							LFUConfig
R-0h							R/W-0h

Table 3-337. LFU_LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Reserved
13	SWConfig2_PORESETn	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: SYSRSn
12	SWConfig1_PORESETn	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: SYSRSn
11	SWConfig2_XRSn	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: SYSRSn
10	SWConfig1_XRSn	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: SYSRSn
9	SWConfig2_SYSRSn	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: SYSRSn
8	SWConfig1_SYSRSn	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: SYSRSn
7-1	RESERVED	R	0h	Reserved
0	LFUConfig	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: SYSRSn

3.15.19.10 LFU_COMMIT Register (Offset = 1Eh) [Reset = 0h]

LFU_COMMIT is shown in Figure 3-305 and described in Table 3-338.

Return to the Summary Table.

LFU Commit Configuration

Figure 3-305. LFU_COMMIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED		SWConfig2_PORESETn	SWConfig1_PORESETn	SWConfig2_XRSn	SWConfig1_XRSn	SWConfig2_SYSRSn	SWConfig1_SYSRSn
R-0h		R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h
7	6	5	4	3	2	1	0
RESERVED							LFUConfig
R-0h							R/WSonce-0h

Table 3-338. LFU_COMMIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Reserved
13	SWConfig2_PORESETn	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register lock configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: SYSRSn
12	SWConfig1_PORESETn	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register lock configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: SYSRSn
11	SWConfig2_XRSn	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register lock configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: SYSRSn
10	SWConfig1_XRSn	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register lock configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: SYSRSn
9	SWConfig2_SYSRSn	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register lock configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: SYSRSn
8	SWConfig1_SYSRSn	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register lock configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: SYSRSn
7-1	RESERVED	R	0h	Reserved

Table 3-338. LFU_COMMIT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	LFUConfig	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register lock configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: SYSRSn

3.15.20 Register to Driverlib Function Mapping

3.15.20.1 CPUTIMER Registers to Driverlib Functions

Table 3-339. CPUTIMER Registers to Driverlib Functions

File	Driverlib Function
TIM	
cputimer.h	CPUTimer_getTimerCount
PRD	
cputimer.h	CPUTimer_setPeriod
TCR	
cputimer.c	CPUTimer_setEmulationMode
cputimer.h	CPUTimer_clearOverflowFlag
cputimer.h	CPUTimer_disableInterrupt
cputimer.h	CPUTimer_enableInterrupt
cputimer.h	CPUTimer_reloadTimerCounter
cputimer.h	CPUTimer_stopTimer
cputimer.h	CPUTimer_resumeTimer
cputimer.h	CPUTimer_startTimer
cputimer.h	CPUTimer_getTimerOverflowStatus
TPR	
cputimer.h	CPUTimer_setPreScaler
TPRH	
cputimer.h	CPUTimer_setPreScaler

3.15.20.2 DCSM Registers to Driverlib Functions

Table 3-340. DCSM Registers to Driverlib Functions

File	Driverlib Function
Z1OTP_LINKPOINTER1	
-	
Z1OTP_LINKPOINTER2	
-	
Z1OTP_LINKPOINTER3	
-	
Z1OTP_JLM_ENABLE	
-	
Z1OTP_GPREG1	
-	
Z1OTP_GPREG2	
-	
Z1OTP_GPREG3	
-	
Z1OTP_GPREG4	
-	
Z1OTP_PSWDLOCK	
-	
Z1OTP_CRCLOCK	
-	

Table 3-340. DCSM Registers to Driverlib Functions (continued)

File	Driverlib Function
Z1OTP_JTAGPSWDH0	
-	
Z1OTP_JTAGPSWDH1	
-	
Z1OTP_CMACKEY0	
-	
Z1OTP_CMACKEY1	
-	
Z1OTP_CMACKEY2	
-	
Z1OTP_CMACKEY3	
-	
Z2OTP_LINKPOINTER1	
-	
Z2OTP_LINKPOINTER2	
-	
Z2OTP_LINKPOINTER3	
-	
Z2OTP_GPREG1	
-	
Z2OTP_GPREG2	
-	
Z2OTP_GPREG3	
-	
Z2OTP_GPREG4	
-	
Z2OTP_PSWDLOCK	
-	
Z2OTP_CRCLOCK	
-	
Z1_LINKPOINTER	
dcsm.c	DCSM_unlockZone1CSM
dcsm.c	DCSM_readZone1CSMPwd
dcsm.h	DCSM_getZone1LinkPointerError
Z1_OTPSECLOCK	
dcsm.h	DCSM_getZone1OTPSecureLockStatus
Z1_JLM_ENABLE	
-	
Z1_LINKPOINTERERR	
dcsm.h	DCSM_getZone1LinkPointerError
Z1_GPREG1	
-	
Z1_GPREG2	
-	
Z1_GPREG3	

Table 3-340. DCSM Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
Z1_GPREG4	
-	
Z1_CSMKEY0	
dcsm.c	DCSM_unlockZone1CSM
dcsm.c	DCSM_writeZone1CSM
Z1_CSMKEY1	
dcsm.c	DCSM_unlockZone1CSM
dcsm.c	DCSM_writeZone1CSM
Z1_CSMKEY2	
dcsm.c	DCSM_unlockZone1CSM
dcsm.c	DCSM_writeZone1CSM
Z1_CSMKEY3	
dcsm.c	DCSM_unlockZone1CSM
dcsm.c	DCSM_writeZone1CSM
Z1_CR	
dcsm.h	DCSM_secureZone1
dcsm.h	DCSM_getZone1CSMSecurityStatus
dcsm.h	DCSM_getZone1ControlStatus
Z1_GRABSECT1R	
-	
Z1_GRABSECT2R	
-	
Z1_GRABSECT3R	
-	
Z1_GRABRAM1R	
-	
Z1_EXEONLYSECT1R	
dcsm.c	DCSM_getZone1FlashEXEStatus
Z1_EXEONLYSECT2R	
dcsm.c	DCSM_getZone1FlashEXEStatus
Z1_EXEONLYRAM1R	
dcsm.c	DCSM_getZone1RAMEXEStatus
Z1_JTAGKEY0	
-	
Z1_JTAGKEY1	
-	
Z1_JTAGKEY2	
-	
Z1_JTAGKEY3	
-	
Z1_CMACKEY0	
-	
Z1_CMACKEY1	
-	

Table 3-340. DCSM Registers to Driverlib Functions (continued)

File	Driverlib Function
Z1_CMACKEY2	
-	
Z1_CMACKEY3	
-	
Z2_LINKPOINTER	
dcsm.c	DCSM_unlockZone2CSM
dcsm.c	DCSM_readZone2CSMPwd
dcsm.h	DCSM_getZone2LinkPointerError
Z2_OTPSECLOCK	
dcsm.h	DCSM_getZone2OTPSecureLockStatus
Z2_LINKPOINTERERR	
dcsm.h	DCSM_getZone2LinkPointerError
Z2_GPREG1	
-	
Z2_GPREG2	
-	
Z2_GPREG3	
-	
Z2_GPREG4	
-	
Z2_CSMKEY0	
dcsm.c	DCSM_unlockZone2CSM
dcsm.c	DCSM_writeZone2CSM
Z2_CSMKEY1	
dcsm.c	DCSM_unlockZone2CSM
dcsm.c	DCSM_writeZone2CSM
Z2_CSMKEY2	
dcsm.c	DCSM_unlockZone2CSM
dcsm.c	DCSM_writeZone2CSM
Z2_CSMKEY3	
dcsm.c	DCSM_unlockZone2CSM
dcsm.c	DCSM_writeZone2CSM
Z2_CR	
dcsm.h	DCSM_secureZone2
dcsm.h	DCSM_getZone2CSMSecurityStatus
dcsm.h	DCSM_getZone2ControlStatus
Z2_GRABSECT1R	
-	
Z2_GRABSECT2R	
-	
Z2_GRABSECT3R	
-	
Z2_GRABRAM1R	
-	
Z2_EXEONLYSECT1R	

Table 3-340. DCSM Registers to Driverlib Functions (continued)

File	Driverlib Function
dcsm.c	DCSM_getZone2FlashEXEStatus
Z2_EXEONLYSECT2R	
dcsm.c	DCSM_getZone2FlashEXEStatus
Z2_EXEONLYRAM1R	
dcsm.c	DCSM_getZone2RAMEXEStatus
FLSEM	
dcsm.c	DCSM_claimZoneSemaphore
dcsm.c	DCSM_releaseZoneSemaphore
SECTSTAT1	
dcsm.h	DCSM_getFlashSectorZone
SECTSTAT2	
dcsm.h	DCSM_getFlashSectorZone
SECTSTAT3	
dcsm.h	DCSM_getFlashSectorZone
RAMSTAT1	
dcsm.h	DCSM_getRAMZone
SECERRSTAT	
dcsm.h	DCSM_getFlashErrorStatus
SECERRCLR	
dcsm.h	DCSM_clearFlashErrorStatus
SECERRFRC	
dcsm.h	DCSM_forceFlashErrorStatus

3.15.20.3 MEMCFG Registers to Driverlib Functions

Table 3-341. MEMCFG Registers to Driverlib Functions

File	Driverlib Function
DXLOCK	
memcfg.c	MemCfg_lockConfig
memcfg.c	MemCfg_unlockConfig
DXCOMMIT	
memcfg.c	MemCfg_commitConfig
DXACCPROT0	
memcfg.c	MemCfg_setProtection
DXTEST	
memcfg.c	MemCfg_setTestMode
DXINIT	
memcfg.c	MemCfg_initSections
memcfg.c	MemCfg_getInitStatus
DXINITDONE	
memcfg.c	MemCfg_getInitStatus
DXRAMTEST_LOCK	
memcfg.c	MemCfg_lockTestConfig
memcfg.c	MemCfg_unlockTestConfig
LSXLOCK	
memcfg.c	MemCfg_lockConfig

Table 3-341. MEMCFG Registers to Driverlib Functions (continued)

File	Driverlib Function
memcfg.c	MemCfg_unlockConfig
LSXCOMMIT	
memcfg.c	MemCfg_commitConfig
LSXMSEL	
memcfg.c	MemCfg_setLSRAMMasterSel
LSXCLAPGM	
memcfg.h	MemCfg_setCLAMemType
LSXACCPROT0	
memcfg.c	MemCfg_setProtection
LSXACCPROT1	
-	
LSXTEST	
memcfg.c	MemCfg_setTestMode
LSXINIT	
memcfg.c	MemCfg_initSections
memcfg.c	MemCfg_getInitStatus
LSXINITDONE	
memcfg.c	MemCfg_getInitStatus
LSXRAMTEST_LOCK	
memcfg.c	MemCfg_lockTestConfig
memcfg.c	MemCfg_unlockTestConfig
GSXLOCK	
memcfg.c	MemCfg_lockConfig
memcfg.c	MemCfg_unlockConfig
GSXCOMMIT	
memcfg.c	MemCfg_commitConfig
GSXACCPROT0	
memcfg.c	MemCfg_setProtection
GSXTEST	
memcfg.c	MemCfg_setTestMode
GSXINIT	
memcfg.c	MemCfg_initSections
memcfg.c	MemCfg_getInitStatus
GSXINITDONE	
memcfg.c	MemCfg_getInitStatus
GSXRAMTEST_LOCK	
memcfg.c	MemCfg_lockTestConfig
memcfg.c	MemCfg_unlockTestConfig
MSGXLOCK	
memcfg.c	MemCfg_lockConfig
memcfg.c	MemCfg_unlockConfig
MSGXCOMMIT	
memcfg.c	MemCfg_commitConfig
MSGXTEST	
memcfg.c	MemCfg_setTestMode

Table 3-341. MEMCFG Registers to Driverlib Functions (continued)

File	Driverlib Function
MSGXINIT	
memcfg.c	MemCfg_initSections
memcfg.c	MemCfg_getInitStatus
MSGXINITDONE	
memcfg.c	MemCfg_getInitStatus
MSGXRAMTEST_LOCK	
memcfg.c	MemCfg_lockTestConfig
memcfg.c	MemCfg_unlockTestConfig
ROM_LOCK	
memcfg.c	MemCfg_lockTestConfig
memcfg.c	MemCfg_unlockTestConfig
ROM_TEST	
memcfg.c	MemCfg_setTestMode
ROM_FORCE_ERROR	
memcfg.c	MemCfg_forceMemError
NMAVFLG	
memcfg.h	MemCfg_getViolationInterruptStatus
NMAVSET	
memcfg.h	MemCfg_forceViolationInterrupt
NMAVCLR	
memcfg.h	MemCfg_clearViolationInterruptStatus
NMAVINTEN	
memcfg.h	MemCfg_enableViolationInterrupt
memcfg.h	MemCfg_disableViolationInterrupt
NMCPURDAVADDR	
memcfg.c	MemCfg_getViolationAddress
NMCPUWRAVADDR	
memcfg.c	MemCfg_getViolationAddress
NMCPUFAVADDR	
-	
NMDMAWRAVADDR	
-	
NMCLA1RDAVADDR	
-	
NMCLA1WRAVADDR	
-	
NMCLA1FAVADDR	
-	
NMDMARDAVADDR	
-	
MAVFLG	
memcfg.h	MemCfg_getViolationInterruptStatus
MAVSET	
memcfg.h	MemCfg_forceViolationInterrupt
MAVCLR	

Table 3-341. MEMCFG Registers to Driverlib Functions (continued)

File	Driverlib Function
memcfg.h	MemCfg_clearViolationInterruptStatus
MAVINTEN	
memcfg.h	MemCfg_enableViolationInterrupt
memcfg.h	MemCfg_disableViolationInterrupt
MCPUFAVADDR	
memcfg.c	MemCfg_getViolationAddress
MCPUWRAVADDR	
-	
MDMAWRAVADDR	
-	
MHICWRAVADDR	
-	
NMHICRDAVADDR	
-	
NMHICWRAVADDR	
-	
UCERRFLG	
memcfg.h	MemCfg_getUncorrErrorStatus
UCERRSET	
memcfg.h	MemCfg_forceUncorrErrorStatus
UCERRCLR	
memcfg.h	MemCfg_clearUncorrErrorStatus
UCCPUREADDR	
memcfg.c	MemCfg_getUncorrErrorAddress
UCDMAREADDR	
memcfg.c	MemCfg_getUncorrErrorAddress
UCCLA1READDR	
-	
UCHICAREADDR	
-	
CERRFLG	
memcfg.h	MemCfg_getCorrErrorStatus
CERRSET	
memcfg.c	MemCfg_getCorrErrorAddress
memcfg.h	MemCfg_forceCorrErrorStatus
CERRCLR	
memcfg.c	MemCfg_getCorrErrorAddress
memcfg.h	MemCfg_clearCorrErrorStatus
CCPUREADDR	
memcfg.c	MemCfg_getCorrErrorAddress
CDMAREADDR	
-	
CCLA1READDR	
-	
CERRCNT	

Table 3-341. MEMCFG Registers to Driverlib Functions (continued)

File	Driverlib Function
memcfg.h	MemCfg_getCorrErrorCount
CERRTHRES	
memcfg.h	MemCfg_setCorrErrorThreshold
CEINTFLG	
memcfg.h	MemCfg_getCorrErrorInterruptStatus
CEINTCLR	
memcfg.h	MemCfg_clearCorrErrorInterruptStatus
CEINTSET	
memcfg.h	MemCfg_forceCorrErrorInterrupt
CEINTEN	
memcfg.h	MemCfg_enableCorrErrorInterrupt
memcfg.h	MemCfg_disableCorrErrorInterrupt
CHICREADDR	
-	
CPU_RAM_TEST_ERROR_STS	
memcfg.h	MemCfg_getDiagErrorStatus
memcfg.h	MemCfg_clearDiagErrorStatus
CPU_RAM_TEST_ERROR_STS_CLR	
memcfg.h	MemCfg_clearDiagErrorStatus
CPU_RAM_TEST_ERROR_ADDR	
memcfg.h	MemCfg_getDiagErrorAddress

3.15.20.4 NMI Registers to Driverlib Functions

Table 3-342. NMI Registers to Driverlib Functions

File	Driverlib Function
CFG	
sysctl.h	SysCtl_enableNMIGlobalInterrupt
FLG	
sysctl.h	SysCtl_getNMIStatus
sysctl.h	SysCtl_getNMIFlagStatus
sysctl.h	SysCtl_isNMIFlagSet
sysctl.h	SysCtl_clearNMIStatus
sysctl.h	SysCtl_clearAllNMIFlags
sysctl.h	SysCtl_forceNMIFlags
FLGCLR	
sysctl.h	SysCtl_clearNMIStatus
sysctl.h	SysCtl_clearAllNMIFlags
FLGFRC	
sysctl.h	SysCtl_forceNMIFlags
WDCNT	
sysctl.h	SysCtl_getNMIWatchdogCounter
WDPRD	
sysctl.h	SysCtl_setNMIWatchdogPeriod
sysctl.h	SysCtl_getNMIWatchdogPeriod
SHDFLG	

Table 3-342. NMI Registers to Driverlib Functions (continued)

File	Driverlib Function
sysctl.h	SysCtl_getNMIShadowFlagStatus
sysctl.h	SysCtl_isNMIShadowFlagSet
ERRORSTS	
sysctl.h	SysCtl_isErrorTriggered
sysctl.h	SysCtl_getErrorPinStatus
sysctl.h	SysCtl_forceError
sysctl.h	SysCtl_clearError
ERRORSTSCLR	
sysctl.h	SysCtl_clearError
ERRORSTSFRC	
sysctl.h	SysCtl_forceError
ERRORCTL	
sysctl.h	SysCtl_selectErrPinPolarity
ERRORLOCK	
sysctl.h	SysCtl_lockErrControl

3.15.20.5 PIE Registers to Driverlib Functions

Table 3-343. PIE Registers to Driverlib Functions

File	Driverlib Function
CTRL	
interrupt.c	Interrupt_initModule
interrupt.h	Interrupt_defaultHandler
interrupt.h	Interrupt_enablePIE
interrupt.h	Interrupt_disablePIE
ACK	
interrupt.c	Interrupt_disable
interrupt.h	Interrupt_clearACKGroup
IER1	
interrupt.c	Interrupt_initModule
interrupt.c	Interrupt_enable
interrupt.c	Interrupt_disable
IFR1	
interrupt.c	Interrupt_initModule
IER2	
interrupt.c	Interrupt_initModule
IFR2	
interrupt.c	Interrupt_initModule
IER3	
interrupt.c	Interrupt_initModule
IFR3	
interrupt.c	Interrupt_initModule
IER4	
interrupt.c	Interrupt_initModule
IFR4	
interrupt.c	Interrupt_initModule

Table 3-343. PIE Registers to Driverlib Functions (continued)

File	Driverlib Function
IER5	
interrupt.c	Interrupt_initModule
IFR5	
interrupt.c	Interrupt_initModule
IER6	
interrupt.c	Interrupt_initModule
IFR6	
interrupt.c	Interrupt_initModule
IER7	
interrupt.c	Interrupt_initModule
IFR7	
interrupt.c	Interrupt_initModule
IER8	
interrupt.c	Interrupt_initModule
IFR8	
interrupt.c	Interrupt_initModule
IER9	
interrupt.c	Interrupt_initModule
IFR9	
interrupt.c	Interrupt_initModule
IER10	
interrupt.c	Interrupt_initModule
IFR10	
interrupt.c	Interrupt_initModule
IER11	
interrupt.c	Interrupt_initModule
IFR11	
interrupt.c	Interrupt_initModule
IER12	
interrupt.c	Interrupt_initModule
IFR12	
interrupt.c	Interrupt_initModule

3.15.20.6 SYSCTL Registers to Driverlib Functions

Table 3-344. SYSCTL Registers to Driverlib Functions

File	Driverlib Function
PARTIDL	
sysctl.c	SysCtl_getDeviceParametric
PARTIDH	
sysctl.c	SysCtl_getDeviceParametric
REVID	
sysctl.h	SysCtl_getDeviceRevision
FUSEERR	
sysctl.h	SysCtl_getEfuseError
SOFTPRES0	

Table 3-344. SYSCTL Registers to Driverlib Functions (continued)

File	Driverlib Function
sysctl.h	SysCtl_resetPeripheral
SOFTPRES2	
-	See SOFTPRES0
SOFTPRES3	
-	See SOFTPRES0
SOFTPRES4	
-	See SOFTPRES0
SOFTPRES6	
-	See SOFTPRES0
SOFTPRES7	
-	See SOFTPRES0
SOFTPRES8	
-	See SOFTPRES0
SOFTPRES9	
-	See SOFTPRES0
SOFTPRES10	
-	See SOFTPRES0
SOFTPRES13	
-	See SOFTPRES0
SOFTPRES14	
-	See SOFTPRES0
SOFTPRES16	
-	See SOFTPRES0
SOFTPRES17	
-	
SOFTPRES18	
-	
SOFTPRES19	
-	
SOFTPRES20	
-	
SOFTPRES21	
-	
SOFTPRES25	
-	
SOFTPRES26	
-	
SOFTPRES27	
-	
TAP_STATUS	
-	
ECAPTYPE	
sysctl.c	SysCtl_configureType
sysctl.c	SysCtl_isConfigTypeLocked
SDFMTYPE	

Table 3-344. SYSCTL Registers to Driverlib Functions (continued)

File	Driverlib Function
sysctl.c	SysCtl_configureType
CLKCFGLOCK1	
sysctl.c	SysCtl_lockClkConfig
CLKSRCCTL1	
sysctl.c	SysCtl_getClock
sysctl.c	SysCtl_setClock
sysctl.c	SysCtl_selectXTAL
sysctl.c	SysCtl_selectXTALSingleEnded
sysctl.c	SysCtl_selectOscSource
sysctl.h	SysCtl_turnOnOsc
sysctl.h	SysCtl_turnOffOsc
sysctl.h	SysCtl_enableWatchdogInHalt
sysctl.h	SysCtl_disableWatchdogInHalt
CLKSRCCTL2	
can.h	CAN_selectClockSource
CLKSRCCTL3	
sysctl.h	SysCtl_selectClockOutSource
SYSPLLCTL1	
sysctl.c	SysCtl_getClock
sysctl.c	SysCtl_setClock
SYSPLLMULT	
sysctl.c	SysCtl_getClock
sysctl.c	SysCtl_setClock
SYSPLLSTS	
sysctl.c	SysCtl_setClock
SYSCLKDIVSEL	
sysctl.c	SysCtl_getClock
sysctl.c	SysCtl_setClock
sysctl.h	SysCtl_setPLLSysClk
AUXCLKDIVSEL	
sysctl.h	SysCtl_setMCANClk
XCLKOUTDIVSEL	
sysctl.h	SysCtl_setXClk
LOSPCP	
sysctl.c	SysCtl_getLowSpeedClock
sysctl.h	SysCtl_setLowSpeedClock
MCDCR	
sysctl.h	SysCtl_enableMCD
sysctl.h	SysCtl_disableMCD
sysctl.h	SysCtl_isMCDClockFailureDetected
sysctl.h	SysCtl_resetMCD
sysctl.h	SysCtl_connectMCDClockSource
sysctl.h	SysCtl_disconnectMCDClockSource
X1CNT	
sysctl.c	SysCtl_pollX1Counter

Table 3-344. SYSCTL Registers to Driverlib Functions (continued)

File	Driverlib Function
sysctl.h	SysCtl_getExternalOscCounterValue
sysctl.h	SysCtl_clearExternalOscCounterValue
XTALCR	
sysctl.c	SysCtl_setClock
sysctl.c	SysCtl_selectXTAL
sysctl.c	SysCtl_selectXTALSingleEnded
sysctl.h	SysCtl_setExternalOscMode
sysctl.h	SysCtl_turnOnOsc
sysctl.h	SysCtl_turnOffOsc
XTALCR2	
sysctl.c	SysCtl_selectXTAL
CLKFAILCFG	
-	
CPUSYSLOCK1	
sysctl.c	SysCtl_lockSysConfig
CPUSYSLOCK2	
-	
PIEVERRADDR	
sysctl.h	SysCtl_getPIEVErrAddr
PCLKCR0	
sysctl.h	SysCtl_enablePeripheral
sysctl.h	SysCtl_disablePeripheral
PCLKCR2	
-	See PCLKCR0
PCLKCR3	
-	See PCLKCR0
PCLKCR4	
-	See PCLKCR0
PCLKCR6	
-	See PCLKCR0
PCLKCR7	
-	See PCLKCR0
PCLKCR8	
-	See PCLKCR0
PCLKCR9	
-	See PCLKCR0
PCLKCR10	
-	See PCLKCR0
PCLKCR13	
-	See PCLKCR0
PCLKCR14	
-	See PCLKCR0
PCLKCR16	
-	See PCLKCR0
PCLKCR17	

Table 3-344. SYSCTL Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
PCLKCR18	
-	
PCLKCR19	
-	
PCLKCR20	
-	
PCLKCR21	
-	
PCLKCR25	
-	
PCLKCR26	
-	
PCLKCR27	
-	
SIMRESET	
sysctl.h	SysCtl_simulateReset
LPMCR	
sysctl.h	SysCtl_enterIdleMode
sysctl.h	SysCtl_enterStandbyMode
sysctl.h	SysCtl_enterHaltMode
sysctl.h	SysCtl_setStandbyQualificationPeriod
sysctl.h	SysCtl_enableWatchdogStandbyWakeup
sysctl.h	SysCtl_disableWatchdogStandbyWakeup
GPIOLPMSEL0	
sysctl.h	SysCtl_enableLPMWakeupPin
sysctl.h	SysCtl_disableLPMWakeupPin
GPIOLPMSEL1	
sysctl.h	SysCtl_enableLPMWakeupPin
sysctl.h	SysCtl_disableLPMWakeupPin
TMR2CLKCTL	
cputimer.h	CPUTimer_selectClockSource
sysctl.h	SysCtl_setCputimer2Clk
RESCCLR	
sysctl.h	SysCtl_clearResetCause
sysctl.h	SysCtl_clearWatchdogResetStatus
RESC	
sysctl.h	SysCtl_getResetCause
sysctl.h	SysCtl_clearResetCause
sysctl.h	SysCtl_getWatchdogResetStatus
sysctl.h	SysCtl_clearWatchdogResetStatus
MCANWAKESTATUS	
sysctl.h	SysCtl_isMCANWakeStatusSet
sysctl.h	SysCtl_clearMCANWakeStatus
MCANWAKESTATUSCLR	

Table 3-344. SYSCTL Registers to Driverlib Functions (continued)

File	Driverlib Function
sysctl.h	SysCtl_clearMCANWakeStatus
CLKSTOPREQ	
-	
CLKSTOPACK	
-	
CLA1TASKSRCSELLOCK	
-	
DMACHSRCSELLOCK	
-	
CLA1TASKSRCSEL1	
cla.c	CLA_setTriggerSource
CLA1TASKSRCSEL2	
cla.c	CLA_setTriggerSource
DMACHSRCSEL1	
dma.c	DMA_configMode
DMACHSRCSEL2	
dma.c	DMA_configMode
ADCA_AC	
-	
ADCB_AC	
-	
ADCC_AC	
-	
CMPSS1_AC	
-	
CMPSS2_AC	
-	
CMPSS3_AC	
-	
CMPSS4_AC	
-	
DACA_AC	
-	
DACB_AC	
-	
EPWM1_AC	
-	
EPWM2_AC	
-	
EPWM3_AC	
-	
EPWM4_AC	
-	
EPWM5_AC	
-	

Table 3-344. SYSCTL Registers to Driverlib Functions (continued)

File	Driverlib Function
EPWM6_AC	
-	
EPWM7_AC	
-	
EPWM8_AC	
-	
EQEP1_AC	
-	
EQEP2_AC	
-	
ECAP1_AC	
-	
ECAP2_AC	
-	
ECAP3_AC	
-	
SDFM1_AC	
-	
SDFM2_AC	
-	
CLB1_AC	
-	
CLB2_AC	
-	
CLB3_AC	
-	
CLB4_AC	
-	
SCIA_AC	
-	
SCIB_AC	
-	
SPIA_AC	
-	
SPIB_AC	
-	
I2CA_AC	
-	
I2CB_AC	
-	
PMBUS_A_AC	
-	
LIN_A_AC	
-	
LIN_B_AC	

Table 3-344. SYSCTL Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
DCANA_AC	
-	
MCANA_AC	
-	
FSIATX_AC	
-	
FSIARX_AC	
-	
HRPWM_A_AC	
-	
HIC_A_AC	
-	
AESA_AC	
-	
PERIPH_AC_LOCK	
sysctl.h	SysCtl_lockAccessControlRegs
SYNCSELECT	
sysctl.h	SysCtl_setSyncOutputConfig
ADCSOCOUTSELECT	
sysctl.h	SysCtl_enableExtADCSOCSource
sysctl.h	SysCtl_disableExtADCSOCSource
SYNCSOCLOCK	
sysctl.h	SysCtl_lockExtADCSOCSelect
sysctl.h	SysCtl_lockSyncSelect
LFUCONFIG	
sysctl.h	SysCtl_setLFUCPU
sysctl.h	SysCtl_getLFUCPU
sysctl.h	SysCtl_setLFUCLA1
sysctl.h	SysCtl_getLFUCLA1
sysctl.h	SysCtl_swapPieVectorAndLS01
sysctl.h	SysCtl_swapPieVector
sysctl.h	SysCtl_swapLS01
LFUSTATUS	
sysctl.h	SysCtl_isPieVectorSwap
sysctl.h	SysCtl_isLS01Swap
SWCONFIG1_SYSRSN	
sysctl.h	SysCtl_setLFUUserRegister
sysctl.h	SysCtl_getLFUUserRegister
SWCONFIG2_SYSRSN	
-	
SWCONFIG1_XRSN	
-	
SWCONFIG2_XRSN	
-	

Table 3-344. SYSCTL Registers to Driverlib Functions (continued)

File	Driverlib Function
SWCONFIG1_PORESETN	
-	
SWCONFIG2_PORESETN	
-	
LFU_LOCK	
sysctl.h	SysCtl_lockLFUConfigRegister
sysctl.h	SysCtl_lockLFUUserRegister
sysctl.h	SysCtl_unlockLFUConfigRegister
sysctl.h	SysCtl_unlockLFUUserRegister
LFU_COMMIT	
sysctl.h	SysCtl_commitLFUConfigRegister
sysctl.h	SysCtl_commitLFUUserRegister
SYS_ERR_INT_FLG	
sysctl.h	SysCtl_getInterruptStatus
SYS_ERR_INT_CLR	
sysctl.h	SysCtl_clearInterruptStatus
SYS_ERR_INT_SET	
sysctl.h	SysCtl_setInterruptStatus
SYS_ERR_MASK	
sysctl.h	SysCtl_getInterruptStatusMask
sysctl.h	SysCtl_setInterruptStatusMask

3.15.20.7 WWD Registers to Driverlib Functions

Table 3-345. WWD Registers to Driverlib Functions

File	Driverlib Function
SCSR	
sysctl.h	SysCtl_setWatchdogMode
sysctl.h	SysCtl_isWatchdogInterruptActive
sysctl.h	SysCtl_clearWatchdogOverride
WDCNTR	
sysctl.h	SysCtl_getWatchdogCounterValue
WDKEY	
sysctl.h	SysCtl_serviceWatchdog
sysctl.h	SysCtl_enableWatchdogReset
sysctl.h	SysCtl_resetWatchdog
WDCR	
sysctl.h	SysCtl_resetDevice
sysctl.h	SysCtl_disableWatchdog
sysctl.h	SysCtl_enableWatchdog
sysctl.h	SysCtl_setWatchdogPredivider
sysctl.h	SysCtl_setWatchdogPrescaler
WDWCR	
sysctl.h	SysCtl_setWatchdogWindowValue

3.15.20.8 XINT Registers to Driverlib Functions

Table 3-346. XINT Registers to Driverlib Functions

File	Driverlib Function
1CR	
gpio.c	GPIO_setInterruptPin
gpio.h	GPIO_setInterruptType
gpio.h	GPIO_getInterruptType
gpio.h	GPIO_enableInterrupt
gpio.h	GPIO_disableInterrupt
gpio.h	GPIO_getInterruptCounter
2CR	
-	See 1CR
3CR	
-	See 1CR
4CR	
-	See 1CR
5CR	
-	See 1CR
1CTR	
gpio.h	GPIO_getInterruptCounter
2CTR	
-	
3CTR	
-	

Chapter 4

ROM Code and Peripheral Booting

This chapter explains the boot procedure, the available boot modes, and the various details of the ROM code including memory maps, initializations, reset handling, and status information.

4.1 Introduction

The purpose of this chapter is to explain the boot read-only memory (ROM) code functionality for the CPU core, including the boot procedure. It also discusses the functions and features of the boot ROM code, and provides details about the ROM memory-map contents. On every reset, the device executes a boot sequence in the ROM depending on the reset type and boot configuration. This sequence initializes the device to run the application code. For the CPU, the boot ROM also contains peripheral bootloaders that can be used to load an application into RAM. These bootloaders can be disabled for safety or security purposes.

See Table 4-1 for details on available boot features for the C28x CPU. Additionally, Table 4-2 shows the sizes of the various ROMs on the device.

For details on the security APIs provided, refer to Section 4.8.11.

Various tables are provided in ROM for use in software library, refer to Section 4.8.8 for more details.

Table 4-1. Boot System Overview

Boot Feature	CPU
Initial boot process	Device reset
Boot mode selection	GPIOs
Boot modes supported	Flash boot Secure Flash boot RAM boot Live firmware update (LFU) boot Secure LFU boot
Peripheral boot loaders supported	Parallel IO SCI / Wait CAN CAN-FD I2C SPI

Table 4-2. ROM Memory

ROM	CPU Size
Unsecure boot ROM	64 KB
Secure ROM	48 KB
CLA data ROM	8 KB

4.2 ROM Related Collateral

Foundational Materials

- Bootloading 101 (Video)

Expert Materials

- *C2000 Software Controlled Firmware Update Process Application Report*

4.3 Device Boot Sequence

Table 4-3 describes the general boot ROM procedure each time the CPU core is reset.

During boot, boot ROM code updates a boot status location in RAM that details the actions taken during this process. Refer to Section 4.8.13 for more details.

Table 4-3. Device Boot ROM Sequence

Step	CPU Action
1	After reset, check for HWBIST reset. If it is a HWBIST reset, immediately branch and return to the user application. If it is not a HWBIST reset, then continue boot and check the FUSE error register for any errors and handle accordingly.
2	Clock configuration and Flash power-up
3	Peripheral trimming and device configuration registers are loaded from OTP.
4	On power-on reset (POR), all RAMs are initialized.
5	Non-maskable interrupt (NMI) handling is enabled and DCSM initialization is performed.
6	Device calibration is performed; trimming the specified peripherals with set OTP values.
7	Determine if polling the GPIO pins are needed for determining the boot mode and, if so, read the boot mode GPIO pins to determine the boot mode to run.
8	Based on the boot mode and options, the appropriate boot sequence is executed. Refer to Section 4.6.1 for a flow chart of the boot sequences.

4.4 Device Boot Modes

This section explains the default boot modes, as well as all the available boot modes supported on this device. The boot ROM uses the boot mode select, general purpose input/output (GPIO) pins to determine the boot mode configuration.

4.4.1 Default Boot Modes

Table 4-4 shows the boot mode options available for selection by the default boot mode select pins. Users have the option to program the device to customize the boot modes selectable in the boot-up table as well as the boot mode select pin GPIOs used.

Table 4-4. Device Default Boot Modes

Boot Mode	GPIO24 (Default boot mode select pin 1)	GPIO32 (Default boot mode select pin 0)
Parallel IO	0	0
SCI / Wait Boot(1)	0	1
CAN	1	0
Flash	1	1

(1) SCI boot mode is used as a wait boot mode as long as SCI continues to wait for an 'A' or 'a' during the SCI autobaud lock process.

Refer to Section 4.8.9.1 for functional details of the boot modes.

Refer to Section 4.8.10 for GPIOs used for selecting the boot modes.

Refer to Section 4.5 for details of boot configurations.

Note

All the peripheral boot modes that are supported use the first instance of the peripheral module (SCIA, SPIA, I2CA, CANA, and so forth). Whenever these boot modes are referred to in this chapter, such as SCI boot, it is actually referring to the first module instance, which means the SCI boot on the SCIA port. The same applies to the other peripheral boot modes.

4.4.2 Custom Boot Modes

Once the user programs a custom boot table in user OTP, an entry in the custom table is used for booting. Users can customize the boot mode select pins in the end system design by programming the BOOTPIN_CONFIG location in user OTP. This allows customers to use 0, 1, 2, or 3 boot mode select pins as needed. You can also customize the boot definition table and indicate which location to boot from by programming the boot mode definition table in the BOOTDEF location of user OTP. Table 4-5 shows the options for various boot modes.

Table 4-5. Custom Boot Modes

Boot Mode Number	Boot Modes
0	Parallel
1	SCI / Wait
2	CAN
3	Flash
4	Wait
5	RAM
6	SPI
7	I2C
10	Secure Flash

4.5 Device Boot Configurations

This section details what boot configurations are available and how to configure them. This device supports from **one** boot mode select pins up to **three** boot mode select pins as well as from **one** configured boot mode up to **eight** configured boot modes.

To change and configure the device from the default settings to custom settings for your application, use the following process:

1. Determine all the various ways you want application to be able to boot. (For example: Primary boot option of Flash boot for your main application, secondary boot option of CAN boot for firmware updates, tertiary boot option of SCI boot for debugging, etc)
2. Based on the number of boot modes needed, determine how many boot mode select pins (BMSPs) are required to select between your selected boot modes. (For example: **Two** BMSPs are required to select between **three** boot mode options)
3. Assign the required BMSPs to a physical GPIO pin. (For example, BMSP0 to GPIO10, BMSP1 to GPIO51, and BMSP2 left as default which is disabled). Refer to Section 4.5.1 for all the details on performing these configurations.
4. Assign the determined boot mode definitions to indexes in your custom boot table that correlate to the decoded value of the BMSPs. For example, BOOTDEF0=Boot to Flash, BOOTDEF1=CAN Boot, BOOTDEF2=SCI Boot; all other BOOTDEFx are left as default/nothing). Refer to Section 4.5.2 for all the details on setting up and configuring the custom boot mode table.

Additionally, Section 4.5.3 provides some example use cases on how to configure the BMSPs and custom boot tables.

4.5.1 Configuring Boot Mode Pins

This section explains how the boot mode select pinsARE customized by the user, by programming the BOOTPIN-CONFIG location (refer to Table 4-6) in the user-configurable dual-zone security module (DCSM) OTP. The location in the DCSM OTP is Z1-OTP-BOOTPIN-CONFIG or Z2-OTP-BOOTPIN-CONFIG. When debugging, EMU-BOOTPIN-CONFIG is the emulation equivalent of Z1-OTP-BOOTPIN-CONFIG/Z2-OTP-BOOTPIN-CONFIG, and can be programmed to experiment with different boot modes without writing to OTP. The device can be programmed to use **zero**, **one**, **two** or **three** boot mode select pins as needed.

Note

When using Z2-OTP-BOOTPIN-CONFIG, the configurations programmed in this location take priority over the configurations in Z1-OTP-BOOTPIN-CONFIG. It is **recommended** to use Z1-OTP-BOOTPIN-CONFIG first and then if OTP configurations need to be altered, switch to using Z2-OTP-BOOTPIN-CONFIG.

Table 4-6. BOOTPIN-CONFIG Bit Fields

Bit	Name	Description
31:24	Key	Write 0x5A to these 8-bits to tell the boot ROM code that the bits in this register are valid.
23:16	Boot Mode Select Pin 2 (BMSP2)	Refer to BMSP0 description.
15:8	Boot Mode Select Pin 1 (BMSP1)	Refer to BMSP0 description.
7:0	Boot Mode Select Pin 0 (BMSP0)	Set to the GPIO pin to be used during boot (up to 255). 0x0 = GPIO0, 0x01 = GPIO1, and so on. Writing 0xFF disables this BMSP and this pin is no longer used to select the boot mode.

Note

GPIO 20, 21, 224 to 253 are analog pins, but digital inputs are possible on these pins provided the software writes to the GPIOHAMSEL register bits.

The following GPIOs **cannot** be used as a BMSP. If selected for a particular BMSP, the boot ROM will automatically select the factory default GPIOs for BMSP0 and BMSP1. Factory default for BMSP2 is 0xFF, which disables the BMSP.

- GPIO 36 and GPIO 38 (Not available on any package)
- GPIO 62 to GPIO 223 (Not available on any package)
- GPIO 20 and GPIO 21 are analog pins on 80-pin package only

Table 4-7. Standalone Boot Mode Select Pin Decoding

BOOTPIN_CONFIG Key	BMSP0	BMSP1	BMSP2	Realized Boot Mode
!= 0x5A	Don't Care	Don't Care	Don't Care	Boot as defined by the factory default BMSPs
= 0x5A	0xFF	0xFF	0xFF	Boot as defined in the boot table for boot mode 0 (All BMSPs disabled)
	Valid GPIO	0xFF	0xFF	Boot as defined by the value of BMSP0 (BMSP1 and BMSP2 disabled)
	0xFF	Valid GPIO	0xFF	Boot as defined by the value of BMSP1 (BMSP0 and BMSP2 disabled)
	0xFF	0xFF	Valid GPIO	Boot as defined by the value of BMSP2 (BMSP0 and BMSP1 disabled)
	Valid GPIO	Valid GPIO	0xFF	Boot as defined by the values of BMSP0 and BMSP1 (BMSP2 disabled)
	Valid GPIO	0xFF	Valid GPIO	Boot as defined by the values of BMSP0 and BMSP2 (BMSP1 disabled)
	0xFF	Valid GPIO	Valid GPIO	Boot as defined by the values of BMSP1 and BMSP2 (BMSP0 disabled)
	Valid GPIO	Valid GPIO	Valid GPIO	Boot as defined by the values of BMSP0, BMSP1, and BMSP2
	Invalid GPIO	Valid GPIO	Valid GPIO	BMSP0 is reset to the factory default BMSP0 GPIO Boot as defined by the values of BMSP0, BMSP1, and BMSP2
	Valid GPIO	Invalid GPIO	Valid GPIO	BMSP1 is reset to the factory default BMSP1 GPIO Boot as defined by the values of BMSP0, BMSP1, and BMSP2
	Valid GPIO	Valid GPIO	Invalid GPIO	BMSP2 is reset to the factory default state, which is disabled Boot as defined by the values of BMSP0 and BMSP1

Note

When decoding the boot mode, BMSP0 is the least-significant-bit and BMSP2 is the most-significant-bit of the boot table index value. It is **recommended** when disabling BMSPs to start with disabling BMSP2. For example, in an instance when only using BMSP2 (BMSP1 and BMSP0 are disabled), then only the boot table indexes of 0 and 4 will be selectable. In the instance when using only BMSP0, then the selectable boot table indexes are 0 and 1.

4.5.2 Configuring Boot Mode Table Options

This section explains how to configure the boot definition table, BOOTDEF, for the device and the associated boot options. The 64-bit location is located in user-configurable DCSM OTP in the Z1-OTP-BOOTDEF-LOW and Z1-OTP-BOOTDEF-HIGH locations. When debugging, EMU-BOOTDEF-LOW and EMU-BOOTDEF-HIGH are the emulation equivalents of Z1-OTP-BOOTDEF-LOW and Z1-OTP-BOOTDEF-HIGH, and can be programmed to experiment with different boot mode options without writing to OTP. The range of customization to the boot definition table depends on how many boot mode select pins (BMSP) are being used. For example, 0 BMSPs equals to 1 table entry, 1 BMSP equals to 2 table entries, 2 BMSPs equals to 4 table entries, and 3 BMSPs equals to 8 table entries. Refer to Section 4.5.3 for examples on how to setup the BOOTPIN_CONFIG and BOOTDEF values.

Note

The locations Z2-OTP-BOOTDEF-LOW and Z2-OTP-BOOTDEF-HIGH will be used instead of Z1-OTP-BOOTDEF-LOW and Z1-OTP-BOOTDEF-HIGH locations when Z2-OTP-BOOTPIN-CONFIG is configured. Refer to Section 4.5.1 for more details on BOOTPIN_CONFIG usage.

Table 4-8. BOOTDEF Bit Fields

BOOTDEF Name	Byte Position	Name	Description
BOOT_DEF0	7:0	[3:0] BOOT_DEF0 Mode	Set the boot mode number from Section 4.4.2. Any unsupported boot mode will cause the device to either go to wait boot (debugger connected) or boot to Flash (standalone).
		[7:4] BOOT_DEF0 Options	Set alternate / additional boot options. This can include changing the GPIOs for a particular boot peripheral or specifying a different Flash entry point. Refer to Section 4.8.10 for valid BOOTDEF values to set in the table.
BOOT_DEF1	15:8	BOOT_DEF1 Mode/Options	Refer to BOOT_DEF0 description.
BOOT_DEF2	23:16	BOOT_DEF2 Mode/Options	
BOOT_DEF3	31:24	BOOT_DEF3 Mode/Options	
BOOT_DEF4	39:32	BOOT_DEF4 Mode/Options	
BOOT_DEF5	47:40	BOOT_DEF5 Mode/Options	
BOOT_DEF6	55:48	BOOT_DEF6 Mode/Options	
BOOT_DEF7	63:56	BOOT_DEF7 Mode/Options	

4.5.3 Boot Mode Example Use Cases

This section demonstrates some use cases for configuring the boot mode select pins and boot modes.

4.5.3.1 Zero Boot Mode Select Pins

This use case demonstrates a scenario for an application that does not use any boot mode select pins and always has the device boot to Flash.

1. Program the BOOTPIN_CONFIG location in OTP as follows:
   - Set BOOTPIN_CONFIG.BMSP0 to 0xFF
   - Set BOOTPIN_CONFIG.BMSP1 to 0xFF
   - Set BOOTPIN_CONFIG.BMSP2 to 0xFF
   - Set BOOTPIN_CONFIG.KEY to 0x5A for boot ROM to treat these register bits as valid and use the custom boot table.
2. Program the BOOTDEF location options for the device. This essentially sets up a device-specific boot mode table. Refer to Section 4.8.10 for valid BOOTDEF values to set in the table.
   - Set BOOTDEF.BOOTDEF0 to 0x03 for booting to Flash (entry address option 0). This sets Flash boot to boot table index 0.
   - Refer to Section 4.8.2 for the available Flash entry points.

Table 4-9. Zero Boot Pin Boot Table Result

Boot Mode Table Number	Boot Mode
0	Flash Boot (0x03)

4.5.3.2 One Boot Mode Select Pin

This use case demonstrates a scenario for an application using one boot mode select pin to select between booting to Flash or using CAN boot.

1. Program the BOOTPIN_CONFIG location in OTP as follows:
   - Set BOOTPIN_CONFIG.BMSP0 to a user specified GPIO, such as 0x0 for GPIO0
   - Set BOOTPIN_CONFIG.BMSP1 to 0xFF
   - Set BOOTPIN_CONFIG.BMSP2 to 0xFF
   - Set BOOTPIN_CONFIG.KEY to 0x5A for boot ROM to treat these register bits as valid and use the custom boot table.
2. Program the BOOTDEF location options for the device. This essentially sets up a device-specific boot mode table. Refer to Section 4.8.10 for valid BOOTDEF values to set in the table.
   - Set BOOTDEF.BOOTDEF0 to 0x02 for CAN booting. This sets CAN boot to boot table index 0.
   - Set BOOTDEF.BOOTDEF1 to 0x03 for booting to Flash (entry address option 0). This sets Flash boot to boot table index 1.

Table 4-10. One Boot Pin Boot Table Result

Boot Mode Table Number	Boot Mode
0	CAN Boot (0x02)
1	Flash Boot (0x03)

4.5.3.3 Three Boot Mode Select Pins

This use case demonstrates a scenario for an application using three boot mode select pins to select between various boot modes in the custom boot table.

1. Program the BOOTPIN_CONFIG location in OTP as follows:
   - Set BOOTPIN_CONFIG.BMSP0 to a user specified GPIO, such as 0x0 for GPIO0
   - Set BOOTPIN_CONFIG.BMSP1 to a user specified GPIO, such as 0x1 for GPIO1
   - Set BOOTPIN_CONFIG.BMSP2 to a user specified GPIO, such as 0x2 for GPIO2
   - Set BOOTPIN_CONFIG.KEY to 0x5A for boot ROM to treat these register bits as valid and use the custom boot table.
2. Program the BOOTDEF location options for the device. This essentially sets up a device-specific boot mode table. Refer to Section 4.8.10 for valid BOOTDEF values to set in the table.
   - Set BOOTDEF.BOOTDEF0 to 0x02 for CAN booting. This sets CAN boot to boot table index 0.
   - Set BOOTDEF.BOOTDEF1 to 0x03 for booting to Flash (entry address option 0). This sets Flash boot to boot table index 1.
   - Set BOOTDEF.BOOTDEF2 to 0x24 for booting to wait boot (alternate option). This sets wait boot to boot table index 2.
   - Set BOOTDEF.BOOTDEF3 to 0x66 for SPI booting (alternate GPIO option 3). This sets SPI boot to boot table index 3.
   - Set BOOTDEF.BOOTDEF4 to 0x43 for booting to Flash (entry address option 2). This sets Flash boot to boot table index 4.

Table 4-11. Three Boot Pins Boot Table Result

Boot Mode Table Number	Boot Mode
0	CAN Boot (0x02)
1	Flash Boot (0x03)
2	Wait Boot - Alt (0x24)
3	SPI - Alt3 (0x66)
4	Flash Boot - Alt2 (0x43)
5, 6, 7	Not used in this example

4.6 Device Boot Flow Diagrams

This section details the boot flow diagrams for standalone and emulation boot flows.

4.6.1 Boot Flow

Upon reset, the CPU follows the boot flow shown in Figure 4-1. Depending on whether a JTAG debugger is connected to the device, the CPU either continues booting following the emulation boot flow or the standalone boot flow.

Note

BOR follows same flow as POR.

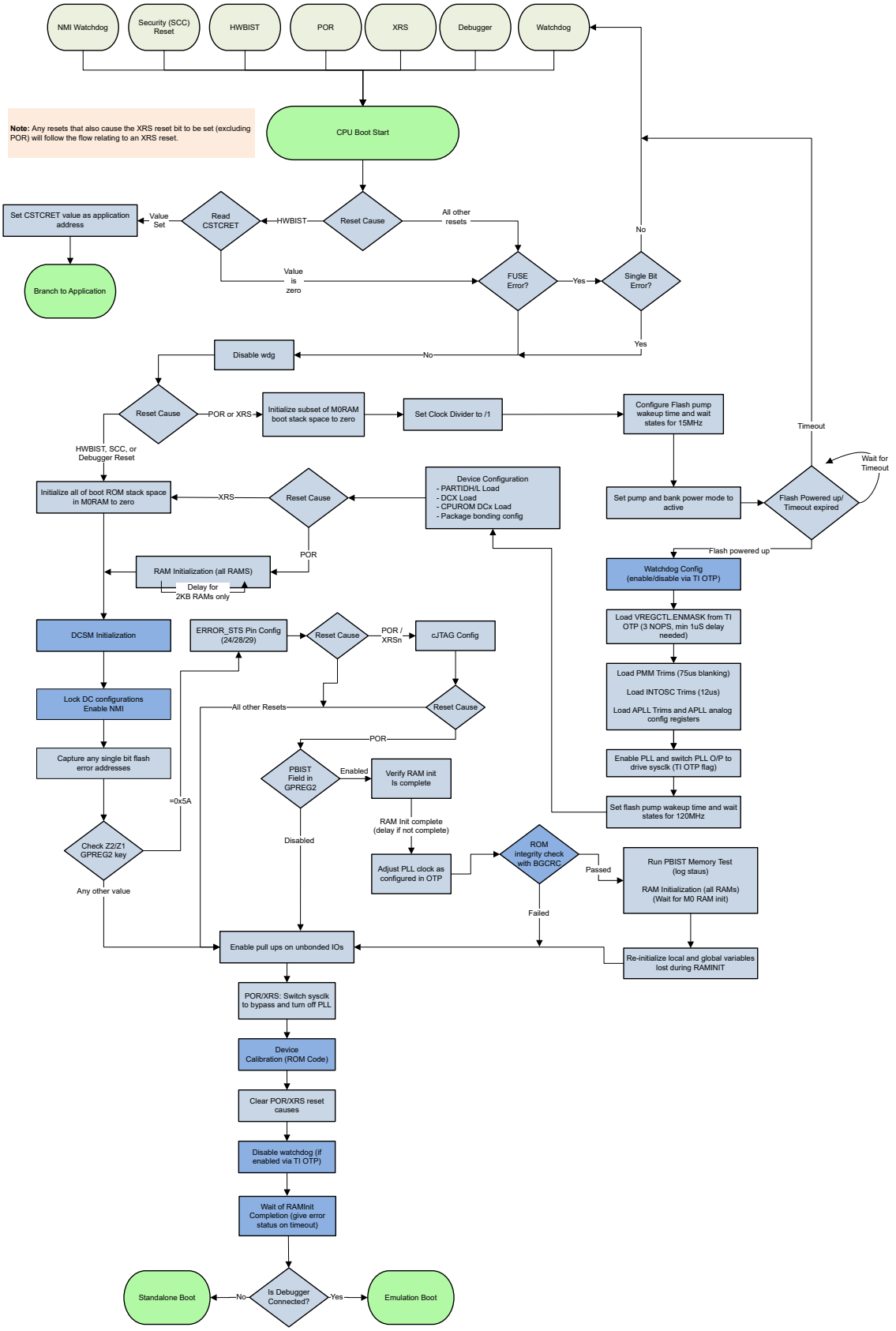


Figure 4-1. Device Boot Flow

4.6.2 Emulation Boot Flow

Figure 4-2 shows the emulation boot flow when JTAG debugger is connected.

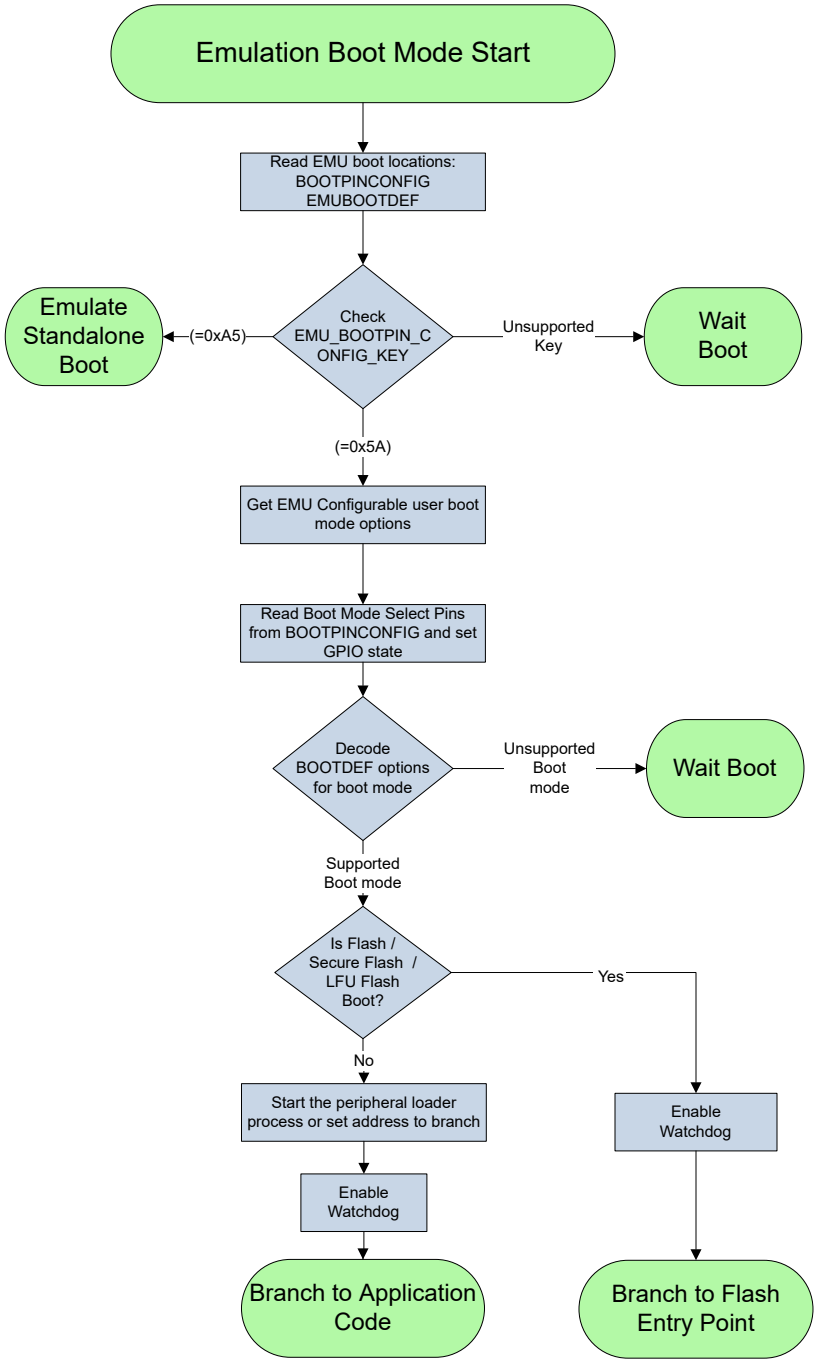


Figure 4-2. Emulation Boot Flow

4.6.3 Standalone Boot Flow

Figure 4-3 shows the standalone boot flow when no JTAG debugger is connected to the device.

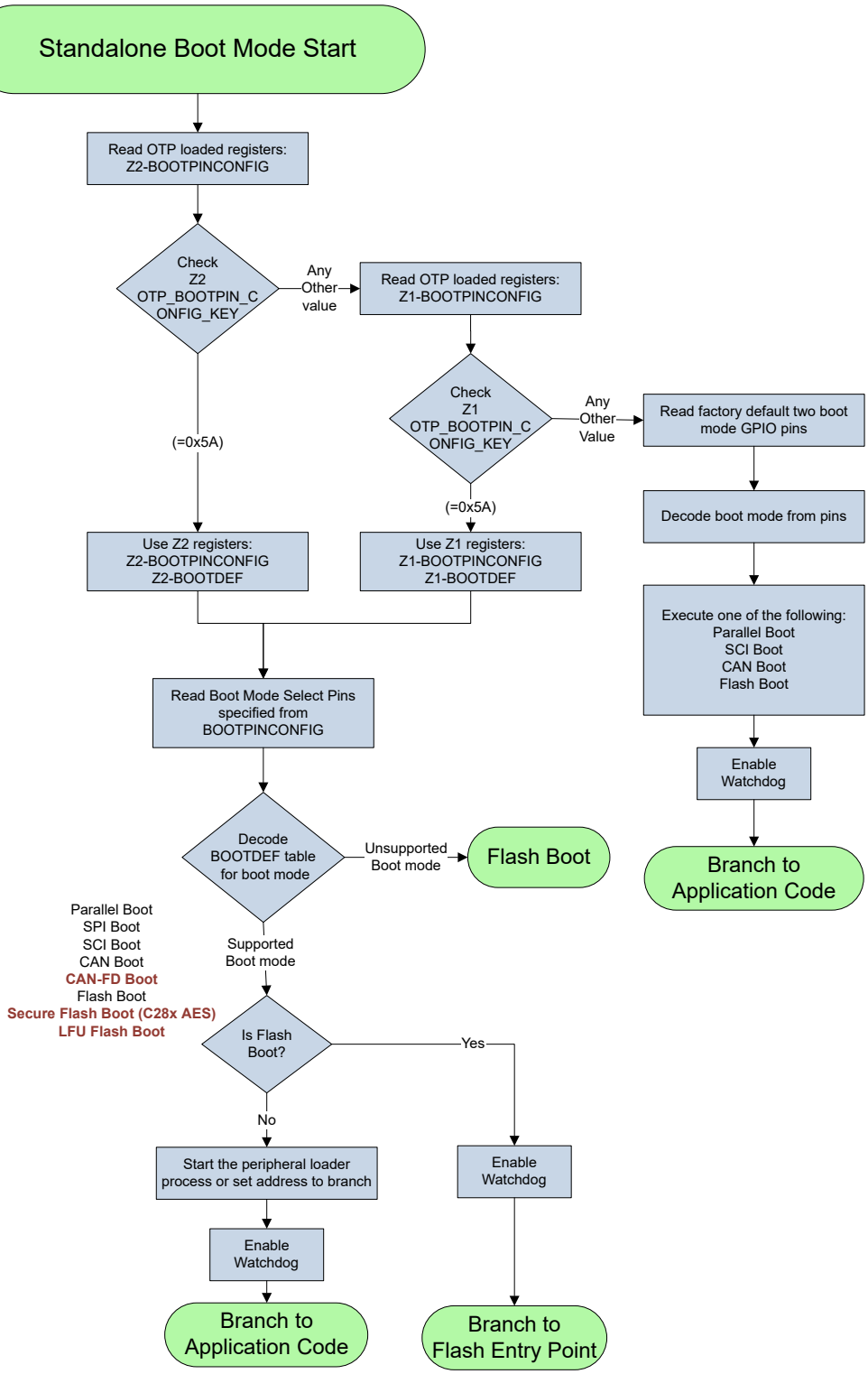


Figure 4-3. CPU Standalone Boot Flow

4.7 Device Reset and Exception Handling

4.7.1 Reset Causes and Handling

Table 4-12 explains the actions each boot ROM performs upon reset for a specific reset cause.

Table 4-12. Boot ROM Reset Causes and Actions

Reset Source	Boot ROM Action
Power on Reset (POR)	1. Configure Clock Divider 2. Flash Power Up 3. Device configuration and trimming 4. RAM Initialization 5. Continue default boot flow
External Reset (XRS) Includes: • Watchdog Reset • NMI Watchdog Reset • SIMRESET XRS	1. Configure Clock Divider 2. Flash Power Up 3. Device configuration and trimming 4. Clear RAM for boot stack 5. Continue default boot flow
Hardware Built-In Self Test (HWBIST)	1. Read HWBIST return address 2. If set, branch to address 3. If not set, continue boot following "Debugger" boot flow
Secure Copy Code (SCC) Reset	1. Clear RAM for boot stack 2. Continue default boot flow
SIMRESET	1. Clear RAM for boot stack 2. Continue default boot flow
Debugger Reset	1. Clear RAM for boot stack 2. Continue default boot flow

4.7.2 Exceptions and Interrupts Handling

Table 4-13 explains the actions boot ROM performs if any exceptions occur during boot. The exception handling philosophy in most cases, is to log the error and continue booting to reach the application.

Table 4-13. Boot ROM Exceptions and Actions

Exception Event Source	Boot ROM Action	Event Logged
Single-bit error in FUSEERR	Ignore and continue to boot	No
Multi-bit error in FUSEERR	Reset the device	No
Clock Fail	Clear the NMI flag and continue to boot	Yes
RAM Uncorrectable Error ROM Parity Error	Perform RAM initialization and reset the device	Yes(1)
Flash Uncorrectable Error	Reset the device	Yes
HWBIST Error	Clear the NMI flag and continue to boot	Yes
Embedded Real-time Analysis and Diagnostic (ERAD) NMI	Clear the NMI flag and continue to boot	Yes
RL NMI (CLB)	Clear the NMI flag and continue to boot	Yes
Software NMI error (Software self test error)	Reset the device	No
ITRAP Exception	Record memory address of where the illegal instruction was executed and let device reset	Yes
Unsupported PIE Interrupts	Ignore and continue to boot	No

(1) A RAM uncorrectable error or ROM parity error will clear the boot status information stored in RAM because a RAM initialization is performed to attempt to correct the error. Since the boot status information is erased, this exception can be identified in that a NMIWD reset occurred and all the RAMs are erased.

4.8 Boot ROM Description

This section explains the details regarding the device boot ROMs.

4.8.1 Boot ROM Configuration Registers

The boot ROM code involves several memory addresses and registers used during execution. There are two sets of configurations; one for emulation and one for standalone boot flow. The emulation locations located in RAM emulate the OTP configurations and can be written to as many times as needed. The user configurable DCSM OTP locations used in the standalone boot flow program the device OTP and hence can only be written once. Table 4-14 details these locations. For bit field configuration details for BOOTPIN-CONFIG and BOOTDEF, see Section 4.5.1 and Section 4.5.2.

Additionally, the boot ROM supports boot configurations from DCSM zone 1 and zone 2 registers. Zone 2 configurations will supercede zone 1 configurations, so it is recommended to use zone 1 configurations and use zone 2 as a secondary option.

Table 4-14. Boot ROM Registers

Boot Flow	Register Name	Boot ROM Name	Register Address	User OTP Address
Emulation	-	EMU-BOOTPIN-CONFIG	0x0000 0D00	-
	-	EMU-GPREG2	0x0000 0D02	-
	-	EMU-BOOTDEF-LOW	0x0000 0D04	-
	-	EMU-BOOTDEF-HIGH	0x0000 0D06	-
Standalone (Using Z1)	Z1-GPREG1	Z1-OTP-BOOTPIN-CONFIG	0x0005 F008	0x0007 8008
	Z1-GPREG2	Z1-OTP-BOOT-GPREG2	0x0005 F00A	0x0007 800A
	Z1-GPREG3	Z1-OTP-BOOTDEF-LOW	0x0005 F00C	0x0007 800C
	Z1-GPREG4	Z1-OTP-BOOTDEF-HIGH	0x0005 F00E	0x0007 800E
Standalone (Using Z2)	Z2-GPREG1	Z2-OTP-BOOTPIN-CONFIG	0x0005 F088	0x0007 8208
	Z2-GPREG2	Z2-OTP-BOOT-GPREG2	0x0005 F08A	0x0007 820A
	Z2-GPREG3	Z2-OTP-BOOTDEF-LOW	0x0005 F08C	0x0007 820C
	Z2-GPREG4	Z2-OTP-BOOTDEF-HIGH	0x0005 F08E	0x0007 820E

4.8.1.1 GPREG2 Usage and MPOST Configuration

Table 4-15 explains how the bit field values from the user configurable DCSM OTP location, Z1-OTP-BOOT-GPREG2 or Z2-OTP-BOOT-GPREG2, are decoded by boot ROM.

Table 4-15. DCSM Z1/Z2 GPREG2 Bit Fields

Bit	Name	Description	Boot ROM Action
31:24	Key	Write 0x5A to indicate to the boot ROM code that the bits in this register are valid.	If user sets to 0x5A, boot ROM uses the values in this register. If set to any other value, boot ROM ignores values in this register.
23:8	Reserved	Reserved	No Action
7:6	MPOST	0x0 = Run MPOST with PLL disabled (10-MHz internal oscillator)	When configured to a valid value, MPOST POR memory self-test will be run on all device memories
		0x1 = Run MPOST with PLL enabled for 95 MHz	
		0x2 = Run MPOST with PLL enabled for 47.5 MHz	
		0x3 = Disable MPOST	
5:4	ERROR_STS_PIN config	0x0 – GPIO24, MUX Option 13	This indicates which GPIO pin is supposed to be used as ERROR_PIN and boot ROM configures the mux as such for the said pin. The ERROR_STS pin mux configuration is locked by the boot ROM, but not committed.
		0x1 – GPIO28, MUX Option 13	
		0x2 – GPIO29, MUX Option 13	
		0x3 – ERROR_STS function disabled (default)	
3:0	CJTAGNODEID	CJTAGNODEID[3:0]	Boot ROM takes this values and programs the lower 4 bits of the CJTAGNODEID register.

4.8.2 Entry Points

This sections gives details about the entry point addresses for various boot modes. These entry points direct the boot ROM what address to branch to at the end of booting as per the selected boot mode.

Table 4-16 gives details about the entry point addresses for Flash boot mode.

Table 4-17 gives details about the entry point addresses for RAM boot mode.

Table 4-16. Flash Entry Point Addresses

Option	BOOTDEFx Value	Flash Sector	Address	Packages Supported
0	0x03	CPU Bank 0 Sector 0	0x0008 0000	All
1	0x23	CPU Bank 0 Sector 8	0x0008 8000	All
2	0x43	CPU Bank 0 End of Sector 15	0x0008 FFF0	All
3	0x63	CPU Bank 1 Sector 0	0x0009 0000	All
4	0x83	CPU Bank 1 End of Sector 7	0x0009 7FF0	All
5	0xA3	CPU Bank 1 End of Sector 15	0x0009 FFF0	All
6	0xC3	CPU Bank 2 Sector 0	0x000A 0000	All
7	0xE3	CPU Bank 2 End of Sector 15	0x000A FFF0	All

Table 4-17. RAM Entry Point Address

Option	BOOTDEFx Value	RAM Entry Point	Package Supported
0	0x05	0x0000 0000	All

4.8.3 Wait Points

The wait mode puts the CPU in a loop in the boot ROM code and won't branch to the user application code. The device can enter wait boot mode either through manually being set or because of some issue during boot up. Using wait boot mode is recommended when using a debugger to avoid any JTAG issues. There will an ESTOP provided for debugging during Wait boot.

Table 4-18. Wait Boot Options

Option	BOOTDEFx Value	Watchdog Status	Package Supported
0 (default)	0x04	Enabled	All
1	0x24	Disabled	All

During boot ROM execution, there are situations where the CPU may enter a wait loop in the code. This state can occur for a variety of reasons. details the address ranges that the CPU PC register value will fall between if it has entered one of these instances.

Following are the actions for entering wait boot mode:

- Wait boot is set by the user as the boot mode.
- Boot mode is unrecognizable and a debugger is connected to the device.
- The emulation BOOTPIN_CONFIG key isn't equal to 0xA5 or 0x5A.
- An error occurs during emulation boot and the boot mode pins are decoded with a value not recognized as a valid boot mode.

Table 4-19. Wait Point Addresses

Address Range	Description
0x3FB8B9 – 0x3FB8C0	In Wait Boot Mode
0x3FC7D0 – 0x3FC7D8	In SCI Boot waiting on autobaud lock
0x3FEDFE – 0x3FEEC8	In NMI Handler
0x3FEEC9 – 0x3FEEF9	In ITRAP ISR
0x3FCB96 - 0x3FCB9A	In Parallel boot waiting for control signal

4.8.4 Secure Flash Boot

Secure Flash boot mode is similar to Flash boot mode in that the boot flow branches to the configured memory address in Flash except only after the Flash memory contents have been authenticated. The Flash authentication uses a Cipher-based Message Authentication Protocol (CMAC) to authenticate 16 KB of Flash starting from the configured Flash entry point address. The CMAC calculation requires a user-defined 128-bit key programmed in the CPU User OTP Zone 1 Header OTP CMACKEY bit field. Additionally, the user must calculate the golden CMAC tag based on the 16 KB Flash memory range and store it along with the user code at a hardcoded address in Flash. During secure Flash boot, the calculated CMAC tag is compared to the user golden CMAC tag in Flash to determine the pass/fail status of the CMAC authentication. When authentication passes, boot flow continues and branches to Flash to begin executing the application. When authentication fails, the device is reset.

For the available secure Flash boot entry address options, refer to Section 4.8.2.

For generating the secure Flash golden CMAC tag for CPU, refer to the *TMS320C28x Assembly Language Tools User's Guide* within section "Using Secure Flash Boot on TMS320F2838x Devices" for instructions.

Note

Both the CMAC golden signature and CMAC key are stored in the most-significant double format, but each 32-bit section is in little-endian format.

Key: 2B7E1516 28AED2A6 ABF71588 09CF4F3C

(MSB is 2B and LSB is 3C)

CMACKEY0 = 0x2B7E1516

CMACKEY1 = 0x28AED2A6

CMACKEY2 = 0xABF71588

CMACKEY3 = 0x09CF4F3C

Note

User must ensure that the Flash sector that encompasses the configured Flash entry point and the first 16 KB of Flash is assigned to Zone 1 for the core setup for secure Flash boot.

Recommended to use device JTAGLOCK when using secure Flash boot.

APIs for CMAC calculation and authentication is provided as part of ROM. Details are available in Section 4.8.11

Table 4-20. Secure Flash Boot Details

Details	Location Address
CMAC Signature Address	Flash Entry Point Address + 0x2
CMAC Key Address (128-bit key)	DCSM Z1 OTP CMACKEY0/1/2/3
Flash Entry Point (Bank 0, Sector 0)	0x0008 0000
Flash Entry Point (Bank 0, Sector 8)	0x0008 8000
Flash Entry Point (Bank 0, End of Sector 15)	0x0008 FFF0
Flash Entry Point (Bank 1, Sector 0)	0x0009 0000
Flash Entry Point (Bank 1, End of Sector 7)	0x0009 7FF0
Flash Entry Point (Bank 1, End of Sector 15)	0x0009 FFF0
Flash Entry Point (Bank 2, Sector 0)	0x000A 0000
Address Range for CMAC Calculation	Start: Flash Entry Point Address End: Flash Entry Point Address + 16 KB

Table 4-21. Secure Flash Tag and Key Details

Name	Address	Details
CMAC Golden Tag (128-bit)	**CPU:** *Flash Entry Point Address* + 0x2	Located in Flash, offset from the entry point address, by 2 words (CPU). When CMAC calculations are performed, the golden tag location in memory is considered all 0xFs. Refer to Example 4-1 for an example regarding linker configuration on CPU. Lower memory contains the tag's MSW and higher memory contains the LSW. **Example (on CPU):** Tag = 0x00112233 44556677 8899AABB CCDDEEFF Address 0x0 = 0x00112233 Address 0x2 = 0x44556677 Address 0x4 = 0x8899AABB Address 0x6 = 0xCCDDEEFF

Table 4-21. Secure Flash Tag and Key Details (continued)

Name	Address	Details
CMAC 128-Bit Key	0x0007 8018	Located in CPU Zone 1 User Header OTP (CMACKEY0, CMACKEY1, CMACKEY2, CMACKEY3) CMACKEY0 contains the key's MSW and CMACKEY3 contains the LSW. **Example:** Key = 0x00112233 44556677 8899AABB CCDDEEFF CMACKEY0 = 0x00112233 CMACKEY1 = 0x44556677 CMACKEY2 = 0x8899AABB CMACKEY3 = 0xCCDDEEFF

Table 4-22. Secure Flash Authentication Failure Actions

CPU	Action on Failed Authentication
C28x CPU	1. Emulation only - Halt debugger (ESTOP) 2. Wait in endless loop (for device reset due to WD reset)

Table 4-23. Secure Flash on all CPUs Recommended Flow

Step	Action
1	Secure Flash boot CPU
2	Any Flash beyond the first 16 KB from the entry point that is planned for use should be authenticated by the user using a different CMAC golden tag embedded at an address somewhere within the already authenticated 16 KB of Flash.

Example 4-1. Secure Flash CPU1 Linker File Example

```
MEMORY
{
/* Code Start branch to _c_int00 */
BEGIN : origin = 0x80000, length = 0x0002
/* User calculated golden CMAC tag for Flash Sector 0 */
GOLDEN_CMAC_TAG : origin = 0x80002, length = 0x0008
/* Flash Sector 0 containing application code */
FLASH_SECTOR_0 : origin = 0x8000A, length = 0x1FF6
.
.
.
}
```

4.8.5 Live Firmware Update (LFU) Flash Boot

Live Firmware Update (LFU) boot mode is required to handle the Live Firmware Update feature. In this boot mode, the boot loader reads the version number from images present in multiple banks and identifies the latest image. The boot ROM then hands off the execution to the application with the latest version. To support LFU boot mode each Flash bank containing an application image needs a few fields as shown in Table 4-24.

Table 4-24. LFU Application Image Format

Image Address Offset	Content
0x0	Application entry point (32-bit)
0xA	Key (32-bit) Valid Key = 0x5A5A5A5A
0xC	Firmware version number (32-bit)

Application entry point: This is the code execution start address of the image stored in Flash.

Key: This 32-bit field determines if this image is valid. The image in a bank is considered valid only if the location contains the value 0x5A5A5A5A. In case all banks have invalid keys, an error is flagged in boot_status variable and program jumps to a while loop in standalone boot mode (ESTOP in emulation boot mode).

Firmware version number: This 32-bit field is the version number of the firmware or application. 0xFFFF FFFF is considered as the initial value and this needs to be decremented after every update. The image with lower version number is the latest application. If all valid images have same version number, then bank-0 (or the lowest numbered bank) is chosen.

For example, if bank-0 has invalid **Key** and bank-1 and bank-2 have valid keys, then the one having lowest **Firmware version number** will be selected for boot. If both are same, then bank-1 will be selected.

Table 4-25 shows the entry points for LFU boot mode.

Table 4-25. LFU Entry Point Addresses

Option	BOOTDEFx Value	Bank 0	Bank 1	Bank 2
0	0x0B	0x0008 0000	0x0009 0000	0x000A 0000
1	0x2B	0x0008 8000	0x0009 8000	0x000A 8000
2	0x4B	0x0008 FFF0	0x0009 FFF0	0x000A FFF0
3	0x6B	0x0008 8000	0x0009 0000	0x000A 0000
4	0x8B	0x0008 EFF0	0x0009 7FF0	0x0008 7FF0

For example, if Option 0 with Bank 0 has the application image:

[0008 0000-0008 0001] = Application entry point

[0008 000A-0008 000B] = Key

[0008 000C-0008 000D] = Firmware version number

4.8.6 Secure LFU Flash Boot

Secure LFU Flash boot performs the same Flash bank selection as the non-secure LFU described in Section 4.8.5. Once the bank entry address is determined, this address + 16 KB is authenticated using a CMAC algorithm. The CPUBROM_verifySecureFlash and CMAC algorithms are reused from the Section 4.8.4 to perform this authentication.

The flow is:

1. Secure LFU boot mode selected.
2. Perform LFU bank selection.
3. Perform CMAC on selected entry address + 16 KB.
4. Boot to Flash bank.

To support Secure LFU Flash boot mode, the application image has to have a few fields as shown in Table 4-26.

Table 4-26. Secure LFU Application Image Format

Image Address Offset	Content
0x0	Application entry point (32-bit)
0x2	Golden CMAC Tag (128-bits)
0xA	Key (32-bit) Valid Key = 0x5A5A5A5A
0xC	Firmware version number (32-bit)

Application entry point: This is the code execution start address of the image stored in Flash.

Key: This 32-bit field determines if this image is valid. The image in a bank is considered valid only if the location contains the value 0x5A5A5A5A. In case all banks have invalid keys, an error is flagged in boot_status variable and program jumps to a while loop in standalone boot mode (ESTOP in emulation boot mode).

Firmware version number: This 32-bit field is the version number of the application. 0xFFFF FFFF is considered as the initial value and this needs to be decremented after every update. The image with lower version number is the latest application. If all valid images have same version number, then bank-0 (or the lowest numbered bank) is chosen.

Golden CMAC Tag: This 128-bit field stores the calculated golden CMAC tag that is used as part of the secure Flash CMAC authentication.

Table 4-27 shows the entry points for LFU boot mode.

Table 4-27. Secure LFU Entry Point Addresses

Option	BOOTDEFx Value	Bank 0	Bank 1	Bank 2
0	0x0C	0x0008 0000	0x0009 0000	0x000A 0000
1	0x2C	0x0008 8000	0x0009 8000	0x000A 8000
3	0x6C	0x0008 8000	0x0009 0000	0x000A 0000
4	0x8C	0x0008 EFF0	0x0009 7FF0	0x0008 7FF0

Note

Secure LFU boot mode has the same bank addresses as non-secure LFU boot mode, except boot option 2 is not supported due to CMAC requirements.

For example, if Option 0 with Bank 0 has the application image:

[0008 0000-0008 0001] = Application entry point

[0008 0002-0008 0009] = Golden CMAC Tag

[0008 000A-0008 000B] = Key

[0008 000C-0008 000D] = Firmware version number

4.8.7 Memory Maps

4.8.7.1 Boot ROM Memory Maps

Table 4-28 details the ROM memory maps.

Table 4-28. Boot ROM Memory Map

Memory	Origin Address	Length (Words)
ROM Signature	0x003F 0000	0x0002
Golden CRC	0x003F 0002	0x0002
Full ROM Checksum	0x003F 0040	0x0042
Version	0x003F 0082	0x0004
IQmath Tables	0x003F 812C	0x1674
FPU32 Fast Tables	0x003F 97A0	0x081A
FPU32 Twiddle Tables	0x003F 9FBA	0x0DF8
Boot Code	0x003F ADB2	0x4000
Interrupt Handlers	0x003F EDB2	0x01AE
CPU Fast Data ROM (InstaSPIN)	0x003F EF60	0x0100
RTS Lib	0x003F F060	0x0098
Full ROM Checksum	0x003F 8380	0x0042
CRC Table	0x003F F0F8	0x0008
Vector Table	0x003F FFBE	0x0042

Table 4-29. Secure ROM Memory Map

Memory	Origin Address	Length
Zone 1 Secure-Copy / CRC Code	0x3F2000	0x0800
Zone 1 Secure Flash Boot	0x3F2800	0x0800
Zone 2 Secure-Copy / CRC Code	0x3F3000	0x0800
Z1 Secure EXE ROM - InstaSPIN	0x3F3800	0x47F0
RESERVED	0x3EFFF0	0x0010

4.8.7.2 CLA Data ROM Memory Maps

Note

In Table 4-30, **Load** refers to the memory addresses where the C28x CPU can view the data. **Run** refers to the CLA memory addresses that the CLA uses to access the data.

Table 4-30. CLA Data ROM Memory Map

Memory	Origin Address	Length (Words)
FFT Tables (Load)	0x0100 1070	0x0800
Data (Load)	0x0100 1870	0x078A
Version (Load)	0x0100 1FFA	0x0006
FFT Tables (Run)	0x0000 F070	0x0800
Data (Run)	0x0000 F870	0x078A
Version (Run)	0x0000 FFFA	0x0006

4.8.7.3 Reserved RAM Memory Maps

Table 4-31 details memory usage in RAM that is reserved for boot ROM to use. These memory sections should be reserved in the user application.

Table 4-31. Reserved RAM Memory Map

Memory	Description	Origin Address	Length (Words)
RAM	Boot Status, Boot Mode, MPOST Status, Boot Stack	0x0000 0002	0x0126

4.8.8 ROM Tables

Table 4-32 details the boot ROM symbol libraries that can be integrated into an application to use the available ROM functions and tables.

Table 4-32. ROM Symbol Tables

ROM Symbols	Library Name	Location
ROM Bootloaders and Functions	F28003xCPU_BootROM_Symbols	Under */libraries/boot_rom* in C2000Ware
FPU32 Tables	F28003xCPU_BootROM_Symbols	
IQmath	F28003xCPU_IQMathROM_Symbols	

4.8.9 Boot Modes and Loaders

The available boot modes and bootloaders supported on this device are detailed in this section.

4.8.9.1 Boot Modes

This section details the available boot modes that do not involve a peripheral boot loader. Table 4-33 details the available boot modes that do not involve a peripheral boot loader.

Table 4-33. Boot Mode Availability

Boot Mode	CPU Support
Flash Boot	C28x CPU
RAM Boot	C28x CPU
Wait Boot	C28x CPU
Secure Flash Boot	C28x CPU
LFU Flash Boot	C28x CPU
Secure LFU Flash Boot	C28x CPU

4.8.9.1.1 Flash Boot

Flash boot mode branches to the configured memory address in Flash. Refer to Section 4.8.2 for all the available Flash address options.

4.8.9.1.2 RAM Boot

RAM boot mode branches to the configured memory address in RAM. Refer to Section 4.8.2 for all the available RAM address options.

4.8.9.1.3 Wait Boot

Wait boot mode branches to the memory address as mentioned in Section 4.8.3.

Table 4-34. Wait Boot Options

Option	BOOTDEFx Value	Watchdog Status	Package Supported
0 (default)	0x04	Enabled	All
1	0x24	Disabled	All

4.8.9.2 Bootloaders

This section details the available boot modes that use a peripheral boot loader. For more specific details on the supported data stream structure used by the following bootloaders, refer to Section 4.9.1.

4.8.9.2.1 SCI Boot Mode

The SCI boot mode asynchronously transfers code from SCI-A to internal memory. This boot mode only supports an incoming 8-bit data stream and follows the data flow as shown in Figure 4-4.

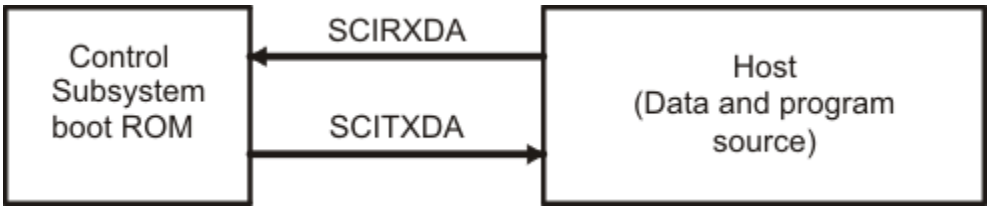


Figure 4-4. Overview of SCI Bootloader Operation

The device communicates with the external host by communication through the SCI-A peripheral. The autobaud feature of the SCI port is used to lock baud rates with the host. For this reason the SCI loader is very flexible and you can use a number of different baud rates to communicate with the device.

After each data transfer, the bootloader will echo back the 8-bit character received to the host. This allows the host to check that each character was received by the bootloader.

At higher baud rates, the slew rate of the incoming data bits can be affected by transceiver and connector performance. While normal serial communications may work well, this slew rate may limit reliable auto-baud detection at higher baud rates (typically beyond 100kbaud) and cause the auto-baud lock feature to fail. To avoid this, the following is recommended:

1. Achieve a baud-lock between the host and SCI bootloader using a lower baud rate.
2. Load the incoming application or custom loader at this lower baud rate.
3. The host may then handshake with the loaded application to set the SCI baud rate register to the desired high baud rate.

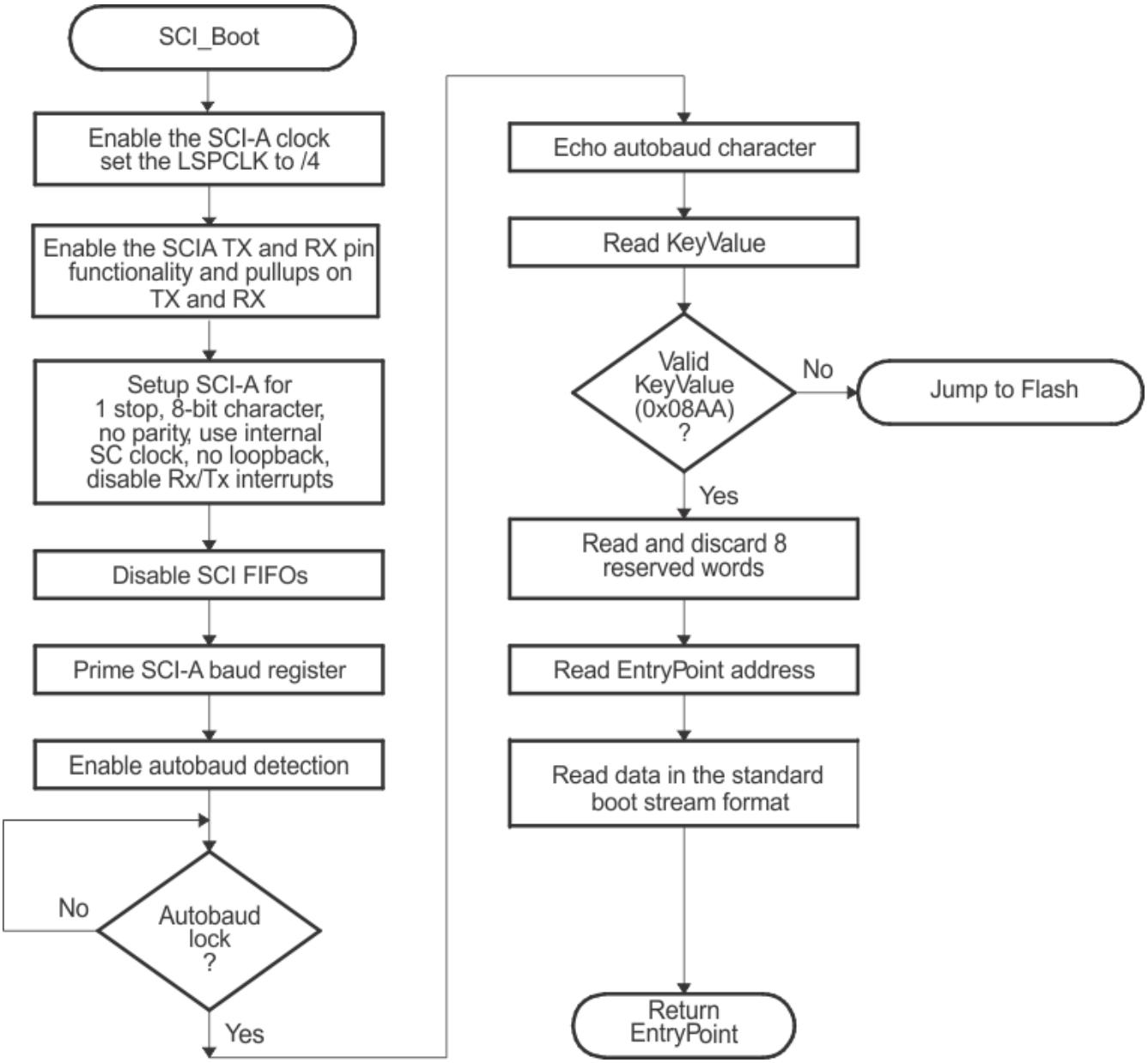


Figure 4-5. Overview of SCI Boot Function

4.8.9.2.2 SPI Boot Mode

The SPI loader expects an SPI-compatible 16-bit or 24-bit addressable serial EEPROM or serial Flash device to be present on the SPI-A pins as shown in Figure 4-6. The SPI bootloader supports an 8-bit data stream. It does not support a 16-bit data stream.

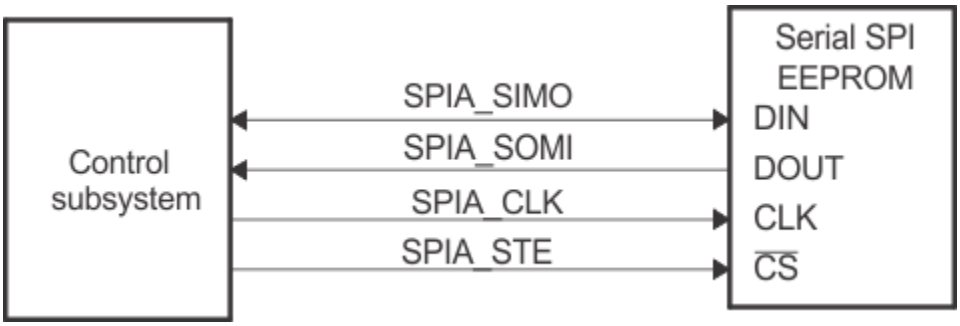


Figure 4-6. Overview of SPI Bootloader Operation

The SPI boot ROM loader initializes the SPI module to interface to a serial SPI EEPROM or Flash. Devices of this type include, but are not limited to, the Xicor X25320 (4Kx8) and Xicor X25256 (32Kx8) SPI serial SPI EEPROMs and the Atmel AT25F1024A serial Flash.

The SPI boot ROM loader initializes the SPI with the following settings: FIFO enabled, 8-bit character, internal SPICLK master mode and talk mode, clock phase = 1, polarity = 0, using the slowest baud rate.

If the download is to be performed from an SPI port on another device, then that device must be set up to operate in the slave mode and mimic a serial SPI EEPROM. Immediately after entering the SPI_Boot function, the pin functions for the SPI pins are set to primary and the SPI is initialized. The initialization is done at the slowest speed possible. Once the SPI is initialized and the key value read, you could specify a change in baud rate or low speed peripheral clock. Table 4-35 shows the 8-bit data stream used by the SPI.

Table 4-35. SPI 8-Bit Data Stream

Byte	Contents
1	LSB: AA (KeyValue for memory width = 8-bits)
2	MSB: 08h (KeyValue for memory width = 8-bits)
3	LSB: LOSPCP
4	MSB: SPIBRR
5	LSB: reserved for future use
6	MSB: reserved for future use
...	Reserved
17	LSB: reserved for future use
18	MSB: reserved for future use
19	LSB: Upper half (MSW) of Entry point PC[23:16]
20	MSB: Upper half (MSW) of Entry point PC[31:24] (Note: Always 0x00)
21	LSB: Lower half (LSW) of Entry point PC[7:0]
22	MSB: Lower half (LSW) of Entry point PC[15:8]
...	....
...	Data for this section.
	...
...	Blocks of data in the format size/destination address/data as shown in the generic data stream description
...	...
...	Data for this section.
n	LSB: 00h

Table 4-35. SPI 8-Bit Data Stream (continued)

Byte	Contents
n+1	MSB: 00h - indicates the end of the source

The data transfer is done in "burst" mode from the serial SPI EEPROM. The transfer is carried out entirely in byte mode (SPI at 8 bits/character). A step-by-step description of the sequence is:

1. The SPI-A port is initialized.
2. The GPIO pin, as defined by SPI option configured from Table 4-45, is used as a chip-select for the serial SPI EEPROM or Flash.
3. The SPI-A outputs a read command for the serial SPI EEPROM or Flash.
4. The SPI-A sends the serial SPI EEPROM an address 0x0000; that is, the host requires that the EEPROM or Flash must have the downloadable packet starting at address 0x0000 in the EEPROM or Flash. The loader is compatible with both 16-bit addresses and 24-bit addresses.
5. The next word fetched must match the key value for an 8-bit data stream (0x08AA). The least significant byte of this word is the byte read first and the most significant byte is the next byte fetched. This is true of all word transfers on the SPI. If the key value does not match, then the load is aborted and the bootloader jumps to Flash.
6. The next two bytes fetched can be used to change the value of the low speed peripheral clock register (LOSPCP) and the SPI baud rate register (SPIBRR). The first byte read is the LOSPCP value and the second byte read is the SPIBRR value. The next seven words are reserved for future enhancements. The SPI bootloader reads these seven words and discards them.
7. The next two words makeup the 32-bit entry point address where execution will continue after the boot load process is complete. This is typically the entry point for the program being downloaded through the SPI port.
8. Multiple blocks of code and data are then copied into memory from the external serial SPI EEPROM through the SPI port. The blocks of code are organized in the standard data stream structure presented earlier. This is done until a block size of 0x0000 is encountered. At that point in time the entry point address is returned to the calling routine that then exits the bootloader and resumes execution at the address specified.

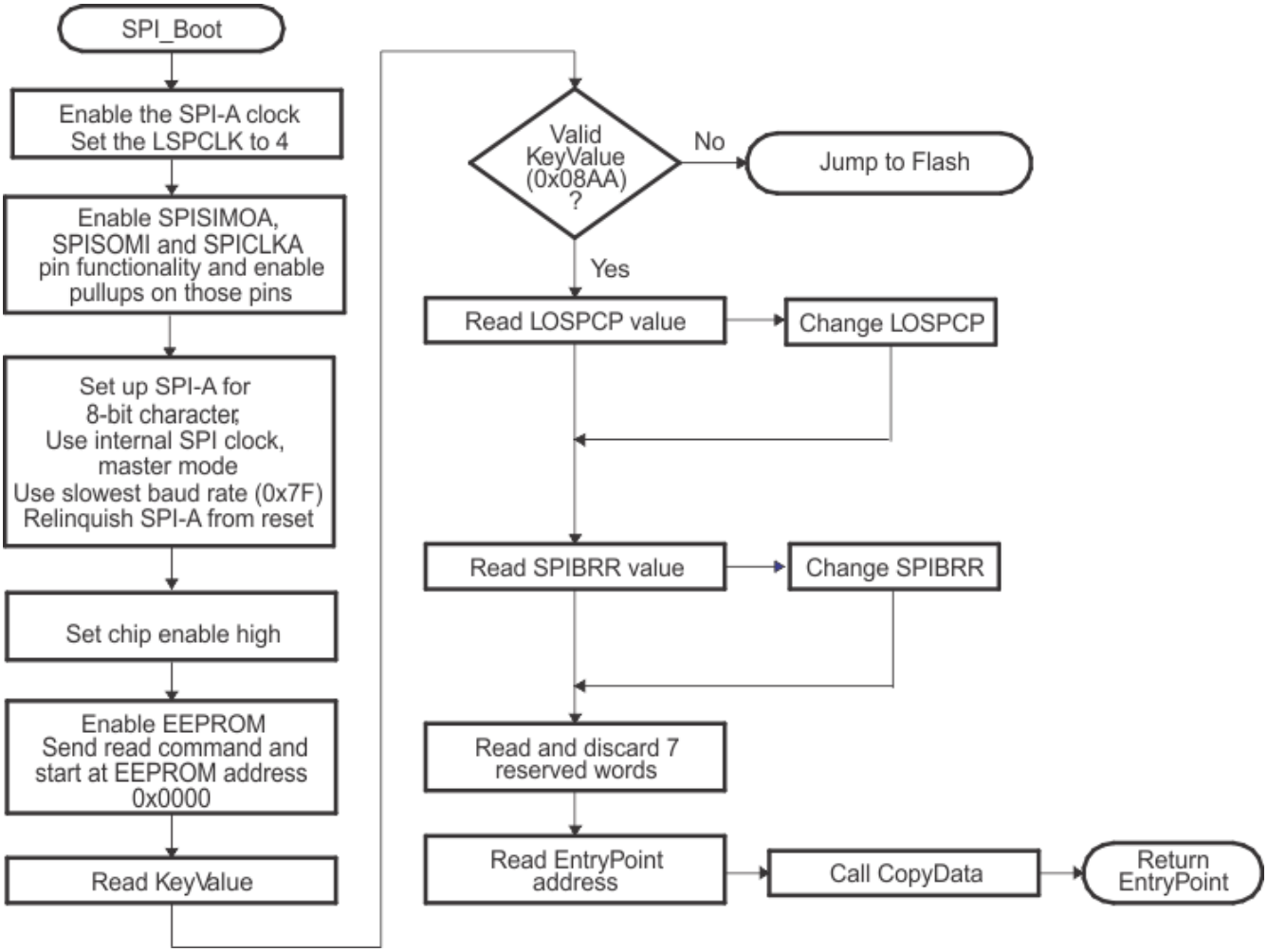


Figure 4-7. Data Transfer from EEPROM Flow

4.8.9.2.3 I2C Boot Mode

The I2C bootloader expects an 8-bit wide I2C-compatible EEPROM device to be present at address 0x50 on the I2C-A bus as shown in Figure 4-8. The EEPROM must adhere to conventional I2C EEPROM protocol, as described in this section, with a 16-bit base address architecture.

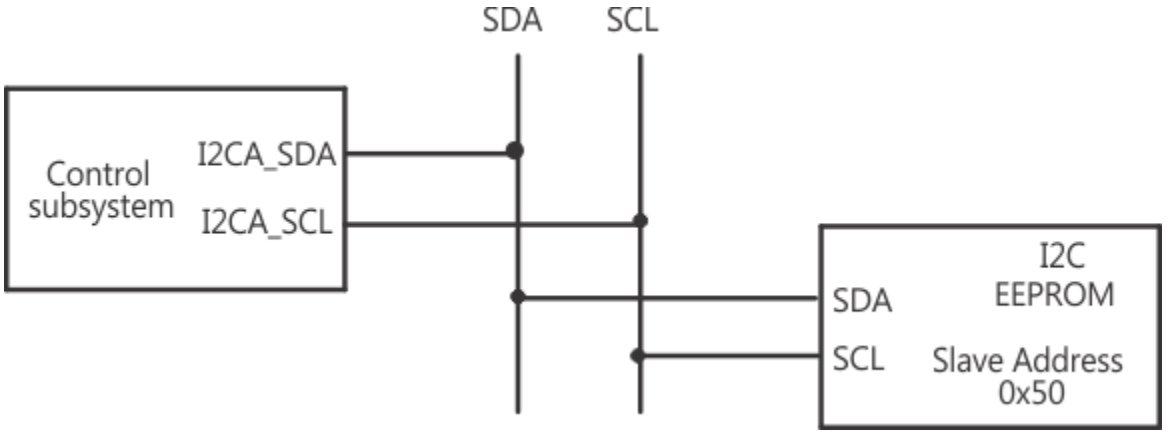


Figure 4-8. EEPROM Device at Address 0x50

If the download is to be performed from a device other than an EEPROM, then that device must be set up to operate in the slave mode and mimic the I2C EEPROM. Immediately after entering the I2C boot function, the GPIO pins are configured for I2C-A operation and the I2C is initialized. The following requirements must be met when booting from the I2C module:

- The input frequency to the device must be in the appropriate range.
- The EEPROM must be at slave address 0x50.

The bit-period prescalers (I2CCLKH and I2CCLKL) are configured by the bootloader to run the I2C at a 50 percent duty cycle at 100-kHz bit rate (standard I2C mode) when the system clock is 10 MHz. These registers can be modified after receiving the first few bytes from the EEPROM. This allows the communication to be increased up to a 400-kHz bit rate (fast I2C mode) during the remaining data reads.

Arbitration, bus busy, and slave signals are not checked. Therefore, no other master is allowed to control the bus during this initialization phase. If the application requires another master during I2C boot mode, that master must be configured to hold off sending any I2C messages until the application software signals that it is past the bootloader portion of initialization.

The non-acknowledgment bit is checked only during the first message sent to initialize the EEPROM base address. This is to make sure that an EEPROM is present at address 0x50 before continuing. If an EEPROM is not present, the non-acknowledgment bit is not checked during the address phase of the data read messages (I2C_Get Word). If a non acknowledgment is received during the data read messages, the I2C bus will hang. Table 4-36 shows the 8-bit data stream used by the I2C.

The I2C EEPROM protocol required by the I2C bootloader is shown in Figure 4-10 and Figure 4-11. The first communication, which sets the EEPROM address pointer to 0x0000 and reads the KeyValue (0x08AA) from it, is shown in Figure 4-10. All subsequent reads are shown in Figure 4-11 and are read two bytes at a time.

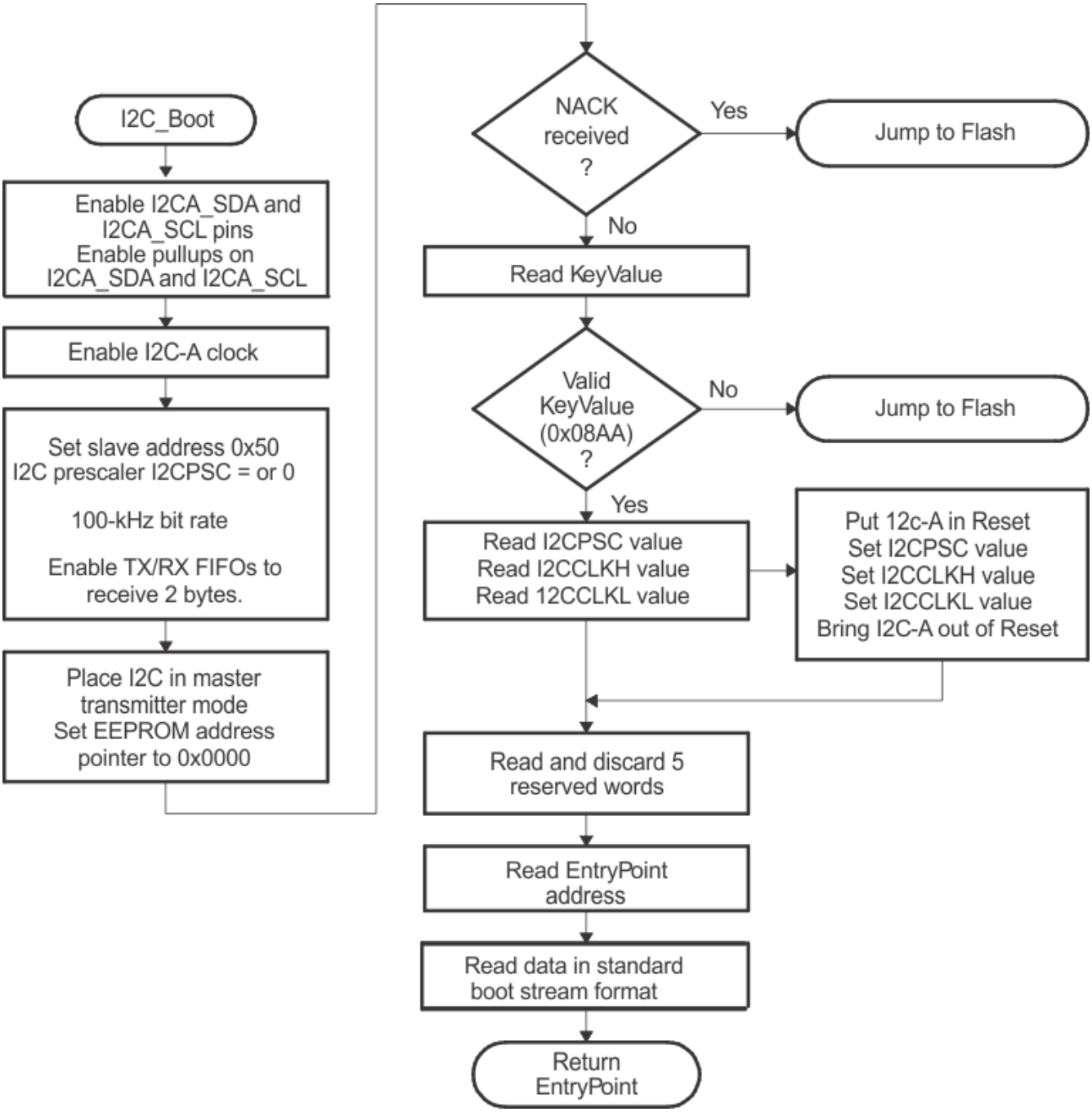


Figure 4-9. Overview of I2C Boot Function

Table 4-36. I2C 8-Bit Data Stream

Byte	Contents
1	LSB: AA (KeyValue for memory width = 8 bits)
2	MSB: 08h (KeyValue for memory width = 8 bits)
3	LSB: I2CPSC[7:0]
4	Reserved
5	LSB: I2CCLKH[7:0]
6	MSB: I2CCLKH[15:8]
7	LSB: I2CCLKL[7:0]
8	MSB: I2CCLKL[15:8]
...	...
...	Data for this section.
	...
17	LSB: Reserved for future use
18	MSB: Reserved for future use
19	LSB: Upper half of entry point PC
20	MSB: Upper half of entry point PC[22:16] (Note: Always 0x00)
21	LSB: Lower half of entry point PC[15:8]
22	MSB: Lower half of entry point PC[7:0]
...	...
...	Data for this section.
	...
	Blocks of data in the format size/destination address/data as shown in the generic data stream description.
...	...
...	Data for this section.
n	LSB: 00h
n+1	MSB: 00h - indicates the end of the source

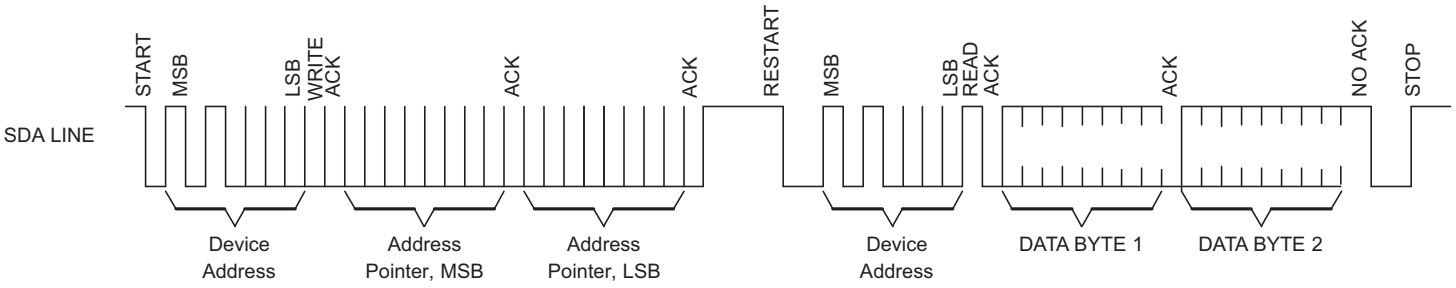


Figure 4-10. Random Read

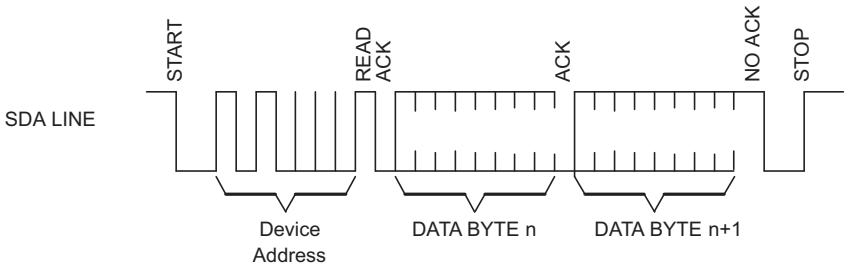


Figure 4-11. Sequential Read

4.8.9.2.4 Parallel Boot Mode

The parallel general purpose I/O (GPIO) boot mode asynchronously transfers code from host to C28x's internal memory. Each value is eight bits long and follows the same data flow as outlined in Figure 4-12.

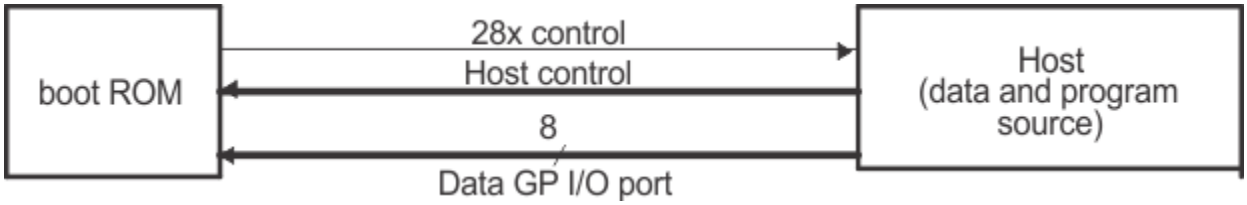


Figure 4-12. Overview of Parallel GPIO Bootloader Operation

The control subsystem communicates with the external host device by polling/driving the Host Control and 28x control lines. The handshake protocol shown in Figure 4-13 must be used to successfully transfer each word via GPIO [D0:D7]. This protocol is very robust and allows for a slower or faster host to communicate with the master subsystem.

Two consecutive 8-bit words are read to form a single 16-bit word. The least significant byte (LSB) is read first followed by the most significant byte (MSB). In this case, data is read from GPIO[D0:D7].

The 8-bit data stream is shown in Table 4-37.

Table 4-37. Parallel GPIO Boot 8-Bit Data Stream

Bytes		GPIO[D0:D7] (Byte 1 of 2)	GPIO[D0:D7] (Byte 2 of 2)	Description
1	2	AA	08	0x08AA (KeyValue for memory width = 16 bits)
3	4	00	00	8 reserved words (words 2 - 9)
...	...	...	...	...
17	18	00	00	Last reserved word
19	20	BB	00	Entry point PC[22:16]
21	22	DD	CC	Entry point PC[15:0] (PC = 0x00BBCCDD)
23	24	NN	MM	Block size of the first block of data to load = 0xMMNN words
25	26	BB	AA	Destination address of first block Addr[31:16]
27	28	DD	CC	Destination address of first block Addr[15:0] (Addr = 0xAABBCCDD)
29	30	BB	AA	First word of the first block in the source being loaded = 0xAABB
...				...
...				Data for this section.
				...
.		BB	AA	Last word of the first block of the source being loaded = 0xAABB
.		NN	MM	Block size of the 2nd block to load = 0xMMNN words
.		BB	AA	Destination address of second block Addr[31:16]
.		DD	CC	Destination address of second block Addr[15:0]
.		BB	AA	First word of the second block in the source being loaded
.				…
n	n+1	BB	AA	Last word of the last block of the source being loaded (More sections if required)
n+2	n+3	00	00	Block size of 0000h - indicates end of the source program

The device first signals the host that it is ready to begin data transfer by pulling the 28x Control pin low. The host load then initiates the data transfer by pulling the DSP control pin low. The complete protocol is shown in Figure 4-13.

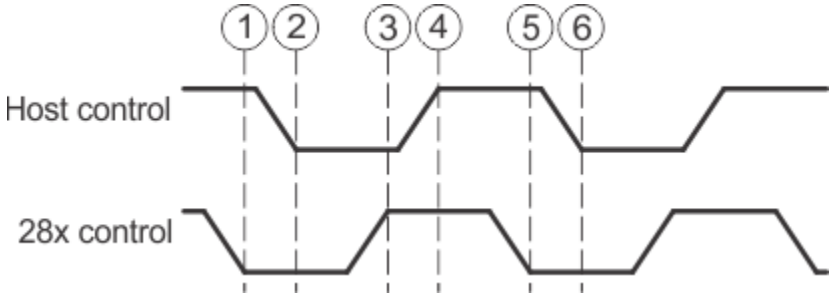


Figure 4-13. Parallel GPIO Bootloader Handshake Protocol

1. The device indicates it is ready to start receiving data by pulling the 28x control pin low.
2. The bootloader waits until the host puts data on GPIO [D0:D7]. The host signals to the device that data is ready by pulling the host control pin low.
3. The device reads the data and signals the host that the read is complete by pulling 28x control high.
4. The bootloader waits until the host acknowledges the device by pulling host control high.
5. The device again indicates it is ready for more data by pulling the 28x control pin low.

This process is repeated for each data value to be sent.

Figure 4-14 shows an overview of the Parallel GPIO bootloader flow.

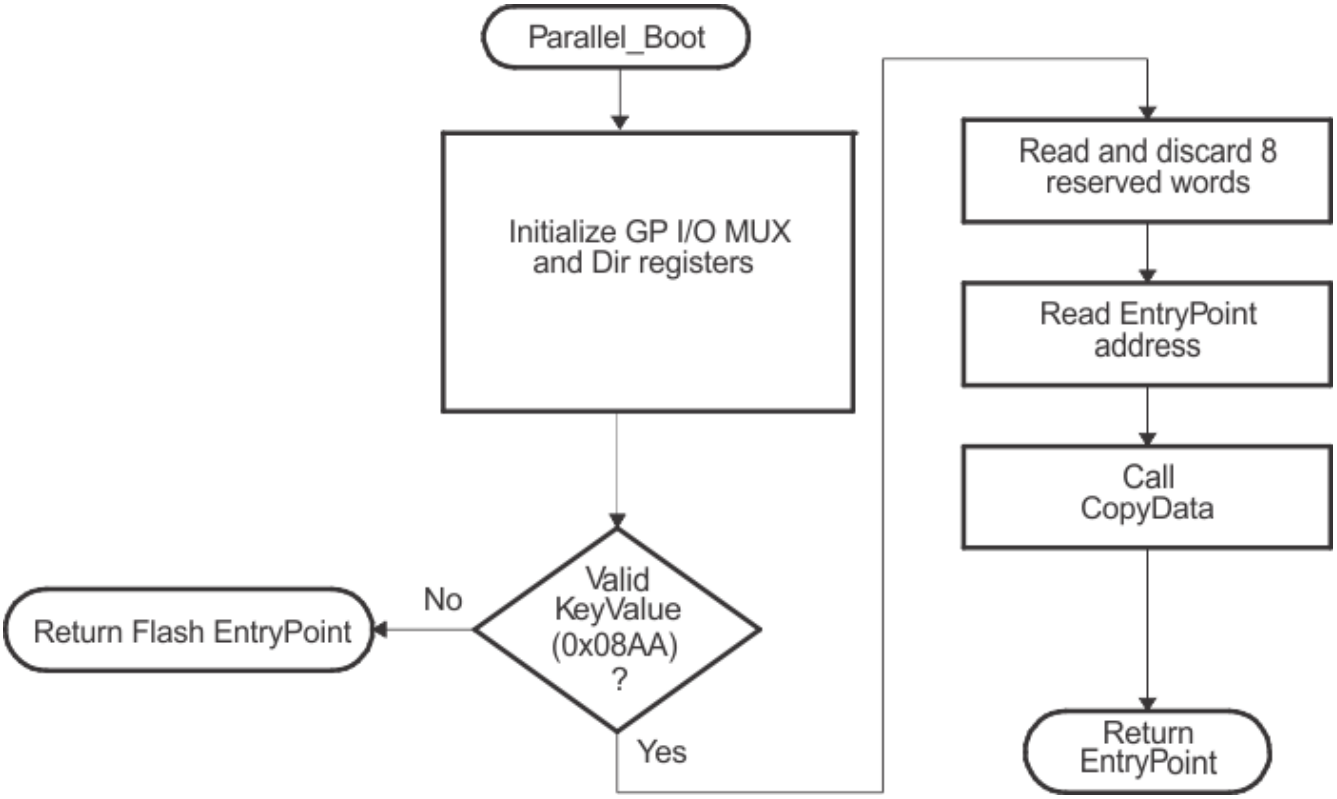


Figure 4-14. Overview of Parallel GPIO Boot Function

Figure 4-15 shows the transfer flow from the host side. The operating speed of the CPU and host are not critical in this mode as the host will wait for the device and the device will in turn wait for the host. In this manner the protocol will work with both a host running faster and a host running slower than the device.

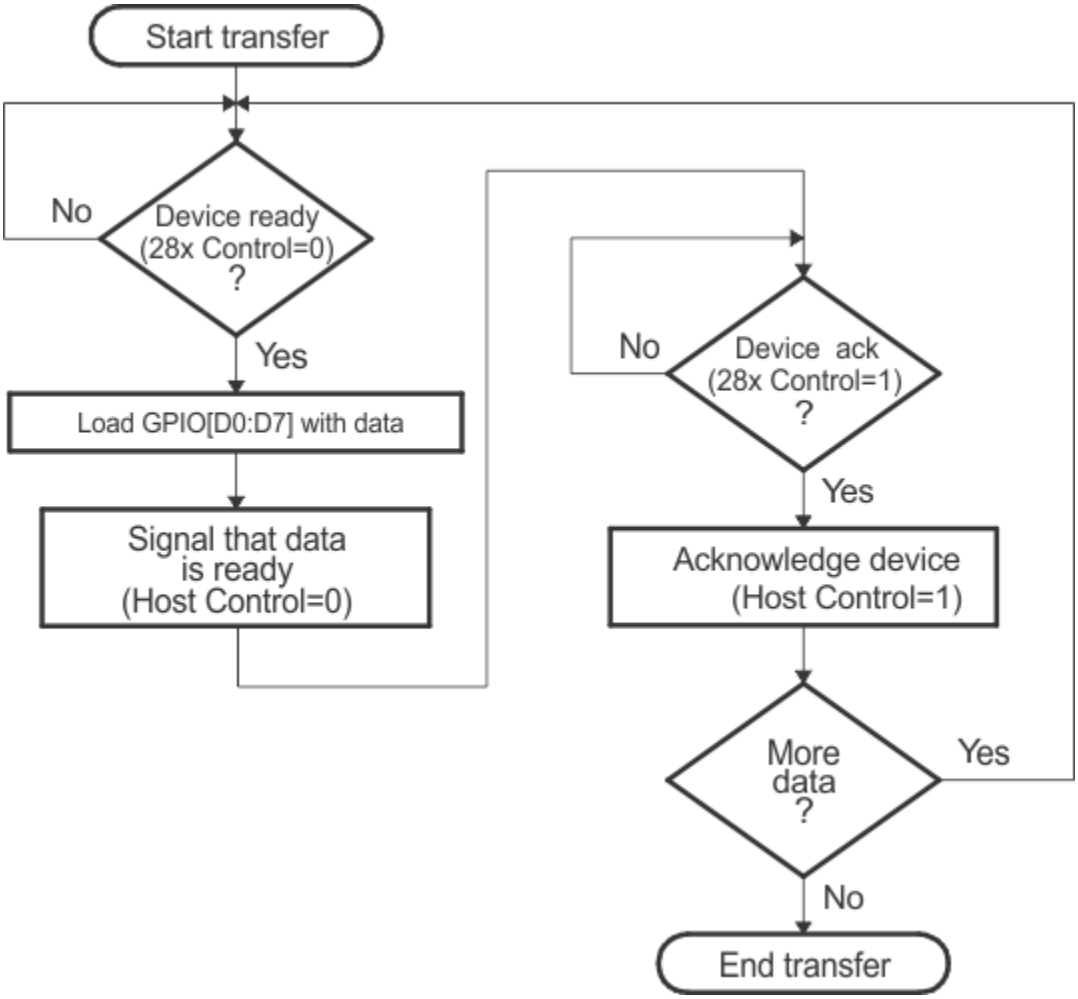


Figure 4-15. Parallel GPIO Mode - Host Transfer Flow

Figure 4-16 shows the flow used to read a single word of data from the parallel port.

- **8-bit data stream**

  The 8-bit routine, shown in Figure 4-16, discards the upper eight bits of the first read from the port and treats the lower eight bits masked with D7 in bit position 7 and D6 in bit position six as the least significant byte (LSB) of the word to be fetched. The routine will then perform a second read to fetch the most significant byte (MSB). The routine will then perform a second read to fetch the most significant byte (MSB). It then combines the MSB and LSB into a single 16-bit value to be passed back to the calling routine.

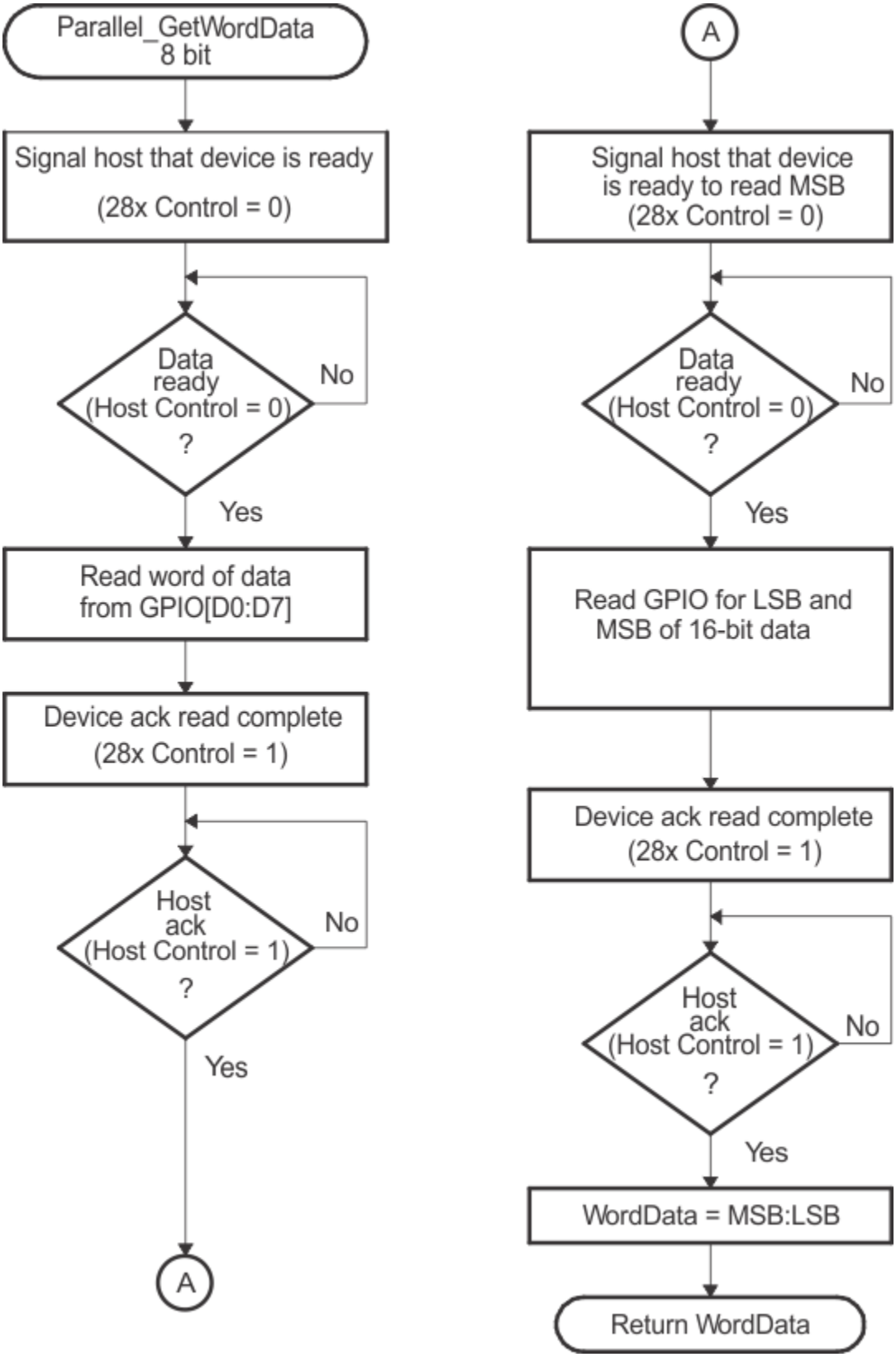


Figure 4-16. 8-Bit Parallel GetWord Function

4.8.9.2.5 CAN Boot Mode

The CAN bootloader asynchronously transfers code from CAN-A to internal memory as shown in Figure 4-17. The host can be any CAN node. The communication is first done with 11-bit standard identifiers (with a MSGID of 0x1) using two bytes per data frame. The host can download a kernel to reconfigure the CAN if higher data throughput is desired.

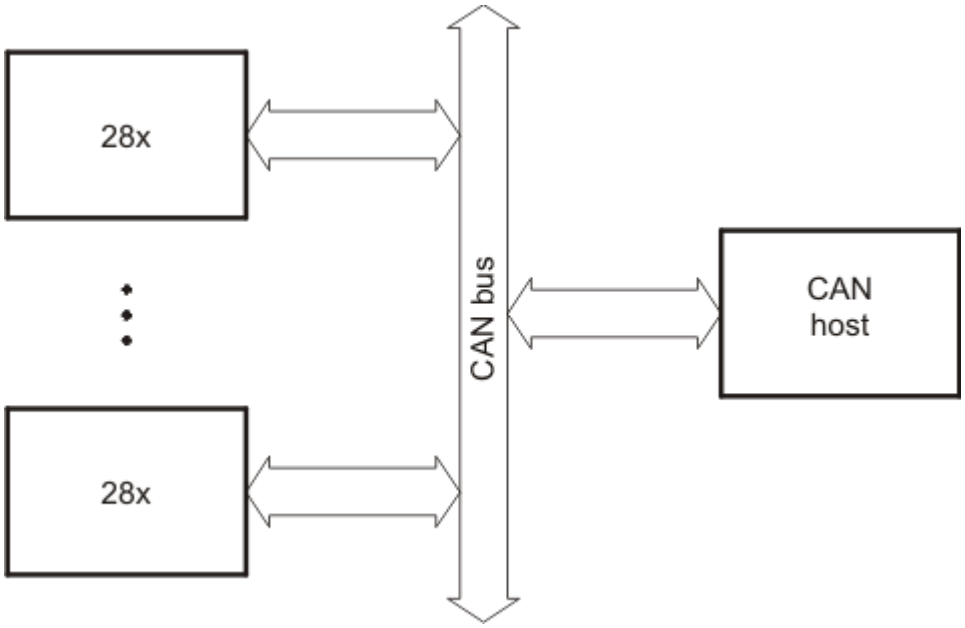


Figure 4-17. Overview of CAN-A Bootloader Operation

The bit timing registers are programmed in such a way that a 100 kbps bit rate is achieved with a 20 MHz external oscillator, as shown in Table 4-38.

Table 4-38. Bit-Rate Value for Internal Oscillators

OSCCLK	SYSCLK	Bit Rate
20 MHz	10 MHz	100 kbps

The SYSCLKOUT values shown are the reset values with the default PLL setting. The BRP and bit-time values are hard-coded to 10 and 20, respectively.

Note

The CAN boot loader uses XTAL as the bit clock source and INTOSC2 as the system clock source.

Mailbox 1 is programmed with a standard MSGID of 0x1 for boot-loader communication. The CAN host should transmit only two bytes at a time, LSB first and MSB next. For example, to transmit the word 0x08AA to the device, transmit AA first, followed by 08. The program flow of the CAN bootloader is identical to the SCI bootloader. The data sequence for the CAN bootloader is shown in Table 4-39.

Table 4-39. CAN 8-Bit Data Stream

Bytes		Byte 1 of 2	Byte 2 of 2	Description
1	2	AA	08	0x08AA (KeyValue for memory width = 16 bits)
3	4	00	00	reserved
5	6	00	00	reserved
7	8	00	00	reserved
9	10	00	00	reserved
11	12	00	00	reserved
13	14	00	00	reserved
15	16	00	00	reserved
17	18	00	00	reserved
19	20	BB	AA	Entry point PC[22:16]
21	22	DD	CC	Entry point PC[15:0] (PC = 0xAABBCCDD)
23	24	NN	MM	Block size of the first block of data to load = 0xMMNN words
25	26	BB	AA	Destination address of first block Addr[31:16]
27	28	DD	CC	Destination address of the first block Addr[15:0] (Addr = 0xAABBCCDD)
29	30	BB	AA	First word of the first block in the source being loaded = 0xAABB
...				....
...				Data for this section.
				...
.		BB	AA	Last word of the first block of the source being loaded = 0xAABB
.		NN	MM	Block size of the second block to load = 0xMMNN words
.		BB	AA	Destination address of the second block Addr[31:16]
.		DD	CC	Destination address of the second block Addr[15:0]
.		BB	AA	First word of the second block in the source being loaded
.				…
n	n+1	BB	AA	Last word of the last block of the source being loaded (More sections if required)
n+2	n+3	00	00	Block size of 0000h - indicates end of the source program

4.8.9.2.6 CAN-FD Boot Mode

The CAN-FD bootloader asynchronously transfers code from CAN-FD to internal memory and follows same bootloader execution flow as Section 4.8.9.2.5. The host can be any CAN-FD node. The communication is first done with 11-bit standard identifiers (with a MSGID of 0x1) using two bytes per data frame. It uses a fixed 64-byte payload size and default bit rate of 1-Mbps for nominal phase and 2-Mbps for data phase. Bit data timing can be optionally reconfigured after receiving first data segment. It supports same debug boot mode and GOIP option-0 as CAN bootloader.

Mailbox 1 is programmed with a standard MSGID of 0x1 for boot-loader communication. The CAN-FD host should transmit only two bytes at a time, LSB first and MSB next. For example, to transmit the word 0x08AA to the device, transmit AA first, followed by 08. The program flow of the CAN-FD bootloader is identical to the CAN bootloader. The data sequence for the CAN-FD bootloader is shown in Table 4-40.

Table 4-40. CAN-FD 8-Bit Data Stream

Bytes		Byte 1 of 2	Byte 2 of 2	Description
1	2	AA	08	0x08AA (KeyValue for memory width = 16 bits)
3	4	XX (for example, BB)	XX (for example, AA)	0: Ignored Nonzero: Custom nominal bit register timing [31:16]
5	6	XX (for example, DD)	XX (for example, CC)	0: Ignored Nonzero: Custom nominal bit register timing [15:0] (**NBTR** = 0xAABBCCDD)
7	8	XX (for example, BB)	XX (for example, AA)	0: Ignored Nonzero: Custom nominal bit register timing [31:16]
9	10	XX (for example, DD)	XX (for example, CC)	0: Ignored Nonzero: Custom nominal bit register timing [15:0] (**DBTR** = 0xAABBCCDD)
11	12	00	00	reserved
13	14	00	00	reserved
15	16	00	00	reserved
17	18	00	00	reserved
19	20	BB	AA	Entry point PC[22:16]
21	22	DD	CC	Entry point PC[15:0] (**PC** = 0xAABBCCDD)
23	24	NN	MM	Block size of the first block of data to load = 0xMMNN words
25	26	BB	AA	Destination address of first block Addr[31:16]
27	28	DD	CC	Destination address of the first block Addr[15:0] (Addr = 0xAABBCCDD)
29	30	BB	AA	First word of the first block in the source being loaded = 0xAABB
... ...				.... Data for this section. ...
.		BB	AA	Last word of the first block of the source being loaded = 0xAABB
.		NN	MM	Block size of the second block to load = 0xMMNN words
.		BB	AA	Destination address of the second block Addr[31:16]
.		DD	CC	Destination address of the second block Addr[15:0]
.		BB	AA	First word of the second block in the source being loaded
.				…
n	n+1	BB	AA	Last word of the last block of the source being loaded (More sections if required)
n+2	n+3	00	00	Block size of 0000h - indicates end of the source program

4.8.10 GPIO Assignments

This section details the GPIOs and boot option values used for boot mode set in the BOOT_DEF memory location located at Z1-OTP-BOOTDEF-LOW/ Z2-OTP-BOOTDEF-LOW and Z1-OTP-BOOTDEF-HIGH/ Z2-OTP-BOOTDEF-HIGH. Refer to Section 4.5.2 on how to configure BOOT_DEF. When selecting a boot mode option, make sure to verify that the necessary pins are available in the pin mux options for the specific device package being used.

Table 4-41. SCI Boot Options

Option	BOOTDEF Value	SCITXDA GPIO	SCIRXDA GPIO
0 (default)	0x01	GPIO29	GPIO28
1	0x21	GPIO16	GPIO17
2	0x41	GPIO8	GPIO9
3	0x61	GPIO2	GPIO3
4	0x81	GPIO16	GPIO3

Table 4-42. CAN Boot Options

Option	BOOTDEF Value	CANTXA GPIO	CANRXA GPIO
0 (default)	0x02	GPIO4	GPIO5
1	0x22	GPIO32	GPIO33
2	0x42	GPIO2	GPIO3
3	0x62	GPIO13	GPIO12

Table 4-43. CAN-FD Boot Options

Option	BOOTDEFx Value	CANTXA GPIO	CANRXA GPIO
0	0x08	GPIO4	GPIO5
1	0x28	GPIO1	GPIO0
2	0x48	GPIO13	GPIO12

Table 4-44. I2C Boot Options

Option	BOOTDEF Value	SDAA GPIO	SCLA GPIO
0	0x07	GPIO32	GPIO33
1	0x27	GPIO0	GPIO1
2	0x47	GPIO10	GPIO8

Table 4-45. SPI Boot Options

Option	BOOTDEF Value	SPISIMOA	SPISOMIA	SPICLKA	SPISTEA
0	0x06	GPIO2	GPIO1	GPIO3	GPIO5
1	0x26	GPIO16	GPIO1	GPIO3	GPIO0
2	0x46	GPIO8	GPIO10	GPIO9	GPIO11
3	0x66	GPIO8	GPIO17	GPIO9	GPIO11

Table 4-46. Parallel Boot Options

Option	BOOTDEF Value	D0-D7 GPIO	C28x (DSP) Control GPIO	Host Control GPIO
0 (default)	0x00	D0 - GPIO0	GPIO16	GPIO29
		D1 - GPIO1		
		D2 - GPIO2		
		D3 - GPIO3		
		D4 - GPIO4		
		D5 - GPIO5		
		D6 - GPIO6		
		D7 - GPIO7		
1	0x20	D0 - GPIO0	GPIO16	GPIO11
		D1 - GPIO1		
		D2 - GPIO2		
		D3 - GPIO3		
		D4 - GPIO4		
		D5 - GPIO5		
		D6 - GPIO6		
		D7 - GPIO7		

4.8.11 Secure ROM Function APIs

There are a few functions that are available within Secure ROM to be called by the application to perform EXEONLY Flash/RAM tasks in a secure manner.

Note

The application should disable interrupts before calling one of the EXEONLY function APIs.

If a vector fetch request is given by the CPU (C28) while its program counter (PC) is within the EXEONLY function API code of the Secure ROM, a reset will fire (RSN if from C28). The consequence of this is if an NMI or ITRAP or Bus Fault occurs while the PC is executing one of the EXEONLY API functions, the NMI/ITRAP/Fault cannot be serviced because a reset will be fired to the subsystem.

The **secure copy code zone 1 and zone 2 functions** allow EXEONLY Flash to be copied to EXEONLY RAM in a secure manner. The source must be from EXEONLY Flash and the destination to EXEONLY RAM. There is no support to copy EXEONLY ROM or EXEONLY RAM to RAM. Both Flash and RAM must be set to EXEONLY and configured for the same zone. Additionally, the copy size must not cross over the Flash sector boundary. Any violations of these requirements will result in a failure status returned. Upon successful copy of the data, the number of 16-bit words copied is returned.

Table 4-47. Secure Copy Code Function

CPU	Function Prototype	Function Parameters	Function Return Value
CPU (C28x)	uint16_t **SecureCopyCodeZ1**(uint32_t size, uint16_t *dst, uint16_t *src) uint16_t **SecureCopyCodeZ2**(uint32_t size, uint16_t *dst, uint16_t *src)	*size* : The number of 16-bit words to copy *dst* : The destination memory address in EXEONLY RAM *src* : The source memory address in EXEONLY Flash	0xXXXX : Returns the number of 16-bit words copied 0x0000 : Indicates one of the following: Copy length is zero; Copy size crosses over Flash sector boundary; Flash and RAM don't belong to the same zone; Flash and/or RAM aren't set to EXEONLY; Error occurred during data copy

The **secure CRC calculation zone 1 and zone 2 functions** allow a safety CRC check of EXEONLY memory in a secure manner. The CRC length provided must be a value from 1 to 8 where 1 represents a CRC size of 32 16-bit words and 8 represents a CRC size of 4096 16-bit words. The source address specifies the starting address for the CRC and the destination address is the location that the resulting CRC value will be stored. The source and destination memories must be configured for the same zone. Additionally, the CRC length must not cross over the Flash sector or RAM block boundary. Any violations of these requirements will result in a failure status returned. Upon successful CRC, the number of 16-bit words CRC'd is returned.

Table 4-48. Secure CRC Calculation Function

CPU	Function Prototype	Function Parameters	Function Return Value
CPU (C28x)	uint16_t **SecureCRCCalcZ1**(uint16_t len_id, uint16_t *dst, uint16_t *src) uint16_t **SecureCRCCalcZ2**(uint16_t size, uint16_t *dst, uint16_t *src)	*len_id* : A number from 1 to 8 which corresponds to length options of 32, 64, 128, 256, 512, 1024, 2048, or 4096 16-bit words *dst* : The destination memory address for resulting CRC *src* : The source memory address to begin CRC calculation	0xXXXX : Returns the number of 16-bit words CRC'd 0x0000 : Indicates one of the following: Invalid length option; Source address isn't modulo of length value; Destination address isn't within secure RAM; CRC size crosses over Flash sector or RAM block boundary; The source and destination memory don't belong to the same zone; On CM, CRCLOCK is enabled

The **CMAC calculate and compare function** allows to calculate CMAC signature of a Flash memory block and compare against a golden signature. This is used in the secure boot mode to authenticate the boot image.

Table 4-49. Secure CRC Calculation Function

CPU	Function Prototype	Function Parameters	Function Return Value
CPU (C28x)	uint32_t **CPU1BROM_calculateCMAC**(uint32_t startAddress, uint32_t endAddress, uint32_t signatureAddress)	**startAddress**: Starting address of memory for which CMAC has to be calculated **endAddress**: Ending address of memory for which CMAC has to be calculated **signatureAddress**: Address of location where golden CMAC signature is stored	**0xFFFFFFFFU**: Calculated CMAC signature did not match golden signature (fail) **0xA5A5A5A5U**: Memory range provided isn't aligned to 128-bit boundary or length is zero **0xE1E1E1E1U**: AES Engine timed out **0x00000000U**: No Error

4.8.12 Clock Initializations

During boot-up, the boot ROM initializes the device clocking, depending upon the reset source, to assist in faster boot time response. Clock configurations are performed by the boot ROM code only for POR and XRS reset types. For all other resets, the boot ROM starts executing with the clocks that were already set up before reset.

Note

CPU performs clock configurations during boot up. If the PLL is used during the boot process, it will be bypassed by the boot ROM code before branching to the user application.

Table 4-50. CPU Boot Clock Sources

Source	Frequency	Description
INTOSC2	10 MHz	Default clock source
INTOSC1	10 MHz	Set as clock source if missing clock is detected at power up or right after device reset.
SYSPLL	115 MHz, 57.5 MHz	Enabled optionally as part of main boot flow or as part of MPOST POR memory test boot flow. PLL is bypassed and disabled after memory test has completed. See more details regarding enabling MPOST POR memory test in Section 4.8.1.1.

Table 4-51. CPU Clock State After Boot

Reset Source	Clock State
POR/XRS	1. Using INTOSC2 2. System clock divider set to /1
All other Resets	Maintain clocks setup before device reset.

4.8.13 Boot Status Information

Boot ROM keeps a record of the various actions and events that occur during boot ROM execution. The reason for this is because NMI and other exceptions are enabled by default in the device and must be handled accordingly. Boot ROM stores the boot status information in a RAM location so that the user application can read this boot status and take the necessary actions per application's needs to handle these events.

4.8.13.1 Booting Status

Boot ROM health and booting status is written to a 32-bit address in M0RAM. This status is cleared on a POR or XRS reset. The previous status is retained on any other reset. For example, you should clear the status before performing a debugger device reset in order to view the latest boot ROM actions reflected in the status.

Table 4-52. Boot Status Address

Description	Address
Boot ROM Status	0x0000 0002

Table 4-53. Boot Status Bit Fields

Value	Description
0x8000 0000	HWBIST reset is handled successfully
0x2000 0000	EFuse Single Bit Error
0x1000 0000	LFU Flash Boot Error
0x0800 0000	Flash Verification Error
0x0400 0000	DCSM initialization LP Error
0x0200 0000	DCSM Initialization Invalid LP
0x0100 0000	SYSPLL enabled successfully
0x0080 0000	HWBIST NMI occurred
0x0040 0000	Missing clock NMI occurred
0x0020 0000	RAM Uncorrectable Error NMI occurred
0x0010 0000	Flash Uncorrectable Error NMI occurred
0x0008 0000	RL NMI occured
0x0004 0000	ERAD NMI occurred
0x0002 0000	Boot ROM detected a PIE mismatch
0x0001 0000	Boot ROM detected an ITRAP
0x0000 8000	Boot ROM has completed running
0x0000 2000	Boot ROM handled POR
0x0000 1000	Boot ROM handled XRS
0x0000 0800	Boot ROM handled all the resets
0x0000 0400	POR memory test has completed
0x0000 0200	DCSM initialization has completed
0x0000 0100	RAM Initialization Complete
0x0000 000D	Secure LFU boot has started
0x0000 000C	LFU boot has started
0x0000 000B	Wait boot has started
0x0000 000A	CAN-FD boot has started
0x0000 0009	CAN boot has started
0x0000 0008	I2C boot has started
0x0000 0007	SPI boot has started
0x0000 0006	SCI boot has started
0x0000 0005	RAM boot has started
0x0000 0004	Parallel boot has started
0x0000 0003	Secure Flash boot has started
0x0000 0002	Flash boot has started
0x0000 0001	Boot ROM has started running

4.8.13.2 Boot Mode and MPOST (Memory Power On Self-Test) Status

Once the boot mode is decoded during the boot flow, the boot mode value is written to RAM. Additionally when running the MPOST POR memory test, the test result is written to RAM.

For more information, see the *C2000™ Memory Power-On Self-Test (M-POST) Application Report*.

Table 4-54. Boot Mode and MPOST Status Addresses

Description	Address
Boot Mode	0x0000 0004
MPOST POR Memory Test Result	0x0000 0006

4.8.14 ROM Version

The ROM revision and release date information is stored at the ROM locations specified in Table 4-55. Reading a revision number value of “0x100” represents version “1.0”, “0x101” represents version “1.1”, and so on. Reading a revision date value of “0x0119” represents “01/19” or “January 2019”.

Table 4-55. Boot ROM Version Information

Contents	Address
Revision Number	0x003F 8044
Revision Date	0x003F 8045
Build Number	0x003F 8046

4.9 Application Notes for Using the Bootloaders

4.9.1 Bootloader Data Stream Structure

This section details the data transfer protocols or stream structures that allow boot data transfer between boot ROM and host device. This data transfer protocol is compatible to the respective bootloaders on C2000 devices.

Table 4-56 and Example 4-2 show the structure of the data stream incoming to the bootloader. The basic structure is the same for all the bootloaders and is based on the C54x source data stream generated by the C54x hex utility. The C28x hex utility (hex2000.exe) has been updated to support this structure. The hex2000.exe utility is included with the C2000 code generation tools. All values in the data stream structure are in hex. Refer to for more details on using the C28x hex utility to convert a project to this format.

The first 16-bit word in the data stream is known as the key value. The key value is used to tell the bootloader the width of the incoming stream: 8 or 16 bits. Note that not all bootloaders will accept both 8 and 16-bit streams. Please refer to the detailed information on each loader for the valid data stream width. For an 8-bit data stream, the key value is 0x08AA and for a 16-bit stream it is 0x10AA. If a bootloader receives an invalid key value, then the load is aborted.

The next eight words are used to initialize register values or otherwise enhance the bootloader by passing values to it. If a bootloader does not use these values then they are reserved for future use and the bootloader simply reads the value and then discards it. Currently only the SPI and I2C and parallel bootloaders use these words to initialize registers.

The tenth and eleventh words comprise the 22-bit entry point address. This address is used to initialize the PC after the boot load is complete. This address is most likely the entry point of the program downloaded by the bootloader.

The twelfth word in the data stream is the size of the first data block to be transferred. The size of the block is defined as 8-bit data stream format. For example, to transfer a block of 20 8-bit data values from an 8-bit data stream, the block size would be 0x000A to indicate 10 16-bit words.

The next two words tell the loader the destination address of the block of data. Following the size and address will be the 16-bit words that makeup that block of data.

This pattern of block size/destination address repeats for each block of data to be transferred. Once all the blocks have been transferred, a block size of 0x0000 signals to the loader that the transfer is complete. At this point, the loader returns the entry point address to the calling routine, which will clean up and exit. Execution then continues at the entry point address as determined by the input data stream contents.

Table 4-56. LSB/MSB Loading Sequence in 8-Bit Data Stream

Byte		Contents	
		LSB (First Byte of 2)	MSB (Second Byte of 2)
1	2	LSB: AA (KeyValue for memory width = 8 bits)	MSB: 08h (KeyValue for memory width = 8 bits)
3	4	LSB: Register initialization value or reserved	MSB: Register initialization value or reserved
5	6	LSB: Register initialization value or reserved	MSB: Register initialization value or reserved
7	8	LSB: Register initialization value or reserved	MSB: Register initialization value or reserved
...	...	...	...
...	...	...	...
17	18	LSB: Register initialization value or reserved	MSB: Register initialization value or reserved
19	20	LSB: Upper half of Entry point PC[23:16]	MSB: Upper half of entry point PC[31:24] (Always 0x00)
21	22	LSB: Lower half of Entry point PC[7:0]	MSB: Lower half of Entry point PC[15:8]
23	24	LSB: Block size in words of the first block to load. If the block size is 0, this indicates the end of the source program. Otherwise another block follows. For example, a block size of 0x000A would indicate 10 words or 20 bytes in the block.	MSB: block size
25	26	LSB: MSW destination address, first block Addr[23:16]	MSB: MSW destination address, first block Addr[31:24]
27	28	LSB: LSW destination address, first block Addr[7:0]	MSB: LSW destination address, first block Addr[15:8]
29	30	LSB: First word of the first block being loaded	MSB: First word of the first block being loaded
...	...	...	...
...	...	...	...
.	.	LSB: Last word of the first block to load	MSB: Last word of the first block to load
.	.	LSB: Block size of the second block	MSB: Block size of the second block
.	.	LSB: MSW destination address, second block Addr[23:16]	MSB: MSW destination address, second block Addr[31:24]
.	.	LSB: LSW destination address, second block Addr[7:0]	MSB: LSW destination address, second block Addr[15:8]
.	.	LSB: First word of the second block being loaded	MSB: First word of the second block being loaded
...	...	...	...
...	...	...	...
.	.	LSB: Last word of the second block	MSB: Last word of the second block
.	.	LSB: Block size of the last block	MSB: Block size of the last block
.	.	LSB: MSW of destination address of last block Addr[23:16]	MSB: MSW destination address, last block Addr[31:24]
.	.	LSB: LSW destination address, last block Addr[7:0]	MSB: LSW destination address, last block Addr[15:8]
.	.	LSB: First word of the last block being loaded	MSB: First word of the last block being loaded
...	...	...	...
...	...	...	...
.	.	LSB: Last word of the last block	MSB: Last word of the last block
n	n+1	LSB: 00h	MSB: 00h - indicates the end of the source

Example 4-2. Data Stream Structure 8-bit

```
AA 08       ; 0x08AA 8bit key value
00 00 00 00 ; 8 reserved words
00 00 00 00
00 00 00 00
00 00 00 00
3F 00 00 80 ; 0x003F8000 EntryAddr, starting point after boot load completes
05 00       ; 0x0005 First block consists of 5 16-bit words
3F 00 10 90 ; 0x003F9010 First block will be loaded starting at 0x3F9010
01 00       ; Data loaded = 0x0001 0x0002 0x0003 0x0004 0x0005
02 00
03 00
04 00
05 00
02 00       ; 0x0002 - 2nd block consists of 2 16bit words
3F 00 00 80 ; 0x003F8000 2nd block will be loaded starting at 0x3F8000
00 77       ; Data loaded = 0x7700 0x7625
25 76
00 00       ; 0x0000 Size of 0 indicates end of data stream
After load has completed the following memory values will have been initialized as follows:
Location Value
0x3F9010 0x0001
0x3F9011 0x0002
0x3F9012 0x0003
0x3F9013 0x0004
0x3F9014 0x0005
0x3F8000 0x7700
0x3F8001 0x7625
PC Begins execution at 0x3F8000
```

4.9.2 The C2000 Hex Utility

To use the features of the bootloader, you must generate a data stream and boot table as described in Section 4.9.1. The hex conversion utility tool, included with the 28x code generation tools, can generate the required data stream including the required boot table. This section describes the hex2000 utility. An example of a file conversion performed by hex2000 is described in Example 4-3.

The hex utility supports creation of the boot table required for the SCI, SPI, I2C, CAN, and parallel I/O loaders. That is, the hex utility adds the required information to the file such as the key value, reserved bits, entry point, address, block start address, block length and terminating value. The contents of the boot table vary slightly depending on the boot mode and the options selected when running the hex conversion utility. The actual file format required by the host (ASCII, binary, hex, and so on) will differ from one specific application to another and some additional conversion may be required.

To build the boot table, follow these steps:

1. **Assemble or compile the code.**

   This creates the object files that will then be used by the linker to create a single output file.
2. **Link the file.**

   The linker combines all of the object files into a single output file in common object file format (ELF). The specified linker command file is used by the linker to allocate the code sections to different memory blocks. Each block of the boot table data corresponds to an initialized section in the ELF file. Uninitialized sections are not converted by the hex conversion utility. The following options may be useful:
   - The linker -m option can be used to generate a map file. This map file will show all of the sections that were created, their location in memory, and their length. It can be useful to check this file to make sure that the initialized sections are where you expect them.
   - The linker -w option configures the linker to show, if the linker assigned a section to a memory region automatically. For example, if you have a section in your code called .TI.ramfunc.

3. **Run the hex conversion utility.**

   Choose the appropriate options for the desired boot mode and run the hex conversion utility to convert the ELF file produced by the linker to a boot table.

See the *TMS320C28x Assembly Language Tools User's Guide* and the *TMS320C28x Optimizing C/C++ Compiler User's Guide* for more information on the compiling and linking process.

Table 4-57 summarizes the hex conversion utility options available for the bootloader. See the *TMS320C28x Assembly Language Tools User's Guide* for a detailed description of the hex2000 operations used to generate a boot table. Updates will be made to support the I2C boot. See the Codegen release notes for the latest information.

Table 4-57. Boot Loader Options

Option	Description
-boot	Convert all sections into bootable form (use instead of a SECTIONS directive)
-sci8	Specify the source of the bootloader table as the SCI-A port, 8-bit mode
-spi8	Specify the source of the bootloader table as the SPI-A port, 8-bit mode
-gpio8	Specify the source of the bootloader table as the GPIO port, 8-bit mode
-bootorg value	Specify the source address of the bootloader table
-lospcp value	Specify the initial value for the LOSPCP register. This value is used only for the spi8 boot table format and ignored for all other formats. If the value is greater than 0x7F, the value is truncated to 0x7F.
-spibrr value	Specify the initial value for the SPIBRR register. This value is used only for the spi8 boot table format and ignored for all other formats. If the value is greater than 0x7F, the value is truncated to 0x7F.
-e value	Specify the entry point at which to begin execution after boot loading. The value can be an address or a global symbol. This value is optional. The entry point can be defined at compile time using the linker -e option to assign the entry point to a global symbol. The entry point for a C program is normally _c_int00 unless defined otherwise by the -e linker option.
-i2c8	Specify the source of the bootloader table as the I2C-A port, 8-bit
-i2cpsc value	Specify the value for the I2CPSC register. This value will be loaded and take effect after all I2C options are loaded, prior to reading data from the EEPROM. This value will be truncated to the least significant eight bits and should be set to maintain an I2C module clock of 7-12 MHz.
-i2cclkh value	Specify the value for the I2CCLKH register. This value will be loaded and take effect after all I2C options are loaded, prior to reading data from the EEPROM.
-i2cclkl value	Specify the value for the I2CCLKL register. This value will be loaded and take effect after all I2C options are loaded, prior to reading data from the EEPROM.

Example 4-3. HEX2000.exe Command Syntax

```
C: HEX2000 GPIO34TOG.OUT -boot -gpio8 -a
Where:
- boot  Convert all sections into bootable form.
- gpio8 Use the GPIO in 8-bit mode data format. The eCAN
        uses the same data format as the GPIO in 8bit mode.
- a     Select ASCII-Hex as the output format.
```

Chapter 5

Dual Code Security Module (DCSM)

This chapter explains the dual code security module.

5.1 Introduction

The dual code security module (DCSM) is a security feature incorporated in this device. It prevents access and visibility to on-chip secure memories (and other secure resources) by unauthorized persons. It also prevents duplication and reverse-engineering of proprietary code. The term “secure” means that access to on-chip secure memories and resources is blocked. The term “unsecure” means that access is allowed; that is, the contents of the memory could be read by any means (for example, through a debugging tool such as Code Composer Studio™ IDE.

The CSM has dual-zone security, Zone1 (Z1) and Zone2 (Z2).

5.1.1 DCSM Related Collateral

Getting Started Materials

- *C2000 DCSM Security Tool Application Report*
- *C2000 Unique Device Number Application Report*
- *Enhancing Device Security by Using JTAGLOCK Feature Application Report*

5.2 Functional Description

The security module restricts the CPU access to on-chip secure memory and resources without interrupting or stalling CPU execution. When a read occurs to a secure memory location, the read returns a zero value and CPU execution continues with the next instruction. This, in effect, blocks read and write access to secure memories through the JTAG port or external peripherals.

The code security mechanism offers protection for two zones, Zone1 (Z1) and Zone2 (Z2). The security mechanism for both the zones is identical. Each zone has its own dedicated secure resource and allocated secure resource. The following are different secure resources available on this device:

- **OTP:** Each zone has its own dedicated secure OTP (USER OTP). This contains the security configurations for the individual zone. If a zone is secure, its USER OTP content (including CSM passwords) can be read (execution not allowed) only if the zone is unlocked using the password match flow (PMF).
- **RAM:** All LSx RAMs can be secure RAM on this device. These RAMs can be allocated to either zone by configuring the respective GRABRAM locations in the USER OTP.
- **Flash Sectors:** Flash sectors of each CPU subsystems can be made secure on this device. Each Flash sector can be allocated to either zone by configuring the respective GRABSECT locations in the bank's USER OTP.
- **Secure ROM:** This device also has secure ROM on each CPU subsystem which is EXEONLY-protected. These ROM contains specific function for the user, provided by TI.

Table 5-1 shows the status of a RAM block/Flash sector based on the configuration in the GRABRAM/ GRABSECT register.

The security of each zone is ensured by its own 128-bit (four 32-bit words) password (CSM password). The password for each zone is stored in USER OTP. A zone can be unsecured by executing the password match flow (PMF), described in Section 5.7.4.

There are three types of accesses: data/program reads, JTAG access, and instruction fetches (calls, jumps, code executions, ISRs). Instruction fetches are never blocked. JTAG accesses are always blocked when a memory is secure. Data reads to a secure memory are always blocked unless the program is executing from a memory which belongs to the same zone. Data reads to unsecure memory are always allowed. Table 5-2 shows the levels of security.

Table 5-1. RAM/Flash Status

Zone 1 GRABRAMx/GRABSECTx Bits	Zone 2 GRABRAMx/GRABSECT Bits	Ownership and Accessibility
01	10	RAM block/Flash Sector belongs to Zone1
01	11(2)	RAM block/Flash Sector belongs to Zone1
10	01	RAM block/Flash Sector belongs to Zone2
11(1)	01	RAM block/Flash Sector belongs to Zone2
10	10	RAM block/Flash Sector is unsecure
11(1)	11(2)	RAM block/Flash Sector is unsecure
11	11	RAM block/Flash Sector inaccessible if either of the zone is secure (CSM passwords are programmed). User should never leave these values default (11) if CSM passwords are programmed for even one zone.

(1) Zone1 must be unsecure. Assumption in this case is that user is not using Zone1 so none of the fields, including passwords, in Zone1 USER OTP are programmed by user hence Zone1 will always be unsecure.
(2) Zone2 must be unsecure. Assumption in this case is that user is not using Zone2 so none of the fields, including passwords, in Zone2 USER OTP are programmed by user hence Zone2 will always be unsecure.

Note

You should never program any other values in these fields. Failing any of these conditions for a RAM block/Flash sector makes that RAM block/Flash sector inaccessible.

Table 5-2. Security Levels

PMF Executed With Correct Password?	Operating Mode of the Zone	Program Fetch Location	Security Description
No	Secure	Outside secure memory	Only instruction fetches by the CPU are allowed to secure memory. In other words, code can still be executed, but not read.
No	Secure	Inside secure memory	CPU has full access (except for EXEONLY memories where read is not allowed). JTAG port cannot read the secured memory contents.
Yes	Unsecure	Anywhere	Full access for CPU and JTAG port to secure memory of that zone.

5.2.1 CSM Passwords

Unlike earlier C2000™ devices, on this device ALL_1 value (0xFFFFFFFF, 0xFFFFFFFF, 0xFFFFFFFF, 0xFFFFFFFF) for CSM password for a zone does not unsecure the zone. Instead, if for any zone the CSM password values get loaded as ALL_1 from USER OTP, the device will be in BLOCKED state. Due to this reason TI will program a few bits in the second 32-bit password value (ZxOTP_CSMPSWD1) in every zone select block of each zone with value ‘0’. The default value for this password location is chosen in a manner that the respective ECC value remains ALL_1. Due to this, the CSMPSWD1 value programmed by TI for every zone select block is different. See Table 5-3 for ZxOTP_CSMPSWD1 value, programmed by TI on every device. Since ECC is not programmed, the user will be able to change this value by flipping the bits which are ‘1’ to ‘0’ but leaving the ones which are already programmed by TI as ‘0’. BOOTROM code will write the default password value into the KEYx register to unlock the device as part of device initialization sequence.

If the password locations of a zone have all 128 bits as zeros (ALL_0), that zone becomes permanently secure (LOCKED state), regardless of the contents of the CSMKEYx registers which means the zone cannot be unlocked using PMF, the password match flow described in Section 5.7.4. Therefore, the user should never

use ALL_0 as password. A password of ALL_0 will prevent debug of secure code or reprogramming the Flash sectors. CSMKEYx registers are user-accessible registers that are used to unsecure the zones.

Table 5-3. Default Value of ZxOTP (Programmed by TI)

Zone Select Block	Zone1 USER OTP		Zone2 USER OTP	
	Address	Value	Address	Value
JLM_ENABLE (JTAGLOCK)	0x00078006	0xffff000f	NA	NA
PSWDLOCK	0x00078010	0xfb7fffff	0x00078210	0x1f7fffff
CRCLOCK	0x00078012	0x7fffffff	0x00078212	0x3fffffff
JTAGPSWDH0	0x00078014	0x4bffffff	NA	NA
JTAGPSWDH1	0x00078016	0x3fffffff	NA	NA
Zone_Select_Block0	0x00078022 (CSMPSWD1)	0x4d7fffff	0x00078222 (CSMPSWD1)	0x1f7fffff
Zone_Select_Block0	0x0007803e (JTAGPSWDL1)	0x2bffffff	NA	NA
Zone_Select_Block1	0x00078042 (CSMPSWD1)	0x5f7fffff	0x00078242 (CSMPSWD1)	0xe57fffff
Zone_Select_Block1	0x0007805e (JTAGPSWDL1)	0x27ffffff	NA	NA
Zone_Select_Block2	0x00078062 (CSMPSWD1)	0x1dffffff	0x00078262 (CSMPSWD1)	0x4fffffff
Zone_Select_Block2	0x0007807e (JTAGPSWDL1)	0x7b7fffff	NA	NA
Zone_Select_Block3	0x00078082 (CSMPSWD1)	0xaf7fffff	0x00078282 (CSMPSWD1)	0xe37fffff
Zone_Select_Block3	0x0007809e (JTAGPSWDL1)	0xc9ffffff	NA	NA
Zone_Select_Block4	0x000780a2 (CSMPSWD1)	0x1bffffff	0x000782a2 (CSMPSWD1)	0x57ffffff
Zone_Select_Block4	0x000780be (JTAGPSWDL1)	0x7d7fffff	NA	NA
Zone_Select_Block5	0x000780c2 (CSMPSWD1)	0x17ffffff	0x000782c2 (CSMPSWD1)	0x5bffffff
Zone_Select_Block5	0x000780de (JTAGPSWDL1)	0x6f7fffff	NA	NA
Zone_Select_Block6	0x000780e2 (CSMPSWD1)	0xbd7fffff	0x000782e2 (CSMPSWD1)	0xf17fffff
Zone_Select_Block6	0x000780fe (JTAGPSWDL1)	0x33ffffff	NA	NA
Zone_Select_Block7	0x00078102 (CSMPSWD1)	0x9f7fffff	0x00078302 (CSMPSWD1)	0x3b7fffff
Zone_Select_Block7	0x0007811e (JTAGPSWDL1)	0x0fffffff	NA	NA
Zone_Select_Block8	0x00078122 (CSMPSWD1)	0x2bffffff	0x00078322 (CSMPSWD1)	0x8fffffff
Zone_Select_Block8	0x0007813e (JTAGPSWDL1)	0xbb7fffff	NA	NA
Zone_Select_Block9	0x00078142 (CSMPSWD1)	0x27ffffff	0x00078342 (CSMPSWD1)	0x6bffffff
Zone_Select_Block9	0x0007815e (JTAGPSWDL1)	0x5f7fffff	NA	NA
Zone_Select_Block10	0x00078162 (CSMPSWD1)	0x7b7fffff	0x00078362 (CSMPSWD1)	0x377fffff
Zone_Select_Block10	0x0007817e (JTAGPSWDL1)	0x1dffffff	NA	NA
Zone_Select_Block11	0x00078182 (CSMPSWD1)	0xc9ffffff	0x00078382 (CSMPSWD1)	0x9bffffff
Zone_Select_Block11	0x0007819e (JTAGPSWDL1)	0xaf7fffff	NA	NA
Zone_Select_Block12	0x000781a2 (CSMPSWD1)	0x7d7fffff	0x000783a2 (CSMPSWD1)	0x2f7fffff
Zone_Select_Block12	0x000781be (JTAGPSWDL1)	0x1bffffff	NA	NA
Zone_Select_Block13	0x000781c2 (CSMPSWD1)	0x6f7fffff	0x000783c2 (CSMPSWD1)	0xcb7fffff
Zone_Select_Block13	0x000781de (JTAGPSWDL1)	0x17ffffff	NA	NA
Zone_Select_Block14	0x000781e2 (CSMPSWD1)	0x33ffffff	0x000783e2 (CSMPSWD1)	0x97ffffff
Zone_Select_Block14	0x000781fe (JTAGPSWDL1)	0xbd7fffff		

5.2.2 Emulation Code Security Logic (ECSL)

In addition to the CSM, the emulation code security logic (ECSL) has been implemented using a 64-bit password (part of existing CSM password) for each zone to prevent unauthorized users from stepping through secure code. A halt in secure code while the emulator is connected will trip the ECSL and break the emulation connection. To allow emulation of secure code, while maintaining the CSM protection against secure memory reads, you must write the correct 64-bit password into the CSMKEY (0/1) registers, which matches the password value stored in the USER OTP of that zone. This will disable the ECSL for the specific zone.

When initially debugging a device with the password locations in OTP programmed (secured), the emulator takes some time to take control of the CPU. During this time, the CPU will start running and may execute an instruction that performs an access to a protected ECSL area and if the CPU is halted when the program counter (PC) is pointing to a secure location, the ECSL will trip and cause the emulator connection to be broken.

The solution to this problem is to use the Wait Boot Mode boot option. In this mode, the CPU will be in a loop and will not jump to the user application code. Using this BOOTMODE, you can connect to CCS and debug the code.

5.2.3 CPU Secure Logic

The CPU Secure Logic (CPUSL) on this device prevents a hacker from reading the CPU registers in a watch window while code is running in a secure zone. All accesses to CPU registers when the PC points to a secure location are blocked by this logic. The only exception to this is read access to the PC. It is highly recommended not to write into the CPU register in this case, because proper code execution may get affected. If the CSM is unlocked using the CSM password match flow, the CPUSL logic also gets disabled.

5.2.4 Password Lock

The password locations in USER OTP for each zone can be locked by programming the zone's PSWDLOCK field with any value other than "1111" (0xF) at the PSWDLOCK location in OTP. Until the passwords of a zone are locked, password locations will not be secure and can be read from the debugger as well as code running from non-secure memory. This feature can be used by the user to avoid accidental locking of the zone while programming the Flash sectors during the software development phase. On a fresh device the value for password lock fields for all zones at the PSWDLOCK location in OTP will be "1111" which means the password for all zones will be unlocked.

Note

Password unlock only makes password locations non-secure. All other secure memories remains secure as per security settings. Since password locations are non-secure, anyone can read the password and make the zone un-secure by running through PMF, user must program PSWDLOCK locations to lock the password before sending the device in field.

5.2.5 JTAGLOCK

Sometimes you want to disable the JTAG access on a device to avoid any debug access to it. This can be done by using the JTAGLOCK feature on this device. You need to follow a two step process to enable the JTAGLOCK feature (both steps can be performed at the same time).

1. Program the JTAG passwords. This device has a 128-bit JTAG password that needs to be programmed in Z1 USER OTP. JTAG passwords are split into two parts, JTAGPSWDH and JTAGPSWDL. JTAGPSWDH is part of Z1 USER OTP header and JTAGPSWDL is part of Z1 Zone Select Block (ZSB). What this means is program JTAGPSWDH once and change the JTAGPSWDL multiple times, if needed. Code Composer Studio has an integrated tool that you need to use to unlock the JTAGLOCK on device.
2. After programming the JTAG passwords, you need to enable the JTAGLOCK module (JLM) by programming bit [3:0] of Z1OTP_JLM_ENABLE with any value other than 0xF. It is recommended to program all four bits with a value 0x0.

5.2.6 Link Pointer and Zone Select

For each of the two security zones, a dedicated OTP block exists that holds the configuration related to zone's security. The following are user programmable configurations:

- ZxOTP_LINKPOINTER1
- ZxOTP_LINKPOINTER2
- ZxOTP_LINKPOINTER3
- Z1OTP_JLM_ENABLE
- ZxOTP_GPREG1
- ZxOTP_GPREG2
- ZxOTP_GPREG3
- ZxOTP_GPREG4
- ZxOTP_PSWDLOCK
- ZxOTP_CRCLOCK
- Z1OTP_JTAGPSWDH
- Z1OTP_CMACKEY
- ZxOTP_CSMPSWD0
- ZxOTP_CSMPSWD1
- ZxOTP_CSMPSWD2
- ZxOTP_CSMPSWD3
- ZxOTP_GRABSECT1
- ZxOTP_GRABSECT2
- ZxOTP_GRABSECT3
- ZxOTP_GRABRAM1
- ZxOTP_GRABRAM2
- ZxOTP_GRABRAM3
- ZxOTP_EXEONLYSECT1
- ZxOTP_EXEONLYSECT2
- ZxOTP_EXEONLYRAM1
- Z1OTP_JTAGPSWDL

Since OTP cannot be erased, the following configurations are placed in zone select blocks of each zone's OTP Flash of both the banks:

- ZxOTP_CSMPSWD0
- ZxOTP_CSMPSWD1
- ZxOTP_CSMPSWD2
- ZxOTP_CSMPSWD3
- ZxOTP_GRABSECT1
- ZxOTP_GRABSECT2
- ZxOTP_GRABSECT3
- ZxOTP_GRABRAM1
- ZxOTP_GRABRAM2
- ZxOTP_GRABRAM3
- ZxOTP_EXEONLYSECT1
- ZxOTP_EXEONLYSECT2
- ZxOTP_EXEONLYRAM1
- Z1OTP_JTAGPSWDL

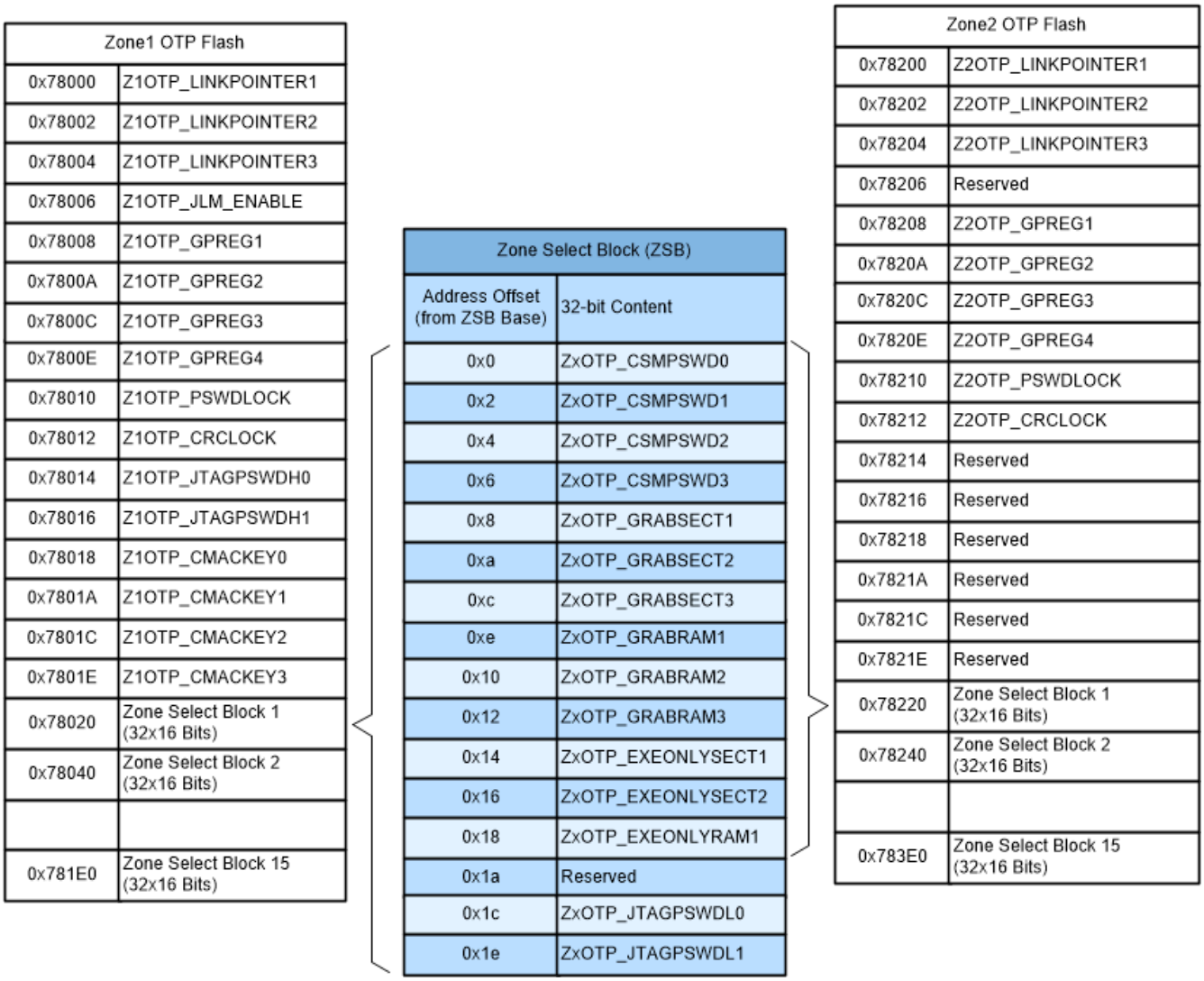
The location of the valid zone select block in OTP is decided based on the value of three 14-bit link pointers (Zx-LINKPOINTERx) programmed in the OTP of each zone. All OTP locations except link pointers and Z1OTP_JLM_ENABLE locations are protected with ECC. Since the link pointer locations are not protected with ECC, three link pointers are provided that need to be programmed with the same value. The final value of the link pointer is resolved in hardware, when a dummy read is done to all the link pointers, by comparing all the three values (bit-wise voting logic). Since in OTP, a '1' can be flipped by the user to '0' but '0' can not be flipped to '1' (no erase operation for OTP), the most significant bit position in the resolved link pointer which is '0', defines the valid base address for the zone select block. While generating the final link pointer value, if the bit pattern is not one of those listed in Figure 5-1, the final link pointer value becomes All_1 (0xFFFF_FFFF), which selects the Zone-Select-Block1 (also known as the default zone select block).

Zx-LINKPOINTER	Selected ZSB	Zone1 ZSB Address	Zone2 ZSB Address
32xxxxxxxxxxxxxxxxx11111111111111	ZSB1	0x78020	0x78220
32xxxxxxxxxxxxxxxxx11111111111110	ZSB2	0x78040	0x78240
32xxxxxxxxxxxxxxxxx11111111111100	ZSB3	0x78060	0x78260
32xxxxxxxxxxxxxxxxx11111111111000	ZSB4	0x78080	0x78280
32xxxxxxxxxxxxxxxxx11111111110000	ZSB5	0x780a0	0x782a0
32xxxxxxxxxxxxxxxxx11111111100000	ZSB6	0x780c0	0x782c0
32xxxxxxxxxxxxxxxxx11111111000000	ZSB7	0x780e0	0x782e0
32xxxxxxxxxxxxxxxxx11111110000000	ZSB8	0x78100	0x78300
32xxxxxxxxxxxxxxxxx11111100000000	ZSB9	0x78120	0x78320
32xxxxxxxxxxxxxxxxx11111000000000	ZSB10	0x78140	0x78340
32xxxxxxxxxxxxxxxxx11110000000000	ZSB11	0x78160	0x78360
32xxxxxxxxxxxxxxxxx11100000000000	ZSB12	0x78180	0x78380
32xxxxxxxxxxxxxxxxx11000000000000	ZSB13	0x781a0	0x783a0
32xxxxxxxxxxxxxxxxx10000000000000	ZSB14	0x781c0	0x783c0
32xxxxxxxxxxxxxxxxx00000000000000	ZSB15	0x781e0	0x783e0

Figure 5-1. Storage of Zone-Select Bits in OTP

Note

Address locations for other security settings that are not part of Zone Select blocks can be programmed only once; therefore, you should program them towards end of the development cycle.



Zone1 OTP Flash	
0x78000	Z1OTP_LINKPOINTER1
0x78002	Z1OTP_LINKPOINTER2
0x78004	Z1OTP_LINKPOINTER3
0x78006	Z1OTP_JLM_ENABLE
0x78008	Z1OTP_GPREG1
0x7800A	Z1OTP_GPREG2
0x7800C	Z1OTP_GPREG3
0x7800E	Z1OTP_GPREG4
0x78010	Z1OTP_PSWDLOCK
0x78012	Z1OTP_CRCLOCK
0x78014	Z1OTP_JTAGPSWDH0
0x78016	Z1OTP_JTAGPSWDH1
0x78018	Z1OTP_CMACKEY0
0x7801A	Z1OTP_CMACKEY1
0x7801C	Z1OTP_CMACKEY2
0x7801E	Z1OTP_CMACKEY3
0x78020	Zone Select Block 1 (32x16 Bits)
0x78040	Zone Select Block 2 (32x16 Bits)
0x781E0	Zone Select Block 15 (32x16 Bits)

Zone Select Block (ZSB)	
Address Offset (from ZSB Base)	32-bit Content
0x0	ZxOTP_CSMPSWD0
0x2	ZxOTP_CSMPSWD1
0x4	ZxOTP_CSMPSWD2
0x6	ZxOTP_CSMPSWD3
0x8	ZxOTP_GRABSECT1
0xa	ZxOTP_GRABSECT2
0xc	ZxOTP_GRABSECT3
0xe	ZxOTP_GRABRAM1
0x10	ZxOTP_GRABRAM2
0x12	ZxOTP_GRABRAM3
0x14	ZxOTP_EXEONLYSECT1
0x16	ZxOTP_EXEONLYSECT2
0x18	ZxOTP_EXEONLYRAM1
0x1a	Reserved
0x1c	ZxOTP_JTAGPSWDL0
0x1e	ZxOTP_JTAGPSWDL1

Zone2 OTP Flash	
0x78200	Z2OTP_LINKPOINTER1
0x78202	Z2OTP_LINKPOINTER2
0x78204	Z2OTP_LINKPOINTER3
0x78206	Reserved
0x78208	Z2OTP_GPREG1
0x7820A	Z2OTP_GPREG2
0x7820C	Z2OTP_GPREG3
0x7820E	Z2OTP_GPREG4
0x78210	Z2OTP_PSWDLOCK
0x78212	Z2OTP_CRCLOCK
0x78214	Reserved
0x78216	Reserved
0x78218	Reserved
0x7821A	Reserved
0x7821C	Reserved
0x7821E	Reserved
0x78220	Zone Select Block 1 (32x16 Bits)
0x78240	Zone Select Block 2 (32x16 Bits)
0x783E0	Zone Select Block 15 (32x16 Bits)

Figure 5-2. Location of Zone-Select Block Based on Link-Pointer

CAUTION

USER OTP is ECC protected. You must program the ECC value while programming the security setting in USER OTP. Failing to program the correct ECC value causes the device to be blocked permanently and you will have to replace the device.

5.2.7 C Code Example to Get Zone Select Block Addr for Zone1

```
unsigned long LinkPointer;
unsigned long *Zone1SelBlockPtr;
int Bitpos = 13;
int ZeroFound = 0;
// Read Z1-Linkpointer register of DCSM module.
LinkPointer = *(unsigned long *)0x5F000;
// Bits 31 to 15 as most-sigificant 0 are reserved LinkPointer options
LinkPointer = LinkPointer << 18;
while ((ZeroFound == 0) && (bitpos > -1))
{
if ((LinkPointer & 0x80000000) == 0)
{
ZeroFound = 1;
Zone1SelBlockPtr = (unsigned long *)(0x78000 + ((bitpos + 2)*32));
}
else
{
bitpos--;
LinkPointer = LinkPointer << 1;
}
}
if (ZeroFound == 0)
{
//Default in case there is no zero found.
Zone1SelBlockPtr = (unsigned long *)0x78020;
}
```

5.3 Flash and OTP Erase/Program

On this device, OTP as well as normal Flash, are secure resources. Each zone has its own dedicated OTP, whereas normal Flash sectors can be allocated to any zone based on the value programmed in the GRABSECTx location in OTP. Each zone has its own 128bit CSM passwords. Read and write accesses are not allowed to resources assigned to Z1 by code running from memory allocated to Z2 and vice versa. Before programming any secure Flash sector the user must either unlock the zone to which that particular sector belongs using PMF or execute the Flash programming code from secure memory which belongs to the same zone. The same is the case for erasing any secure Flash sector. To program the security settings in OTP Flash, the user must unlock the CSM of the respective zone. Unless the zone is unlocked, security settings in OTP Flash can not be updated. The OTP content cannot be erased.

A semaphore mechanism is provided to avoid the program/erase conflict between Z1 and Z2. A zone needs to grab this semaphore to successfully complete the erase/program operation on the secure Flash sectors allocated to that zone. A semaphore can be grabbed by a zone by writing the appropriate value in the SEM field of the FLSEM register. For further details of this field, see the register description.

5.4 Secure Copy Code

In some applications, the user may want to copy the code from secure Flash to secure RAM for better performance. The user cannot do this for EXEONLY Flash sectors because EXEONLY secure memories cannot be read from anywhere. TI provides specific “Secure Copy Code” library functions for each zone to enable the user to copy content from EXEONLY secure Flash sectors to EXEONLY RAM blocks. These functions do the copy-code operation in a highly secure environment and allow a copy to be performed only when the following conditions are met:

- The secure RAM block and the secure Flash sector belong to the same zone.
- Both the secure RAM block and the secure Flash sector have EXEONLY protection enabled.

For further usage of these library functions, see the *ROM Code and Peripheral Booting* chapter.

5.5 SecureCRC

Since reads from EXEONLY memories are not allowed, the user cannot calculate the CRC on content in EXEONLY memories using the CRC engine available on this device (for example, VCUCRC, GCRC) or software. In some safety-critical applications, the user may have to calculate the CRC even on these memories. To enable this without compromising on security, TI provides specific “SecureCRC” library functions for each zone. These functions do the CRC calculation in highly secure environment and allow a CRC calculation to be performed only when the following conditions are met:

- The source address should be modulo the number of words (based on length_id) for which the CRC needs to be calculated.
- The destination address should belong to the same zone as the source address.

For further usage of these library functions, see the device-specific Boot ROM documentation.

Note

The user must disable all the interrupts before calling the secure functions in ROM. If there is a vector fetch during secure function execution, the CPU gets reset immediately.

Disclaimer: The Code Security Module (CSM) included on this device was designed to password protect the data stored in the associated memory and is warranted by Texas Instruments (TI), in accordance with its standard terms and conditions, to conform to TI's published specifications for the warranty period applicable for this device. TI DOES NOT, HOWEVER, WARRANT OR REPRESENT THAT THE CSM CANNOT BE COMPROMISED OR BREACHED OR THAT THE DATA STORED IN THE ASSOCIATED MEMORY CANNOT BE ACCESSED THROUGH OTHER MEANS. MOREOVER, EXCEPT AS SET FORTH ABOVE, TI MAKES NO WARRANTIES OR REPRESENTATIONS CONCERNING THE CSM OR OPERATION OF THIS DEVICE, INCLUDING ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT SHALL TI BE LIABLE FOR ANY CONSEQUENTIAL, SPECIAL, INDIRECT, INCIDENTAL, OR PUNITIVE DAMAGES, HOWEVER CAUSED, ARISING IN ANY WAY OUT OF YOUR USE OF THE CSM OR THIS DEVICE, WHETHER OR NOT TI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. EXCLUDED DAMAGES INCLUDE, BUT ARE NOT LIMITED TO LOSS OF DATA, LOSS OF GOODWILL, LOSS OF USE OR INTERRUPTION OF BUSINESS OR OTHER ECONOMIC LOSS.

5.6 CSM Impact on Other On-Chip Resources

On this device, some of the memories are not secure. To avoid any potential hacking when the device is in the default state (post reset), accesses (all types) to all memories (secure as well as non-secure, except BOOT-ROM and OTP) are disabled until proper security initialization is done. This means that after reset none of the memory resources except BOOT_ROM and OTP is accessible to the user.

The following steps are required by CPU after reset (any type of reset) to initialize the security on device.

Security Initialization

- Dummy Read to address location of SECDC (0x703F0, TI-reserved register) in TI OTP
- Dummy Read to address location of Z1OTP_LINKPOINTER1 in Z1 OTP
- Dummy Read to address location of Z1OTP_LINKPOINTER2 in Z1 OTP
- Dummy Read to address location of Z1OTP_LINKPOINTER3 in Z1 OTP
- Dummy Read to address location of Z2OTP_LINKPOINTER1 in Z2 OTP
- Dummy Read to address location of Z2OTP_LINKPOINTER2 in Z2 OTP
- Dummy Read to address location of Z2OTP_LINKPOINTER3 in Z2 OTP
- Dummy Read to address location Z1OTP_JLM_ENABLE in Z1 OTP
- Dummy Read to address location of Z1OTP_GPREG1, Z1OTP_GPREG2, Z1OTP_GPREG3, Z1OTP_GPREG4 in Z1 OTP
- Dummy Read to address location of Z1OTP_PSWDLOCK in Z1 OTP
- Dummy Read to address location of Z1OTP_CRCLOCK in Z1 OTP
- Dummy Read to address location of Z1OTP_JTAGPSWDH0, Z1OTP_JTAGPSWDH1 in Z1 OTP
- Dummy Read to address location of Z2OTP_GPREG1, Z2OTP_GPREG2, Z2OTP_GPREG3, Z2OTP_GPREG4 in Z2 OTP
- Dummy Read to address location of Z2OTP_PSWDLOCK in Z2 OTP
- Dummy Read to address location of Z2OTP_CRCLOCK in Z2 OTP
- Read to memory map register of Z1_LINKPOINTER in DCSM module to calculate the address of zone select block for Z1
- Dummy read to address location of Z1OTP_GRABSECT1, Z1OTP_GRABSECT2, Z1OTP_GRABSECT3 in Z1 OTP
- Dummy read to address location of Z1OTP_GRABRAM1, Z1OTP_GRABRAM2, Z1OTP_GRABRAM3 in Z1 OTP
- Dummy read to address location of Z1OTP_EXEONLYSECT1, Z1OTP_EXEONLYSECT2 in Z1 OTP
- Dummy read to address location of Z1OTP_EXEONLYRAM1 in Z1 OTP
- Dummy Read to address location of Z1OTP_JTAGPSWDL0, Z1OTP_JTAGPSWDL1 in Z1 OTP
- Read to memory map register of Z2_LINKPOINTER in DCSM module to calculate the address of zone select block for Z2
- Dummy read to address location of Z2OTP_GRABSECT1, Z2OTP_GRABSECT2, Z2OTP_GRABSECT3 in Z2 OTP
- Dummy read to address location of Z2OTP_GRABRAM1, Z2OTP_GRABRAM2, Z2OTP_GRABRAM3 in Z2 OTP
- Dummy read to address location of Z2OTP_EXEONLYSECT1, Z2OTP_EXEONLYSECT2 in Z2 OTP
- Dummy read to address location of Z2OTP_EXEONLYRAM1 in Z2 OTP

Note

Security Initialization is done by BOOTROM code on all the resets (as part of device initialization) that assert SYSRSn. This will not be part of user application code.

The order of initialization matters; hence, if a memory watch window with the USER OTP address is opened in the debugger (CCS), the security initialization could occur in an incorrect order locking the device down. To avoid this, you should not keep a memory window with USER OTP address opened in the debugger (CCS) when performing a reset.

5.7 Incorporating Code Security in User Applications

Code security is typically not required in the development phase of a project. However, security is needed once a robust code is developed for a zone. Before such a code is programmed in the Flash memory, a CSM password should be chosen to secure the zone. Once a CSM password is in place for a zone, the zone is secured (programming a password at the appropriate locations and either performing a device reset or setting the FORCESEC bit (Zx_CR.31) is the action that secures the device). From that time on, access to debug the contents of secure memory by any means (via JTAG, code running off external/on-chip memory, and so forth) requires a valid password. A password is not needed to run the code out of secure memory (such as in a typical end-user usage); however, access to secure memory contents for debug purposes, requires a password.

5.7.1 Environments That Require Security Unlocking

The following are the typical situations under which unsecuring the zone can be required:

- Code development using debuggers (such as Code Composer Studio™ IDE). This is the most common environment during the design phase of a product.
- Flash programming using TI's Flash utilities such as Code Composer Studio On-Chip Flash Programmer plug-in or the Uniflash tool. Flash programming is common during code development and testing. Once the user supplies the necessary password, the Flash utilities disable the security logic before attempting to program the Flash. In custom programming solutions that use the Flash API supplied by TI, unlocking the CSM can be avoided by executing the Flash programming algorithms from secure memory.
- Custom environment defined by the application
  In addition to the above, access to secure memory contents can be required in situations such as:
  - Using the on-chip bootloader to load code or data into secure SARAM or to erase and program the Flash.
  - Executing code from on-chip unsecure memory and requiring access to secure memory for the lookup table. This is not a suggested operating condition as supplying the password from external code could compromise code security.

The unsecuring sequence is identical in all the above situations. This sequence is referred to as the password match flow (PMF) for simplicity. Figure 5-3 explains the sequence of operation that is required every time the user attempts to unsecure a particular zone. A code example is listed for clarity.

5.7.2 CSM Password Match Flow

Password match flow (PMF) is essentially a sequence of four dummy reads from password locations (PWL) followed by four writes (32-bit writes) to CSMKEY(0/1/2/3) registers. Figure 5-3 shows how PMF helps to initialize the security logic registers and disable security logic.

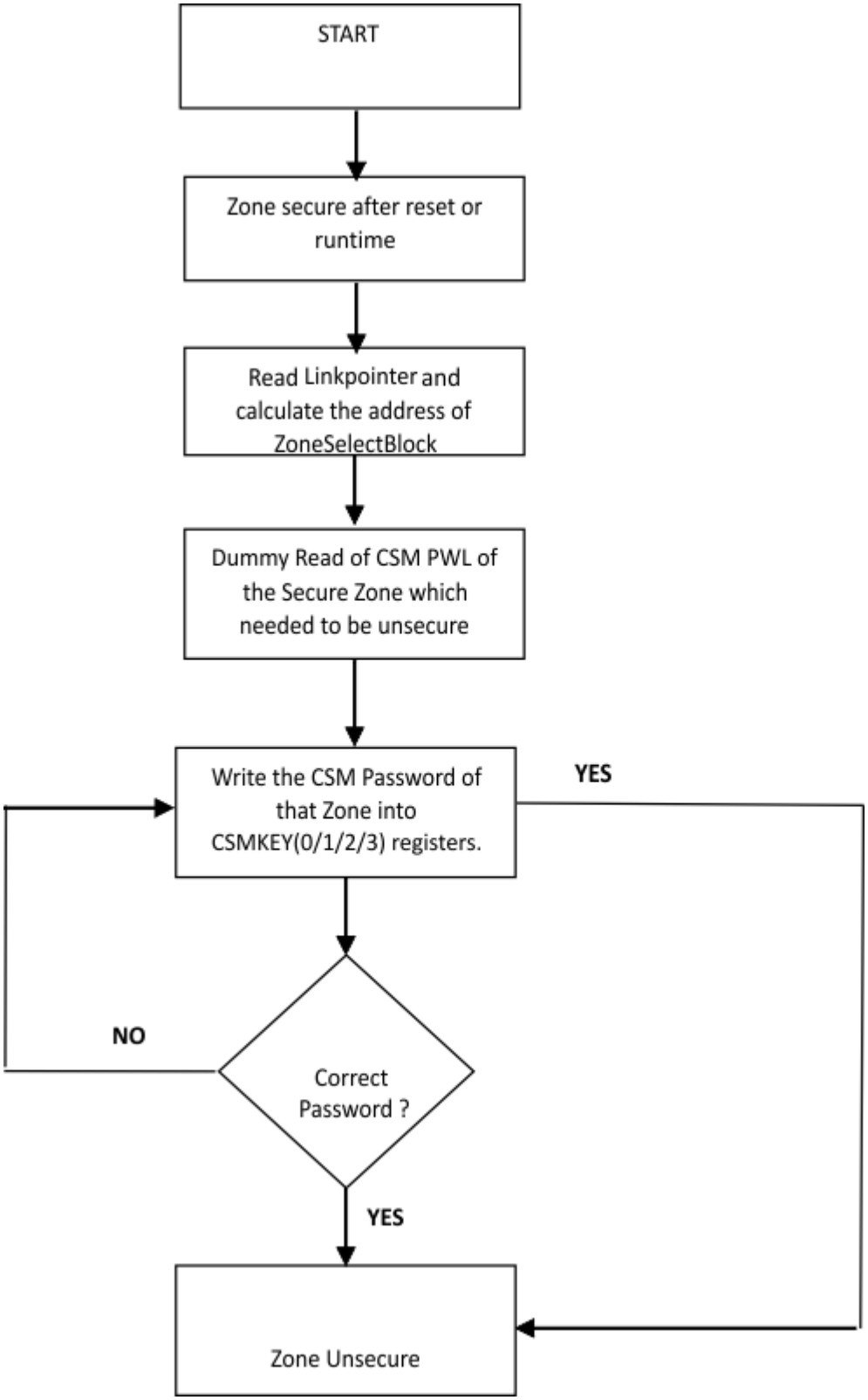


Figure 5-3. CSM Password Match Flow (PMF)

5.7.3 C Code Example to Unsecure C28x Zone1

```
volatile long int *CSM = (volatile long int *)5F090; //CSM register file volatile
long int *CSMPWL = (volatile long int *)0x78020; //CSM Password location (assuming default Zone
select block)
volatile int tmp;
int I;
// Read the 128-bits of the CSM password locations (PWL)
//
for (I=0;I<4; I++) tmp = *CSMPWL++;
// Write the 128-bit password to the CSMKEY registers
// If this password matches that stored in the
// CSLPWL then the CSM will become unsecure. If it does not
// match, then the zone will remain secure.
// An example password of: // 0x1111222233334444555566667777888 is used.
*CSM++ = 0x22221111; // Register Z1_CSMKEY0 at 0x5F090
*CSM++ = 0x44443333; // Register Z1_CSMKEY1 at 0x5F092
*CSM++ = 0x66665555; // Register Z1_CSMKEY2 at 0x5F094
*CSM++ = 0x88887777; // Register Z1_CSMKEY3 at 0x5F096
```

5.7.4 C Code Example to Resecure C28x Zone1

```
volatile int *Z1_CR = 0x5F019; //CSMSCR register
//Set FORCESEC bit
*Z1_CR = 0x8000;
```

5.7.5 Environments That Require ECSL Unlocking

The following are the typical situations under which unsecuring can be required:

- The user develops some main IP, and then outsources peripheral functions to a subcontractor who must be able to run the user code during debug and may halt while main IP code is running. If ECSL is not unlocked, then Code Composer Studio connections will get disconnected, which can be inconvenient for the user. Note that unlocking ECSL does not enable access to secure code but only avoids disconnection of CCS (JTAG).

5.7.6 ECSL Password Match Flow

A password match flow (PMF) is essentially a sequence of eight dummy reads from password locations (PWL) followed by two writes to KEY registers. Figure 5-4 shows how the PMF helps to initialize the security logic registers and disable security logic.

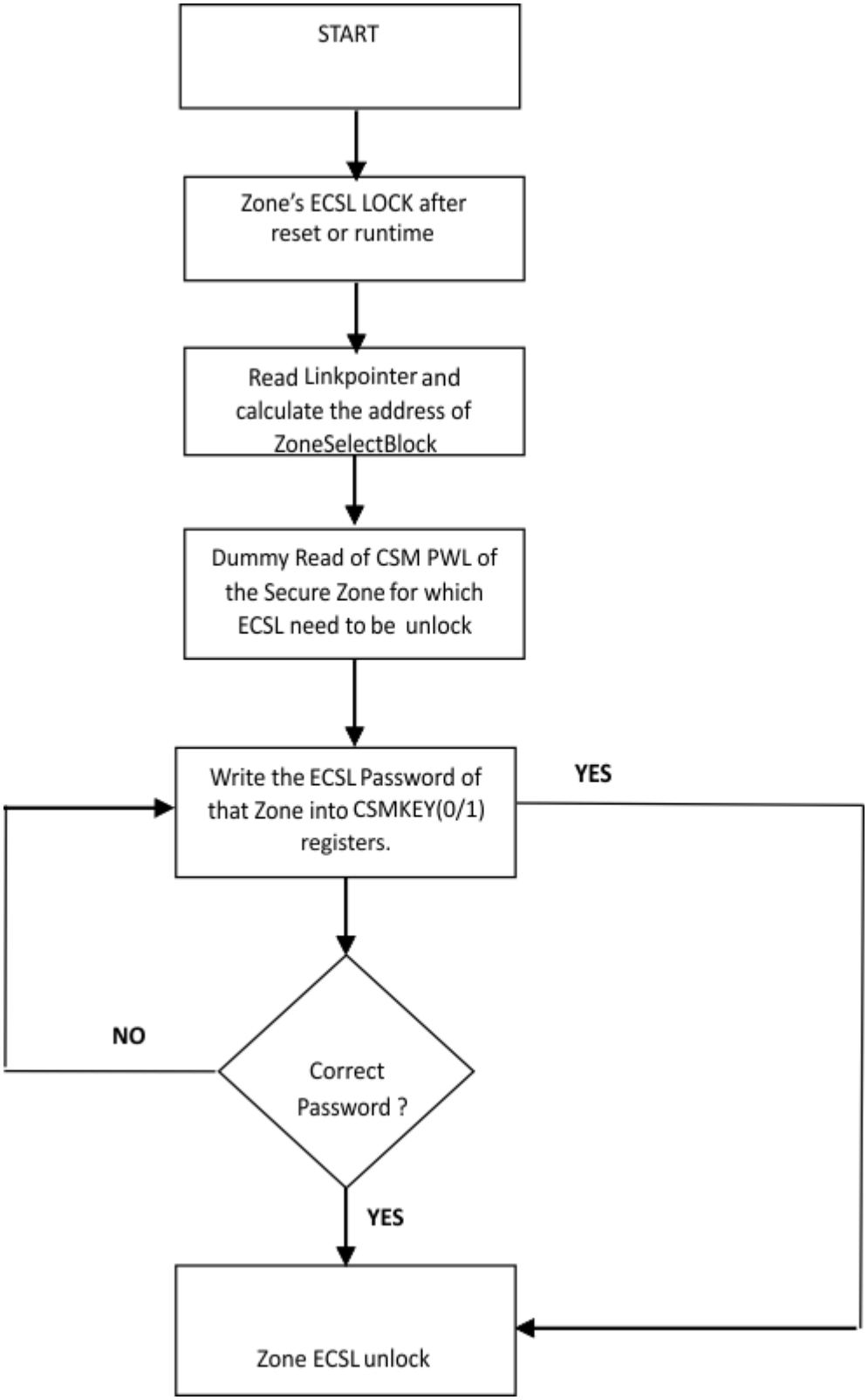


Figure 5-4. ECSL Password Match Flow (PMF)

5.7.7 ECSL Disable Considerations for any Zone

A zone with ECSL enabled should have a predetermined ECSL password stored in the ECSL password locations in Flash (same as lower 64 bits of CSM passwords). The following are steps to disable the ECSL for any particular zone:

- Perform a dummy read of CSM password locations of that Zone.
- Write the password into the CSMKEY0/1 registers, corresponding to that Zone.
- If the password is correct, the ECSL gets disabled; otherwise, it stays enabled.

5.7.7.1 C Code Example to Disable ECSL for C28x Zone1

```
volatile long int *ECSL = (volatile int *)0x5F090; //ECSL register file
volatile long int *ECSLPWL = (volatile int *)0x78028; //ECSL Password location (assuming default
Zone sel block)
volatile int tmp;
int I;
// Read the 64-bits of the password locations (PWL).
for (I=0;I<2; I++) tmp = *ECSLPWL++;
// Write the 64-bit password to the CSMKEYx registers
// If this password matches that stored in the
// CSMPWL then ECSL will get disable. If it does not
// match, then the zone will remain secure.
// An example password of: // 0x1111222233334444 is used.
*ECSL++ = 0x22221111; // Register Z1_CSMKEY0 at 0x5F090
*ECSL++ = 0x44443333; // Register Z1_CSMKEY1 at 0x5F092
```

5.7.8 Device Unique ID

TI OTP contains a 256-bit value that is made up of both random and sequential parts. This value can be used as a seed for code encryption. The starting address of the value is 0x7020E. The first 192 bits are random, the next 32 bits are sequential, and the last 32 bits are a checksum value.

5.8 DCSM Registers

This section describes the various DCSM registers.

5.8.1 DCSM Base Address Table

Table 5-4. DCSM Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
DcsmZ1Regs	DCSM_Z1_REGS	DCSM_Z1_BASE	0x0005_F000	YES	-	-	-	YES
DcsmZ2Regs	DCSM_Z2_REGS	DCSM_Z2_BASE	0x0005_F080	YES	-	-	-	YES
DcsmCommonRegs	DCSM_COMMON_REGS	DCSMCOMMON_BASE	0x0005_F0C0	YES	-	-	-	YES
DcsmZ1OtpRegs	DCSM_Z1_OTP	DCSM_Z1OTP_BASE	0x0007_8000	YES	-	-	-	-
DcsmZ2OtpRegs	DCSM_Z2_OTP	DCSM_Z2OTP_BASE	0x0007_8200	YES	-	-	-	-

5.8.2 DCSM_Z1_REGS Registers

Table 5-5 lists the memory-mapped registers for the DCSM_Z1_REGS registers. All register offset addresses not listed in Table 5-5 should be considered as reserved locations and the register contents should not be modified.

Table 5-5. DCSM_Z1_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	Z1_LINKPOINTER	Zone 1 Link Pointer		Go
2h	Z1_OTPSECLOCK	Zone 1 OTP Secure Lock		Go
4h	Z1_JLM_ENABLE	Zone 1 JTAGLOCK Enable Register		Go
6h	Z1_LINKPOINTERERR	Link Pointer Error		Go
8h	Z1_GPREG1	Zone 1 General Purpose Register-1		Go
Ah	Z1_GPREG2	Zone 1 General Purpose Register-2		Go
Ch	Z1_GPREG3	Zone 1 General Purpose Register-3		Go
Eh	Z1_GPREG4	Zone 1 General Purpose Register-4		Go
10h	Z1_CSMKEY0	Zone 1 CSM Key 0		Go
12h	Z1_CSMKEY1	Zone 1 CSM Key 1		Go
14h	Z1_CSMKEY2	Zone 1 CSM Key 2		Go
16h	Z1_CSMKEY3	Zone 1 CSM Key 3		Go
18h	Z1_CR	Zone 1 CSM Control Register		Go
1Ah	Z1_GRABSECT1R	Zone 1 Grab Flash Status Register 1		Go
1Ch	Z1_GRABSECT2R	Zone 1 Grab Flash Status Register 2		Go
1Eh	Z1_GRABSECT3R	Zone 1 Grab Flash Status Register 3		Go
20h	Z1_GRABRAM1R	Zone 1 Grab RAM Status Register 1		Go
26h	Z1_EXEONLYSECT1R	Zone 1 Execute Only Flash Status Register 1		Go
28h	Z1_EXEONLYSECT2R	Zone 1 Execute Only Flash Status Register 2		Go
2Ah	Z1_EXEONLYRAM1R	Zone 1 Execute Only RAM Status Register 1		Go
2Eh	Z1_JTAGKEY0	JTAG Unlock Key Register 0		Go
30h	Z1_JTAGKEY1	JTAG Unlock Key Register 1		Go
32h	Z1_JTAGKEY2	JTAG Unlock Key Register 2		Go
34h	Z1_JTAGKEY3	JTAG Unlock Key Register 3		Go
36h	Z1_CMACKEY0	Secure Boot CMAC Key Status Register 0		Go
38h	Z1_CMACKEY1	Secure Boot CMAC Key Status Register 1		Go
3Ah	Z1_CMACKEY2	Secure Boot CMAC Key Status Register 2		Go
3Ch	Z1_CMACKEY3	Secure Boot CMAC Key Status Register 3		Go

Complex bit access types are encoded to fit into small table cells. Table 5-6 shows the codes that are used for access types in this section.

Table 5-6. DCSM_Z1_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value

Table 5-6. DCSM_Z1_REGS Access Type Codes (continued)

Access Type	Code	Description
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

5.8.2.1 Z1_LINKPOINTER Register (Offset = 0h) [Reset = FFFFC000h]

Z1_LINKPOINTER is shown in Figure 5-5 and described in Table 5-7.

Return to the Summary Table.

Zone 1 Link Pointer

Figure 5-5. Z1_LINKPOINTER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																		LINKPOINTER													
R-0h																		R-0h													

Table 5-7. Z1_LINKPOINTER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Reserved
13-0	LINKPOINTER	R	0h	This is resolved Link-Pointer value which is generated by looking at the three physical Link-Pointer values loaded from OTP Reset type: SYSRSn

5.8.2.2 Z1_OTPSECLOCK Register (Offset = 2h) [Reset = 1h]

Z1_OTPSECLOCK is shown in Figure 5-6 and described in Table 5-8.

Return to the Summary Table.

Zone 1 OTP Secure Lock

Figure 5-6. Z1_OTPSECLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				CRCLOCK			
R-0h				R-0h			
7	6	5	4	3	2	1	0
PSWDLOCK				RESERVED			JTAGLOCK
R-0h				R-0h			R-1h

Table 5-8. Z1_OTPSECLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-12	RESERVED	R	0h	Reserved
11-8	CRCLOCK	R	0h	Value in this field gets loaded from Z1_CRCLOCK[3:0] when a read is issued to address location of Z1_CRCLOCK in OTP. 1111 : VCU has ability to calculate CRC on secure memories. Other Value : VCU doesn't have ability to calculate CRC on secure memories. Reset type: XRSn
7-4	PSWDLOCK	R	0h	Value in this field gets loaded from Z1_PSWDLOCK[3:0] when a read is issued to address location of Z1_PSWDLOCK in OTP. 1111 : CSM password locations in OTP are not protected and can be read from debugger as well as code running from anywhere. Other Value : CSM password locations in OTP are protected and can't be read without unlocking CSM of that zone. Reset type: XRSn
3-1	RESERVED	R	0h	Reserved
0	JTAGLOCK	R	1h	Reflects the state of the JTAGLOCK feature. 0 : JTAG is not locked 1 : JTAG is locked Reset type: PORESETn

5.8.2.3 Z1_JLM_ENABLE Register (Offset = 4h) [Reset = Fh]

Z1_JLM_ENABLE is shown in Figure 5-7 and described in Table 5-9.

Return to the Summary Table.

Zone 1 JTAGLOCK Enable Register

Figure 5-7. Z1_JLM_ENABLE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				Z1_JLM_ENABLE			
R-0h				R-Fh			

Table 5-9. Z1_JLM_ENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3-0	Z1_JLM_ENABLE	R	Fh	Zone1 JLM_ENABLE register. The value in this field gets loaded from Z1OTP_JLM_ENABLE[3:0] when a read is issued to address location of Z1OTP_JLM_ENABLE in OTP. If Z1OTP_JLM_ENABLE[31:0] is equal to all ones during the load, the JTAGLOCK is not bypassed (is enabled). If the value of Z1OTP_JLM_ENABLE[31:0] is not all ones during the load, the JTAGLOCK is governed as follows by the Z1_JLM_ENABLE bits: 1111 : JTAG/Emulation access is allowed (JTAGLOCK is not enabled) Other values: JTAGLOCK is governed by the JTAGKEY==JTAGPSWD match condition Reset type: PORESETn

5.8.2.4 Z1_LINKPOINTERERR Register (Offset = 6h) [Reset = 0h]

Z1_LINKPOINTERERR is shown in Figure 5-8 and described in Table 5-10.

Return to the Summary Table.

Link Pointer Error

Figure 5-8. Z1_LINKPOINTERERR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED		Z1_LINKPOINTERERR													
R-0h		R-0h													

Table 5-10. Z1_LINKPOINTERERR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Reserved
13-0	Z1_LINKPOINTERERR	R	0h	These bits indicate errors during formation of the resolved Link-Pointer value after the three physical Link-Pointer values loaded of OTP in flash. 0 : No Error. Other : Error on bit positions which is set to 1. Reset type: SYSRSn

5.8.2.5 Z1_GPREG1 Register (Offset = 8h) [Reset = 0h]

Z1_GPREG1 is shown in Figure 5-9 and described in Table 5-11.

Return to the Summary Table.

Zone 1 General Purpose Register-1

Figure 5-9. Z1_GPREG1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPREG1																															
R-0h																															

Table 5-11. Z1_GPREG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPREG1	R	0h	This field gets loaded with the contents of Z1OTP_GPREG1 locations in Zone-1-USER-OTP when a dummy read is issued to that address. Users can use this register to load any general purpose non-voliate content from USER-OTP of Zone-1 Reset type: SYSRSn

5.8.2.6 Z1_GPREG2 Register (Offset = Ah) [Reset = 0h]

Z1_GPREG2 is shown in Figure 5-10 and described in Table 5-12.

Return to the Summary Table.

Zone 1 General Purpose Register-2

Figure 5-10. Z1_GPREG2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPREG2																															
R-0h																															

Table 5-12. Z1_GPREG2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPREG2	R	0h	This field gets loaded with the contents of Z1OTP_GPREG2 locations in Zone-1-USER-OTP when a dummy read is issued to that address. Users can use this register to load any general purpose non-voliate content from USER-OTP of Zone-1 Reset type: SYSRSn

5.8.2.7 Z1_GPREG3 Register (Offset = Ch) [Reset = 0h]

Z1_GPREG3 is shown in Figure 5-11 and described in Table 5-13.

Return to the Summary Table.

Zone 1 General Purpose Register-3

Figure 5-11. Z1_GPREG3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPREG3																															
R-0h																															

Table 5-13. Z1_GPREG3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPREG3	R	0h	This field gets loaded with the contents of Z1OTP_GPREG3 locations in Zone-1-USER-OTP when a dummy read is issued to that address. Users can use this register to load any general purpose non-voliate content from USER-OTP of Zone-1 Reset type: SYSRSn

5.8.2.8 Z1_GPREG4 Register (Offset = Eh) [Reset = 0h]

Z1_GPREG4 is shown in Figure 5-12 and described in Table 5-14.

Return to the Summary Table.

Zone 1 General Purpose Register-4

Figure 5-12. Z1_GPREG4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPREG4																															
R-0h																															

Table 5-14. Z1_GPREG4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPREG4	R	0h	This field gets loaded with the contents of Z1OTP_GPREG4 locations in Zone-1-USER-OTP when a dummy read is issued to that address. Users can use this register to load any general purpose non-voliate content from USER-OTP of Zone-1 Reset type: SYSRSn

5.8.2.9 Z1_CSMKEY0 Register (Offset = 10h) [Reset = 0h]

Z1_CSMKEY0 is shown in Figure 5-13 and described in Table 5-15.

Return to the Summary Table.

Zone 1 CSM Key 0

Figure 5-13. Z1_CSMKEY0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1_CSMKEY0																															
R/W-0h																															

Table 5-15. Z1_CSMKEY0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1_CSMKEY0	R/W	0h	To unlock Zone1, user needs to write this regsiter with exact value as Z1_CSMPSWD0, programmed in OTP (zone gets unlock only if 128 bit password in OTP match with value written in four CSMKEY registers.) Reset type: SYSRSn

5.8.2.10 Z1_CSMKEY1 Register (Offset = 12h) [Reset = 0h]

Z1_CSMKEY1 is shown in Figure 5-14 and described in Table 5-16.

Return to the Summary Table.

Zone 1 CSM Key 1

Figure 5-14. Z1_CSMKEY1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1_CSMKEY1																															
R/W-0h																															

Table 5-16. Z1_CSMKEY1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1_CSMKEY1	R/W	0h	To unlock Zone1, user needs to write this regsiter with exact value as Z1_CSMPSWD1, programmed in OTP (zone gets unlock only if 128 bit password in OTP match with value written in four CSMKEY registers.) Reset type: SYSRSn

5.8.2.11 Z1_CSMKEY2 Register (Offset = 14h) [Reset = 0h]

Z1_CSMKEY2 is shown in Figure 5-15 and described in Table 5-17.

Return to the Summary Table.

Zone 1 CSM Key 2

Figure 5-15. Z1_CSMKEY2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1_CSMKEY2																															
R/W-0h																															

Table 5-17. Z1_CSMKEY2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1_CSMKEY2	R/W	0h	To unlock Zone1, user needs to write this regsiter with exact value as Z1_CSMPSWD2, programmed in OTP (zone gets unlock only if 128 bit password in OTP match with value written in four CSMKEY registers.) Reset type: SYSRSn

5.8.2.12 Z1_CSMKEY3 Register (Offset = 16h) [Reset = 0h]

Z1_CSMKEY3 is shown in Figure 5-16 and described in Table 5-18.

Return to the Summary Table.

Zone 1 CSM Key 3

Figure 5-16. Z1_CSMKEY3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1_CSMKEY3																															
R/W-0h																															

Table 5-18. Z1_CSMKEY3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1_CSMKEY3	R/W	0h	To unlock Zone1, user needs to write this regsiter with exact value as Z1_CSMPSWD3, programmed in OTP (zone gets unlock only if 128 bit password in OTP match with value written in four CSMKEY registers.) Reset type: SYSRSn

5.8.2.13 Z1_CR Register (Offset = 18h) [Reset = 00080000h]

Z1_CR is shown in Figure 5-17 and described in Table 5-19.

Return to the Summary Table.

Zone 1 CSM Control Register

Figure 5-17. Z1_CR Register

31	30	29	28	27	26	25	24
FORCESEC	RESERVED						
R-0/W-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED	ARMED	UNSECURE	ALLONE	ALLZERO	RESERVED		
R-0h	R-0h	R-0h	R-0h	R-1h	R-0h		
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	RESERVED	RESERVED	RESERVED
R-0h				R-0h	R-0h	R-0h	R-0h

Table 5-19. Z1_CR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	FORCESEC	R-0/W	0h	A write of '1' to this bit relocks the zone by clearing the zone's CSMKEYx registers. If the zone's CSMPSWDx OTP locations were updated prior to using this bit, it is suggested that all of the new password registers be dummy loaded from OTP immediately to avoid a mix between old and new password data. Reset type: SYSRSn
30-24	RESERVED	R	0h	Reserved
23	RESERVED	R	0h	Reserved
22	ARMED	R	0h	0 : Dummy read to CSM Password locations in OTP hasn't been performed. 1 : Dummy read to CSM Password locations in OTP has been performed. Reset type: SYSRSn
21	UNSECURE	R	0h	Indiacates the state of Zone. 0 : Zone is in lock(secure) state. 1 : Zone is in unlock(unsecure) state. Reset type: SYSRSn
20	ALLONE	R	0h	Indicates the state of CSM passwords. 0 : Zone CSM Passwords are not all ones. 1 : Zone CSM Passwords are all ones. Reset type: SYSRSn
19	ALLZERO	R	1h	Indicates the state of CSM passowrds. 0 : CSM Passwords are not all zeros. 1 : CSM Passwords are all zero and device is permanently locked. Reset type: SYSRSn
18-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	RESERVED	R	0h	Reserved
2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved

Table 5-19. Z1_CR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	RESERVED	R	0h	Reserved

5.8.2.14 Z1_GRABSECT1R Register (Offset = 1Ah) [Reset = 0h]

Z1_GRABSECT1R is shown in Figure 5-18 and described in Table 5-20.

Return to the Summary Table.

Zone 1 Grab Flash Status Register 1

Figure 5-18. Z1_GRABSECT1R Register

31	30	29	28	27	26	25	24
GRAB_SECT15		GRAB_SECT14		GRAB_SECT13		GRAB_SECT12	
R-0h		R-0h		R-0h		R-0h	
23	22	21	20	19	18	17	16
GRAB_SECT11		GRAB_SECT10		GRAB_SECT9		GRAB_SECT8	
R-0h		R-0h		R-0h		R-0h	
15	14	13	12	11	10	9	8
GRAB_SECT7		GRAB_SECT6		GRAB_SECT5		GRAB_SECT4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
GRAB_SECT3		GRAB_SECT2		GRAB_SECT1		GRAB_SECT0	
R-0h		R-0h		R-0h		R-0h	

Table 5-20. Z1_GRABSECT1R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GRAB_SECT15	R	0h	Value in this field gets loaded from Z1_GRABSECT1[31:30] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 15 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 15 to Zone1. 10 : No request for Bank 0 Flash Sector 15 11 : No request for Bank 0 Flash Sector 15 when this zone is UNLOCKED. Else Bank 0 Flash Sector 15 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
29-28	GRAB_SECT14	R	0h	Value in this field gets loaded from Z1_GRABSECT1[29:28] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 14 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 14 to Zone1. 10 : No request for Bank 0 Flash Sector 14 11 : No request for Bank 0 Flash Sector 14 when this zone is UNLOCKED. Else Bank 0 Flash Sector 14 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
27-26	GRAB_SECT13	R	0h	Value in this field gets loaded from Z1_GRABSECT1[27:26] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 13 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 13 to Zone1. 10 : No request for Bank 0 Flash Sector 13 11 : No request for Bank 0 Flash Sector 13 when this zone is UNLOCKED. Else Bank 0 Flash Sector 13 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 5-20. Z1_GRABSECT1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-24	GRAB_SECT12	R	0h	Value in this field gets loaded from Z1_GRABSECT1[25:24] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 12 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 12 to Zone1. 10 : No request for Bank 0 Flash Sector 12 11 : No request for Bank 0 Flash Sector 12 when this zone is UNLOCKED. Else Bank 0 Flash Sector 12 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
23-22	GRAB_SECT11	R	0h	Value in this field gets loaded from Z1_GRABSECT1[23:22] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 11 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 11 to Zone1. 10 : No request for Bank 0 Flash Sector 11 11 : No request for Bank 0 Flash Sector 11 when this zone is UNLOCKED. Else Bank 0 Flash Sector 11 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
21-20	GRAB_SECT10	R	0h	Value in this field gets loaded from Z1_GRABSECT1[21:20] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 10 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 10 to Zone1. 10 : No request for Bank 0 Flash Sector 10 11 : No request for Bank 0 Flash Sector 10 when this zone is UNLOCKED. Else Bank 0 Flash Sector 10 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
19-18	GRAB_SECT9	R	0h	Value in this field gets loaded from Z1_GRABSECT1[19:18] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 9 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 9 to Zone1. 10 : No request for Bank 0 Flash Sector 9 11 : No request for Bank 0 Flash Sector 9 when this zone is UNLOCKED. Else Bank 0 Flash Sector 9 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
17-16	GRAB_SECT8	R	0h	Value in this field gets loaded from Z1_GRABSECT1[17:16] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 8 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 8 to Zone1. 10 : No request for Bank 0 Flash Sector 8 11 : No request for Bank 0 Flash Sector 8 when this zone is UNLOCKED. Else Bank 0 Flash Sector 8 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
15-14	GRAB_SECT7	R	0h	Value in this field gets loaded from Z1_GRABSECT1[15:14] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 7 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 7 to Zone1. 10 : No request for Bank 0 Flash Sector 7 11 : No request for Bank 0 Flash Sector 7 when this zone is UNLOCKED. Else Bank 0 Flash Sector 7 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
13-12	GRAB_SECT6	R	0h	Value in this field gets loaded from Z1_GRABSECT1[13:12] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 6 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 6 to Zone1. 10 : No request for Bank 0 Flash Sector 6 11 : No request for Bank 0 Flash Sector 6 when this zone is UNLOCKED. Else Bank 0 Flash Sector 6 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 5-20. Z1_GRABSECT1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11-10	GRAB_SECT5	R	0h	Value in this field gets loaded from Z1_GRABSECT1[11:10] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 5 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 5 to Zone1. 10 : No request for Bank 0 Flash Sector 5 11 : No request for Bank 0 Flash Sector 5 when this zone is UNLOCKED. Else Bank 0 Flash Sector 5 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
9-8	GRAB_SECT4	R	0h	Value in this field gets loaded from Z1_GRABSECT1[9:8] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 4 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 4 to Zone1. 10 : No request for Bank 0 Flash Sector 4 11 : No request for Bank 0 Flash Sector 4 when this zone is UNLOCKED. Else Bank 0 Flash Sector 4 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
7-6	GRAB_SECT3	R	0h	Value in this field gets loaded from Z1_GRABSECT1[7:6] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 3 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 3 to Zone1. 10 : No request for Bank 0 Flash Sector 3 11 : No request for Bank 0 Flash Sector 3 when this zone is UNLOCKED. Else Bank 0 Flash Sector 3 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
5-4	GRAB_SECT2	R	0h	Value in this field gets loaded from Z1_GRABSECT1[5:4] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 2 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 2 to Zone1. 10 : No request for Bank 0 Flash Sector 2 11 : No request for Bank 0 Flash Sector 2 when this zone is UNLOCKED. Else Bank 0 Flash Sector 2 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
3-2	GRAB_SECT1	R	0h	Value in this field gets loaded from Z1_GRABSECT1[3:2] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 1 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 1 to Zone1. 10 : No request for Bank 0 Flash Sector 1 11 : No request for Bank 0 Flash Sector 1 when this zone is UNLOCKED. Else Bank 0 Flash Sector 1 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
1-0	GRAB_SECT0	R	0h	Value in this field gets loaded from Z1_GRABSECT1[1:0] when a read is issued to address location of Z1_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 0 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 0 to Zone1. 10 : No request for Bank 0 Flash Sector 0 11 : No request for Bank 0 Flash Sector 0 when this zone is UNLOCKED. Else Bank 0 Flash Sector 0 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

5.8.2.15 Z1_GRABSECT2R Register (Offset = 1Ch) [Reset = 0h]

Z1_GRABSECT2R is shown in Figure 5-19 and described in Table 5-21.

Return to the Summary Table.

Zone 1 Grab Flash Status Register 2

Figure 5-19. Z1_GRABSECT2R Register

31	30	29	28	27	26	25	24
GRAB_SECT15		GRAB_SECT14		GRAB_SECT13		GRAB_SECT12	
R-0h		R-0h		R-0h		R-0h	
23	22	21	20	19	18	17	16
GRAB_SECT11		GRAB_SECT10		GRAB_SECT9		GRAB_SECT8	
R-0h		R-0h		R-0h		R-0h	
15	14	13	12	11	10	9	8
GRAB_SECT7		GRAB_SECT6		GRAB_SECT5		GRAB_SECT4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
GRAB_SECT3		GRAB_SECT2		GRAB_SECT1		GRAB_SECT0	
R-0h		R-0h		R-0h		R-0h	

Table 5-21. Z1_GRABSECT2R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GRAB_SECT15	R	0h	Value in this field gets loaded from Z1_GRABSECT2[31:30] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 15 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 15 to Zone1. 10 : No request for Bank 1 Flash Sector 15 11 : No request for Bank 1 Flash Sector 15 when this zone is UNLOCKED. Else Bank 1 Flash Sector 15 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
29-28	GRAB_SECT14	R	0h	Value in this field gets loaded from Z1_GRABSECT2[29:28] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 14 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 14 to Zone1. 10 : No request for Bank 1 Flash Sector 14 11 : No request for Bank 1 Flash Sector 14 when this zone is UNLOCKED. Else Bank 1 Flash Sector 14 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
27-26	GRAB_SECT13	R	0h	Value in this field gets loaded from Z1_GRABSECT2[27:26] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 13 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 13 to Zone1. 10 : No request for Bank 1 Flash Sector 13 11 : No request for Bank 1 Flash Sector 13 when this zone is UNLOCKED. Else Bank 1 Flash Sector 13 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 5-21. Z1_GRABSECT2R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-24	GRAB_SECT12	R	0h	Value in this field gets loaded from Z1_GRABSECT2[25:24] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 12 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 12 to Zone1. 10 : No request for Bank 1 Flash Sector 12 11 : No request for Bank 1 Flash Sector 12 when this zone is UNLOCKED. Else Bank 1 Flash Sector 12 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
23-22	GRAB_SECT11	R	0h	Value in this field gets loaded from Z1_GRABSECT2[23:22] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 11 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 11 to Zone1. 10 : No request for Bank 1 Flash Sector 11 11 : No request for Bank 1 Flash Sector 11 when this zone is UNLOCKED. Else Bank 1 Flash Sector 11 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
21-20	GRAB_SECT10	R	0h	Value in this field gets loaded from Z1_GRABSECT2[21:20] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 10 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 10 to Zone1. 10 : No request for Bank 1 Flash Sector 10 11 : No request for Bank 1 Flash Sector 10 when this zone is UNLOCKED. Else Bank 1 Flash Sector 10 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
19-18	GRAB_SECT9	R	0h	Value in this field gets loaded from Z1_GRABSECT2[19:18] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 9 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 9 to Zone1. 10 : No request for Bank 1 Flash Sector 9 11 : No request for Bank 1 Flash Sector 9 when this zone is UNLOCKED. Else Bank 1 Flash Sector 9 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
17-16	GRAB_SECT8	R	0h	Value in this field gets loaded from Z1_GRABSECT2[17:16] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 8 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 8 to Zone1. 10 : No request for Bank 1 Flash Sector 8 11 : No request for Bank 1 Flash Sector 8 when this zone is UNLOCKED. Else Bank 1 Flash Sector 8 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
15-14	GRAB_SECT7	R	0h	Value in this field gets loaded from Z1_GRABSECT2[15:14] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 7 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 7 to Zone1. 10 : No request for Bank 1 Flash Sector 7 11 : No request for Bank 1 Flash Sector 7 when this zone is UNLOCKED. Else Bank 1 Flash Sector 7 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
13-12	GRAB_SECT6	R	0h	Value in this field gets loaded from Z1_GRABSECT2[13:12] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 6 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 6 to Zone1. 10 : No request for Bank 1 Flash Sector 6 11 : No request for Bank 1 Flash Sector 6 when this zone is UNLOCKED. Else Bank 1 Flash Sector 6 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 5-21. Z1_GRABSECT2R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11-10	GRAB_SECT5	R	0h	Value in this field gets loaded from Z1_GRABSECT2[11:10] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 5 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 5 to Zone1. 10 : No request for Bank 1 Flash Sector 5 11 : No request for Bank 1 Flash Sector 5 when this zone is UNLOCKED. Else Bank 1 Flash Sector 5 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
9-8	GRAB_SECT4	R	0h	Value in this field gets loaded from Z1_GRABSECT2[9:8] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 4 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 4 to Zone1. 10 : No request for Bank 1 Flash Sector 4 11 : No request for Bank 1 Flash Sector 4 when this zone is UNLOCKED. Else Bank 1 Flash Sector 4 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
7-6	GRAB_SECT3	R	0h	Value in this field gets loaded from Z1_GRABSECT2[7:6] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 3 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 3 to Zone1. 10 : No request for Bank 1 Flash Sector 3 11 : No request for Bank 1 Flash Sector 3 when this zone is UNLOCKED. Else Bank 1 Flash Sector 3 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
5-4	GRAB_SECT2	R	0h	Value in this field gets loaded from Z1_GRABSECT2[5:4] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 2 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 2 to Zone1. 10 : No request for Bank 1 Flash Sector 2 11 : No request for Bank 1 Flash Sector 2 when this zone is UNLOCKED. Else Bank 1 Flash Sector 2 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
3-2	GRAB_SECT1	R	0h	Value in this field gets loaded from Z1_GRABSECT2[3:2] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 1 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 1 to Zone1. 10 : No request for Bank 1 Flash Sector 1 11 : No request for Bank 1 Flash Sector 1 when this zone is UNLOCKED. Else Bank 1 Flash Sector 1 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
1-0	GRAB_SECT0	R	0h	Value in this field gets loaded from Z1_GRABSECT2[1:0] when a read is issued to address location of Z1_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 0 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 0 to Zone1. 10 : No request for Bank 1 Flash Sector 0 11 : No request for Bank 1 Flash Sector 0 when this zone is UNLOCKED. Else Bank 1 Flash Sector 0 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

5.8.2.16 Z1_GRABSECT3R Register (Offset = 1Eh) [Reset = 0h]

Z1_GRABSECT3R is shown in Figure 5-20 and described in Table 5-22.

Return to the Summary Table.

Zone 1 Grab Flash Status Register 3

Figure 5-20. Z1_GRABSECT3R Register

31	30	29	28	27	26	25	24
GRAB_SECT15		GRAB_SECT14		GRAB_SECT13		GRAB_SECT12	
R-0h		R-0h		R-0h		R-0h	
23	22	21	20	19	18	17	16
GRAB_SECT11		GRAB_SECT10		GRAB_SECT9		GRAB_SECT8	
R-0h		R-0h		R-0h		R-0h	
15	14	13	12	11	10	9	8
GRAB_SECT7		GRAB_SECT6		GRAB_SECT5		GRAB_SECT4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
GRAB_SECT3		GRAB_SECT2		GRAB_SECT1		GRAB_SECT0	
R-0h		R-0h		R-0h		R-0h	

Table 5-22. Z1_GRABSECT3R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GRAB_SECT15	R	0h	Value in this field gets loaded from Z1_GRABSECT3[31:30] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 15 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 15 to Zone1. 10 : No request for Bank 2 Flash Sector 15 11 : No request for Bank 2 Flash Sector 15 when this zone is UNLOCKED. Else Bank 2 Flash Sector 15 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
29-28	GRAB_SECT14	R	0h	Value in this field gets loaded from Z1_GRABSECT3[29:28] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 14 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 14 to Zone1. 10 : No request for Bank 2 Flash Sector 14 11 : No request for Bank 2 Flash Sector 14 when this zone is UNLOCKED. Else Bank 2 Flash Sector 14 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
27-26	GRAB_SECT13	R	0h	Value in this field gets loaded from Z1_GRABSECT3[27:26] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 13 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 13 to Zone1. 10 : No request for Bank 2 Flash Sector 13 11 : No request for Bank 2 Flash Sector 13 when this zone is UNLOCKED. Else Bank 2 Flash Sector 13 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 5-22. Z1_GRABSECT3R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-24	GRAB_SECT12	R	0h	Value in this field gets loaded from Z1_GRABSECT3[25:24] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 12 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 12 to Zone1. 10 : No request for Bank 2 Flash Sector 12 11 : No request for Bank 2 Flash Sector 12 when this zone is UNLOCKED. Else Bank 2 Flash Sector 12 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
23-22	GRAB_SECT11	R	0h	Value in this field gets loaded from Z1_GRABSECT3[23:22] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 11 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 11 to Zone1. 10 : No request for Bank 2 Flash Sector 11 11 : No request for Bank 2 Flash Sector 11 when this zone is UNLOCKED. Else Bank 2 Flash Sector 11 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
21-20	GRAB_SECT10	R	0h	Value in this field gets loaded from Z1_GRABSECT3[21:20] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 10 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 10 to Zone1. 10 : No request for Bank 2 Flash Sector 10 11 : No request for Bank 2 Flash Sector 10 when this zone is UNLOCKED. Else Bank 2 Flash Sector 10 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
19-18	GRAB_SECT9	R	0h	Value in this field gets loaded from Z1_GRABSECT3[19:18] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 9 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 9 to Zone1. 10 : No request for Bank 2 Flash Sector 9 11 : No request for Bank 2 Flash Sector 9 when this zone is UNLOCKED. Else Bank 2 Flash Sector 9 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
17-16	GRAB_SECT8	R	0h	Value in this field gets loaded from Z1_GRABSECT3[17:16] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 8 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 8 to Zone1. 10 : No request for Bank 2 Flash Sector 8 11 : No request for Bank 2 Flash Sector 8 when this zone is UNLOCKED. Else Bank 2 Flash Sector 8 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
15-14	GRAB_SECT7	R	0h	Value in this field gets loaded from Z1_GRABSECT3[15:14] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 7 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 7 to Zone1. 10 : No request for Bank 2 Flash Sector 7 11 : No request for Bank 2 Flash Sector 7 when this zone is UNLOCKED. Else Bank 2 Flash Sector 7 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
13-12	GRAB_SECT6	R	0h	Value in this field gets loaded from Z1_GRABSECT3[13:12] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 6 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 6 to Zone1. 10 : No request for Bank 2 Flash Sector 6 11 : No request for Bank 2 Flash Sector 6 when this zone is UNLOCKED. Else Bank 2 Flash Sector 6 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 5-22. Z1_GRABSECT3R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11-10	GRAB_SECT5	R	0h	Value in this field gets loaded from Z1_GRABSECT3[11:10] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 5 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 5 to Zone1. 10 : No request for Bank 2 Flash Sector 5 11 : No request for Bank 2 Flash Sector 5 when this zone is UNLOCKED. Else Bank 2 Flash Sector 5 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
9-8	GRAB_SECT4	R	0h	Value in this field gets loaded from Z1_GRABSECT3[9:8] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 4 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 4 to Zone1. 10 : No request for Bank 2 Flash Sector 4 11 : No request for Bank 2 Flash Sector 4 when this zone is UNLOCKED. Else Bank 2 Flash Sector 4 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
7-6	GRAB_SECT3	R	0h	Value in this field gets loaded from Z1_GRABSECT3[7:6] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 3 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 3 to Zone1. 10 : No request for Bank 2 Flash Sector 3 11 : No request for Bank 2 Flash Sector 3 when this zone is UNLOCKED. Else Bank 2 Flash Sector 3 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
5-4	GRAB_SECT2	R	0h	Value in this field gets loaded from Z1_GRABSECT3[5:4] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 2 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 2 to Zone1. 10 : No request for Bank 2 Flash Sector 2 11 : No request for Bank 2 Flash Sector 2 when this zone is UNLOCKED. Else Bank 2 Flash Sector 2 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
3-2	GRAB_SECT1	R	0h	Value in this field gets loaded from Z1_GRABSECT3[3:2] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 1 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 1 to Zone1. 10 : No request for Bank 2 Flash Sector 1 11 : No request for Bank 2 Flash Sector 1 when this zone is UNLOCKED. Else Bank 2 Flash Sector 1 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
1-0	GRAB_SECT0	R	0h	Value in this field gets loaded from Z1_GRABSECT3[1:0] when a read is issued to address location of Z1_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 0 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 0 to Zone1. 10 : No request for Bank 2 Flash Sector 0 11 : No request for Bank 2 Flash Sector 0 when this zone is UNLOCKED. Else Bank 2 Flash Sector 0 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

5.8.2.17 Z1_GRABRAM1R Register (Offset = 20h) [Reset = 0h]

Z1_GRABRAM1R is shown in Figure 5-21 and described in Table 5-23.

Return to the Summary Table.

Zone 1 Grab RAM Status Register 1

Figure 5-21. Z1_GRABRAM1R Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
GRAB_RAM7		GRAB_RAM6		GRAB_RAM5		GRAB_RAM4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
GRAB_RAM3		GRAB_RAM2		GRAB_RAM1		GRAB_RAM0	
R-0h		R-0h		R-0h		R-0h	

Table 5-23. Z1_GRABRAM1R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-14	GRAB_RAM7	R	0h	Value in this field gets loaded from Z1_GRABRAM1[15:14] when a read is issued to address location of Z1_GRABRAM1 in OTP. 00 : Invalid. LS7 RAM is inaccessible. 01 : Request to allocate LS7 RAM to Zone1. 10 : No request for LS7 RAM 11 : No request for LS7 RAM when this zone is UNLOCKED. Else LS7 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
13-12	GRAB_RAM6	R	0h	Value in this field gets loaded from Z1_GRABRAM1[13:12] when a read is issued to address location of Z1_GRABRAM1 in OTP. 00 : Invalid. LS6 RAM is inaccessible. 01 : Request to allocate LS6 RAM to Zone1. 10 : No request for LS6 RAM 11 : No request for LS6 RAM when this zone is UNLOCKED. Else LS6 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
11-10	GRAB_RAM5	R	0h	Value in this field gets loaded from Z1_GRABRAM1[11:10] when a read is issued to address location of Z1_GRABRAM1 in OTP. 00 : Invalid. LS5 RAM is inaccessible. 01 : Request to allocate LS5 RAM to Zone1. 10 : No request for LS5 RAM 11 : No request for LS5 RAM when this zone is UNLOCKED. Else LS5 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
9-8	GRAB_RAM4	R	0h	Value in this field gets loaded from Z1_GRABRAM1[9:8] when a read is issued to address location of Z1_GRABRAM1 in OTP. 00 : Invalid. LS4 RAM is inaccessible. 01 : Request to allocate LS4 RAM to Zone1. 10 : No request for LS4 RAM 11 : No request for LS4 RAM when this zone is UNLOCKED. Else LS4 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 5-23. Z1_GRABRAM1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	GRAB_RAM3	R	0h	Value in this field gets loaded from Z1_GRABRAM1[7:6] when a read is issued to address location of Z1_GRABRAM1 in OTP. 00 : Invalid. LS3 RAM is inaccessible. 01 : Request to allocate LS3 RAM to Zone1. 10 : No request for LS3 RAM 11 : No request for LS3 RAM when this zone is UNLOCKED. Else LS3 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
5-4	GRAB_RAM2	R	0h	Value in this field gets loaded from Z1_GRABRAM1[5:4] when a read is issued to address location of Z1_GRABRAM1 in OTP. 00 : Invalid. LS2 RAM is inaccessible. 01 : Request to allocate LS2 RAM to Zone1. 10 : No request for LS2 RAM 11 : No request for LS2 RAM when this zone is UNLOCKED. Else LS2 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
3-2	GRAB_RAM1	R	0h	Value in this field gets loaded from Z1_GRABRAM1[3:2] when a read is issued to address location of Z1_GRABRAM1 in OTP. 00 : Invalid. LS1 RAM is inaccessible. 01 : Request to allocate LS1 RAM to Zone1. 10 : No request for LS1 RAM 11 : No request for LS1 RAM when this zone is UNLOCKED. Else LS1 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
1-0	GRAB_RAM0	R	0h	Value in this field gets loaded from Z1_GRABRAM1[1:0] when a read is issued to address location of Z1_GRABRAM1 in OTP. 00 : Invalid. LS0 RAM is inaccessible. 01 : Request to allocate LS0 RAM to Zone1. 10 : No request for LS0 RAM 11 : No request for LS0 RAM when this zone is UNLOCKED. Else LS0 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn

5.8.2.18 Z1_EXEONLYSECT1R Register (Offset = 26h) [Reset = 0h]

Z1_EXEONLYSECT1R is shown in Figure 5-22 and described in Table 5-24.

Return to the Summary Table.

Zone 1 Execute Only Flash Status Register 1

Figure 5-22. Z1_EXEONLYSECT1R Register

31	30	29	28	27	26	25	24
EXEONLY_BANK1_SECT15	EXEONLY_BANK1_SECT14	EXEONLY_BANK1_SECT13	EXEONLY_BANK1_SECT12	EXEONLY_BANK1_SECT11	EXEONLY_BANK1_SECT10	EXEONLY_BANK1_SECT9	EXEONLY_BANK1_SECT8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
EXEONLY_BANK1_SECT7	EXEONLY_BANK1_SECT6	EXEONLY_BANK1_SECT5	EXEONLY_BANK1_SECT4	EXEONLY_BANK1_SECT3	EXEONLY_BANK1_SECT2	EXEONLY_BANK1_SECT1	EXEONLY_BANK1_SECT0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
EXEONLY_BANK0_SECT15	EXEONLY_BANK0_SECT14	EXEONLY_BANK0_SECT13	EXEONLY_BANK0_SECT12	EXEONLY_BANK0_SECT11	EXEONLY_BANK0_SECT10	EXEONLY_BANK0_SECT9	EXEONLY_BANK0_SECT8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
EXEONLY_BANK0_SECT7	EXEONLY_BANK0_SECT6	EXEONLY_BANK0_SECT5	EXEONLY_BANK0_SECT4	EXEONLY_BANK0_SECT3	EXEONLY_BANK0_SECT2	EXEONLY_BANK0_SECT1	EXEONLY_BANK0_SECT0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 5-24. Z1_EXEONLYSECT1R Register Field Descriptions

Bit	Field	Type	Reset	Description
31	EXEONLY_BANK1_SECT15	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[31] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 15 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled forBank 1 Flash Sector 15 (only if it's allocated to Zone1) Reset type: SYSRSn
30	EXEONLY_BANK1_SECT14	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[30] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 14 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled forBank 1 Flash Sector 14 (only if it's allocated to Zone1) Reset type: SYSRSn
29	EXEONLY_BANK1_SECT13	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[29] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 13 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled forBank 1 Flash Sector 13 (only if it's allocated to Zone1) Reset type: SYSRSn
28	EXEONLY_BANK1_SECT12	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[28] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 12 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled forBank 1 Flash Sector 12 (only if it's allocated to Zone1) Reset type: SYSRSn

Table 5-24. Z1_EXEONLYSECT1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	EXEONLY_BANK1_SECT11	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[27] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 11 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled forBank 1 Flash Sector 11 (only if it's allocated to Zone1) Reset type: SYSRSn
26	EXEONLY_BANK1_SECT10	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[26] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 10 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled forBank 1 Flash Sector 10 (only if it's allocated to Zone1) Reset type: SYSRSn
25	EXEONLY_BANK1_SECT9	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[25] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 9 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled forBank 1 Flash Sector 9 (only if it's allocated to Zone1) Reset type: SYSRSn
24	EXEONLY_BANK1_SECT8	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[24] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 8 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled forBank 1 Flash Sector 8 (only if it's allocated to Zone1) Reset type: SYSRSn
23	EXEONLY_BANK1_SECT7	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[23] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 7 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled forBank 1 Flash Sector 7 (only if it's allocated to Zone1) Reset type: SYSRSn
22	EXEONLY_BANK1_SECT6	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[22] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 6 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled forBank 1 Flash Sector 6 (only if it's allocated to Zone1) Reset type: SYSRSn
21	EXEONLY_BANK1_SECT5	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[21] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 5 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled forBank 1 Flash Sector 5 (only if it's allocated to Zone1) Reset type: SYSRSn
20	EXEONLY_BANK1_SECT4	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[20] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 4 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled forBank 1 Flash Sector 4 (only if it's allocated to Zone1) Reset type: SYSRSn

Table 5-24. Z1_EXEONLYSECT1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	EXEONLY_BANK1_SECT3	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[19] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 3 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled forBank 1 Flash Sector 3 (only if it's allocated to Zone1) Reset type: SYSRSn
18	EXEONLY_BANK1_SECT2	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[18] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 2 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled forBank 1 Flash Sector 2 (only if it's allocated to Zone1) Reset type: SYSRSn
17	EXEONLY_BANK1_SECT1	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[17] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 1 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled forBank 1 Flash Sector 1 (only if it's allocated to Zone1) Reset type: SYSRSn
16	EXEONLY_BANK1_SECT0	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[16] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 0 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled forBank 1 Flash Sector 0 (only if it's allocated to Zone1) Reset type: SYSRSn
15	EXEONLY_BANK0_SECT15	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[15] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 15 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 15 (only if it's allocated to Zone1) Reset type: SYSRSn
14	EXEONLY_BANK0_SECT14	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[14] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 14 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 14 (only if it's allocated to Zone1) Reset type: SYSRSn
13	EXEONLY_BANK0_SECT13	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[13] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 13 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 13 (only if it's allocated to Zone1) Reset type: SYSRSn
12	EXEONLY_BANK0_SECT12	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[12] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 12 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 12 (only if it's allocated to Zone1) Reset type: SYSRSn

Table 5-24. Z1_EXEONLYSECT1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	EXEONLY_BANK0_SECT 11	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[11] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 11 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 11 (only if it's allocated to Zone1) Reset type: SYSRSn
10	EXEONLY_BANK0_SECT 10	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[10] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 10 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 10 (only if it's allocated to Zone1) Reset type: SYSRSn
9	EXEONLY_BANK0_SECT 9	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[9] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 9 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 9 (only if it's allocated to Zone1) Reset type: SYSRSn
8	EXEONLY_BANK0_SECT 8	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[8] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 8 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 8 (only if it's allocated to Zone1) Reset type: SYSRSn
7	EXEONLY_BANK0_SECT 7	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[7] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 7 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 7 (only if it's allocated to Zone1) Reset type: SYSRSn
6	EXEONLY_BANK0_SECT 6	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[6] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 6 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 6 (only if it's allocated to Zone1) Reset type: SYSRSn
5	EXEONLY_BANK0_SECT 5	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[5] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 5 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 5 (only if it's allocated to Zone1) Reset type: SYSRSn
4	EXEONLY_BANK0_SECT 4	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[4] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 4 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 4 (only if it's allocated to Zone1) Reset type: SYSRSn

Table 5-24. Z1_EXEONLYSECT1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	EXEONLY_BANK0_SECT3	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[3] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 3 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 3 (only if it's allocated to Zone1) Reset type: SYSRSn
2	EXEONLY_BANK0_SECT2	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[2] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 2 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 2 (only if it's allocated to Zone1) Reset type: SYSRSn
1	EXEONLY_BANK0_SECT1	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[1] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 1 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 1 (only if it's allocated to Zone1) Reset type: SYSRSn
0	EXEONLY_BANK0_SECT0	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT1[0] when a read is issued to Z1_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 0 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 0 (only if it's allocated to Zone1) Reset type: SYSRSn

5.8.2.19 Z1_EXEONLYSECT2R Register (Offset = 28h) [Reset = 0h]

Z1_EXEONLYSECT2R is shown in Figure 5-23 and described in Table 5-25.

Return to the Summary Table.

Zone 1 Execute Only Flash Status Register 2

Figure 5-23. Z1_EXEONLYSECT2R Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
EXEONLY_BANK2_SECT15	EXEONLY_BANK2_SECT14	EXEONLY_BANK2_SECT13	EXEONLY_BANK2_SECT12	EXEONLY_BANK2_SECT11	EXEONLY_BANK2_SECT10	EXEONLY_BANK2_SECT9	EXEONLY_BANK2_SECT8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
EXEONLY_BANK2_SECT7	EXEONLY_BANK2_SECT6	EXEONLY_BANK2_SECT5	EXEONLY_BANK2_SECT4	EXEONLY_BANK2_SECT3	EXEONLY_BANK2_SECT2	EXEONLY_BANK2_SECT1	EXEONLY_BANK2_SECT0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 5-25. Z1_EXEONLYSECT2R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	EXEONLY_BANK2_SECT15	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[15] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for BANK 2 Flash Sector 15 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for BANK 2 Flash Sector 15 (only if it's allocated to Zone1) Reset type: SYSRSn
14	EXEONLY_BANK2_SECT14	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[14] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for BANK 2 Flash Sector 14 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for BANK 2 Flash Sector 14 (only if it's allocated to Zone1) Reset type: SYSRSn
13	EXEONLY_BANK2_SECT13	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[13] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for BANK 2 Flash Sector 13 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for BANK 2 Flash Sector 13 (only if it's allocated to Zone1) Reset type: SYSRSn
12	EXEONLY_BANK2_SECT12	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[12] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for BANK 2 Flash Sector 12 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for BANK 2 Flash Sector 12 (only if it's allocated to Zone1) Reset type: SYSRSn

Table 5-25. Z1_EXEONLYSECT2R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	EXEONLY_BANK2_SECT11	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[11] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for BANK 2 Flash Sector 11 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for BANK 2 Flash Sector 11 (only if it's allocated to Zone1) Reset type: SYSRSn
10	EXEONLY_BANK2_SECT10	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[10] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for BANK 2 Flash Sector 10 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for BANK 2 Flash Sector 10 (only if it's allocated to Zone1) Reset type: SYSRSn
9	EXEONLY_BANK2_SECT9	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[9] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for BANK 2 Flash Sector 9 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for BANK 2 Flash Sector 9 (only if it's allocated to Zone1) Reset type: SYSRSn
8	EXEONLY_BANK2_SECT8	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[8] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for BANK 2 Flash Sector 8 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for BANK 2 Flash Sector 8 (only if it's allocated to Zone1) Reset type: SYSRSn
7	EXEONLY_BANK2_SECT7	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[7] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for BANK 2 Flash Sector 7 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for BANK 2 Flash Sector 7 (only if it's allocated to Zone1) Reset type: SYSRSn
6	EXEONLY_BANK2_SECT6	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[6] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for BANK 2 Flash Sector 6 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for BANK 2 Flash Sector 6 (only if it's allocated to Zone1) Reset type: SYSRSn
5	EXEONLY_BANK2_SECT5	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[5] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for BANK 2 Flash Sector 5 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for BANK 2 Flash Sector 5 (only if it's allocated to Zone1) Reset type: SYSRSn
4	EXEONLY_BANK2_SECT4	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[4] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for BANK 2 Flash Sector 4 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for BANK 2 Flash Sector 4 (only if it's allocated to Zone1) Reset type: SYSRSn

Table 5-25. Z1_EXEONLYSECT2R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	EXEONLY_BANK2_SECT3	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[3] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for BANK 2 Flash Sector 3 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for BANK 2 Flash Sector 3 (only if it's allocated to Zone1) Reset type: SYSRSn
2	EXEONLY_BANK2_SECT2	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[2] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for BANK 2 Flash Sector 2 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for BANK 2 Flash Sector 2 (only if it's allocated to Zone1) Reset type: SYSRSn
1	EXEONLY_BANK2_SECT1	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[1] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for BANK 2 Flash Sector 1 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for BANK 2 Flash Sector 1 (only if it's allocated to Zone1) Reset type: SYSRSn
0	EXEONLY_BANK2_SECT0	R	0h	Value in this field gets loaded from Z1_EXEONLYSECT2[0] when a read is issued to Z1_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for BANK 2 Flash Sector 0 (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for BANK 2 Flash Sector 0 (only if it's allocated to Zone1) Reset type: SYSRSn

5.8.2.20 Z1_EXEONLYRAM1R Register (Offset = 2Ah) [Reset = 0h]

Z1_EXEONLYRAM1R is shown in Figure 5-24 and described in Table 5-26.

Return to the Summary Table.

Zone 1 Execute Only RAM Status Register 1

Figure 5-24. Z1_EXEONLYRAM1R Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
EXEONLY_RAM7	EXEONLY_RAM6	EXEONLY_RAM5	EXEONLY_RAM4	EXEONLY_RAM3	EXEONLY_RAM2	EXEONLY_RAM1	EXEONLY_RAM0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 5-26. Z1_EXEONLYRAM1R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	EXEONLY_RAM7	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[7] when a read is issued to Z1_EXEONLYRAM address location in OTP. 0 : Execute-Only protection is enabled for LS7 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for LS7 RAM (only if it's allocated to Zone1) Reset type: SYSRSn
6	EXEONLY_RAM6	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[6] when a read is issued to Z1_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for LS6 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for LS6 RAM (only if it's allocated to Zone1) Reset type: SYSRSn
5	EXEONLY_RAM5	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[5] when a read is issued to Z1_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for LS5 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for LS5 RAM (only if it's allocated to Zone1) Reset type: SYSRSn
4	EXEONLY_RAM4	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[4] when a read is issued to Z1_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for LS4 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for LS4 RAM (only if it's allocated to Zone1) Reset type: SYSRSn

Table 5-26. Z1_EXEONLYRAM1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	EXEONLY_RAM3	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[3] when a read is issued to Z1_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for LS3 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for LS3 RAM (only if it's allocated to Zone1) Reset type: SYSRSn
2	EXEONLY_RAM2	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[2] when a read is issued to Z1_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for LS2 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for LS2 RAM (only if it's allocated to Zone1) Reset type: SYSRSn
1	EXEONLY_RAM1	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[1] when a read is issued to Z1_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for LS1 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for LS1 RAM (only if it's allocated to Zone1) Reset type: SYSRSn
0	EXEONLY_RAM0	R	0h	Value in this field gets loaded from Z1_EXEONLYRAM1[0] when a read is issued to Z1_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for LS0 RAM (only if it's allocated to Zone1) 1 : Execute-Only protection is disabled for LS0 RAM (only if it's allocated to Zone1) Reset type: SYSRSn

5.8.2.21 Z1_JTAGKEY0 Register (Offset = 2Eh) [Reset = 0h]

Z1_JTAGKEY0 is shown in Figure 5-25 and described in Table 5-27.

Return to the Summary Table.

JTAG Unlock Key Register 0

Figure 5-25. Z1_JTAGKEY0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY0																															
R-0h																															

Table 5-27. Z1_JTAGKEY0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY0	R	0h	Value in this field is scanned in via the JLM scan chain. JTAGKEY is compared to a hidden register that gets dummy loaded when a read is issued to the JTAGPSWD locations in OTP. Reset type: PORESETn

5.8.2.22 Z1_JTAGKEY1 Register (Offset = 30h) [Reset = 0h]

Z1_JTAGKEY1 is shown in Figure 5-26 and described in Table 5-28.

Return to the Summary Table.

JTAG Unlock Key Register 1

Figure 5-26. Z1_JTAGKEY1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY1																															
R-0h																															

Table 5-28. Z1_JTAGKEY1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY1	R	0h	Value in this field is scanned in via the JLM scan chain. JTAGKEY is compared to a hidden register that gets dummy loaded when a read is issued to the JTAGPSWD locations in OTP. Reset type: PORESETn

5.8.2.23 Z1_JTAGKEY2 Register (Offset = 32h) [Reset = 0h]

Z1_JTAGKEY2 is shown in Figure 5-27 and described in Table 5-29.

Return to the Summary Table.

JTAG Unlock Key Register 2

Figure 5-27. Z1_JTAGKEY2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY2																															
R-0h																															

Table 5-29. Z1_JTAGKEY2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY2	R	0h	Value in this field is scanned in via the JLM scan chain. JTAGKEY is compared to a hidden register that gets dummy loaded when a read is issued to the JTAGPSWD locations in OTP. Reset type: PORESETn

5.8.2.24 Z1_JTAGKEY3 Register (Offset = 34h) [Reset = 0h]

Z1_JTAGKEY3 is shown in Figure 5-28 and described in Table 5-30.

Return to the Summary Table.

JTAG Unlock Key Register 3

Figure 5-28. Z1_JTAGKEY3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY3																															
R-0h																															

Table 5-30. Z1_JTAGKEY3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY3	R	0h	Value in this field is scanned in via the JLM scan chain. JTAGKEY is compared to a hidden register that gets dummy loaded when a read is issued to the JTAGPSWD locations in OTP. Reset type: PORESETn

5.8.2.25 Z1_CMACKEY0 Register (Offset = 36h) [Reset = 0h]

Z1_CMACKEY0 is shown in Figure 5-29 and described in Table 5-31.

Return to the Summary Table.

Secure Boot CMAC Key Status Register 0

Figure 5-29. Z1_CMACKEY0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY0																															
R-0h																															

Table 5-31. Z1_CMACKEY0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY0	R	0h	Value in this field gets loaded from CMACKEY0 when a read is issued to its address in OTP. Reset type: SYSRSn

5.8.2.26 Z1_CMACKEY1 Register (Offset = 38h) [Reset = 0h]

Z1_CMACKEY1 is shown in Figure 5-30 and described in Table 5-32.

Return to the Summary Table.

Secure Boot CMAC Key Status Register 1

Figure 5-30. Z1_CMACKEY1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY1																															
R-0h																															

Table 5-32. Z1_CMACKEY1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY1	R	0h	Value in this field gets loaded from CMACKEY1 when a read is issued to its address in OTP. Reset type: SYSRSn

5.8.2.27 Z1_CMACKEY2 Register (Offset = 3Ah) [Reset = 0h]

Z1_CMACKEY2 is shown in Figure 5-31 and described in Table 5-33.

Return to the Summary Table.

Secure Boot CMAC Key Status Register 2

Figure 5-31. Z1_CMACKEY2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY2																															
R-0h																															

Table 5-33. Z1_CMACKEY2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY2	R	0h	Value in this field gets loaded from CMACKEY2 when a read is issued to its address in OTP. Reset type: SYSRSn

5.8.2.28 Z1_CMACKEY3 Register (Offset = 3Ch) [Reset = 0h]

Z1_CMACKEY3 is shown in Figure 5-32 and described in Table 5-34.

Return to the Summary Table.

Secure Boot CMAC Key Status Register 3

Figure 5-32. Z1_CMACKEY3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY3																															
R-0h																															

Table 5-34. Z1_CMACKEY3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY3	R	0h	Value in this field gets loaded from CMACKEY3 when a read is issued to its address in OTP. Reset type: SYSRSn

5.8.3 DCSM_Z2_REGS Registers

Table 5-35 lists the memory-mapped registers for the DCSM_Z2_REGS registers. All register offset addresses not listed in Table 5-35 should be considered as reserved locations and the register contents should not be modified.

Table 5-35. DCSM_Z2_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	Z2_LINKPOINTER	Zone 2 Link Pointer		Go
2h	Z2_OTPSECLOCK	Zone 2 OTP Secure Lock		Go
6h	Z2_LINKPOINTERERR	Link Pointer Error		Go
8h	Z2_GPREG1	Zone 2 General Purpose Register-1		Go
Ah	Z2_GPREG2	Zone 2 General Purpose Register-2		Go
Ch	Z2_GPREG3	Zone 2 General Purpose Register-3		Go
Eh	Z2_GPREG4	Zone 2 General Purpose Register-4		Go
10h	Z2_CSMKEY0	Zone 2 CSM Key 0		Go
12h	Z2_CSMKEY1	Zone 2 CSM Key 1		Go
14h	Z2_CSMKEY2	Zone 2 CSM Key 2		Go
16h	Z2_CSMKEY3	Zone 2 CSM Key 3		Go
18h	Z2_CR	Zone 2 CSM Control Register		Go
1Ah	Z2_GRABSECT1R	Zone 2 Grab Flash Status Register 1		Go
1Ch	Z2_GRABSECT2R	Zone 2 Grab Flash Status Register 2		Go
1Eh	Z2_GRABSECT3R	Zone 2 Grab Flash Status Register 3		Go
20h	Z2_GRABRAM1R	Zone 2 Grab RAM Status Register 1		Go
26h	Z2_EXEONLYSECT1R	Zone 2 Execute Only Flash Status Register 1		Go
28h	Z2_EXEONLYSECT2R	Zone 2 Execute Only Flash Status Register 2		Go
2Ah	Z2_EXEONLYRAM1R	Zone 2 Execute Only RAM Status Register 1		Go

Complex bit access types are encoded to fit into small table cells. Table 5-36 shows the codes that are used for access types in this section.

Table 5-36. DCSM_Z2_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.

Table 5-36. DCSM_Z2_REGS Access Type Codes (continued)

Access Type	Code	Description
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

5.8.3.1 Z2_LINKPOINTER Register (Offset = 0h) [Reset = FFFFC000h]

Z2_LINKPOINTER is shown in Figure 5-33 and described in Table 5-37.

Return to the Summary Table.

Zone 2 Link Pointer

Figure 5-33. Z2_LINKPOINTER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																		LINKPOINTER													
R-0h																		R-0h													

Table 5-37. Z2_LINKPOINTER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Reserved
13-0	LINKPOINTER	R	0h	This is resolved Link-Pointer value which is generated by looking at the three physical Link-Pointer values loaded from OTP Reset type: SYSRSn

5.8.3.2 Z2_OTPSECLOCK Register (Offset = 2h) [Reset = 1h]

Z2_OTPSECLOCK is shown in Figure 5-34 and described in Table 5-38.

Return to the Summary Table.

Zone 2 OTP Secure Lock

Figure 5-34. Z2_OTPSECLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				CRCLOCK			
R-0h				R-0h			
7	6	5	4	3	2	1	0
PSWDLOCK				RESERVED			JTAGLOCK
R-0h				R-0h			R-1h

Table 5-38. Z2_OTPSECLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-12	RESERVED	R	0h	Reserved
11-8	CRCLOCK	R	0h	Value in this field gets loaded from Z2_CRCLOCK[3:0] when a read is issued to address location of Z2_CRCLOCK in OTP. 1111 : VCU has ability to calculate CRC on secure memories. Other Value : VCU doesn't have ability to calculate CRC on secure memories. Reset type: XRSn
7-4	PSWDLOCK	R	0h	Value in this field gets loaded from Z2_PSWDLOCK[3:0] when a read is issued to address location of Z1_PSWDLOCK in OTP. 1111 : CSM password locations in OTP are not protected and can be read from debugger as well as code running from anywhere. Other Value : CSM password locations in OTP are protected and can't be read without unlocking CSM of that zone. Reset type: XRSn
3-1	RESERVED	R	0h	Reserved
0	JTAGLOCK	R	1h	Reflects the state of the JTAGLOCK feature. 0 : JTAG is not locked 1 : JTAG is locked This bit is a copy of the Z1_OTPSECLOCK.JTAGLOCK bit. Reset type: PORESETn

5.8.3.3 Z2_LINKPOINTERERR Register (Offset = 6h) [Reset = 0h]

Z2_LINKPOINTERERR is shown in Figure 5-35 and described in Table 5-39.

Return to the Summary Table.

Link Pointer Error

Figure 5-35. Z2_LINKPOINTERERR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED		Z2_LINKPOINTERERR													
R-0h		R-0h													

Table 5-39. Z2_LINKPOINTERERR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-14	RESERVED	R	0h	Reserved
13-0	Z2_LINKPOINTERERR	R	0h	These bits indicate errors during formation of the resolved Link-Pointer value after the three physical Link-Pointer values loaded of OTP in flash. 0 : No Error. Other : Error on bit positions which is set to 1. Reset type: SYSRSn

5.8.3.4 Z2_GPREG1 Register (Offset = 8h) [Reset = 0h]

Z2_GPREG1 is shown in Figure 5-36 and described in Table 5-40.

Return to the Summary Table.

Zone 2 General Purpose Register-1

Figure 5-36. Z2_GPREG1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPREG1																															
R-0h																															

Table 5-40. Z2_GPREG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPREG1	R	0h	This field gets loaded with the contents of Z2OTP_GPREG1 locations in Zone-2-USER-OTP when a dummy read is issued to that address. Users can use this register to load any general purpose non-voliate content from USER-OTP of Zone-2 Reset type: SYSRSn

5.8.3.5 Z2_GPREG2 Register (Offset = Ah) [Reset = 0h]

Z2_GPREG2 is shown in Figure 5-37 and described in Table 5-41.

Return to the Summary Table.

Zone 2 General Purpose Register-2

Figure 5-37. Z2_GPREG2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPREG2																															
R-0h																															

Table 5-41. Z2_GPREG2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPREG2	R	0h	This field gets loaded with the contents of Z2OTP_GPREG2 locations in Zone-2-USER-OTP when a dummy read is issued to that address. Users can use this register to load any general purpose non-voliate content from USER-OTP of Zone-2 Reset type: SYSRSn

5.8.3.6 Z2_GPREG3 Register (Offset = Ch) [Reset = 0h]

Z2_GPREG3 is shown in Figure 5-38 and described in Table 5-42.

Return to the Summary Table.

Zone 2 General Purpose Register-3

Figure 5-38. Z2_GPREG3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPREG3																															
R-0h																															

Table 5-42. Z2_GPREG3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPREG3	R	0h	This field gets loaded with the contents of Z2OTP_GPREG3 locations in Zone-2-USER-OTP when a dummy read is issued to that address. Users can use this register to load any general purpose non-voliate content from USER-OTP of Zone-2 Reset type: SYSRSn

5.8.3.7 Z2_GPREG4 Register (Offset = Eh) [Reset = 0h]

Z2_GPREG4 is shown in Figure 5-39 and described in Table 5-43.

Return to the Summary Table.

Zone 2 General Purpose Register-4

Figure 5-39. Z2_GPREG4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPREG4																															
R-0h																															

Table 5-43. Z2_GPREG4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	GPREG4	R	0h	This field gets loaded with the contents of Z2OTP_GPREG4 locations in Zone-2-USER-OTP when a dummy read is issued to that address. Users can use this register to load any general purpose non-voliate content from USER-OTP of Zone-2 Reset type: SYSRSn

5.8.3.8 Z2_CSMKEY0 Register (Offset = 10h) [Reset = 0h]

Z2_CSMKEY0 is shown in Figure 5-40 and described in Table 5-44.

Return to the Summary Table.

Zone 2 CSM Key 0

Figure 5-40. Z2_CSMKEY0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2_CSMKEY0																															
R/W-0h																															

Table 5-44. Z2_CSMKEY0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2_CSMKEY0	R/W	0h	To unlock Zone2, user needs to write this regsiter with exact value as Z2_CSMPSWD0, programmed in OTP (zone gets unlock only if 128 bit password in OTP match with value written in four CSMKEY registers.) Reset type: SYSRSn

5.8.3.9 Z2_CSMKEY1 Register (Offset = 12h) [Reset = 0h]

Z2_CSMKEY1 is shown in Figure 5-41 and described in Table 5-45.

Return to the Summary Table.

Zone 2 CSM Key 1

Figure 5-41. Z2_CSMKEY1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2_CSMKEY1																															
R/W-0h																															

Table 5-45. Z2_CSMKEY1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2_CSMKEY1	R/W	0h	To unlock Zone2, user needs to write this regsiter with exact value as Z2_CSMPSWD1, programmed in OTP (zone gets unlock only if 128 bit password in OTP match with value written in four CSMKEY registers.) Reset type: SYSRSn

5.8.3.10 Z2_CSMKEY2 Register (Offset = 14h) [Reset = 0h]

Z2_CSMKEY2 is shown in Figure 5-42 and described in Table 5-46.

Return to the Summary Table.

Zone 2 CSM Key 2

Figure 5-42. Z2_CSMKEY2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2_CSMKEY2																															
R/W-0h																															

Table 5-46. Z2_CSMKEY2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2_CSMKEY2	R/W	0h	To unlock Zone2, user needs to write this regsiter with exact value as Z2_CSMPSWD2, programmed in OTP (zone gets unlock only if 128 bit password in OTP match with value written in four CSMKEY registers.) Reset type: SYSRSn

5.8.3.11 Z2_CSMKEY3 Register (Offset = 16h) [Reset = 0h]

Z2_CSMKEY3 is shown in Figure 5-43 and described in Table 5-47.

Return to the Summary Table.

Zone 2 CSM Key 3

Figure 5-43. Z2_CSMKEY3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2_CSMKEY3																															
R/W-0h																															

Table 5-47. Z2_CSMKEY3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2_CSMKEY3	R/W	0h	To unlock Zone2, user needs to write this regsiter with exact value as Z2_CSMPSWD3, programmed in OTP (zone gets unlock only if 128 bit password in OTP match with value written in four CSMKEY registers.) Reset type: SYSRSn

5.8.3.12 Z2_CR Register (Offset = 18h) [Reset = 00080000h]

Z2_CR is shown in Figure 5-44 and described in Table 5-48.

Return to the Summary Table.

Zone 2 CSM Control Register

Figure 5-44. Z2_CR Register

31	30	29	28	27	26	25	24
FORCESEC	RESERVED						
R-0/W-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED	ARMED	UNSECURE	ALLONE	ALLZERO	RESERVED		
R-0h	R-0h	R-0h	R-0h	R-1h	R-0h		
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	RESERVED	RESERVED	RESERVED
R-0h				R-0h	R-0h	R-0h	R-0h

Table 5-48. Z2_CR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	FORCESEC	R-0/W	0h	A write of '1' to this bit relocks the zone by clearing the zone's CSMKEYx registers. If the zone's CSMPSWDx OTP locations were updated prior to using this bit, it is suggested that all of the new password registers be dummy loaded from OTP immediately to avoid a mix between old and new password data. Reset type: SYSRSn
30-24	RESERVED	R	0h	Reserved
23	RESERVED	R	0h	Reserved
22	ARMED	R	0h	0 : Dummy read to CSM Password locations in OTP hasn't been performed. 1 : Dummy read to CSM Password locations in OTP has been performed. Reset type: SYSRSn
21	UNSECURE	R	0h	Indiacates the state of Zone. 0 : Zone is in lock(secure) state. 1 : Zone is in unlock(unsecure) state. Reset type: SYSRSn
20	ALLONE	R	0h	Indicates the state of CSM passwords. 0 : Zone CSM Passwords are not all ones. 1 : Zone CSM Passwords are all ones. Reset type: SYSRSn
19	ALLZERO	R	1h	Indicates the state of CSM passowrds. 0 : CSM Passwords are not all zeros. 1 : CSM Passwords are all zero and device is permanently locked. Reset type: SYSRSn
18-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3	RESERVED	R	0h	Reserved
2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved

Table 5-48. Z2_CR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	RESERVED	R	0h	Reserved

5.8.3.13 Z2_GRABSECT1R Register (Offset = 1Ah) [Reset = 0h]

Z2_GRABSECT1R is shown in Figure 5-45 and described in Table 5-49.

Return to the Summary Table.

Zone 2 Grab Flash Status Register 1

Figure 5-45. Z2_GRABSECT1R Register

31	30	29	28	27	26	25	24
GRAB_SECT15		GRAB_SECT14		GRAB_SECT13		GRAB_SECT12	
R-0h		R-0h		R-0h		R-0h	
23	22	21	20	19	18	17	16
GRAB_SECT11		GRAB_SECT10		GRAB_SECT9		GRAB_SECT8	
R-0h		R-0h		R-0h		R-0h	
15	14	13	12	11	10	9	8
GRAB_SECT7		GRAB_SECT6		GRAB_SECT5		GRAB_SECT4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
GRAB_SECT3		GRAB_SECT2		GRAB_SECT1		GRAB_SECT0	
R-0h		R-0h		R-0h		R-0h	

Table 5-49. Z2_GRABSECT1R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GRAB_SECT15	R	0h	Value in this field gets loaded from Z2_GRABSECT1[31:30] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 15 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 15 to Zone2. 10 : No request for Bank 0 Flash Sector 15 11 : No request for Bank 0 Flash Sector 15 when this zone is UNLOCKED. Else Bank 0 Flash Sector 15 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
29-28	GRAB_SECT14	R	0h	Value in this field gets loaded from Z2_GRABSECT1[29:28] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 14 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 14 to Zone2. 10 : No request for Bank 0 Flash Sector 14 11 : No request for Bank 0 Flash Sector 14 when this zone is UNLOCKED. Else Bank 0 Flash Sector 14 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
27-26	GRAB_SECT13	R	0h	Value in this field gets loaded from Z2_GRABSECT1[27:26] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 13 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 13 to Zone2. 10 : No request for Bank 0 Flash Sector 13 11 : No request for Bank 0 Flash Sector 13 when this zone is UNLOCKED. Else Bank 0 Flash Sector 13 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 5-49. Z2_GRABSECT1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-24	GRAB_SECT12	R	0h	Value in this field gets loaded from Z2_GRABSECT1[25:24] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 12 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 12 to Zone2. 10 : No request for Bank 0 Flash Sector 12 11 : No request for Bank 0 Flash Sector 12 when this zone is UNLOCKED. Else Bank 0 Flash Sector 12 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
23-22	GRAB_SECT11	R	0h	Value in this field gets loaded from Z2_GRABSECT1[23:22] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 11 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 11 to Zone2. 10 : No request for Bank 0 Flash Sector 11 11 : No request for Bank 0 Flash Sector 11 when this zone is UNLOCKED. Else Bank 0 Flash Sector 11 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
21-20	GRAB_SECT10	R	0h	Value in this field gets loaded from Z2_GRABSECT1[21:20] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 10 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 10 to Zone2. 10 : No request for Bank 0 Flash Sector 10 11 : No request for Bank 0 Flash Sector 10 when this zone is UNLOCKED. Else Bank 0 Flash Sector 10 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
19-18	GRAB_SECT9	R	0h	Value in this field gets loaded from Z2_GRABSECT1[19:18] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 9 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 9 to Zone2. 10 : No request for Bank 0 Flash Sector 9 11 : No request for Bank 0 Flash Sector 9 when this zone is UNLOCKED. Else Bank 0 Flash Sector 9 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
17-16	GRAB_SECT8	R	0h	Value in this field gets loaded from Z2_GRABSECT1[17:16] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 8 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 8 to Zone2. 10 : No request for Bank 0 Flash Sector 8 11 : No request for Bank 0 Flash Sector 8 when this zone is UNLOCKED. Else Bank 0 Flash Sector 8 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
15-14	GRAB_SECT7	R	0h	Value in this field gets loaded from Z2_GRABSECT1[15:14] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 7 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 7 to Zone2. 10 : No request for Bank 0 Flash Sector 7 11 : No request for Bank 0 Flash Sector 7 when this zone is UNLOCKED. Else Bank 0 Flash Sector 7 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
13-12	GRAB_SECT6	R	0h	Value in this field gets loaded from Z2_GRABSECT1[13:12] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 6 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 6 to Zone2. 10 : No request for Bank 0 Flash Sector 6 11 : No request for Bank 0 Flash Sector 6 when this zone is UNLOCKED. Else Bank 0 Flash Sector 6 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 5-49. Z2_GRABSECT1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11-10	GRAB_SECT5	R	0h	Value in this field gets loaded from Z2_GRABSECT1[11:10] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 5 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 5 to Zone2. 10 : No request for Bank 0 Flash Sector 5 11 : No request for Bank 0 Flash Sector 5 when this zone is UNLOCKED. Else Bank 0 Flash Sector 5 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
9-8	GRAB_SECT4	R	0h	Value in this field gets loaded from Z2_GRABSECT1[9:8] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 4 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 4 to Zone2. 10 : No request for Bank 0 Flash Sector 4 11 : No request for Bank 0 Flash Sector 4 when this zone is UNLOCKED. Else Bank 0 Flash Sector 4 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
7-6	GRAB_SECT3	R	0h	Value in this field gets loaded from Z2_GRABSECT1[7:6] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 3 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 3 to Zone2. 10 : No request for Bank 0 Flash Sector 3 11 : No request for Bank 0 Flash Sector 3 when this zone is UNLOCKED. Else Bank 0 Flash Sector 3 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
5-4	GRAB_SECT2	R	0h	Value in this field gets loaded from Z2_GRABSECT1[5:4] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 2 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 2 to Zone2. 10 : No request for Bank 0 Flash Sector 2 11 : No request for Bank 0 Flash Sector 2 when this zone is UNLOCKED. Else Bank 0 Flash Sector 2 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
3-2	GRAB_SECT1	R	0h	Value in this field gets loaded from Z2_GRABSECT1[3:2] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 1 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 1 to Zone2. 10 : No request for Bank 0 Flash Sector 1 11 : No request for Bank 0 Flash Sector 1 when this zone is UNLOCKED. Else Bank 0 Flash Sector 1 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
1-0	GRAB_SECT0	R	0h	Value in this field gets loaded from Z2_GRABSECT1[1:0] when a read is issued to address location of Z2_GRABSECT1 in OTP. 00 : Invalid. Bank 0 Flash Sector 0 is inaccessible. 01 : Request to allocate Bank 0 Flash Sector 0 to Zone2. 10 : No request for Bank 0 Flash Sector 0 11 : No request for Bank 0 Flash Sector 0 when this zone is UNLOCKED. Else Bank 0 Flash Sector 0 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

5.8.3.14 Z2_GRABSECT2R Register (Offset = 1Ch) [Reset = 0h]

Z2_GRABSECT2R is shown in Figure 5-46 and described in Table 5-50.

Return to the Summary Table.

Zone 2 Grab Flash Status Register 2

Figure 5-46. Z2_GRABSECT2R Register

31	30	29	28	27	26	25	24
GRAB_SECT15		GRAB_SECT14		GRAB_SECT13		GRAB_SECT12	
R-0h		R-0h		R-0h		R-0h	
23	22	21	20	19	18	17	16
GRAB_SECT11		GRAB_SECT10		GRAB_SECT9		GRAB_SECT8	
R-0h		R-0h		R-0h		R-0h	
15	14	13	12	11	10	9	8
GRAB_SECT7		GRAB_SECT6		GRAB_SECT5		GRAB_SECT4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
GRAB_SECT3		GRAB_SECT2		GRAB_SECT1		GRAB_SECT0	
R-0h		R-0h		R-0h		R-0h	

Table 5-50. Z2_GRABSECT2R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GRAB_SECT15	R	0h	Value in this field gets loaded from Z2_GRABSECT2[31:30] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 15 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 15 to Zone2. 10 : No request for Bank 1 Flash Sector 15 11 : No request for Bank 1 Flash Sector 15 when this zone is UNLOCKED. Else Bank 1 Flash Sector 15 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
29-28	GRAB_SECT14	R	0h	Value in this field gets loaded from Z2_GRABSECT2[29:28] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 14 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 14 to Zone2. 10 : No request for Bank 1 Flash Sector 14 11 : No request for Bank 1 Flash Sector 14 when this zone is UNLOCKED. Else Bank 1 Flash Sector 14 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
27-26	GRAB_SECT13	R	0h	Value in this field gets loaded from Z2_GRABSECT2[27:26] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 13 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 13 to Zone2. 10 : No request for Bank 1 Flash Sector 13 11 : No request for Bank 1 Flash Sector 13 when this zone is UNLOCKED. Else Bank 1 Flash Sector 13 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 5-50. Z2_GRABSECT2R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-24	GRAB_SECT12	R	0h	Value in this field gets loaded from Z2_GRABSECT2[25:24] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 12 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 12 to Zone2. 10 : No request for Bank 1 Flash Sector 12 11 : No request for Bank 1 Flash Sector 12 when this zone is UNLOCKED. Else Bank 1 Flash Sector 12 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
23-22	GRAB_SECT11	R	0h	Value in this field gets loaded from Z2_GRABSECT2[23:22] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 11 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 11 to Zone2. 10 : No request for Bank 1 Flash Sector 11 11 : No request for Bank 1 Flash Sector 11 when this zone is UNLOCKED. Else Bank 1 Flash Sector 11 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
21-20	GRAB_SECT10	R	0h	Value in this field gets loaded from Z2_GRABSECT2[21:20] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 10 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 10 to Zone2. 10 : No request for Bank 1 Flash Sector 10 11 : No request for Bank 1 Flash Sector 10 when this zone is UNLOCKED. Else Bank 1 Flash Sector 10 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
19-18	GRAB_SECT9	R	0h	Value in this field gets loaded from Z2_GRABSECT2[19:18] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 9 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 9 to Zone2. 10 : No request for Bank 1 Flash Sector 9 11 : No request for Bank 1 Flash Sector 9 when this zone is UNLOCKED. Else Bank 1 Flash Sector 9 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
17-16	GRAB_SECT8	R	0h	Value in this field gets loaded from Z2_GRABSECT2[17:16] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 8 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 8 to Zone2. 10 : No request for Bank 1 Flash Sector 8 11 : No request for Bank 1 Flash Sector 8 when this zone is UNLOCKED. Else Bank 1 Flash Sector 8 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
15-14	GRAB_SECT7	R	0h	Value in this field gets loaded from Z2_GRABSECT2[15:14] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 7 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 7 to Zone2. 10 : No request for Bank 1 Flash Sector 7 11 : No request for Bank 1 Flash Sector 7 when this zone is UNLOCKED. Else Bank 1 Flash Sector 7 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
13-12	GRAB_SECT6	R	0h	Value in this field gets loaded from Z2_GRABSECT2[13:12] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 6 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 6 to Zone2. 10 : No request for Bank 1 Flash Sector 6 11 : No request for Bank 1 Flash Sector 6 when this zone is UNLOCKED. Else Bank 1 Flash Sector 6 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 5-50. Z2_GRABSECT2R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11-10	GRAB_SECT5	R	0h	Value in this field gets loaded from Z2_GRABSECT2[11:10] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 5 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 5 to Zone2. 10 : No request for Bank 1 Flash Sector 5 11 : No request for Bank 1 Flash Sector 5 when this zone is UNLOCKED. Else Bank 1 Flash Sector 5 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
9-8	GRAB_SECT4	R	0h	Value in this field gets loaded from Z2_GRABSECT2[9:8] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 4 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 4 to Zone2. 10 : No request for Bank 1 Flash Sector 4 11 : No request for Bank 1 Flash Sector 4 when this zone is UNLOCKED. Else Bank 1 Flash Sector 4 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
7-6	GRAB_SECT3	R	0h	Value in this field gets loaded from Z2_GRABSECT2[7:6] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 3 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 3 to Zone2. 10 : No request for Bank 1 Flash Sector 3 11 : No request for Bank 1 Flash Sector 3 when this zone is UNLOCKED. Else Bank 1 Flash Sector 3 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
5-4	GRAB_SECT2	R	0h	Value in this field gets loaded from Z2_GRABSECT2[5:4] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 2 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 2 to Zone2. 10 : No request for Bank 1 Flash Sector 2 11 : No request for Bank 1 Flash Sector 2 when this zone is UNLOCKED. Else Bank 1 Flash Sector 2 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
3-2	GRAB_SECT1	R	0h	Value in this field gets loaded from Z2_GRABSECT2[3:2] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 1 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 1 to Zone2. 10 : No request for Bank 1 Flash Sector 1 11 : No request for Bank 1 Flash Sector 1 when this zone is UNLOCKED. Else Bank 1 Flash Sector 1 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
1-0	GRAB_SECT0	R	0h	Value in this field gets loaded from Z2_GRABSECT2[1:0] when a read is issued to address location of Z2_GRABSECT2 in OTP. 00 : Invalid. Bank 1 Flash Sector 0 is inaccessible. 01 : Request to allocate Bank 1 Flash Sector 0 to Zone2. 10 : No request for Bank 1 Flash Sector 0 11 : No request for Bank 1 Flash Sector 0 when this zone is UNLOCKED. Else Bank 1 Flash Sector 0 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

5.8.3.15 Z2_GRABSECT3R Register (Offset = 1Eh) [Reset = 0h]

Z2_GRABSECT3R is shown in Figure 5-47 and described in Table 5-51.

Return to the Summary Table.

Zone 2 Grab Flash Status Register 3

Figure 5-47. Z2_GRABSECT3R Register

31	30	29	28	27	26	25	24
GRAB_SECT15		GRAB_SECT14		GRAB_SECT13		GRAB_SECT12	
R-0h		R-0h		R-0h		R-0h	
23	22	21	20	19	18	17	16
GRAB_SECT11		GRAB_SECT10		GRAB_SECT9		GRAB_SECT8	
R-0h		R-0h		R-0h		R-0h	
15	14	13	12	11	10	9	8
GRAB_SECT7		GRAB_SECT6		GRAB_SECT5		GRAB_SECT4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
GRAB_SECT3		GRAB_SECT2		GRAB_SECT1		GRAB_SECT0	
R-0h		R-0h		R-0h		R-0h	

Table 5-51. Z2_GRABSECT3R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GRAB_SECT15	R	0h	Value in this field gets loaded from Z2_GRABSECT3[31:30] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 15 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 15 to Zone2. 10 : No request for Bank 2 Flash Sector 15 11 : No request for Bank 2 Flash Sector 15 when this zone is UNLOCKED. Else Bank 2 Flash Sector 15 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
29-28	GRAB_SECT14	R	0h	Value in this field gets loaded from Z2_GRABSECT3[29:28] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 14 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 14 to Zone2. 10 : No request for Bank 2 Flash Sector 14 11 : No request for Bank 2 Flash Sector 14 when this zone is UNLOCKED. Else Bank 2 Flash Sector 14 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
27-26	GRAB_SECT13	R	0h	Value in this field gets loaded from Z2_GRABSECT3[27:26] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 13 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 13 to Zone2. 10 : No request for Bank 2 Flash Sector 13 11 : No request for Bank 2 Flash Sector 13 when this zone is UNLOCKED. Else Bank 2 Flash Sector 13 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 5-51. Z2_GRABSECT3R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25-24	GRAB_SECT12	R	0h	Value in this field gets loaded from Z2_GRABSECT3[25:24] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 12 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 12 to Zone2. 10 : No request for Bank 2 Flash Sector 12 11 : No request for Bank 2 Flash Sector 12 when this zone is UNLOCKED. Else Bank 2 Flash Sector 12 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
23-22	GRAB_SECT11	R	0h	Value in this field gets loaded from Z2_GRABSECT3[23:22] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 11 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 11 to Zone2. 10 : No request for Bank 2 Flash Sector 11 11 : No request for Bank 2 Flash Sector 11 when this zone is UNLOCKED. Else Bank 2 Flash Sector 11 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
21-20	GRAB_SECT10	R	0h	Value in this field gets loaded from Z2_GRABSECT3[21:20] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 10 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 10 to Zone2. 10 : No request for Bank 2 Flash Sector 10 11 : No request for Bank 2 Flash Sector 10 when this zone is UNLOCKED. Else Bank 2 Flash Sector 10 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
19-18	GRAB_SECT9	R	0h	Value in this field gets loaded from Z2_GRABSECT3[19:18] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 9 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 9 to Zone2. 10 : No request for Bank 2 Flash Sector 9 11 : No request for Bank 2 Flash Sector 9 when this zone is UNLOCKED. Else Bank 2 Flash Sector 9 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
17-16	GRAB_SECT8	R	0h	Value in this field gets loaded from Z2_GRABSECT3[17:16] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 8 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 8 to Zone2. 10 : No request for Bank 2 Flash Sector 8 11 : No request for Bank 2 Flash Sector 8 when this zone is UNLOCKED. Else Bank 2 Flash Sector 8 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
15-14	GRAB_SECT7	R	0h	Value in this field gets loaded from Z2_GRABSECT3[15:14] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 7 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 7 to Zone2. 10 : No request for Bank 2 Flash Sector 7 11 : No request for Bank 2 Flash Sector 7 when this zone is UNLOCKED. Else Bank 2 Flash Sector 7 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
13-12	GRAB_SECT6	R	0h	Value in this field gets loaded from Z2_GRABSECT3[13:12] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 6 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 6 to Zone2. 10 : No request for Bank 2 Flash Sector 6 11 : No request for Bank 2 Flash Sector 6 when this zone is UNLOCKED. Else Bank 2 Flash Sector 6 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 5-51. Z2_GRABSECT3R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11-10	GRAB_SECT5	R	0h	Value in this field gets loaded from Z2_GRABSECT3[11:10] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 5 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 5 to Zone2. 10 : No request for Bank 2 Flash Sector 5 11 : No request for Bank 2 Flash Sector 5 when this zone is UNLOCKED. Else Bank 2 Flash Sector 5 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
9-8	GRAB_SECT4	R	0h	Value in this field gets loaded from Z2_GRABSECT3[9:8] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 4 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 4 to Zone2. 10 : No request for Bank 2 Flash Sector 4 11 : No request for Bank 2 Flash Sector 4 when this zone is UNLOCKED. Else Bank 2 Flash Sector 4 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
7-6	GRAB_SECT3	R	0h	Value in this field gets loaded from Z2_GRABSECT3[7:6] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 3 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 3 to Zone2. 10 : No request for Bank 2 Flash Sector 3 11 : No request for Bank 2 Flash Sector 3 when this zone is UNLOCKED. Else Bank 2 Flash Sector 3 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
5-4	GRAB_SECT2	R	0h	Value in this field gets loaded from Z2_GRABSECT3[5:4] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 2 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 2 to Zone2. 10 : No request for Bank 2 Flash Sector 2 11 : No request for Bank 2 Flash Sector 2 when this zone is UNLOCKED. Else Bank 2 Flash Sector 2 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
3-2	GRAB_SECT1	R	0h	Value in this field gets loaded from Z2_GRABSECT3[3:2] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 1 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 1 to Zone2. 10 : No request for Bank 2 Flash Sector 1 11 : No request for Bank 2 Flash Sector 1 when this zone is UNLOCKED. Else Bank 2 Flash Sector 1 is inaccessible if this zone is LOCKED. Reset type: SYSRSn
1-0	GRAB_SECT0	R	0h	Value in this field gets loaded from Z2_GRABSECT3[1:0] when a read is issued to address location of Z2_GRABSECT3 in OTP. 00 : Invalid. Bank 2 Flash Sector 0 is inaccessible. 01 : Request to allocate Bank 2 Flash Sector 0 to Zone2. 10 : No request for Bank 2 Flash Sector 0 11 : No request for Bank 2 Flash Sector 0 when this zone is UNLOCKED. Else Bank 2 Flash Sector 0 is inaccessible if this zone is LOCKED. Reset type: SYSRSn

5.8.3.16 Z2_GRABRAM1R Register (Offset = 20h) [Reset = 0h]

Z2_GRABRAM1R is shown in Figure 5-48 and described in Table 5-52.

Return to the Summary Table.

Zone 2 Grab RAM Status Register 1

Figure 5-48. Z2_GRABRAM1R Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
GRAB_RAM7		GRAB_RAM6		GRAB_RAM5		GRAB_RAM4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
GRAB_RAM3		GRAB_RAM2		GRAB_RAM1		GRAB_RAM0	
R-0h		R-0h		R-0h		R-0h	

Table 5-52. Z2_GRABRAM1R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-14	GRAB_RAM7	R	0h	Value in this field gets loaded from Z2_GRABRAM1[15:14] when a read is issued to address location of Z2_GRABRAM1 in OTP. 00 : Invalid. LS7 RAM is inaccessible. 01 : Request to allocate LS7 RAM to Zone2. 10 : No request for LS7 RAM 11 : No request for LS7 RAM when this zone is UNLOCKED. Else LS7 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
13-12	GRAB_RAM6	R	0h	Value in this field gets loaded from Z2_GRABRAM1[13:12] when a read is issued to address location of Z2_GRABRAM1 in OTP. 00 : Invalid. LS6 RAM is inaccessible. 01 : Request to allocate LS6 RAM to Zone2. 10 : No request for LS6 RAM 11 : No request for LS6 RAM when this zone is UNLOCKED. Else LS6 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
11-10	GRAB_RAM5	R	0h	Value in this field gets loaded from Z2_GRABRAM1[11:10] when a read is issued to address location of Z2_GRABRAM1 in OTP. 00 : Invalid. LS5 RAM is inaccessible. 01 : Request to allocate LS5 RAM to Zone2. 10 : No request for LS5 RAM 11 : No request for LS5 RAM when this zone is UNLOCKED. Else LS5 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
9-8	GRAB_RAM4	R	0h	Value in this field gets loaded from Z2_GRABRAM1[9:8] when a read is issued to address location of Z2_GRABRAM1 in OTP. 00 : Invalid. LS4 RAM is inaccessible. 01 : Request to allocate LS4 RAM to Zone2. 10 : No request for LS4 RAM 11 : No request for LS4 RAM when this zone is UNLOCKED. Else LS4 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn

Table 5-52. Z2_GRABRAM1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	GRAB_RAM3	R	0h	Value in this field gets loaded from Z2_GRABRAM1[7:6] when a read is issued to address location of Z2_GRABRAM1 in OTP. 00 : Invalid. LS3 RAM is inaccessible. 01 : Request to allocate LS3 RAM to Zone2. 10 : No request for LS3 RAM 11 : No request for LS3 RAM when this zone is UNLOCKED. Else LS3 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
5-4	GRAB_RAM2	R	0h	Value in this field gets loaded from Z2_GRABRAM1[5:4] when a read is issued to address location of Z2_GRABRAM1 in OTP. 00 : Invalid. LS2 RAM is inaccessible. 01 : Request to allocate LS2 RAM to Zone2. 10 : No request for LS2 RAM 11 : No request for LS2 RAM when this zone is UNLOCKED. Else LS2 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
3-2	GRAB_RAM1	R	0h	Value in this field gets loaded from Z2_GRABRAM1[3:2] when a read is issued to address location of Z2_GRABRAM1 in OTP. 00 : Invalid. LS1 RAM is inaccessible. 01 : Request to allocate LS1 RAM to Zone2. 10 : No request for LS1 RAM 11 : No request for LS1 RAM when this zone is UNLOCKED. Else LS1 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn
1-0	GRAB_RAM0	R	0h	Value in this field gets loaded from Z2_GRABRAM1[1:0] when a read is issued to address location of Z2_GRABRAM1 in OTP. 00 : Invalid. LS0 RAM is inaccessible. 01 : Request to allocate LS0 RAM to Zone2. 10 : No request for LS0 RAM 11 : No request for LS0 RAM when this zone is UNLOCKED. Else LS0 RAM is inaccessible if this zone is LOCKED. Reset type: SYSRSn

5.8.3.17 Z2_EXEONLYSECT1R Register (Offset = 26h) [Reset = 0h]

Z2_EXEONLYSECT1R is shown in Figure 5-49 and described in Table 5-53.

Return to the Summary Table.

Zone 2 Execute Only Flash Status Register 1

Figure 5-49. Z2_EXEONLYSECT1R Register

31	30	29	28	27	26	25	24
EXEONLY_BANK1_SECT15	EXEONLY_BANK1_SECT14	EXEONLY_BANK1_SECT13	EXEONLY_BANK1_SECT12	EXEONLY_BANK1_SECT11	EXEONLY_BANK1_SECT10	EXEONLY_BANK1_SECT9	EXEONLY_BANK1_SECT8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
EXEONLY_BANK1_SECT7	EXEONLY_BANK1_SECT6	EXEONLY_BANK1_SECT5	EXEONLY_BANK1_SECT4	EXEONLY_BANK1_SECT3	EXEONLY_BANK1_SECT2	EXEONLY_BANK1_SECT1	EXEONLY_BANK1_SECT0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
EXEONLY_BANK0_SECT15	EXEONLY_BANK0_SECT14	EXEONLY_BANK0_SECT13	EXEONLY_BANK0_SECT12	EXEONLY_BANK0_SECT11	EXEONLY_BANK0_SECT10	EXEONLY_BANK0_SECT9	EXEONLY_BANK0_SECT8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
EXEONLY_BANK0_SECT7	EXEONLY_BANK0_SECT6	EXEONLY_BANK0_SECT5	EXEONLY_BANK0_SECT4	EXEONLY_BANK0_SECT3	EXEONLY_BANK0_SECT2	EXEONLY_BANK0_SECT1	EXEONLY_BANK0_SECT0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 5-53. Z2_EXEONLYSECT1R Register Field Descriptions

Bit	Field	Type	Reset	Description
31	EXEONLY_BANK1_SECT15	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[31] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 15 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 1 Flash Sector 15 (only if it's allocated to Zone2) Reset type: SYSRSn
30	EXEONLY_BANK1_SECT14	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[30] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 14 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 1 Flash Sector 14 (only if it's allocated to Zone2) Reset type: SYSRSn
29	EXEONLY_BANK1_SECT13	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[29] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 13 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 1 Flash Sector 13 (only if it's allocated to Zone2) Reset type: SYSRSn
28	EXEONLY_BANK1_SECT12	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[28] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 12 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 1 Flash Sector 12 (only if it's allocated to Zone2) Reset type: SYSRSn

Table 5-53. Z2_EXEONLYSECT1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	EXEONLY_BANK1_SECT11	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[27] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 11 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 1 Flash Sector 11 (only if it's allocated to Zone2) Reset type: SYSRSn
26	EXEONLY_BANK1_SECT10	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[26] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 10 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 1 Flash Sector 10 (only if it's allocated to Zone2) Reset type: SYSRSn
25	EXEONLY_BANK1_SECT9	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[25] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 9 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 1 Flash Sector 9 (only if it's allocated to Zone2) Reset type: SYSRSn
24	EXEONLY_BANK1_SECT8	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[24] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 8 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 1 Flash Sector 8 (only if it's allocated to Zone2) Reset type: SYSRSn
23	EXEONLY_BANK1_SECT7	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[23] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 7 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 1 Flash Sector 7 (only if it's allocated to Zone2) Reset type: SYSRSn
22	EXEONLY_BANK1_SECT6	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[22] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 6 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 1 Flash Sector 6 (only if it's allocated to Zone2) Reset type: SYSRSn
21	EXEONLY_BANK1_SECT5	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[21] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 5 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 1 Flash Sector 5 (only if it's allocated to Zone2) Reset type: SYSRSn
20	EXEONLY_BANK1_SECT4	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[20] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 4 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 1 Flash Sector 4 (only if it's allocated to Zone2) Reset type: SYSRSn

Table 5-53. Z2_EXEONLYSECT1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	EXEONLY_BANK1_SECT3	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[19] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 3 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 1 Flash Sector 3 (only if it's allocated to Zone2) Reset type: SYSRSn
18	EXEONLY_BANK1_SECT2	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[18] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 2 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 1 Flash Sector 2 (only if it's allocated to Zone2) Reset type: SYSRSn
17	EXEONLY_BANK1_SECT1	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[17] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 1 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 1 Flash Sector 1 (only if it's allocated to Zone2) Reset type: SYSRSn
16	EXEONLY_BANK1_SECT0	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[16] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 1 Flash Sector 0 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 1 Flash Sector 0 (only if it's allocated to Zone2) Reset type: SYSRSn
15	EXEONLY_BANK0_SECT15	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[15] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 15 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 15 (only if it's allocated to Zone2) Reset type: SYSRSn
14	EXEONLY_BANK0_SECT14	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[14] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 14 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 14 (only if it's allocated to Zone2) Reset type: SYSRSn
13	EXEONLY_BANK0_SECT13	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[13] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 13 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 13 (only if it's allocated to Zone2) Reset type: SYSRSn
12	EXEONLY_BANK0_SECT12	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[12] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 12 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 12 (only if it's allocated to Zone2) Reset type: SYSRSn

Table 5-53. Z2_EXEONLYSECT1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	EXEONLY_BANK0_SECT 11	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[11] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 11 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 11 (only if it's allocated to Zone2) Reset type: SYSRSn
10	EXEONLY_BANK0_SECT 10	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[10] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 10 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 10 (only if it's allocated to Zone2) Reset type: SYSRSn
9	EXEONLY_BANK0_SECT 9	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[9] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 9 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 9 (only if it's allocated to Zone2) Reset type: SYSRSn
8	EXEONLY_BANK0_SECT 8	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[8] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 8 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 8 (only if it's allocated to Zone2) Reset type: SYSRSn
7	EXEONLY_BANK0_SECT 7	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[7] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 7 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 7 (only if it's allocated to Zone2) Reset type: SYSRSn
6	EXEONLY_BANK0_SECT 6	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[6] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 6 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 6 (only if it's allocated to Zone2) Reset type: SYSRSn
5	EXEONLY_BANK0_SECT 5	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[5] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 5 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 5 (only if it's allocated to Zone2) Reset type: SYSRSn
4	EXEONLY_BANK0_SECT 4	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[4] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 4 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 4 (only if it's allocated to Zone2) Reset type: SYSRSn

Table 5-53. Z2_EXEONLYSECT1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	EXEONLY_BANK0_SECT3	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[3] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 3 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 3 (only if it's allocated to Zone2) Reset type: SYSRSn
2	EXEONLY_BANK0_SECT2	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[2] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 2 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 2 (only if it's allocated to Zone2) Reset type: SYSRSn
1	EXEONLY_BANK0_SECT1	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[1] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 1 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 1 (only if it's allocated to Zone2) Reset type: SYSRSn
0	EXEONLY_BANK0_SECT0	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT1[0] when a read is issued to Z2_EXEONLYSECT1 address location in OTP. 0 : Execute-Only protection is enabled for Bank 0 Flash Sector 0 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 0 Flash Sector 0 (only if it's allocated to Zone2) Reset type: SYSRSn

5.8.3.18 Z2_EXEONLYSECT2R Register (Offset = 28h) [Reset = 0h]

Z2_EXEONLYSECT2R is shown in Figure 5-50 and described in Table 5-54.

Return to the Summary Table.

Zone 2 Execute Only Flash Status Register 2

Figure 5-50. Z2_EXEONLYSECT2R Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
EXEONLY_BANK2_SECT15	EXEONLY_BANK2_SECT14	EXEONLY_BANK2_SECT13	EXEONLY_BANK2_SECT12	EXEONLY_BANK2_SECT11	EXEONLY_BANK2_SECT10	EXEONLY_BANK2_SECT9	EXEONLY_BANK2_SECT8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
EXEONLY_BANK2_SECT7	EXEONLY_BANK2_SECT6	EXEONLY_BANK2_SECT5	EXEONLY_BANK2_SECT4	EXEONLY_BANK2_SECT3	EXEONLY_BANK2_SECT2	EXEONLY_BANK2_SECT1	EXEONLY_BANK2_SECT0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 5-54. Z2_EXEONLYSECT2R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	EXEONLY_BANK2_SECT15	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[15] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for Bank 2 Flash Sector 15 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 2 Flash Sector 15 (only if it's allocated to Zone2) Reset type: SYSRSn
14	EXEONLY_BANK2_SECT14	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[14] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for Bank 2 Flash Sector 14 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 2 Flash Sector 14 (only if it's allocated to Zone2) Reset type: SYSRSn
13	EXEONLY_BANK2_SECT13	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[13] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for Bank 2 Flash Sector 13 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 2 Flash Sector 13 (only if it's allocated to Zone2) Reset type: SYSRSn
12	EXEONLY_BANK2_SECT12	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[12] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for Bank 2 Flash Sector 12 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 2 Flash Sector 12 (only if it's allocated to Zone2) Reset type: SYSRSn

Table 5-54. Z2_EXEONLYSECT2R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	EXEONLY_BANK2_SECT11	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[11] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for Bank 2 Flash Sector 11 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 2 Flash Sector 11 (only if it's allocated to Zone2) Reset type: SYSRSn
10	EXEONLY_BANK2_SECT10	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[10] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for Bank 2 Flash Sector 10 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 2 Flash Sector 10 (only if it's allocated to Zone2) Reset type: SYSRSn
9	EXEONLY_BANK2_SECT9	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[9] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for Bank 2 Flash Sector 9 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 2 Flash Sector 9 (only if it's allocated to Zone2) Reset type: SYSRSn
8	EXEONLY_BANK2_SECT8	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[8] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for Bank 2 Flash Sector 8 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 2 Flash Sector 8 (only if it's allocated to Zone2) Reset type: SYSRSn
7	EXEONLY_BANK2_SECT7	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[7] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for Bank 2 Flash Sector 7 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 2 Flash Sector 7 (only if it's allocated to Zone2) Reset type: SYSRSn
6	EXEONLY_BANK2_SECT6	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[6] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for Bank 2 Flash Sector 6 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 2 Flash Sector 6 (only if it's allocated to Zone2) Reset type: SYSRSn
5	EXEONLY_BANK2_SECT5	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[5] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for Bank 2 Flash Sector 5 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 2 Flash Sector 5 (only if it's allocated to Zone2) Reset type: SYSRSn
4	EXEONLY_BANK2_SECT4	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[4] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for Bank 2 Flash Sector 4 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 2 Flash Sector 4 (only if it's allocated to Zone2) Reset type: SYSRSn

Table 5-54. Z2_EXEONLYSECT2R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	EXEONLY_BANK2_SECT3	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[3] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for Bank 2 Flash Sector 3 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 2 Flash Sector 3 (only if it's allocated to Zone2) Reset type: SYSRSn
2	EXEONLY_BANK2_SECT2	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[2] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for Bank 2 Flash Sector 2 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 2 Flash Sector 2 (only if it's allocated to Zone2) Reset type: SYSRSn
1	EXEONLY_BANK2_SECT1	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[1] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for Bank 2 Flash Sector 1 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 2 Flash Sector 1 (only if it's allocated to Zone2) Reset type: SYSRSn
0	EXEONLY_BANK2_SECT0	R	0h	Value in this field gets loaded from Z2_EXEONLYSECT2[0] when a read is issued to Z2_EXEONLYSECT2 address location in OTP. 0 : Execute-Only protection is enabled for Bank 2 Flash Sector 0 (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for Bank 2 Flash Sector 0 (only if it's allocated to Zone2) Reset type: SYSRSn

5.8.3.19 Z2_EXEONLYRAM1R Register (Offset = 2Ah) [Reset = 0h]

Z2_EXEONLYRAM1R is shown in Figure 5-51 and described in Table 5-55.

Return to the Summary Table.

Zone 2 Execute Only RAM Status Register 1

Figure 5-51. Z2_EXEONLYRAM1R Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
EXEONLY_RAM7	EXEONLY_RAM6	EXEONLY_RAM5	EXEONLY_RAM4	EXEONLY_RAM3	EXEONLY_RAM2	EXEONLY_RAM1	EXEONLY_RAM0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 5-55. Z2_EXEONLYRAM1R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	EXEONLY_RAM7	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[7] when a read is issued to Z2_EXEONLYRAM address location in OTP. 0 : Execute-Only protection is enabled for LS7 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for LS7 RAM (only if it's allocated to Zone2) Reset type: SYSRSn
6	EXEONLY_RAM6	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[6] when a read is issued to Z2_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for LS6 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for LS6 RAM (only if it's allocated to Zone2) Reset type: SYSRSn
5	EXEONLY_RAM5	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[5] when a read is issued to Z2_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for LS5 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for LS5 RAM (only if it's allocated to Zone2) Reset type: SYSRSn
4	EXEONLY_RAM4	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[4] when a read is issued to Z2_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for LS4 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for LS4 RAM (only if it's allocated to Zone2) Reset type: SYSRSn

Table 5-55. Z2_EXEONLYRAM1R Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	EXEONLY_RAM3	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[3] when a read is issued to Z2_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for LS3 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for LS3 RAM (only if it's allocated to Zone2) Reset type: SYSRSn
2	EXEONLY_RAM2	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[2] when a read is issued to Z2_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for LS2 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for LS2 RAM (only if it's allocated to Zone2) Reset type: SYSRSn
1	EXEONLY_RAM1	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[1] when a read is issued to Z2_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for LS1 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for LS1 RAM (only if it's allocated to Zone2) Reset type: SYSRSn
0	EXEONLY_RAM0	R	0h	Value in this field gets loaded from Z2_EXEONLYRAM1[0] when a read is issued to Z2_EXEONLYRAM1 address location in OTP. 0 : Execute-Only protection is enabled for LS0 RAM (only if it's allocated to Zone2) 1 : Execute-Only protection is disabled for LS0 RAM (only if it's allocated to Zone2) Reset type: SYSRSn

5.8.4 DCSM_COMMON_REGS Registers

Table 5-56 lists the memory-mapped registers for the DCSM_COMMON_REGS registers. All register offset addresses not listed in Table 5-56 should be considered as reserved locations and the register contents should not be modified.

Table 5-56. DCSM_COMMON_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	FLSEM	Flash Wrapper Semaphore Register	EALLOW	Go
8h	SECTSTAT1	Flash Sectors Status Register 1		Go
Ah	SECTSTAT2	Flash Sectors Status Register 2		Go
Ch	SECTSTAT3	Flash Sectors Status Register 3		Go
10h	RAMSTAT1	RAM Status Register 1		Go
18h	SECERRSTAT	Security Error Status Register		Go
1Ah	SECERRCLR	Security Error Clear Register		Go
1Ch	SECERRFRC	Security Error Force Register		Go

Complex bit access types are encoded to fit into small table cells. Table 5-57 shows the codes that are used for access types in this section.

Table 5-57. DCSM_COMMON_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

5.8.4.1 FLSEM Register (Offset = 0h) [Reset = 0h]

FLSEM is shown in Figure 5-52 and described in Table 5-58.

Return to the Summary Table.

Flash Wrapper Semaphore Register

Figure 5-52. FLSEM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY								RESERVED						SEM	
R-0/W-0h								R-0h						R/W-0h	

Table 5-58. FLSEM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	KEY	R-0/W	0h	Writing a value 0xA5 into this field will allow the writing of the SEM bits, else writes are ignored. Reads will return 0. Reset type: SYSRSn
7-2	RESERVED	R	0h	Reserved
1-0	SEM	R/W	0h	00 : Flash Wrapper registers can be written by code running from anywhere without any restriction. 01 : Flash Wrapper registers can be written by code running from Zone1 security zone. 10 : Flash Wrapper registers can be written by code running from Zone2 security zone 11 : Flash Wrapper registers can be written by code running from anywhere without any restriction Allowed State Transitions in this field. 00 TO 11 : Not allowed. 11 TO 00 : Not allowed. 00/11 TO 01 : Code running from Zone1 only can perform this transition. 01 TO 00/11 : Code running from Zone1 only can perform this transition. 00/11 TO 10 : Code running from Zone2 only can perform this transition. 10 TO 00/11 : Code running from Zone2 can perform this transition 10 TO 01 : Not allowed. 01 TO 10 : Not allowed. Reset type: SYSRSn

5.8.4.2 SECTSTAT1 Register (Offset = 8h) [Reset = 0h]

SECTSTAT1 is shown in Figure 5-53 and described in Table 5-59.

Return to the Summary Table.

Flash Sectors Status Register 1

Figure 5-53. SECTSTAT1 Register

31	30	29	28	27	26	25	24
STATUS_SECT15		STATUS_SECT14		STATUS_SECT13		STATUS_SECT12	
R-0h		R-0h		R-0h		R-0h	
23	22	21	20	19	18	17	16
STATUS_SECT11		STATUS_SECT10		STATUS_SECT9		STATUS_SECT8	
R-0h		R-0h		R-0h		R-0h	
15	14	13	12	11	10	9	8
STATUS_SECT7		STATUS_SECT6		STATUS_SECT5		STATUS_SECT4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
STATUS_SECT3		STATUS_SECT2		STATUS_SECT1		STATUS_SECT0	
R-0h		R-0h		R-0h		R-0h	

Table 5-59. SECTSTAT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	STATUS_SECT15	R	0h	Reflects the status of flash Bank 0 Sector 15. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
29-28	STATUS_SECT14	R	0h	Reflects the status of flash Bank 0 Sector 14. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
27-26	STATUS_SECT13	R	0h	Reflects the status of flash Bank 0 Sector 13. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
25-24	STATUS_SECT12	R	0h	Reflects the status of flash Bank 0 Sector 12. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
23-22	STATUS_SECT11	R	0h	Reflects the status of flash Bank 0 Sector 11. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

Table 5-59. SECTSTAT1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	STATUS_SECT10	R	0h	Reflects the status of flash Bank 0 Sector 10. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
19-18	STATUS_SECT9	R	0h	Reflects the status of flash Bank 0 Sector 9. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
17-16	STATUS_SECT8	R	0h	Reflects the status of flash Bank 0 sector 8. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
15-14	STATUS_SECT7	R	0h	Reflects the status of flash Bank 0 Sector 7. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
13-12	STATUS_SECT6	R	0h	Reflects the status of flash Bank 0 Sector 6. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
11-10	STATUS_SECT5	R	0h	Reflects the status of flash Bank 0 Sector 5. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
9-8	STATUS_SECT4	R	0h	Reflects the status of flash Bank 0 Sector 4. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
7-6	STATUS_SECT3	R	0h	Reflects the status of flash Bank 0 Sector 3. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

Table 5-59. SECTSTAT1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-4	STATUS_SECT2	R	0h	Reflects the status of flash Bank 0 Sector 2. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
3-2	STATUS_SECT1	R	0h	Reflects the status of flash Bank 0 sector 1. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
1-0	STATUS_SECT0	R	0h	Reflects the status of flash Bank 0 Sector 0. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

5.8.4.3 SECTSTAT2 Register (Offset = Ah) [Reset = 0h]

SECTSTAT2 is shown in Figure 5-54 and described in Table 5-60.

Return to the Summary Table.

Flash Sectors Status Register 2

Figure 5-54. SECTSTAT2 Register

31	30	29	28	27	26	25	24
STATUS_SECT15		STATUS_SECT14		STATUS_SECT13		STATUS_SECT12	
R-0h		R-0h		R-0h		R-0h	
23	22	21	20	19	18	17	16
STATUS_SECT11		STATUS_SECT10		STATUS_SECT9		STATUS_SECT8	
R-0h		R-0h		R-0h		R-0h	
15	14	13	12	11	10	9	8
STATUS_SECT7		STATUS_SECT6		STATUS_SECT5		STATUS_SECT4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
STATUS_SECT3		STATUS_SECT2		STATUS_SECT1		STATUS_SECT0	
R-0h		R-0h		R-0h		R-0h	

Table 5-60. SECTSTAT2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	STATUS_SECT15	R	0h	Reflects the status of flash Bank 1 Sector 15. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
29-28	STATUS_SECT14	R	0h	Reflects the status of flash Bank 1 Sector 14. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
27-26	STATUS_SECT13	R	0h	Reflects the status of flash Bank 1 Sector 13. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
25-24	STATUS_SECT12	R	0h	Reflects the status of flash Bank 1 Sector 12. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
23-22	STATUS_SECT11	R	0h	Reflects the status of flash Bank 1 Sector 11. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

Table 5-60. SECTSTAT2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	STATUS_SECT10	R	0h	Reflects the status of flash Bank 1 Sector 10. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
19-18	STATUS_SECT9	R	0h	Reflects the status of flash Bank 1 Sector 9. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
17-16	STATUS_SECT8	R	0h	Reflects the status of flash Bank 1 sector 8. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
15-14	STATUS_SECT7	R	0h	Reflects the status of flash Bank 1 Sector 7. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
13-12	STATUS_SECT6	R	0h	Reflects the status of flash Bank 1 Sector 6. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
11-10	STATUS_SECT5	R	0h	Reflects the status of flash Bank 1 Sector 5. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
9-8	STATUS_SECT4	R	0h	Reflects the status of flash Bank 1 Sector 4. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
7-6	STATUS_SECT3	R	0h	Reflects the status of flash Bank 1 Sector 3. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

Table 5-60. SECTSTAT2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-4	STATUS_SECT2	R	0h	Reflects the status of flash Bank 1 Sector 2. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
3-2	STATUS_SECT1	R	0h	Reflects the status of flash Bank 1 sector 1. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
1-0	STATUS_SECT0	R	0h	Reflects the status of flash Bank 1 Sector 0. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

5.8.4.4 SECTSTAT3 Register (Offset = Ch) [Reset = 0h]

SECTSTAT3 is shown in Figure 5-55 and described in Table 5-61.

Return to the Summary Table.

Flash Sectors Status Register 3

Figure 5-55. SECTSTAT3 Register

31	30	29	28	27	26	25	24
STATUS_SECT15		STATUS_SECT14		STATUS_SECT13		STATUS_SECT12	
R-0h		R-0h		R-0h		R-0h	
23	22	21	20	19	18	17	16
STATUS_SECT11		STATUS_SECT10		STATUS_SECT9		STATUS_SECT8	
R-0h		R-0h		R-0h		R-0h	
15	14	13	12	11	10	9	8
STATUS_SECT7		STATUS_SECT6		STATUS_SECT5		STATUS_SECT4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
STATUS_SECT3		STATUS_SECT2		STATUS_SECT1		STATUS_SECT0	
R-0h		R-0h		R-0h		R-0h	

Table 5-61. SECTSTAT3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	STATUS_SECT15	R	0h	Reflects the status of flash Bank 2 Sector 15. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
29-28	STATUS_SECT14	R	0h	Reflects the status of flash Bank 2 Sector 14. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
27-26	STATUS_SECT13	R	0h	Reflects the status of flash Bank 2 Sector 13. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
25-24	STATUS_SECT12	R	0h	Reflects the status of flash Bank 2 Sector 12. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
23-22	STATUS_SECT11	R	0h	Reflects the status of flash Bank 2 Sector 11. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

Table 5-61. SECTSTAT3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	STATUS_SECT10	R	0h	Reflects the status of flash Bank 2 Sector 10. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
19-18	STATUS_SECT9	R	0h	Reflects the status of flash Bank 2 Sector 9. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
17-16	STATUS_SECT8	R	0h	Reflects the status of flash Bank 2 sector 8. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
15-14	STATUS_SECT7	R	0h	Reflects the status of flash Bank 2 Sector 7. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
13-12	STATUS_SECT6	R	0h	Reflects the status of flash Bank 2 Sector 6. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
11-10	STATUS_SECT5	R	0h	Reflects the status of flash Bank 2 Sector 5. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
9-8	STATUS_SECT4	R	0h	Reflects the status of flash Bank 2 Sector 4. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
7-6	STATUS_SECT3	R	0h	Reflects the status of flash Bank 2 Sector 3. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

Table 5-61. SECTSTAT3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-4	STATUS_SECT2	R	0h	Reflects the status of flash Bank 2 Sector 2. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
3-2	STATUS_SECT1	R	0h	Reflects the status of flash Bank 2 sector 1. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
1-0	STATUS_SECT0	R	0h	Reflects the status of flash Bank 2 Sector 0. 00 : Sector is in-accessible 01 : Sector belongs to Zone1. 10 : Sector belongs to Zone2. 11: Sector is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

5.8.4.5 RAMSTAT1 Register (Offset = 10h) [Reset = 0h]

RAMSTAT1 is shown in Figure 5-56 and described in Table 5-62.

Return to the Summary Table.

RAM Status Register 1

Figure 5-56. RAMSTAT1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
STATUS_RAM7		STATUS_RAM6		STATUS_RAM5		STATUS_RAM4	
R-0h		R-0h		R-0h		R-0h	
7	6	5	4	3	2	1	0
STATUS_RAM3		STATUS_RAM2		STATUS_RAM1		STATUS_RAM0	
R-0h		R-0h		R-0h		R-0h	

Table 5-62. RAMSTAT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-14	STATUS_RAM7	R	0h	Reflects the status of LS7 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
13-12	STATUS_RAM6	R	0h	Reflects the status of LS6 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
11-10	STATUS_RAM5	R	0h	Reflects the status of LS5 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
9-8	STATUS_RAM4	R	0h	Reflects the status of LS4 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

Table 5-62. RAMSTAT1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	STATUS_RAM3	R	0h	Reflects the status of LS3 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
5-4	STATUS_RAM2	R	0h	Reflects the status of LS2 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
3-2	STATUS_RAM1	R	0h	Reflects the status of LS1 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn
1-0	STATUS_RAM0	R	0h	Reflects the status of LS0 RAM. 00 : RAM is in-accessible 01 : RAM belongs to Zone1. 10 : RAM belongs to Zone2. 11: RAM is un-secure and code running in both zone have full access to it. Reset type: SYSRSn

5.8.4.6 SECERRSTAT Register (Offset = 18h) [Reset = 0h]

SECERRSTAT is shown in Figure 5-57 and described in Table 5-63.

Return to the Summary Table.

Security Error Status Register

Figure 5-57. SECERRSTAT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ERR
R-0h															R-0h

Table 5-63. SECERRSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	ERR	R	0h	This bit indicates if any error has occurred in the load of any security configuration from USER-OTP. 0: No error has occurred in the load of security information from USER-OTP 1: Error has occurred in the load of security information from USER-OTP Reset type: PORESETn

5.8.4.7 SECERRCLR Register (Offset = 1Ah) [Reset = 0h]

SECERRCLR is shown in Figure 5-58 and described in Table 5-64.

Return to the Summary Table.

Security Error Clear Register

Figure 5-58. SECERRCLR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ERR
R-0h															R-0/W1S-0h

Table 5-64. SECERRCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	ERR	R-0/W1S	0h	A write of '1' clears the SECERRSTST.ERR bit. Write of '0' is ignored. This bit always reads back '0'. Reset type: N/A

5.8.4.8 SECERRFRC Register (Offset = 1Ch) [Reset = 0h]

SECERRFRC is shown in Figure 5-59 and described in Table 5-65.

Return to the Summary Table.

Security Error Force Register

Figure 5-59. SECERRFRC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
KEY															
R-0/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ERR
R-0h															R-0/W1S-0h

Table 5-65. SECERRFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	In order to write to the ERR bits, 0x5a5a must be written to these key bits at the same time. Otherwise, writes are ignored. The key is cleared immediately after writing, so it must be written again for every write to ERR. Reads will return 0. Reset type: N/A
15-1	RESERVED	R	0h	Reserved
0	ERR	R-0/W1S	0h	A write of '1', along with the proper KEY, sets the SECERRSTST.ERR bit. Write of '0' is ignored. This bit always reads back '0'. Reset type: N/A

5.8.5 DCSM_Z1_OTP Registers

Table 5-66 lists the memory-mapped registers for the DCSM_Z1_OTP registers. All register offset addresses not listed in Table 5-66 should be considered as reserved locations and the register contents should not be modified.

Table 5-66. DCSM_Z1_OTP Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	Z1OTP_LINKPOINTER1	Zone 1 Link Pointer1		Go
2h	Z1OTP_LINKPOINTER2	Zone 1 Link Pointer2		Go
4h	Z1OTP_LINKPOINTER3	Zone 1 Link Pointer3		Go
6h	Z1OTP_JLM_ENABLE	Zone 1 JTAGLOCK Enable Register		Go
8h	Z1OTP_GPREG1	Zone 1 General Purpose Register 1		Go
Ah	Z1OTP_GPREG2	Zone 1 General Purpose Register 2		Go
Ch	Z1OTP_GPREG3	Zone 1 General Purpose Register 3		Go
Eh	Z1OTP_GPREG4	Zone 1 General Purpose Register 4		Go
10h	Z1OTP_PSWDLOCK	Secure Password Lock		Go
12h	Z1OTP_CRCLOCK	Secure CRC Lock		Go
14h	Z1OTP_JTAGPSWDH0	JTAG Lock Permanent Password 0		Go
16h	Z1OTP_JTAGPSWDH1	JTAG Lock Permanent Password 1		Go
18h	Z1OTP_CMACKEY0	Secure Boot CMAC Key 0		Go
1Ah	Z1OTP_CMACKEY1	Secure Boot CMAC Key 1		Go
1Ch	Z1OTP_CMACKEY2	Secure Boot CMAC Key 2		Go
1Eh	Z1OTP_CMACKEY3	Secure Boot CMAC Key 3		Go

Complex bit access types are encoded to fit into small table cells. Table 5-67 shows the codes that are used for access types in this section.

Table 5-67. DCSM_Z1_OTP Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

5.8.5.1 Z1OTP_LINKPOINTER1 Register (Offset = 0h) [Reset = FFFFFFFFh]

Z1OTP_LINKPOINTER1 is shown in Figure 5-60 and described in Table 5-68.

Return to the Summary Table.

Zone 1 Link Pointer1

Figure 5-60. Z1OTP_LINKPOINTER1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1OTP_LINKPOINTER1																															
R-FFFFFFFFh																															

Table 5-68. Z1OTP_LINKPOINTER1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1OTP_LINKPOINTER1	R	FFFFFFFFh	Zone1 Link Pointer 1 location in USER OTP. Note: [1] ECC comparison is disabled for this location [2] When this value is loaded into DCSM, if the bits[31:14] !=0, device will remain in BLOCKED state. Before shipping parts to customers, TI would change the value of these bits to 0s. Reset type: N/A

5.8.5.2 Z1OTP_LINKPOINTER2 Register (Offset = 2h) [Reset = FFFFFFFFh]

Z1OTP_LINKPOINTER2 is shown in Figure 5-61 and described in Table 5-69.

Return to the Summary Table.

Zone 1 Link Pointer2

Figure 5-61. Z1OTP_LINKPOINTER2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1OTP_LINKPOINTER2																															
R-FFFFFFFFh																															

Table 5-69. Z1OTP_LINKPOINTER2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1OTP_LINKPOINTER2	R	FFFFFFFFh	Zone1 Link Pointer 2 location in USER OTP. Note: [1] ECC comparison is disabled for this location [2] When this value is loaded into DCSM, if the bits[31:14] !=0, device will remain in BLOCKED state. Before shipping parts to customers, TI would change the value of these bits to 0s. Reset type: N/A

5.8.5.3 Z1OTP_LINKPOINTER3 Register (Offset = 4h) [Reset = FFFFFFFFh]

Z1OTP_LINKPOINTER3 is shown in Figure 5-62 and described in Table 5-70.

Return to the Summary Table.

Zone 1 Link Pointer3

Figure 5-62. Z1OTP_LINKPOINTER3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1OTP_LINKPOINTER3																															
R-FFFFFFFFh																															

Table 5-70. Z1OTP_LINKPOINTER3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1OTP_LINKPOINTER3	R	FFFFFFFFh	Zone1 Link Pointer 3 location in USER OTP. Note: [1] ECC comparison is disabled for this location [2] When this value is loaded into DCSM, if the bits[31:14] !=0, device will remain in BLOCKED state. Before shipping parts to customers, TI would change the value of these bits to 0s. Reset type: N/A

5.8.5.4 Z1OTP_JLM_ENABLE Register (Offset = 6h) [Reset = FFFFFFFFh]

Z1OTP_JLM_ENABLE is shown in Figure 5-63 and described in Table 5-71.

Return to the Summary Table.

Zone 1 JTAGLOCK Enable Register

Figure 5-63. Z1OTP_JLM_ENABLE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1OTP_JLM_ENABLE																															
R-FFFFFFFFh																															

Table 5-71. Z1OTP_JLM_ENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1OTP_JLM_ENABLE	R	FFFFFFFFh	Zone1 JLM_ENABLE register location in USER OTP. Note: When this value is loaded into Z1_JLM_ENABLE, if the value is 32-bit all-1s, the JTAGLOCK will be enabled. Before shipping parts to customers, TI will program the default value to 0xFFFF_000F, which will disable the JTAGLOCK feature. Users should program 0xFFFF_0000 to enable the JTAGLOCK feature. Reset type: N/A

5.8.5.5 Z1OTP_GPREG1 Register (Offset = 8h) [Reset = FFFFFFFFh]

Z1OTP_GPREG1 is shown in Figure 5-64 and described in Table 5-72.

Return to the Summary Table.

Zone 1 General Purpose Register 1

Figure 5-64. Z1OTP_GPREG1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1OTP_GPREG1																															
R-FFFFFFFFh																															

Table 5-72. Z1OTP_GPREG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1OTP_GPREG1	R	FFFFFFFFh	Zone1 General Purpose register location in USER OTP. Reset type: N/A

5.8.5.6 Z1OTP_GPREG2 Register (Offset = Ah) [Reset = FFFFFFFFh]

Z1OTP_GPREG2 is shown in Figure 5-65 and described in Table 5-73.

Return to the Summary Table.

Zone 1 General Purpose Register 2

Figure 5-65. Z1OTP_GPREG2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1OTP_GPREG2																															
R-FFFFFFFFh																															

Table 5-73. Z1OTP_GPREG2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1OTP_GPREG2	R	FFFFFFFFh	Zone1 General Purpose register location in USER OTP. Reset type: N/A

5.8.5.7 Z1OTP_GPREG3 Register (Offset = Ch) [Reset = FFFFFFFFh]

Z1OTP_GPREG3 is shown in Figure 5-66 and described in Table 5-74.

Return to the Summary Table.

Zone 1 General Purpose Register 3

Figure 5-66. Z1OTP_GPREG3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1OTP_GPREG3																															
R-FFFFFFFFh																															

Table 5-74. Z1OTP_GPREG3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1OTP_GPREG3	R	FFFFFFFFh	Zone1 General Purpose register location in USER OTP. Reset type: N/A

5.8.5.8 Z1OTP_GPREG4 Register (Offset = Eh) [Reset = FFFFFFFFh]

Z1OTP_GPREG4 is shown in Figure 5-67 and described in Table 5-75.

Return to the Summary Table.

Zone 1 General Purpose Register 4

Figure 5-67. Z1OTP_GPREG4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1OTP_GPREG4																															
R-FFFFFFFFh																															

Table 5-75. Z1OTP_GPREG4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1OTP_GPREG4	R	FFFFFFFFh	Zone1 General Purpose register location in USER OTP. Reset type: N/A

5.8.5.9 Z1OTP_PSWDLOCK Register (Offset = 10h) [Reset = FFFFFFFFh]

Z1OTP_PSWDLOCK is shown in Figure 5-68 and described in Table 5-76.

Return to the Summary Table.

Secure Password Lock

Figure 5-68. Z1OTP_PSWDLOCK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1OTP_PSWDLOCK																															
R-FFFFFFFFh																															

Table 5-76. Z1OTP_PSWDLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1OTP_PSWDLOCK	R	FFFFFFFFh	Zone1 password lock location in USER OTP. Note: When this value is loaded into DCSM, if the value is 32-bit all-1s, CSMPSWD will remain locked. Before shipping parts to customers, TI would change the value of this location in such a way that the ECC field remains all-1s and also LSB 4-bits remain 4'b1111. Reset type: N/A

5.8.5.10 Z1OTP_CRCLOCK Register (Offset = 12h) [Reset = FFFFFFFFh]

Z1OTP_CRCLOCK is shown in Figure 5-69 and described in Table 5-77.

Return to the Summary Table.

Secure CRC Lock

Figure 5-69. Z1OTP_CRCLOCK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z1OTP_CRCLOCK																															
R-FFFFFFFFh																															

Table 5-77. Z1OTP_CRCLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z1OTP_CRCLOCK	R	FFFFFFFFh	Zone1 CRC lock location in USER OTP. Note: When this value is loaded into DCSM, if the value is 32-bit all-1s, VCU will not have ability to calculate CRC on secured memory content.. Before shipping parts to customers, TI would change the value of this location in such a way that the ECC field remains all-1s and also LSB 4-bits remain 4'b1111. Reset type: N/A

5.8.5.11 Z1OTP_JTAGPSWDH0 Register (Offset = 14h) [Reset = FFFFFFFFh]

Z1OTP_JTAGPSWDH0 is shown in Figure 5-70 and described in Table 5-78.

Return to the Summary Table.

JTAG Lock Permanent Password 0

Figure 5-70. Z1OTP_JTAGPSWDH0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
JTAGPSWDH0																															
R-FFFFFFFFh																															

Table 5-78. Z1OTP_JTAGPSWDH0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	JTAGPSWDH0	R	FFFFFFFFh	JTAG Lock Password High 0 (bits 95:64) location in USER Z1 OTP. This value is dummy loaded into the non-memory-mapped JTAGPSWD register, bits 95:64. TI must program a default value into this location, leaving the ECC bits all 1's. Reset type: N/A

5.8.5.12 Z1OTP_JTAGPSWDH1 Register (Offset = 16h) [Reset = FFFFFFFFh]

Z1OTP_JTAGPSWDH1 is shown in Figure 5-71 and described in Table 5-79.

Return to the Summary Table.

JTAG Lock Permanent Password 1

Figure 5-71. Z1OTP_JTAGPSWDH1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
JTAGPSWDH1																															
R-FFFFFFFFh																															

Table 5-79. Z1OTP_JTAGPSWDH1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	JTAGPSWDH1	R	FFFFFFFFh	JTAG Lock Password High 1 (bits 127:96) location in USER Z1 OTP. This value is dummy loaded into the non-memory-mapped JTAGPSWD register, bits 127:96. Reset type: N/A

5.8.5.13 Z1OTP_CMACKEY0 Register (Offset = 18h) [Reset = FFFFFFFFh]

Z1OTP_CMACKEY0 is shown in Figure 5-72 and described in Table 5-80.

Return to the Summary Table.

Secure Boot CMAC Key 0

Figure 5-72. Z1OTP_CMACKEY0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMACKEY0																															
R-FFFFFFFFh																															

Table 5-80. Z1OTP_CMACKEY0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMACKEY0	R	FFFFFFFFh	Secure Boot CMAC Key 0 (bits 31:0) location in User Z1 OTP. This value is dummy loaded into the CMACKEY0 register. Reset type: N/A

5.8.5.14 Z1OTP_CMACKEY1 Register (Offset = 1Ah) [Reset = FFFFFFFFh]

Z1OTP_CMACKEY1 is shown in Figure 5-73 and described in Table 5-81.

Return to the Summary Table.

Secure Boot CMAC Key 1

Figure 5-73. Z1OTP_CMACKEY1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMACKEY1																															
R-FFFFFFFFh																															

Table 5-81. Z1OTP_CMACKEY1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMACKEY1	R	FFFFFFFFh	Secure Boot CMAC Key 1 (bits 63:32) location in User Z1 OTP. This value is dummy loaded into the CMACKEY1 register. Reset type: N/A

5.8.5.15 Z1OTP_CMACKEY2 Register (Offset = 1Ch) [Reset = FFFFFFFFh]

Z1OTP_CMACKEY2 is shown in Figure 5-74 and described in Table 5-82.

Return to the Summary Table.

Secure Boot CMAC Key 2

Figure 5-74. Z1OTP_CMACKEY2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMACKEY2																															
R-FFFFFFFFh																															

Table 5-82. Z1OTP_CMACKEY2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMACKEY2	R	FFFFFFFFh	Secure Boot CMAC Key 2 (bits 95:64) location in User Z1 OTP. This value is dummy loaded into the CMACKEY2 register. Reset type: N/A

5.8.5.16 Z1OTP_CMACKEY3 Register (Offset = 1Eh) [Reset = FFFFFFFFh]

Z1OTP_CMACKEY3 is shown in Figure 5-75 and described in Table 5-83.

Return to the Summary Table.

Secure Boot CMAC Key 3

Figure 5-75. Z1OTP_CMACKEY3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMACKEY3																															
R-FFFFFFFFh																															

Table 5-83. Z1OTP_CMACKEY3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CMACKEY3	R	FFFFFFFFh	Secure Boot CMAC Key 3 (bits 127:96) location in User Z1 OTP. This value is dummy loaded into the CMACKEY3 register. Reset type: N/A

5.8.6 DCSM_Z2_OTP Registers

Table 5-84 lists the memory-mapped registers for the DCSM_Z2_OTP registers. All register offset addresses not listed in Table 5-84 should be considered as reserved locations and the register contents should not be modified.

Table 5-84. DCSM_Z2_OTP Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	Z2OTP_LINKPOINTER1	Zone 2 Link Pointer1		Go
2h	Z2OTP_LINKPOINTER2	Zone 2 Link Pointer2		Go
4h	Z2OTP_LINKPOINTER3	Zone 2 Link Pointer3		Go
8h	Z2OTP_GPREG1	Zone 2 General Purpose Register 1		Go
Ah	Z2OTP_GPREG2	Zone 2 General Purpose Register 2		Go
Ch	Z2OTP_GPREG3	Zone 2 General Purpose Register 3		Go
Eh	Z2OTP_GPREG4	Zone 2 General Purpose Register 4		Go
10h	Z2OTP_PSWDLOCK	Secure Password Lock		Go
12h	Z2OTP_CRCLOCK	Secure CRC Lock		Go

Complex bit access types are encoded to fit into small table cells. Table 5-85 shows the codes that are used for access types in this section.

Table 5-85. DCSM_Z2_OTP Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

5.8.6.1 Z2OTP_LINKPOINTER1 Register (Offset = 0h) [Reset = FFFFFFFFh]

Z2OTP_LINKPOINTER1 is shown in Figure 5-76 and described in Table 5-86.

Return to the Summary Table.

Zone 2 Link Pointer1

Figure 5-76. Z2OTP_LINKPOINTER1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2OTP_LINKPOINTER1																															
R-FFFFFFFFh																															

Table 5-86. Z2OTP_LINKPOINTER1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2OTP_LINKPOINTER1	R	FFFFFFFFh	Zone2 Link Pointer 1 location in USER OTP. Note: [1] ECC comparison is disabled for this location [2] When this value is loaded into DCSM, if the bits[31:14] !=0, device will remain in BLOCKED state. Before shipping parts to customers, TI would change the value of these bits to 0s. Reset type: N/A

5.8.6.2 Z2OTP_LINKPOINTER2 Register (Offset = 2h) [Reset = FFFFFFFFh]

Z2OTP_LINKPOINTER2 is shown in Figure 5-77 and described in Table 5-87.

Return to the Summary Table.

Zone 2 Link Pointer2

Figure 5-77. Z2OTP_LINKPOINTER2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2OTP_LINKPOINTER2																															
R-FFFFFFFFh																															

Table 5-87. Z2OTP_LINKPOINTER2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2OTP_LINKPOINTER2	R	FFFFFFFFh	Zone2 Link Pointer 2 location in USER OTP. Note: [1] ECC comparison is disabled for this location [2] When this value is loaded into DCSM, if the bits[31:14] !=0, device will remain in BLOCKED state. Before shipping parts to customers, TI would change the value of these bits to 0s. Reset type: N/A

5.8.6.3 Z2OTP_LINKPOINTER3 Register (Offset = 4h) [Reset = FFFFFFFFh]

Z2OTP_LINKPOINTER3 is shown in Figure 5-78 and described in Table 5-88.

Return to the Summary Table.

Zone 2 Link Pointer3

Figure 5-78. Z2OTP_LINKPOINTER3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2OTP_LINKPOINTER3																															
R-FFFFFFFFh																															

Table 5-88. Z2OTP_LINKPOINTER3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2OTP_LINKPOINTER3	R	FFFFFFFFh	Zone2 Link Pointer 3 location in USER OTP. Note: [1] ECC comparison is disabled for this location [2] When this value is loaded into DCSM, if the bits[31:14] !=0, device will remain in BLOCKED state. Before shipping parts to customers, TI would change the value of these bits to 0s. Reset type: N/A

5.8.6.4 Z2OTP_GPREG1 Register (Offset = 8h) [Reset = FFFFFFFFh]

Z2OTP_GPREG1 is shown in Figure 5-79 and described in Table 5-89.

Return to the Summary Table.

Zone 2 General Purpose Register 1

Figure 5-79. Z2OTP_GPREG1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2OTP_GPREG1																															
R-FFFFFFFFh																															

Table 5-89. Z2OTP_GPREG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2OTP_GPREG1	R	FFFFFFFFh	Zone2 General Purpose register location in USER OTP. Reset type: N/A

5.8.6.5 Z2OTP_GPREG2 Register (Offset = Ah) [Reset = FFFFFFFFh]

Z2OTP_GPREG2 is shown in Figure 5-80 and described in Table 5-90.

Return to the Summary Table.

Zone 2 General Purpose Register 2

Figure 5-80. Z2OTP_GPREG2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2OTP_GPREG2																															
R-FFFFFFFFh																															

Table 5-90. Z2OTP_GPREG2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2OTP_GPREG2	R	FFFFFFFFh	Zone2 General Purpose register location in USER OTP. Reset type: N/A

5.8.6.6 Z2OTP_GPREG3 Register (Offset = Ch) [Reset = FFFFFFFFh]

Z2OTP_GPREG3 is shown in Figure 5-81 and described in Table 5-91.

Return to the Summary Table.

Zone 2 General Purpose Register 3

Figure 5-81. Z2OTP_GPREG3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2OTP_GPREG3																															
R-FFFFFFFFh																															

Table 5-91. Z2OTP_GPREG3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2OTP_GPREG3	R	FFFFFFFFh	Zone2 General Purpose register location in USER OTP. Reset type: N/A

5.8.6.7 Z2OTP_GPREG4 Register (Offset = Eh) [Reset = FFFFFFFFh]

Z2OTP_GPREG4 is shown in Figure 5-82 and described in Table 5-92.

Return to the Summary Table.

Zone 2 General Purpose Register 4

Figure 5-82. Z2OTP_GPREG4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2OTP_GPREG4																															
R-FFFFFFFFh																															

Table 5-92. Z2OTP_GPREG4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2OTP_GPREG4	R	FFFFFFFFh	Zone2 General Purpose register location in USER OTP. Reset type: N/A

5.8.6.8 Z2OTP_PSWDLOCK Register (Offset = 10h) [Reset = FFFFFFFFh]

Z2OTP_PSWDLOCK is shown in Figure 5-83 and described in Table 5-93.

Return to the Summary Table.

Secure Password Lock

Figure 5-83. Z2OTP_PSWDLOCK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2OTP_PSWDLOCK																															
R-FFFFFFFFh																															

Table 5-93. Z2OTP_PSWDLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2OTP_PSWDLOCK	R	FFFFFFFFh	Zone2 password lock location in USER OTP. Note: When this value is loaded into DCSM, if the value is 32-bit all-1s, CSMPSWD will remain locked. Before shipping parts to customers, TI would change the value of this location in such a way that the ECC field remains all-1s and also LSB 4-bits remain 4'b1111. Reset type: N/A

5.8.6.9 Z2OTP_CRCLOCK Register (Offset = 12h) [Reset = FFFFFFFFh]

Z2OTP_CRCLOCK is shown in Figure 5-84 and described in Table 5-94.

Return to the Summary Table.

Secure CRC Lock

Figure 5-84. Z2OTP_CRCLOCK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Z2OTP_CRCLOCK																															
R-FFFFFFFFh																															

Table 5-94. Z2OTP_CRCLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Z2OTP_CRCLOCK	R	FFFFFFFFh	Zone2 CRC lock location in USER OTP. Note: When this value is loaded into DCSM, if the value is 32-bit all-1s, VCU will not have ability to calculate CRC on secured memory content.. Before shipping parts to customers, TI would change the value of this location in such a way that the ECC field remains all-1s and also LSB 4-bits remain 4'b1111. Reset type: N/A

Chapter 6

Flash Module

This chapter describes the Flash module.

6.1 Introduction to Flash and OTP Memory

Flash is an electrically erasable/programmable nonvolatile memory that can be programmed and erased many times to ease code development. Flash memory can be used primarily as a program memory for the core, and secondarily as static data memory.

This section describes the proper sequence to configure the wait states and operating mode of Flash. It also includes information on Flash and OTP power modes, how to improve Flash performance by enabling the Flash prefetch/cache mode, and the SECDED safety feature.

6.1.1 FLASH Related Collateral

Foundational Materials

- C2000 Academy - System Design

Getting Started Materials

- [FAQ] FAQ for Flash ECC usage in C2000 devices - Includes ECC test mode, Linker ECC options:
- [FAQ] FAQ on Flash API usage for C2000 devices
- [FAQ] Flash - How to modify an application from RAM configuration to Flash configuration?
- [FAQ] How can we improve the Flash tool performance?
- [FAQ] TI C2000 Device Programming Tools and Services

6.1.2 Features

Features of Flash memory include:

- Three Flash banks (Bank0, 1,2) (refer to the device data manual for the size of the Flash bank)
- 128 bits (bank width) can be programmed at a time along with ECC
- Multiple sectors providing the option of leaving some sectors programmed and only erasing specific sectors
- User-programmable OTP locations (in user-configurable DCSM OTP, also called USER OTP) for configuring security, OTP boot-mode and boot-mode select pins (if the user is unable to use the factory-default boot-mode select pins)
- Flash pump shared by the two banks
- Enhanced performance using the code-prefetch mechanism and data cache in FMC
- Configurable wait states to give the best performance for a given execution speed
- Safety Features:
  - SECDED-single error correction and double error detection is supported in the FMC
  - Address bits are included in ECC
  - Test mode to check the health of ECC logic
- Supports low-power modes for Flash bank and pump for power savings
- Built-in power mode control logic
- Integrated Flash program/erase state machine (FSM) in the FMC
  - Simple Flash API algorithms
  - Fast erase and program times (refer to the device data manual for details)
- Dual Code Security Module (DCSM) to prevents access to the Flash by unauthorized persons (refer to DCSM chapter for details)

6.1.3 Flash Tools

Texas Instruments provides the following tools for Flash:

- Code Composer Studio™ (CCS) IDE - the development environment with integrated Flash plugin. TI recommends performing a debug reset and restart after programming the code into Flash using CCS.
- Flash API Library - a set of software peripheral functions to erase/program Flash
- UniFlash - standalone tool to erase/program/verify the Flash content through JTAG. No CCS is required.
- Users must check and install available updates for CCS On-Chip Flash Plugin and UniFlash tools.

6.1.4 Default Flash Configuration

The following are Flash module configuration settings at power-up:

- Dedicated Flash banks are in sleep mode (BNKPWR bit field in the FBFALLBAC register)
- Shared pump is in sleep mode (PMPPWR bit field in the FPAC1 register)
- ECC is enabled
- Wait-states are set to the maximum (0xF)
- Code-prefetch mechanism and data cache are disabled in all three FMCs
- Bank and pump active grace periods are set to 0x0 (refer to the BAGP field in the FBAC register and PAGP bit field in the FPAC2 register)

Note that boot ROM changes the BNKPWR and PMPPWR bit fields to active mode.

User application software must initialize wait-states using the FRDCNTL register, and configure cache/prefetch features using the FRD_INTF_CTRL register, to achieve optimum system performance. Software that configures Flash settings like wait-states, cache/prefetch features, and so on, must be executed only from RAM memory, **not** from Flash memory.

Note

Before initializing wait-states, turn off the pre-fetch and data caching in the FRD_INTF_CTRL register.

6.2 Flash Bank, OTP, and Pump

There is a dedicated Flash bank for CPU subsystem. Also, there is a one-time programmable (OTP) memory called USER OTP, which the user can program only once and cannot erase. Flash and OTP are uniformly mapped in both program and data memory space.

CPU subsystem have a TI-OTP that contains manufacturing information like settings used by the Flash state machine for erase and program operations, and so on. Users may read TI-OTP but it cannot be programmed or erased. For memory map and size information of the Flash Banks, TI-OTP, USER OTP, and corresponding ECC locations, refer to the device data manual.

Figure 5-2 shows the user-programmable OTP locations in USER-OTP. For more information on the functionality of these fields, refer to the *Dual Code Security Module (DCSM)* and the *ROM Code and Peripheral Booting* chapters.

6.3 Flash Module Controller (FMC)

The CPU interfaces with the FMC, which in turn interfaces with the Flash bank and the pump, to perform erase or program operations, to read data, and execute code from the Flash bank.

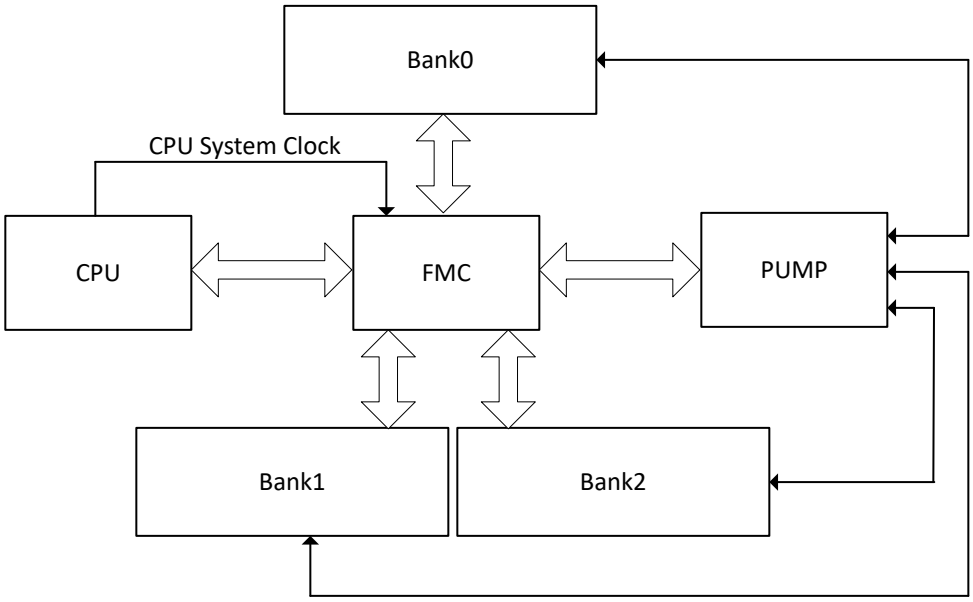


Figure 6-1. FMC Interface with Core, Bank and Pump

6.4 Flash and OTP Power-Down Modes and Wakeup

The Flash bank and pump consume a significant amount of power when active. The Flash module provides a mechanism to power-down Flash banks and pump. Special timers automatically sequence the power-up of the CPU Flash bank. The charge pump module has its own independent power-up timer as well.

The Flash bank and OTP operate in three power modes: Sleep (lowest power), Standby, and Active (highest power)

- Sleep State
  This is the state after a device reset. In this state, a CPU data read or opcode fetch will automatically initiate a change in power mode to the standby state and then to the active state. During this transition time to the active state, the CPU will automatically be stalled.
- Standby State
  This state uses more power than the sleep state, but takes a shorter time to transition to the active or read state. In this state, a CPU data read or opcode fetch will automatically initiate a change in power mode to the active state. During this transition time to the active state, the CPU will automatically be stalled. Once the Flash/OTP has reached the active state, the CPU access will complete as normal.
- Active or Read State
  In this state, the bank and pump are in active power mode state (highest power)

The charge pump operates in two power modes:

- Sleep (lowest power)
- Active (highest power)

Any access to any Flash bank/OTP causes the charge pump to go into active mode, if it is in sleep mode. An erase or program command causes the charge pump and bank to become active. If any bank is in active or in standby mode, the charge pump will be in active mode, independent of the pump power mode control configuration (PMPPWR bit field in the FPAC1 register). While the pump is in sleep state, a charge pump sleep down counter holds a user-configurable value (PSLEEP bit field in the FPAC1 register) and when the charge pump exits sleep power mode, the down counter delays from 0 to PSLEEP prescaled SYSCLK clock cycles (prescaled clock is SYSCLK/2) before putting the charge pump into active power mode.

Following are the number of cycles it will take for the bank and pump to wake up from low power modes.

1. Pump sleep to active = PSLEEP * (SYSCLK/2) cycles
2. Bank sleep to standby = 254 Flash clock cycles
3. Bank standby to active = 55 Flash clock cycles

Where: Flash clock = SYSCLK/(RWAIT+1)

6.5 Active Grace Period

The active grace period (AGP) can be used to optimize the Flash module power consumption versus access time. Faster access times are associated with higher-power modes of operation. At one extreme, the power control logic could attempt to reduce power consumption by putting the bank and charge pump into a low-power mode immediately at the end of every Flash access. However, if accesses are only a few cycles apart, this can actually increase power consumption versus leaving the Flash powered, because the bank and charge pump consume more power during Flash startup and access.

The active grace periods allows the bank and/or charge pump to be maintained in active mode for a specified period following an access. This is done in anticipation of another read within the AGP time, to allow the subsequent read to have a faster access and spend less time dissipating power, than if the bank went into one of the low power modes immediately. If the next access does not occur within the AGP time, the power control logic can automatically put the bank and/or charge pump into a low-power mode to reduce power consumption during long periods of inactivity.

The AGP value is programmed by a set of programmable counters (FBAC and FPAC2) which keep the Flash bank or charge pump in active mode until the counter expires, at which time the bank or charge pump reverts to its fallback power mode as defined in the FBFALLBACK and FPAC1 (refer to PMPPWR bit-field) registers. The application software can configure the fallback power mode to reduce power consumption, or configure it to be active mode to keep the bank active regardless of counter settings (default is SLEEP). The charge pump AGP counter remains in its initialized state when the bank is active, including the AGP counter of the bank. The charge pump AGP counter begins counting when the bank has become inactive.

The application software can check the current power mode of Flash bank and charge pump by reading the FBPRDY register.

6.6 Flash and OTP Performance

Once the Flash bank and pump are in the active power state, a read or fetch access can be classified as a Flash access (access to an address location in Flash) or an OTP access (access to an address location in OTP).

Once the CPU throws an access to a Flash memory address, data is returned after RWAIT+1 number of SYSCLK cycles.

For a USER-OTP access, data is returned after 11 SYSCLK cycles.

RWAIT defines the number of random access wait-states and is configurable using the RWAIT bit-field in the FRDCNTL register. At reset, the RWAIT bit-field defaults to a worst-case wait-state count (15), and therefore needs to be initialized for the appropriate number of wait states to improve performance, based on the CPU clock rate and the access time of the Flash. The Flash supports 0-wait accesses when the RWAIT bits are set to zero. This assumes that the CPU speed is low enough to accommodate the access time.

For a given system clock frequency, RWAIT has to be configured using the following formula:

For C28x Flash Bank: RWAIT = ceiling[(SYSCLK/FCLK)-1]

where SYSCLK is the system operating frequency for CPU1 and CPU2

where FCLK is Flash clock frequency

FCLK should be ≤ $FCLK_{max}$, allowed maximum Flash clock frequency at RWAIT=0.

If RWAIT results in a fractional value when calculated using the above formula, RWAIT has to be rounded up to the nearest integer.

6.7 Flash Read Interface

This section provides details about the data read modes to access Flash bank/OTP and the configuration registers that control the read interface. In addition to a standard read mode, the FMC has a built-in prefetch and cache mechanism to allow increased clock speeds and CPU throughput wherever applicable.

6.7.1 C28x-FMC Flash Read Interface

6.7.1.1 Standard Read Mode

Standard read mode is defined as the read mode in effect when code prefetch-mechanism and data cache are disabled. It is also the default read mode after reset. During this mode, each read access to Flash is decoded by the Flash wrapper to fetch the data from the addressed location and the data is returned after the RWAIT+1 number of cycles (except User OTP).

Prefetch buffers associated with prefetch mechanism and data cache are bypassed in standard read mode; therefore, every access to the Flash/OTP is used by the CPU immediately, and every access creates a unique Flash bank access.

Standard read mode is the recommended mode for lower system frequency operation in which RWAIT can be set to zero to provide single-cycle access operation. The FMC can operate at higher frequencies using standard read mode at the expense of adding wait states. At higher system frequencies, it is recommended to enable cache and prefetch mechanisms to improve performance. Refer to the device specific data manual to determine the maximum Flash frequency allowed in standard read mode (that is, maximum Flash clock frequency with RWAIT=0, $FCLK_{MAX}$).

6.7.1.2 Prefetch Mode

Flash memory is typically used to store application code. During code execution, instructions are fetched from sequential memory addresses, except when a discontinuity occurs. Usually the portion of the code that resides in sequential addresses makes up the majority of the application code and is referred to as linear code. To improve the performance of linear code execution, a Flash prefetch-mechanism has been implemented. Figure 6-2 illustrates how this mode functions.

This prefetch mechanism does a look-ahead prefetch on linear address increments starting from the address of the last instruction fetch. The Flash prefetch mechanism is disabled by default. Setting the PREFETCH_EN bit in the FRD_INTF_CTRL register enables this prefetch mode.

An instruction fetch from the Flash or OTP reads out 128 bits per access. The starting address of the access from Flash is automatically aligned to a 128-bit boundary, such that the instruction location is within the 128 bits to be fetched. With the Flash prefetch mode enabled, the 128 bits read from the instruction fetch are stored in a 128-bit wide by 2-level deep instruction prefetch buffer. The contents of this prefetch buffer are then sent to the CPU for processing as required.

Up to four 32-bit or eight 16-bit instructions can reside within a single 128-bit access. The majority of C28x instructions are 16 bits, so for every 128-bit instruction fetch from the Flash bank, it is likely that there are up to eight instructions in the prefetch buffer ready to process through the CPU. During the time it takes to process these instructions, the Flash prefetch mechanism automatically initiates another access to the Flash bank to prefetch the next 128 bits. In this manner, the Flash prefetch mechanism works in the background to keep the instruction prefetch buffers as full as possible. Using this technique, the overall efficiency of sequential code execution from Flash or OTP is improved significantly.

Note

If the prefetch mechanism is enabled, then the last two rows (16 16-bit words, 256 bits) of the bank that does not have a valid address beyond its boundary should not be used, because the prefetch logic that does a look-ahead prefetch will try to fetch from outside the bank and would result in an ECC error.

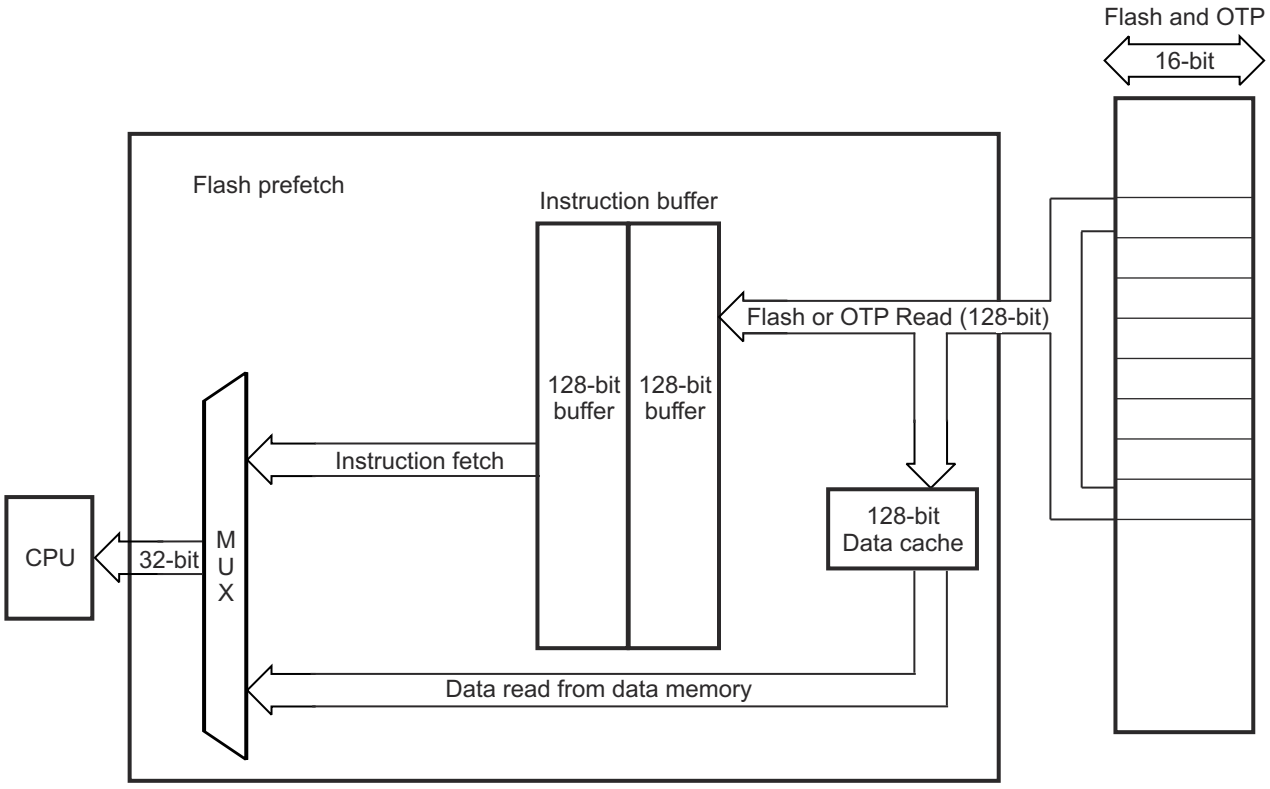


Figure 6-2. Flash Prefetch Mode

The Flash prefetch is aborted only on a PC discontinuity caused by executing an instruction such as a branch, BANZ, call, or loop. When this occurs, the prefetch mechanism is aborted and the contents of the prefetch buffer are flushed. There are two possible scenarios when this occurs:

1. If the destination address is within the Flash or OTP, the prefetch aborts and then resumes at the destination address.
2. If the destination address is outside of the Flash and OTP, the prefetch is aborted and begins again only when a branch is made back into the Flash or OTP. The Flash prefetch mechanism only applies to instruction fetches from program space. Data reads from data memory and from program memory do not utilize the prefetch buffer capability and thus bypass the prefetch buffer. For example, instructions such as MAC, DMAC, and PREAD read a data value from program memory. When this read happens, the prefetch buffer is bypassed but the buffer is not flushed. If an instruction prefetch is already in progress when a data read operation is initiated, then the data read will be stalled until the prefetch completes.

Note that the prefetch mechanism gets bypassed when RWAIT is configured as zero.

6.7.1.2.1 Data Cache

Along with the prefetch mechanism, a data cache of 128-bits wide is also implemented to improve data-space read performance. This data cache will not be filled by the prefetch mechanism. When any kind of data-space read is made by the CPU from an address in the bank, and if the data corresponding to the requested address is not in the data cache, then 128 bits of data will be read from the bank and loaded in the data cache. This data is eventually sent to the CPU for processing. The starting address of the access from Flash is automatically aligned to a 128-bit boundary such that the requested address location is within the 128 bits to be read from the bank. By default, this data cache is disabled and can be enabled by setting DATA_CACHE_EN bit in the FRD_INTF_CTRL register. Note that the data cache gets bypassed when RWAIT is configured as zero.

Some other points to keep in mind when working with Flash/ OTP:

- Reads of the USER OTP locations are hardwired for 10 wait states. The RWAIT bits have no effect on these locations.
- CPU writes to the Flash or OTP memory map areas are ignored. They complete in a single cycle.
- If a security zone is in the locked state and the respective password lock bits are not all 1s, then,
  - Data reads to Zx-CSMPSWD will return 0
  - Program space reads to Zx-CSMPSWD will return 0
  - Program fetches to Zx-CSMPSWD will return 0
- When the Code Security Module (CSM) is secured, reads to the Flash/OTP memory map area from outside the secure zone take the same number of cycles as a normal access. However, the read operation returns a zero.
- The arbitration scheme in FMC prioritizes CPU accesses in the fixed priority order of data read (highest priority), program space read and program fetches/program prefetches (lowest priority).
- When FSM interface is active for erase/program operations, data in the prefetch buffers and data cache in FMC will be flushed.
- When data cache is enabled, the debugger memory window open to Flash/OTP space will invoke data caching. Hence, debugger memory window should not be left open for Flash/OTP space when benchmarking the code for performance.

Note

Flash contents are verified for ECC correctness before they enter prefetch buffer or data cache and not inside the prefetch buffer or data cache itself.

6.8 Erase/Program Flash

Flash memory may be programmed either by using the CCS Flash plugin or by using Uniflash. If these methods are not feasible in an application, the API may be used. The Flash memory should be programmed, erased, and verified only by using the Flash API library. These functions are written, compiled and validated by Texas Instruments. The Flash module contains a Flash state machine (FSM) to perform program and erase operations.

A typical flow to program Flash is:
Erase → Program → Verify

6.8.1 Erase

When the target Flash is erased, it reads as all 1's. This state is called 'blank.' The erase function must be executed before programming. The user should NOT skip erase on sectors that read as 'blank' because these sectors may require additional erasing due to marginally erased bits columns. The FSM provides an Erase Sector command to erase the target sector. The erase function erases the data and the ECC together. Bank erase is also supported in this device.

Note

It is important to provide the correct sector mask for the bank erase command. If the mask is mistakenly chosen to erase an inaccessible sector (belongs to another security zone), the bank erase command will continue attempting to erase the sector endlessly and the FSM will never exit (since erase will not succeed). To avoid such a situation, user must take care to provide the correct mask. However, given that there is a chance of choosing an incorrect mask, TI suggests to initialize the max allowed erase pulses to zero after the max number of pulses are issued by the FSM for the bank erase operation. This will ensure that the FSM will end the bank erase command after trying to erase the inaccessible sector up to the max allowed erase pulses.

The Example_EraseBanks() function in the C2000Ware's Flash API usage example depicts the implementation of this sequence (content of the while loop waiting for the FSM to complete the bank erase command). Users must use this code as-is irrespective of whether or not security is used by the application to also ensure that the FSM exits from bank erase operations in case of an erase-failure.

6.8.2 Program

The FSM provides a command to program the USER OTP and Flash. This command is also used to program ECC check bits.

Note

The main array Flash programming must be aligned to 64-bit address boundaries and each 64-bit word may only be programmed once per write/erase cycle.

The DCSM OTP programming must be aligned to 128-bit address boundaries and each 128-bit word may only be programmed once. The exceptions are:

- The DCSM Zx-LINKPOINTER1 and Zx-LINKPOINTER2 values in the DCSM OTP should be programmed together, and may be programmed one bit at a time as required by the DCSM operation.
- The DCSM Zx-LINKPOINTER3 values in the DCSM OTP may be programmed one bit at a time as required by the DCSM operation.

6.8.3 Verify

After programming, the user must perform verify using API function Fapi_doVerify(). This function verifies the Flash contents against supplied data.

Application software typically perform a CRC check of the Flash memory contents during power-up and at regular intervals during runtime (as needed). Apart from this, ECC logic, when enabled (enabled by default), catches single-bit errors, double-bit errors, and address errors whenever the CPU reads/fetches from a Flash address.

6.9 Error Correction Code (ECC) Protection

CPU1-FMC and CPU2-FMC contain an embedded single error correction and double error detection (SECDED) module. SECDED, when enabled, provides the capability to screen out memory faults. SECDED can detect and correct single-bit data errors and detect address errors/double-bit data errors. For every 64 bits of Flash/OTP data (aligned on a 64-bit memory boundary) that is programmed, eight ECC check bits have to be calculated and programmed in ECC memory space. Refer to the device data manual for the Flash/OTP ECC memory map. SECDED works with a total of eight user-calculated error correction code (ECC) check bits associated with each 64-bit wide data word and its corresponding 128-bit memory-aligned address. Users must program ECC check bits along with Flash data. TI recommends using the AutoEccGeneration option available in the Plugin/API to program ECC. Users can use the Flash API to calculate and program ECC data along with Flash data. Flash API uses hardware ECC logic in the device to generate the ECC data for the given Flash data. The Flash Plugin, the

Flash programming tool integrated with the Code Composer Studio™ IDE, uses the Flash API to generate and program ECC data.

Figure 6-3 illustrates the ECC logic inputs and outputs.

During an instruction fetch or a data read operation, the 19 most significant address bits (three least significant bits of address are not considered), together with the 64-bit data/8-bit ECC read-out of Flash banks/ECC memory map area, pass through the SECDED logic and the eight checkbits are produced in FMC. These eight calculated ECC check bits are then XORed with the stored check bits (user programmed check bits) associated with the address and the read data. The 8-bit output is decoded inside the SECDED module to determine one of three conditions:

- No error occurred
- A correctable error (single bit data error) occurred
- A non-correctable error (double bit data error or address error) occurred

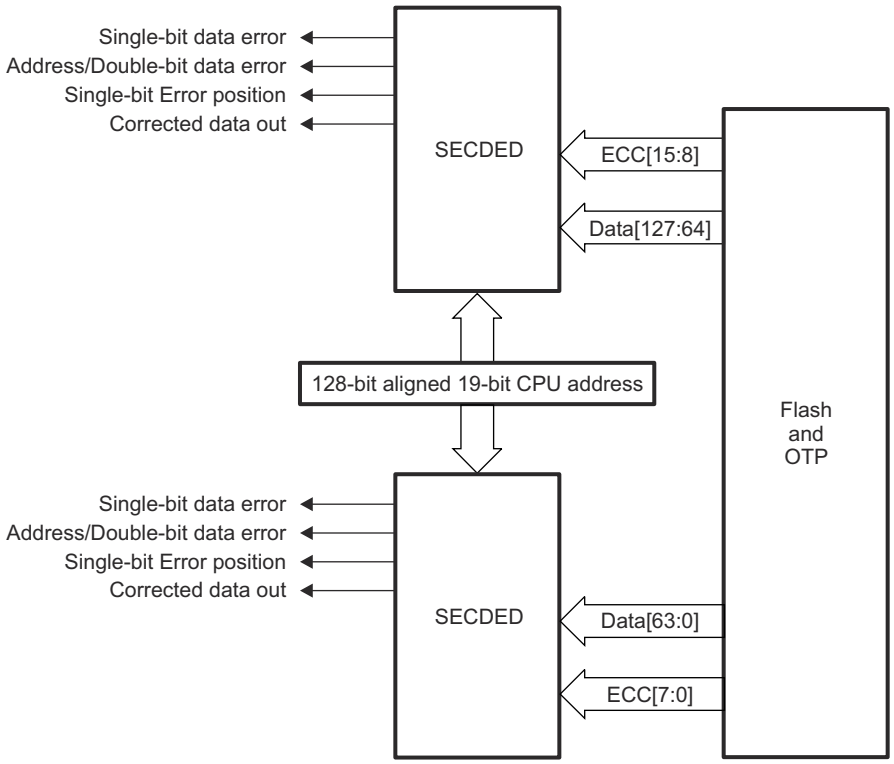


Figure 6-3. ECC Logic Inputs and Outputs

If the SECDED logic finds a single-bit error in the address field, then it is considered to be a non-correctable error.

Note

Since ECC is calculated for an entire 64-bit data, a non 64-bit read such as a byte read or a half-word read will still force the entire 64-bit data to be read and calculated, but only the byte or half-word will be actually used by the CPU.

This ECC (SECDED) feature is enabled at reset. The ECC_ENABLE register can be used to configure (enable/disable) the ECC feature. The ECC for the application code must be programmed. There are two SECDED modules in each FMC. Out of the 128-bit data (aligned on a 128-bit memory boundary) read from the bank/OTP

address, the lower 64 bits of data and corresponding 8 ECC bits (read from user programmable ECC memory area) are fed as inputs to one SECDED module along with 128-bit aligned 19-bit address from where data has been read. The upper 64 bits of data and corresponding 8 ECC bits are fed as inputs to another SECDED module in parallel, along with 128-bit aligned 19-bit address. Each of the SECDED modules evaluate their inputs and determine if there is any single-bit data error or double-bit data error/address error.

ECC logic will be bypassed when the 64 data bits and the associated ECC bits fetched from the bank are either all ones or zeros.

6.9.1 Single-Bit Data Error

This section provides information for both single-bit data errors and single-bit ECC check bit errors. If there is a single bit flip (0 to 1 or 1 to 0) in Flash data or in ECC data, then it is considered as a single-bit data error. The SECDED module detects and corrects single-bit errors, if any, in the 64-bit Flash data or eight ECC check bits read from the Flash/ECC memory map before the read data is provided to the CPU.

When SECDED finds and corrects single bit data errors, the following information is logged in the ECC registers if the ECC feature is enabled:

- Address where the error occurred – if the single-bit error occurs in the lower 64-bits of a 128-bit memory-aligned data, the lower 64-bit memory-aligned address will be captured in the SINGLE_ERR_ADDR_LOW register. If the single-bit error occurs in the upper 64-bits of a 128-bit memory-aligned data, the upper 64-bit memory-aligned address will be captured in the SINGLE_ERR_ADDR_HIGH register.
- Whether the error occurred in data bits or ECC bits – the ERR_TYPE_L and ERR_TYPE_H bit fields in the ERR_POS register indicate whether the error occurred in data bits or ECC bits of the lower 64-bits, or the upper 64-bits respectively, of a 128-bit memory-aligned data.
- Bit position at which error occurred – the ERR_POS_L and ERR_POS_H bit fields in the ERR_POS register indicate the bit position of the error in the lower 64-bits/lower 8-bit ECC, or the upper 64-bits/upper 8-bit ECC respectively, of a 128-bit memory-aligned data.
- Whether the corrected value is 0 (FAIL_0_L, FAIL_0_H flags in ERR_STATUS register)
- Whether the corrected value is 1 (FAIL_1_L, FAIL_1_H flags in ERR_STATUS register)
- A single bit error counter that increments on every single bit error occurrence (ERR_CNT register) until a user-configurable threshold (see ERR_THRESHOLD) is met
- A flag that gets set when one or more single-bit errors occurs after ERR_CNT equals ERR_THRESHOLD (SINGLE_ERR_INT_FLG flag in the ERR_INTFLG register)

When the ERR_CNT value equals THRESHOLD+1 value and a single bit error occurs, the SINGLE_ERR_INT flag is set, and an interrupt (FLASH_CORRECTABLE_ERR on C28x PIE has to be enabled for interrupt, if needed) is fired. The SINGLE_ERR interrupt will not be fired again until the SINGLE_ERR_INTFLG is cleared. If the single error interrupt flag is not cleared using the corresponding error interrupt clear bit in the ERR_INTCLR register, the error interrupt will not come again, as this is an edge-based interrupt.

When multiple single-bit errors get caught by ECC logic, Flash ECC registers will hold the information related to the latest ECC error. When multiple single-bit errors get caught, both FAIL_0_L and FAIL_1_L (and/or FAIL_0_H and FAIL_1_H) might get set, indicating that single-bit fail0/fail1 occurred in different 64-bit aligned addresses.

Although ECC is calculated on 64-bit basis, a read of any address location within a 128-bit aligned Flash memory will cause the single-bit error flag to get set when there is a single-bit error in both or in either one of the lower 64 and upper 64 bits (or corresponding ECC check bits) of that 128-bit data.

6.9.2 Uncorrectable Error

Uncorrectable errors include address errors and double-bit errors in data/ECC. When SECDED finds uncorrectable errors, the following information is logged in ECC registers if the ECC feature is enabled:

- Address where the error occurred – if the uncorrectable error occurs in the lower 64-bits of a 128-bit memory-aligned data, the lower 64-bit memory-aligned address will be captured in the UNC_ERR_ADDR_LOW register. If the uncorrectable error occurs in the upper 64-bits of a 128-bit memory-aligned data, the upper 64-bit memory-aligned address will be captured in the UNC_ERR_ADDR_HIGH register.

- A flag is set indicating that an uncorrectable error occurred – the UNC_ERR_L and UNC_ERR_H flags in the ERR_STATUS register indicate the uncorrectable error occurrence in the lower 64-bits/lower 8-bit ECC, or the upper 64-bits/upper 8-bit ECC, respectively, of a 128-bit memory-aligned data.
- A flag is set indicating that an uncorrectable error interrupt is fired (UNC_ERR_INTFLG in ERR_INTFLG register)

When an uncorrectable error occurs, the UNC_ERR_INTFLG bit is set and an uncorrectable error interrupt is fired. This uncorrectable error interrupt generates an NMI, if enabled. If an uncorrectable error interrupt flag is not cleared using the corresponding error interrupt clear bit in the ERR_INTCLR register, an error interrupt will not come again, as this is an edge based interrupt.

Although ECC is calculated on 64-bit basis, a read of any address location within a 128-bit aligned Flash memory will cause the uncorrectable error flag to get set when there is a uncorrectable error in both or in either one of the lower 64 and upper 64 bits (or corresponding ECC check bits) of that 128-bit data. NMI will occur on the CPU for a read of any address location within a 128-bit aligned Flash memory, when there is an uncorrectable error in both or in either one of the lower 64 and upper 64 bits (or corresponding ECC check bits) of that 128-bit data.

6.9.3 SECDED Logic Correctness Check

Since error detection and correction logic are part of safety-critical logic, safety applications may need to ensure that the SECDED logic is always working properly. For these safety concerns, in order to ensure the correctness of the SECDED logic, an ECC test mode is provided to test the correctness of ECC logic periodically. In this ECC test mode, data/ECC and address inputs to the ECC logic are controlled by the ECC test mode registers FDATAH_TEST, FDATAL_TEST, FECC_TEST, and FADDR_TEST, respectively. Using this test mode, users can introduce single-bit errors, double-bit errors, or address errors and check whether or not SECDED logic is catching those errors. Users can also check if SECDED logic is reporting any false errors when no errors are introduced.

This ECC test mode can be enabled by setting the ECC_TEST_EN bit in the FECC_CTRL register. When ECC test mode is enabled, the CPU cannot read the data from Flash and instead the CPU gets data from the ECC test mode registers (FDATAH_TEST/FDATAL_TEST). This is because ECC test mode registers (FDATAH_TEST, FDATAL_TEST, FECC_TEST) are multiplexed with data from the Flash. Hence, the CPU should not read/fetch from Flash when ECC test mode is enabled. For this reason, ECC test mode code should be executed from RAM and not from Flash.

Only one of the SECDED modules (out of the two SECDED modules that work on lower 64 bits and upper 64 bits of a read 128-bit data) at a time can be tested. The ECC_SELECT bit in the FECC_CTRL register can be configured by users to select one of the SECDED modules for test.

To test the ECC logic using ECC test mode, you can follow the steps:

1. Obtain the ECC for a given Flash address (128-bit aligned) and 64-bit data by using the Auto ECC generation option provided in Flash API.
2. Develop an application to test ECC logic using the above data. In this application:
   - Write the 128-bit aligned 19-bit Flash address in FADDR_TEST
   - Write 64-bit data in FDATAx_TEST (FDATAL_TEST and FDATAH_TEST) registers
   - Write the corresponding 8-bit ECC in the FECC_TEST register
   - In any of the above three steps, users can insert errors (single-bit data error or double-bit data error or address error or single-bit ECC error or double-bit ECC error) so that they can check whether or not ECC logic is able to catch the errors
   - Select the ECC logic block (lower 64-bits or upper 64-bits) which needs to be tested using the ECC_SELECT bit in the FECC_CTRL register
   - Enable ECC test mode usingthe ECC_TEST_EN bit in FECC_CTRL register
   - Write a value of 1 in the DO_ECC_CALC bit in FECC_CTRL register to enable ECC test logic for a single cycle to evaluate the address, data, ECC in FADDR_TEST, FDATAx_TEST and FECC_TEST registers for ECC errors

Once the above ECC test mode registers are written by the user:

- The FECC_OUTH register holds the data output bits 63:32 from the SECDED block under test.
- The FECC_OUTL register holds the data output bits 31:0 from the SECDED block under test.
- The FECC_STATUS register holds the status of single-bit error occurrence, uncorrectable error occurrence, and error position of single- bit error in data/check bits.

6.10 Reserved Locations Within Flash and OTP

When allocating code and data to Flash and OTP memory, keep the following reserved locations in mind:

- The entire OTP has reserved user-configurable locations for security and boot process. For more details on the functionality of these fields, refer to *Dual Code Secutity Module (DCSM)* and the *ROM Code and Peripheral Booting* chapters.
- Refer to the *ROM Code and Peripheral Booting* chapter for reserved locations in Flash for real-time operating system usage and a boot-to-Flash entry point. A boot-to-Flash entry point is reserved for an entry-into-Flash branch instruction. When the boot-to-Flash boot option is used, the boot ROM will jump to this address in Flash. If the user programs a branch instruction here, that will then redirect code execution to the entry point of the application.

6.11 Procedure to Change the Flash Control Registers

During Flash configuration, no accesses to the Flash or OTP can be in progress. This includes instructions still in the CPU pipeline, data reads, and instruction prefetch operations. To be sure that no access takes place during the configuration change, you should follow the procedure shown below for any code that modifies the Flash control registers.

1. Start executing application code from RAM/Flash/OTP.
2. Branch to or call the Flash configuration code (that writes to Flash control registers) in RAM. This is required to properly flush the CPU pipeline before the configuration change. The function that changes the Flash configuration cannot execute from the Flash or OTP. It must reside in RAM.
3. Execute the Flash configuration code (should be located in RAM) that writes to Flash control registers like FRDCNTL, FRD_INTF_CTRL, and so on.
4. At the end of the Flash configuration code execution, wait eight cycles to let the write instructions propagate through the CPU pipeline. This must be done before the return-from-function call is made.
5. Return to the calling function that might reside in RAM or Flash/OTP and continue execution.

6.12 Software

6.12.1 FLASH Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/flash

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

6.12.1.1 Flash ECC Test Mode

FILE: flash_ex2_ecc_test_mode.c

This example demonstrates ECC Test mode.

6.12.1.2 Flash Programming with AutoECC, DataAndECC, DataOnly and EccOnly

FILE: flashapi_ex1_programming.c

This example demonstrates how to program Flash using API's following options
1. AutoEcc generation
2. DataOnly and EccOnly
3. DataAndECC

External Connections
- None.

Watch Variables
- None.

6.13 Flash Registers

This section describes the Flash Module Registers.

6.13.1 FLASH Base Address Table

Table 6-1. FLASH Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
Flash0CtrlRegs	FLASH_CTRL_REGS	FLASH0CTRL_BASE	0x0005_F800	YES	-	-	-	YES
Flash0EccRegs	FLASH_ECC_REGS	FLASH0ECC_BASE	0x0005_FB00	YES	-	-	-	YES

6.13.2 FLASH_CTRL_REGS Registers

Table 6-2 lists the memory-mapped registers for the FLASH_CTRL_REGS registers. All register offset addresses not listed in Table 6-2 should be considered as reserved locations and the register contents should not be modified.

Table 6-2. FLASH_CTRL_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	FRDCNTL	Flash Read Control Register	EALLOW	Go
1Eh	FBAC	Flash Bank Access Control Register	EALLOW	Go
20h	FBFALLBACK	Flash Bank Fallback Power Register	EALLOW	Go
22h	FBPRDY	Flash Bank Pump Ready Register	EALLOW	Go
24h	FPAC1	Flash Pump Access Control Register 1	EALLOW	Go
26h	FPAC2	Flash Pump Access Control Register 2	EALLOW	Go
2Ah	FMSTAT	Flash Module Status Register	EALLOW	Go
180h	FRD_INTF_CTRL	Flash Read Interface Control Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 6-3 shows the codes that are used for access types in this section.

Table 6-3. FLASH_CTRL_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

6.13.2.1 FRDCNTL Register (Offset = 0h) [Reset = F00h]

FRDCNTL is shown in Figure 6-4 and described in Table 6-4.

Return to the Summary Table.

Flash Read Control Register

Figure 6-4. FRDCNTL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				RWAIT				RESERVED							
R-0h				R/W-Fh				R-0h							

Table 6-4. FRDCNTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-12	RESERVED	R	0h	Reserved
11-8	RWAIT	R/W	Fh	Random read waitstate These bits indicate how many waitstates are added to a flash read/fetch access. The RWAIT value can be set anywhere from 0 to 0xF. For a flash access, data is returned in RWAIT+1 SYSCLK cycles. Note: The required wait states for each SYSCLK frequency can be found in the device data manual. Reset type: SYSRSn
7-0	RESERVED	R	0h	Reserved

6.13.2.2 FBAC Register (Offset = 1Eh) [Reset = Fh]

FBAC is shown in Figure 6-5 and described in Table 6-5.

Return to the Summary Table.

Flash Bank Access Control Register

Figure 6-5. FBAC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																BAGP								RESERVED							
R-0h																R/W-0h								R/W-Fh							

Table 6-5. FBAC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	BAGP	R/W	0h	Bank Active Grace Period. These bits contain the starting count value for the BAGP down counter. Any access to a given bank causes its BAGP counter to reload the BAGP value for that bank. After the last access to this flash bank, the down counter delays from 0 to 255 prescaled SYSCLK clock cycles before putting the bank into one of the fallback power modes as determined by the FBFALLBACK register. This value must be greater than 1 when the fallback mode is not ACTIVE. Note: The prescaled clock used for the BAGP down counter is a clock divided by 16 from input SYSCLK. Reset type: SYSRSn
7-0	RESERVED	R/W	Fh	Reserved

6.13.2.3 FBFALLBACK Register (Offset = 20h) [Reset = 0h]

FBFALLBACK is shown in Figure 6-6 and described in Table 6-6.

Return to the Summary Table.

Flash Bank Fallback Power Register

Figure 6-6. FBFALLBACK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		BNKPWR2		BNKPWR1		BNKPWR0	
R-0h		R/W-0h		R/W-0h		R/W-0h	

Table 6-6. FBFALLBACK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-6	RESERVED	R	0h	Reserved
5-4	BNKPWR2	R/W	0h	Fall Back power mode 00 Sleep (Sense amplifiers and sense reference disabled) 01 Standby (Sense amplifiers disabled, but sense reference enabled) 10 Reserved 11 Active (Both sense amplifiers and sense reference enabled) Reset type: SYSRSn
3-2	BNKPWR1	R/W	0h	Fall Back power mode 00 Sleep (Sense amplifiers and sense reference disabled) 01 Standby (Sense amplifiers disabled, but sense reference enabled) 10 Reserved 11 Active (Both sense amplifiers and sense reference enabled) Reset type: SYSRSn
1-0	BNKPWR0	R/W	0h	Fall Back power mode 00 Sleep (Sense amplifiers and sense reference disabled) 01 Standby (Sense amplifiers disabled, but sense reference enabled) 10 Reserved 11 Active (Both sense amplifiers and sense reference enabled) Reset type: SYSRSn

6.13.2.4 FBPRDY Register (Offset = 22h) [Reset = 0h]

FBPRDY is shown in Figure 6-7 and described in Table 6-7.

Return to the Summary Table.

Flash Bank Pump Ready Register

Figure 6-7. FBPRDY Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
PUMPRDY	RESERVED						
R-0h	R-0h						
7	6	5	4	3	2	1	0
RESERVED					BANK2RDY	BANK1RDY	BANK0RDY
R-0h					R-0h	R-0h	R-0h

Table 6-7. FBPRDY Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	PUMPRDY	R	0h	Pump Ready. This is a read-only bit which allows software to determine if the pump is ready for flash access before attempting the actual access. If an access is made to a bank when the pump is not ready, wait states are asserted until it becomes ready. 0 Pump is not ready. 1 Pump is ready, in active power state. Reset type: SYSRSn
14-3	RESERVED	R	0h	Reserved
2	BANK2RDY	R	0h	Bank 2 Ready. This is a read-only register which allows software to determine if the Bank 2 is ready for Flash access before the access is attempted. Note: The user should wait for both the pump and the bank to be ready before attempting an access. 0 Bank 2 is not ready. 1 Bank 2 is in active power mode and is ready for access. Reset type: SYSRSn
1	BANK1RDY	R	0h	Bank 1 Ready. This is a read-only register which allows software to determine if the Bank 1 is ready for Flash access before the access is attempted. Note: The user should wait for both the pump and the bank to be ready before attempting an access. 0 Bank 1 is not ready. 1 Bank 1 is in active power mode and is ready for access. Reset type: SYSRSn
0	BANK0RDY	R	0h	Bank 0 Ready. This is a read-only register which allows software to determine if the Bank 0 is ready for Flash access before the access is attempted. Note: The user should wait for both the pump and the bank to be ready before attempting an access. 0 Bank 0 is not ready. 1 Bank 0 is in active power mode and is ready for access. Reset type: SYSRSn

6.13.2.5 FPAC1 Register (Offset = 24h) [Reset = 00A00000h]

FPAC1 is shown in Figure 6-8 and described in Table 6-8.

Return to the Summary Table.

Flash Pump Access Control Register 1

Figure 6-8. FPAC1 Register

31	30	29	28	27	26	25	24
RESERVED				PSLEEP			
R-0h				R/W-A0h			
23	22	21	20	19	18	17	16
PSLEEP							
R/W-A0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							PMPPWR
R-0h							R/W-0h

Table 6-8. FPAC1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-16	PSLEEP	R/W	A0h	Pump sleep. These bits contain the starting count value for the charge pump sleep down counter. While the charge pump is in sleep mode, the power mode management logic holds the charge pump sleep counter at this value. When the charge pump exits sleep power mode, the down counter delays from 0 to PSLEEP prescaled SYSCLK clock cycles before putting the charge pump into active power mode. Note: The pump sleep down counter uses the same prescaled clock as Bank sleep down counter which is divided by 2 of input SYSCLK. Note: BootROM configures the PSLEEP value as 0x4D4 for 120 MHz operation. Users can modify the PSLEEP value based on their application requirements if needed. Reset type: SYSRSn
15-1	RESERVED	R	0h	Reserved
0	PMPPWR	R/W	0h	Flash Charge Pump Fallback Power Mode. This bit selects what power mode the charge pump enters after the pump active grace period (PAGP) counter has timed out. 0 Sleep (all pump circuits disabled) 1 Active (all pump circuits active) Note for devices with multiple flash banks: As the pump is shared between flash banks, if an access is made either bank, the value of this bit changes to 1 (active). Reset type: SYSRSn

6.13.2.6 FPAC2 Register (Offset = 26h) [Reset = 0h]

FPAC2 is shown in Figure 6-9 and described in Table 6-9.

Return to the Summary Table.

Flash Pump Access Control Register 2

Figure 6-9. FPAC2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																PAGP															
R-0h																R/W-0h															

Table 6-9. FPAC2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	PAGP	R/W	0h	Pump Active Grace Period. This register contains the starting count value for the PAGP mode down counter. Any access to flash memory causes the counter to reload with the PAGP value. After the last access to flash memory, the down counter delays from 0 to 65535 prescaled SYSCLK clock cycles before entering one of the charge pump fallback power modes as determined by PUMPPWR in the FPAC1 register. Note: The PAGP down counter is clocked by the same prescaled clock as the BAGP down counter which is divided by 16 of input SYSCLK. Reset type: SYSRSn

6.13.2.7 FMSTAT Register (Offset = 2Ah) [Reset = 0h]

FMSTAT is shown in Figure 6-10 and described in Table 6-10.

Return to the Summary Table.

Flash Module Status Register

Figure 6-10. FMSTAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						RESERVED	RESERVED
R-0h						R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	PGV	RESERVED	EV	RESERVED	BUSY
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
ERS	PGM	INVDAT	CSTAT	VOLTSTAT	ESUSP	PSUSP	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 6-10. FMSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reserved
17	RESERVED	R	0h	Reserved
16	RESERVED	R	0h	Reserved
15	RESERVED	R	0h	Reserved
14	RESERVED	R	0h	Reserved
13	RESERVED	R	0h	Reserved
12	PGV	R	0h	Program verify When set, indicates that a word is not successfully programmed after the maximum allowed number of program pulses are given for program operation. Reset type: SYSRSn
11	RESERVED	R	0h	Reserved
10	EV	R	0h	Erase verify When set, indicates that a sector is not successfully erased after the maximum allowed number of erase pulses are given for erase operation. Reset type: SYSRSn
9	RESERVED	R	0h	Reserved
8	BUSY	R	0h	When set, this bit indicates that a program, erase, or suspend operation is being processed. Reset type: SYSRSn
7	ERS	R	0h	Erase Active. When set, this bit indicates that the flash module is actively performing an erase operation. This bit is set when erasing starts and is cleared when erasing is complete. It is also cleared when the erase is suspended and set when the erase resumes. Reset type: SYSRSn
6	PGM	R	0h	Program Active. When set, this bit indicates that the flash module is currently performing a program operation. This bit is set when programming starts and is cleared when programming is complete. It is also cleared when programming is suspended and set when programming is resumed. Reset type: SYSRSn

Table 6-10. FMSTAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	INVDAT	R	0h	Invalid Data. When set, this bit indicates that the user attempted to program a "1" where a "0" was already present. Reset type: SYSRSn
4	CSTAT	R	0h	Command Status. Once the FSM starts any failure will set this bit. When set, this bit informs the host that the program, erase, or validate sector command failed and the command was stopped. This bit is cleared by the Clear Status command. For some errors, this will be the only indication of an FSM error because the cause does not fall within the other error bit types. Reset type: SYSRSn
3	VOLTSTAT	R	0h	Core Voltage Status. When set, this bit indicates that the core voltage generator of the pump power upply dipped below the lower limit allowable during a program or erase operation. Reset type: SYSRSn
2	ESUSP	R	0h	When set, this bit indicates that the flash module has received and processed an erase suspend operation. This bit remains set until the erase resume command has been issued or until the Clear_More command is run. Reset type: SYSRSn
1	PSUSP	R	0h	When set, this bit indicates that the flash module has received and processed a program suspend operation. This bit remains set until the program resume command has been issued or until the Clear_More command is run. Reset type: SYSRSn
0	RESERVED	R	0h	Reserved

6.13.2.8 FRD_INTF_CTRL Register (Offset = 180h) [Reset = 0h]

FRD_INTF_CTRL is shown in Figure 6-11 and described in Table 6-11.

Return to the Summary Table.

Flash Read Interface Control Register

Figure 6-11. FRD_INTF_CTRL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						DATA_CACHE_EN	PREFETCH_EN
R-0h						R/W-0h	R/W-0h

Table 6-11. FRD_INTF_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-2	RESERVED	R	0h	Reserved
1	DATA_CACHE_EN	R/W	0h	Data cache enable. 0 A value of 0 disables the data cache. 1 A value of 1 enables the data cache. Reset type: SYSRSn
0	PREFETCH_EN	R/W	0h	Prefetch enable. 0 A value of 0 disables prefetch mechanism. 1 A value of 1 enables pre-fetch mechanism. Reset type: SYSRSn

6.13.3 FLASH_ECC_REGS Registers

Table 6-12 lists the memory-mapped registers for the FLASH_ECC_REGS registers. All register offset addresses not listed in Table 6-12 should be considered as reserved locations and the register contents should not be modified.

Table 6-12. FLASH_ECC_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	ECC_ENABLE	ECC Enable	EALLOW	Go
2h	SINGLE_ERR_ADDR_LOW	Single Error Address Low	EALLOW	Go
4h	SINGLE_ERR_ADDR_HIGH	Single Error Address High	EALLOW	Go
6h	UNC_ERR_ADDR_LOW	Uncorrectable Error Address Low	EALLOW	Go
8h	UNC_ERR_ADDR_HIGH	Uncorrectable Error Address High	EALLOW	Go
Ah	ERR_STATUS	Error Status	EALLOW	Go
Ch	ERR_POS	Error Position	EALLOW	Go
Eh	ERR_STATUS_CLR	Error Status Clear	EALLOW	Go
10h	ERR_CNT	Error Control	EALLOW	Go
12h	ERR_THRESHOLD	Error Threshold	EALLOW	Go
14h	ERR_INTFLG	Error Interrupt Flag	EALLOW	Go
16h	ERR_INTCLR	Error Interrupt Flag Clear	EALLOW	Go
18h	FDATAH_TEST	Data High Test	EALLOW	Go
1Ah	FDATAL_TEST	Data Low Test	EALLOW	Go
1Ch	FADDR_TEST	ECC Test Address	EALLOW	Go
1Eh	FECC_TEST	ECC Test Address	EALLOW	Go
20h	FECC_CTRL	ECC Control	EALLOW	Go
22h	FOUTH_TEST	Test Data Out High	EALLOW	Go
24h	FOUTL_TEST	Test Data Out Low	EALLOW	Go
26h	FECC_STATUS	ECC Status	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 6-13 shows the codes that are used for access types in this section.

Table 6-13. FLASH_ECC_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 6-13. FLASH_ECC_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

6.13.3.1 ECC_ENABLE Register (Offset = 0h) [Reset = Ah]

ECC_ENABLE is shown in Figure 6-12 and described in Table 6-14.

Return to the Summary Table.

ECC Enable

Figure 6-12. ECC_ENABLE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												ENABLE			
R-0h												R/W-Ah			

Table 6-14. ECC_ENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3-0	ENABLE	R/W	Ah	ECC enable. A value of 0xA would enable ECC. Any other value would disable ECC. Reset type: SYSRSn

6.13.3.2 SINGLE_ERR_ADDR_LOW Register (Offset = 2h) [Reset = 0h]

SINGLE_ERR_ADDR_LOW is shown in Figure 6-13 and described in Table 6-15.

Return to the Summary Table.

Single Error Address Low

Figure 6-13. SINGLE_ERR_ADDR_LOW Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERR_ADDR_L																															
R/W-0h																															

Table 6-15. SINGLE_ERR_ADDR_LOW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ERR_ADDR_L	R/W	0h	64-bit aligned address at which a single bit error occurred in the lower 64-bits of a 128-bit aligned memory. Reset type: SYSRSn

6.13.3.3 SINGLE_ERR_ADDR_HIGH Register (Offset = 4h) [Reset = 0h]

SINGLE_ERR_ADDR_HIGH is shown in Figure 6-14 and described in Table 6-16.

Return to the Summary Table.

Single Error Address High

Figure 6-14. SINGLE_ERR_ADDR_HIGH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERR_ADDR_H																															
R/W-0h																															

Table 6-16. SINGLE_ERR_ADDR_HIGH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ERR_ADDR_H	R/W	0h	64-bit aligned address at which a single bit error occurred in the upper 64-bits of a 128-bit aligned memory. Reset type: SYSRSn

6.13.3.4 UNC_ERR_ADDR_LOW Register (Offset = 6h) [Reset = 0h]

UNC_ERR_ADDR_LOW is shown in Figure 6-15 and described in Table 6-17.

Return to the Summary Table.

Uncorrectable Error Address Low

Figure 6-15. UNC_ERR_ADDR_LOW Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UNC_ERR_ADDR_L																															
R/W-0h																															

Table 6-17. UNC_ERR_ADDR_LOW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	UNC_ERR_ADDR_L	R/W	0h	64-bit aligned address at which an uncorrectable error occurred in the lower 64-bits of a 128-bit aligned memory. Reset type: SYSRSn

6.13.3.5 UNC_ERR_ADDR_HIGH Register (Offset = 8h) [Reset = 0h]

UNC_ERR_ADDR_HIGH is shown in Figure 6-16 and described in Table 6-18.

Return to the Summary Table.

Uncorrectable Error Address High

Figure 6-16. UNC_ERR_ADDR_HIGH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UNC_ERR_ADDR_H																															
R/W-0h																															

Table 6-18. UNC_ERR_ADDR_HIGH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	UNC_ERR_ADDR_H	R/W	0h	64-bit aligned address at which an uncorrectable error occurred in the upper 64-bits of a 128-bit aligned memory. Reset type: SYSRSn

6.13.3.6 ERR_STATUS Register (Offset = Ah) [Reset = 0h]

ERR_STATUS is shown in Figure 6-17 and described in Table 6-19.

Return to the Summary Table.

Error Status

Figure 6-17. ERR_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED					UNC_ERR_H	FAIL_1_H	FAIL_0_H
R-0h					R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					UNC_ERR_L	FAIL_1_L	FAIL_0_L
R-0h					R-0h	R-0h	R-0h

Table 6-19. ERR_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Reserved
18	UNC_ERR_H	R	0h	Uncorrectable error. A value of 1 indicates that an un-correctable error occurred in upper 64bits of a 128-bit aligned address. Cleared by writing a 1 to UNC_ERR_H_CLR bit of ERR_STATUS_CLR register. Reset type: SYSRSn
17	FAIL_1_H	R	0h	Fail on 1. 0 Fail on 1 single bit error did not occur in upper 64bits of a 128-bit aligned address. 1 A value of 1 would indicate that a single bit error occurred in upper 64bits of a 128-bit aligned address and the corrected value was 1. Cleared by writing a 1 to FAIL_1_H_CLR bit of ERR_STATUS_CLR register. Note: This bit is updated on every flash access which results in a single bit error, So, in case of multiple single bit error, the status would correspond to the last error which occured. Reset type: SYSRSn
16	FAIL_0_H	R	0h	Fail on 0. 0 Fail on 0 single bit error did not occur in upper 64bits of a 128-bit aligned address. 1 A value of 1 would indicate that a single bit error occurred in upper 64bits of a 128-bit aligned address and the corrected value was 0. Cleared by writing a 1 to FAIL_0_H_CLR bit of ERR_STATUS_CLR register. Note: This bit is updated on every flash access which results in a single bit error, So, in case of multiple single bit error, the status would correspond to the last error which occurred. Reset type: SYSRSn
15-3	RESERVED	R	0h	Reserved
2	UNC_ERR_L	R	0h	Uncorrectable error. A value of 1 indicates that an un-correctable error occurred in lower 64bits of a 128-bit aligned address. Cleared by writing a 1 to UNC_ERR_L_CLR bit of ERR_STATUS_CLR register. Reset type: SYSRSn

Table 6-19. ERR_STATUS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	FAIL_1_L	R	0h	Fail on 1. 0 Fail on 1 single bit error did not occur in lower 64bits of a 128-bit aligned address. 1 A value of 1 would indicate that a single bit error occurred in lower 64bits of a 128-bit aligned address and the corrected value was 1. Cleared by writing a 1 to FAIL_1_L_CLR bit of ERR_STATUS_CLR register. Note: This bit is updated on every flash access which results in a single bit error, So, in case of multiple single bit error, the status would correspond to the last error which occured. Reset type: SYSRSn
0	FAIL_0_L	R	0h	Fail on 0. 0 Fail on 0 single bit error did not occur in lower 64bits of a 128-bit aligned address. 1 Would indicate that a single bit error occurred in lower 64bits of a 128-bit aligned address and the corrected value was 0. Cleared by writing a 1 to FAIL_0_L_CLR bit of ERR_STATUS_CLR register. Note: This bit is updated on every flash access which results in a single bit error, So, in case of multiple single bit error, the status would correspond to the last error which occured. Reset type: SYSRSn

6.13.3.7 ERR_POS Register (Offset = Ch) [Reset = 0h]

ERR_POS is shown in Figure 6-18 and described in Table 6-20.

Return to the Summary Table.

Error Position

Figure 6-18. ERR_POS Register

31	30	29	28	27	26	25	24
RESERVED							ERR_TYPE_H
R-0h							R/W-0h
23	22	21	20	19	18	17	16
RESERVED		ERR_POS_H					
R-0h		R/W-0h					
15	14	13	12	11	10	9	8
RESERVED							ERR_TYPE_L
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED		ERR_POS_L					
R-0h		R/W-0h					

Table 6-20. ERR_POS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24	ERR_TYPE_H	R/W	0h	Error type 0 Indicates that a single bit error occured in upper 64 data bits of a 128-bit aligned address. 1 Indicates that a single bit error occured in ECC check bits of upper 64bits of a 128-bit aligned address. Reset type: SYSRSn
23-22	RESERVED	R	0h	Reserved
21-16	ERR_POS_H	R/W	0h	Error position. Bit position of the single bit error in upper 64bits of a 128-bit aligned address. The position is interpreted depending on whether the ERR_TYPE bit indicates a check bit or a data bit. If ERR_TYPE indicates a check bit error, the error position could range from 0 to 7, else it could range from 0 to 63. Reset type: SYSRSn
15-9	RESERVED	R	0h	Reserved
8	ERR_TYPE_L	R/W	0h	Error type 0 Indicates that a single bit error occured in lower 64 data bits of a 128-bit aligned address. 1 Indicates that a single bit error occured in ECC check bits of lower 64bits of a 128-bit aligned address. Reset type: SYSRSn
7-6	RESERVED	R	0h	Reserved
5-0	ERR_POS_L	R/W	0h	Error position. Bit position of the single bit error in lower 64bits of a 128-bit aligned address. The position is interpreted depending on whether the ERR_TYPE bit indicates a check bit or a data bit. If ERR_TYPE indicates a check bit error, the error position could range from 0 to 7, else it could range from 0 to 63. Reset type: SYSRSn

6.13.3.8 ERR_STATUS_CLR Register (Offset = Eh) [Reset = 0h]

ERR_STATUS_CLR is shown in Figure 6-19 and described in Table 6-21.

Return to the Summary Table.

Error Status Clear

Figure 6-19. ERR_STATUS_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED					UNC_ERR_H_CLR	FAIL_1_H_CLR	FAIL_0_H_CLR
R-0h					R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					UNC_ERR_L_CLR	FAIL_1_L_CLR	FAIL_0_L_CLR
R-0h					R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 6-21. ERR_STATUS_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Reserved
18	UNC_ERR_H_CLR	R-0/W1S	0h	Uncorrectable error clear. Writing a 1 to this bit will clear the UNC_ERR_H bit of ERR_STATUS register. Writes of 0 have no effect. Read returns 0. Reset type: SYSRSn
17	FAIL_1_H_CLR	R-0/W1S	0h	Fail on 1 clear. Writing a 1 to this bit will clear the FAIL_1_H bit of ERR_STATUS register. Writes of 0 have no effect. Read returns 0. Reset type: SYSRSn
16	FAIL_0_H_CLR	R-0/W1S	0h	Fail on 0 clear. Writing a 1 to this bit will clear the FAIL_0_H bit of ERR_STATUS register. Writes of 0 have no effect. Read returns 0. Reset type: SYSRSn
15-3	RESERVED	R	0h	Reserved
2	UNC_ERR_L_CLR	R-0/W1S	0h	Uncorrectable error clear. Writing a 1 to this bit will clear the UNC_ERR_L bit of ERR_STATUS register. Writes of 0 have no effect. Read returns 0. Reset type: SYSRSn
1	FAIL_1_L_CLR	R-0/W1S	0h	Fail on 1 clear. Writing a 1 to this bit will clear the FAIL_1_L bit of ERR_STATUS register. Writes of 0 have no effect. Read returns 0. Reset type: SYSRSn
0	FAIL_0_L_CLR	R-0/W1S	0h	Fail on 0 clear. Writing a 1 to this bit will clear the FAIL_0_L bit of ERR_STATUS register. Writes of 0 have no effect. Read returns 0. Reset type: SYSRSn

6.13.3.9 ERR_CNT Register (Offset = 10h) [Reset = 0h]

ERR_CNT is shown in Figure 6-20 and described in Table 6-22.

Return to the Summary Table.

Error Control

Figure 6-20. ERR_CNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																ERR_CNT															
R-0h																R/W-0h															

Table 6-22. ERR_CNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	ERR_CNT	R/W	0h	Single bit error count. This counter increments with every single bit ECC error occurrence. Upon reaching the threshold value counter stops counting on single bit errors. ERR_CNT can be cleared (irrespective of whether threshold is met or not) using "Single Err Int Clear" bit. This is applicable for ECC logic test mode and normal operational mode. Reset type: SYSRSn

6.13.3.10 ERR_THRESHOLD Register (Offset = 12h) [Reset = 0h]

ERR_THRESHOLD is shown in Figure 6-21 and described in Table 6-23.

Return to the Summary Table.

Error Threshold

Figure 6-21. ERR_THRESHOLD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																ERR_THRESHOLD															
R-0h																R/W-0h															

Table 6-23. ERR_THRESHOLD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	ERR_THRESHOLD	R/W	0h	Single bit error threshold. Sets the threshold for single bit errors. When the ERR_CNT value equals the THRESHOLD value and a single bit error occurs, SINGLE_ERR_INT flag is set, and an interrupt is fired. Reset type: SYSRSn

6.13.3.11 ERR_INTFLG Register (Offset = 14h) [Reset = 0h]

ERR_INTFLG is shown in Figure 6-22 and described in Table 6-24.

Return to the Summary Table.

Error Interrupt Flag

Figure 6-22. ERR_INTFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						UNC_ERR_INTFLG	SINGLE_ERR_INTFLG
R-0h						R-0h	R-0h

Table 6-24. ERR_INTFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-2	RESERVED	R	0h	Reserved
1	UNC_ERR_INTFLG	R	0h	Uncorrectable bit error interrupt flag. When a Un-correctable error occurs, this bit is set and the UNC_ERR_INT interrupt is fired. When UNC_ERR_INTCLR bit of ERR_INTCLR register is written a value of 1 this bit is cleared. Reset type: SYSRSn
0	SINGLE_ERR_INTFLG	R	0h	Single bit error interrupt flag. When the ERR_CNT value equals the ERR_THRESHOLD value and a single bit error occurs then SINGLE_ERR_INT flag is set and SINGLE_ERR_INT interrupt is fired. When SINGLE_ERR_INTCLR bit of ERR_INTCLR register is written a value of 1 this bit is cleared. Reset type: SYSRSn

6.13.3.12 ERR_INTCLR Register (Offset = 16h) [Reset = 0h]

ERR_INTCLR is shown in Figure 6-23 and described in Table 6-25.

Return to the Summary Table.

Error Interrupt Flag Clear

Figure 6-23. ERR_INTCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						UNC_ERR_INTCLR	SINGLE_ERR_INTCLR
R-0h						R-0/W1S-0h	R-0/W1S-0h

Table 6-25. ERR_INTCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-2	RESERVED	R	0h	Reserved
1	UNC_ERR_INTCLR	R-0/W1S	0h	Uncorrectable bit error interrupt flag clear. Writing a 1 to this bit will clear UNC_ERR_INT_FLG. Writes of 0 have no effect. Reset type: SYSRSn
0	SINGLE_ERR_INTCLR	R-0/W1S	0h	Single bit error interrupt flag clear. Writing a 1 to this bit will clear SINGLE_ERR_INT_FLG. Writes of 0 have no effect. Reset type: SYSRSn

6.13.3.13 FDATAH_TEST Register (Offset = 18h) [Reset = 0h]

FDATAH_TEST is shown in Figure 6-24 and described in Table 6-26.

Return to the Summary Table.

Data High Test

Figure 6-24. FDATAH_TEST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FDATAH																															
R/W-0h																															

Table 6-26. FDATAH_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FDATAH	R/W	0h	High double word of selected 64-bit data. User-configurable bits 63:32 of the selected data block in ECC test mode. Reset type: SYSRSn

6.13.3.14 FDATAL_TEST Register (Offset = 1Ah) [Reset = 0h]

FDATAL_TEST is shown in Figure 6-25 and described in Table 6-27.

Return to the Summary Table.

Data Low Test

Figure 6-25. FDATAL_TEST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FDATAL																															
R/W-0h																															

Table 6-27. FDATAL_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FDATAL	R/W	0h	Low double word of selected 64-bit data. User-configurable bits 31:0 of the selected data block in ECC test mode. Reset type: SYSRSn

6.13.3.15 FADDR_TEST Register (Offset = 1Ch) [Reset = 0h]

FADDR_TEST is shown in Figure 6-26 and described in Table 6-28.

Return to the Summary Table.

ECC Test Address

Figure 6-26. FADDR_TEST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED										ADDRH					
R-0h										R/W-0h					
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDRL													RESERVED		
R/W-0h													R-0h		

Table 6-28. FADDR_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R	0h	Reserved
21-16	ADDRH	R/W	0h	Address for selected 64-bit data. User-configurable address bits of the selected data in ECC test mode. Left-shift the address by one bit (to provide byte address) and ignore the three least significant bits of the address and write the bits 21:16 in remaining address bits in this field. Reset type: SYSRSn
15-3	ADDRL	R/W	0h	Address for selected 64-bit data. User-configurable address bits of the selected data in ECC test mode. Left-shift the address by one bit (to provide byte address) and ignore the three least significant bits of the address and write the bits 15:3 in remaining address bits in this field. Reset type: SYSRSn
2-0	RESERVED	R	0h	Reserved

6.13.3.16 FECC_TEST Register (Offset = 1Eh) [Reset = 0h]

FECC_TEST is shown in Figure 6-27 and described in Table 6-29.

Return to the Summary Table.

ECC Test Address

Figure 6-27. FECC_TEST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																RESERVED								ECC							
R-0h																R-0h								R/W-0h							

Table 6-29. FECC_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7-0	ECC	R/W	0h	8-bit ECC for selected 64-bit data. User-configurable ECC bits of the selected 64-bit data block in ECC test mode. Reset type: SYSRSn

6.13.3.17 FECC_CTRL Register (Offset = 20h) [Reset = 0h]

FECC_CTRL is shown in Figure 6-28 and described in Table 6-30.

Return to the Summary Table.

ECC Control

Figure 6-28. FECC_CTRL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					DO_ECC_CALC	ECC_SELECT	ECC_TEST_EN
R-0h					R-0/W1S-0h	R/W-0h	R/W-0h

Table 6-30. FECC_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-3	RESERVED	R	0h	Reserved
2	DO_ECC_CALC	R-0/W1S	0h	Enable ECC calculation. ECC logic will calculate ECC in one cycle for the data and address written in ECC test registers when ECC test logic is enabled by setting ECC_TEST_EN. Reset type: SYSRSn
1	ECC_SELECT	R/W	0h	ECC block select. 0 Selects the ECC block on bits [63:0] of bank data. 1 Selects the ECC block on bits [127:64] of bank data. Reset type: SYSRSn
0	ECC_TEST_EN	R/W	0h	ECC test mode enable. 0 ECC test mode disabled 1 ECC test mode enabled Reset type: SYSRSn

6.13.3.18 FOUTH_TEST Register (Offset = 22h) [Reset = 0h]

FOUTH_TEST is shown in Figure 6-29 and described in Table 6-31.

Return to the Summary Table.

Test Data Out High

Figure 6-29. FOUTH_TEST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATAOUTH																															
R-0h																															

Table 6-31. FOUTH_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATAOUTH	R	0h	High double word test data out. Holds bits 63:32 of the data out of the selected ECC block. Reset type: SYSRSn

6.13.3.19 FOUTL_TEST Register (Offset = 24h) [Reset = 0h]

FOUTL_TEST is shown in Figure 6-30 and described in Table 6-32.

Return to the Summary Table.

Test Data Out Low

Figure 6-30. FOUTL_TEST Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATAOUTL																															
R-0h																															

Table 6-32. FOUTL_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATAOUTL	R	0h	Low double word test data out. Holds bits 31:0 of the data out of the selected ECC block. Reset type: SYSRSn

6.13.3.20 FECC_STATUS Register (Offset = 26h) [Reset = 0h]

FECC_STATUS is shown in Figure 6-31 and described in Table 6-33.

Return to the Summary Table.

ECC Status

Figure 6-31. FECC_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							ERR_TYPE
R-0h							R-0h
7	6	5	4	3	2	1	0
DATA_ERR_POS						UNC_ERR	SINGLE_ERR
R-0h						R-0h	R-0h

Table 6-33. FECC_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-9	RESERVED	R	0h	Reserved
8	ERR_TYPE	R	0h	Test mode ECC single bit error indicator. When 1, indicates that the single bit error is in check bits. When 0, indicates that the single bit error is in data bits (If SINGLE_ERR field is also set). Reset type: SYSRSn
7-2	DATA_ERR_POS	R	0h	Test mode single bit error position. Holds the bit position where the single bit error occurred. The position is interpreted depending on whether the CHK_ERR bit indicates a check bit or a data bit. If CHK_ERR indicates a check bit error, the error position could range from 0 to 7, or it could range from 0 to 63. Reset type: SYSRSn
1	UNC_ERR	R	0h	Test mode ECC double bit error. When 1 indicates that the ECC test resulted in an uncorrectable bit error. Reset type: SYSRSn
0	SINGLE_ERR	R	0h	Test mode ECC single bit error. When 1 indicates that the ECC test resulted in a single bit error. Reset type: SYSRSn

6.13.4 FLASH Registers to Driverlib Functions

Table 6-34. FLASH Registers to Driverlib Functions

File	Driverlib Function
FRDCNTL	
flash.h	Flash_setWaitstates
FBAC	
flash.h	Flash_setBankActiveGracePeriod
FBFALLBACK	
flash.h	Flash_setBankPowerMode
FBPRDY	
flash.h	Flash_isBankReady
flash.h	Flash_isPumpReady
FPAC1	
flash.h	Flash_setPumpPowerMode
flash.h	Flash_setPumpWakeupTime
FPAC2	
flash.h	Flash_setPumpActiveGracePeriod
FMSTAT	
-	
FRD_INTF_CTRL	
flash.h	Flash_enablePrefetch
flash.h	Flash_disablePrefetch
flash.h	Flash_enableCache
flash.h	Flash_disableCache
ECC_ENABLE	
flash.h	Flash_enableECC
flash.h	Flash_disableECC
SINGLE_ERR_ADDR_LOW	
flash.h	Flash_getSingleBitErrorAddressLow
SINGLE_ERR_ADDR_HIGH	
flash.h	Flash_getSingleBitErrorAddressHigh
UNC_ERR_ADDR_LOW	
flash.h	Flash_getUncorrectableErrorAddressLow
UNC_ERR_ADDR_HIGH	
flash.h	Flash_getUncorrectableErrorAddressHigh
ERR_STATUS	
flash.h	Flash_getLowErrorStatus
flash.h	Flash_getHighErrorStatus
flash.h	Flash_clearLowErrorStatus
flash.h	Flash_clearHighErrorStatus
ERR_POS	
flash.h	Flash_getLowErrorPosition
flash.h	Flash_getHighErrorPosition
flash.h	Flash_clearLowErrorPosition
flash.h	Flash_clearHighErrorPosition
flash.h	Flash_getLowErrorType
flash.h	Flash_getHighErrorType

Table 6-34. FLASH Registers to Driverlib Functions (continued)

File	Driverlib Function
flash.h	Flash_clearLowErrorType
flash.h	Flash_clearHighErrorType
ERR_STATUS_CLR	
flash.h	Flash_clearLowErrorStatus
flash.h	Flash_clearHighErrorStatus
ERR_CNT	
flash.h	Flash_getErrorCount
ERR_THRESHOLD	
flash.h	Flash_setErrorThreshold
ERR_INTFLG	
flash.h	Flash_getInterruptFlag
ERR_INTCLR	
flash.h	Flash_clearSingleErrorInterruptFlag
flash.h	Flash_clearUncorrectableInterruptFlag
FDATAH_TEST	
flash.h	Flash_setDataHighECCTest
FDATAL_TEST	
flash.h	Flash_setDataLowECCTest
FADDR_TEST	
flash.h	Flash_setECCTestAddress
FECC_TEST	
flash.h	Flash_setECCTestECCBits
FECC_CTRL	
flash.h	Flash_enableECCTestMode
flash.h	Flash_disableECCTestMode
flash.h	Flash_selectLowECCBlock
flash.h	Flash_selectHighECCBlock
flash.h	Flash_performECCCalculation
FOUTH_TEST	
flash.h	Flash_getTestDataOutHigh
FOUTL_TEST	
flash.h	Flash_getTestDataOutLow
FECC_STATUS	
flash.h	Flash_getECCTestStatus
flash.h	Flash_getECCTestErrorPosition
flash.h	Flash_getECCTestSingleBitErrorType

Chapter 7

Control Law Accelerator (CLA)

The Control Law Accelerator (CLA) Type-2 is an independent, fully-programmable, 32-bit floating-point math processor that brings concurrent control-loop execution to the C28x family. The low interrupt latency of the CLA allows it to read ADC samples "just-in-time." This significantly reduces the ADC sample to output delay to enable faster system response and higher MHz control loops. By using the CLA to service time-critical control loops, the main CPU is free to perform other system tasks such as communications and diagnostics. This chapter provides an overview of the architectural structure and components of the control law accelerator.

7.1 Introduction

The Control Law Accelerator extends the capabilities of the C28x CPU by adding parallel processing. Time-critical control loops serviced by the CLA can achieve low ADC sample to output delay. Thus, the CLA enables faster system response and higher frequency control loops. Utilizing the CLA for time-critical tasks frees up the main CPU to perform other system and communication functions concurrently.

7.1.1 CLA Related Collateral

Foundational Materials

- C2000 Academy - CLA
- C2000 CLA C Compiler Series (Video)
- CLA Hands On Workshop (Video)
- CLA usage in Valley Switching Boost Power Factor Correction (PFC) Reference Design (Video)
- *Enhancing the Computational Performance of the C2000 Microcontroller Family Application Report*

Getting Started Materials

- CLA Software Development Guide
- *Software Examples to Showcase Unique Capabilities of TI's C2000 CLA Application Report*

Expert Materials

- *Digital Control of Two Phase Interleaved PFC and Motor Drive Using MCU With CLA Application Report*
- *Sensorless Field Oriented Control:3-Phase Perm.Magnet Synch. Motors With CLA Application Report*

7.1.2 Features

The following is a list of major features of the CLA:

- C compilers are available for CLA software development.
- Clocked at the same rate as the main CPU (SYSCLKOUT).
- An independent architecture allowing CLA algorithm execution independent of the main C28x CPU.
  - Complete bus architecture:
    - Program Address Bus (PAB) and Program Data Bus (PDB)
    - Data Read Address Bus (DRAB), Data Read Data Bus (DRDB), Data Write Address Bus (DWAB), and Data Write Data Bus (DWDB)
  - Independent eight stage pipeline.
  - 16-bit program counter (MPC)
  - Four 32-bit result registers (MR0-MR3)
  - Two 16-bit auxiliary registers (MAR0, MAR1)
  - Status register (MSTF)
- Instruction set includes:
  - IEEE single-precision (32-bit) floating point math operations
  - Floating-point math with parallel load or store
  - Floating-point multiply with parallel add or subtract
  - 1/X and 1/sqrt(X) estimations
  - Data type conversions.
  - Conditional branch and call
  - Data load/store operations
- The CLA program code can consist of up to eight tasks or interrupt service routines
  - The start address of each task is specified by the MVECT registers.
  - No limit on task size as long as the tasks fit within the configurable CLA program memory space.
  - One task is serviced at a time until its completion. There is no nesting of tasks.
  - Upon task completion a task-specific interrupt is flagged within the PIE.
  - When a task finishes the next highest-priority pending task is automatically started.

- Task trigger mechanisms:
  - C28x CPU via the IACK instruction
  - Task1 to Task8: trigger sources from peripherals connected to the shared bus on which the CLA assumes secondary ownership.
- Memory and Shared Peripherals:
  - Two dedicated message RAMs for communication between the CLA and the main CPU.
  - The C28x CPU can map CLA program and data memory to the main CPU space or CLA space.

7.1.3 Block Diagram

Figure 7-1 is a block diagram.

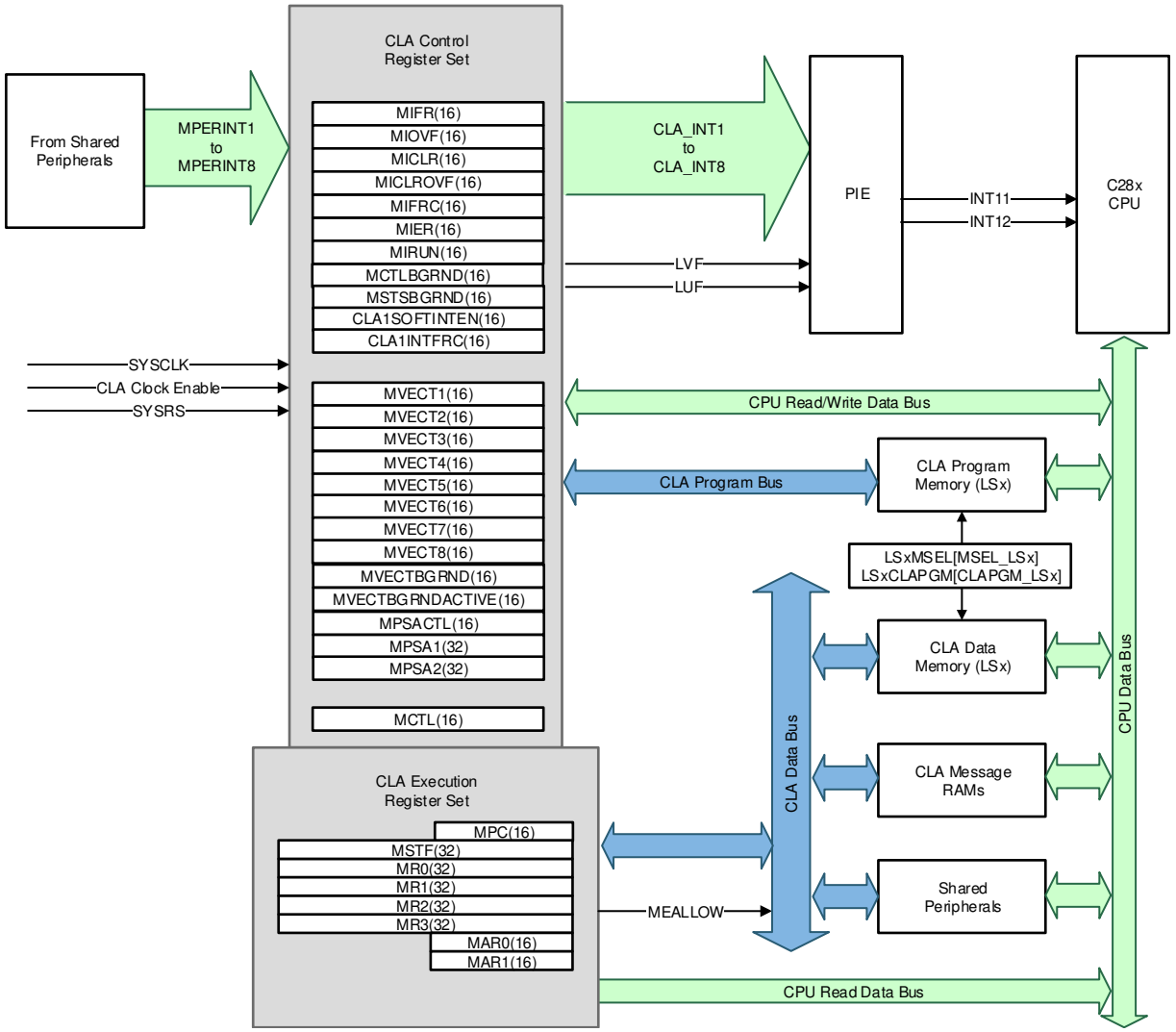


Figure 7-1. CLA (Type 2) Block Diagram

7.2 CLA Interface

This section describes how the C28x main CPU can interface to the CLA and vice versa.

7.2.1 CLA Memory

The CLA can access three types of memory: program, data and message RAMs. The behavior and arbitration for each type of memory is described in this chapter. The CLA RAMs are protected by the DCSM module. Refer to the *Dual Code Security Module (DCSM)* chapter for more details on the security scheme.

- **CLA Program Memory**

  The CLA program can be loaded with any of the local shared memories (LSxRAM). At reset, all memory blocks are mapped to the CPU. While mapped to the CPU space, the CPU can copy the CLA program code into the memory. During debug, the memory can also be loaded directly by the Code Composer Studio™ IDE.

  Once the memory is initialized with CLA code, the CPU maps it to the CLA program space by:
  1. Assigning ownership of the memory block to the CLA by writing a 1 to the memory block's MemCfgRegs.LSxMSEL[MSEL_LSx] bit.
  2. Specifying the memory block as a code block for the CLA by writing a 1 to the MemCfgRegs.LSxCLAPGM[CLAPGM_LSx] bit.

  When a memory block is configured as CLA program memory, debug accesses are allowed only on cycles where the CLA is not fetching a new instruction. A detailed explanation of the memory configurations and access arbitration (CPU, CLA, and DEBUG) process can be found in the Memory Controller Module section of the *System Control and Interrupts* chapter.

  All CLA program fetches are performed as 32-bit read operations and all opcodes must be aligned to an even address. Since all CLA opcodes are 32-bits, this alignment occurs naturally.

- **CLA Data Memory**

  Any of the device's LSxRAMs can serve as data memory blocks to the CLA. At reset, all blocks are mapped to the CPU memory space, whereby the CPU can initialize the memory with data tables, coefficients, and so on, for the CLA to use.

  Once the memory is initialized with CLA data, the CPU maps it to the CLA data space by
  1. Assigning ownership of the memory block to the CLA by writing a 1 to the memory block's MemCfgRegs.LSxMSEL[MSEL_LSx] bit.
  2. Specifying the memory block as a data block for the CLA by writing a 0 to the MemCfgRegs.LSxCLAPGM[CLAPGM_LSx] bit. The value of this bit at reset is 0.

  When a memory block is configured as CLA data memory, CLA read and write accesses are arbitrated along with CPU accesses. The user has the option of turning on CPU fetch or write protection to the memory by writing to the appropriate bits of the MemCfgRegs.LSxACCPROTx registers. A detailed explanation of the memory configurations and access arbitration (CPU, CLA, and DEBUG) process can be found in the Memory Controller Module section of the *System Control and Interrupts* chapter.

- **CLA Shared Message RAMs**

  There are two memory blocks for data sharing and communication between the CLA and the CPU . The message RAMs are always mapped to both CPU and CLA memory spaces, and only data access is allowed; no program fetches can be performed.
  - **CLA to CPU Message RAM**
    The CLA can use this block to pass data to the CPU. This block is both readable and writable by the CLA. This block is also readable by the CPU but writes by the CPU are ignored.
  - **CPU to CLA Message RAM**
    The CPU can use this block to pass data and messages to the CLA. This message RAM is both readable and writable by the CPU. The CLA can perform reads but writes by the CLA are ignored.

7.2.2 CLA Memory Bus

The CLA has dedicated bus architecture similar to that of the C28x CPU where there are separate program read, data read, and data write buses. Thus, there can be simultaneous instruction fetch, data read, and data write in a single cycle. Like the C28x CPU, the CLA expects memory logic to align any 32-bit read or write to an even address. If the address-generation logic generates an odd address, the CLA will begin reading or writing at the previous even address. This alignment does not affect the address values generated by the address-generation logic.

- **CLA Program Bus**

  The CLA program bus has an access range of 32-bit instructions. Since all CLA instructions are 32 bits, this bus always fetches 32 bits at a time and the opcodes must be even-word aligned. The amount of program space available for the CLA is limited to the number of blocks. This number is device-dependent and will be described in the device-specific data manual.

- **CLA Data Read Bus**

  The CLA data read bus has a 64K x 16 address range. The bus can perform 16 or 32-bit reads and will automatically stall if there are memory access conflicts. The data read bus has access to both the message RAMs, CLA data memory, and the shared peripherals.

- **CLA Data Write Bus**

  The CLA data write bus has a 64K x 16 address range. This bus can perform 16 or 32-bit writes. The bus will automatically stall if there are memory access conflicts. The data write bus has access to the CLA to CPU message RAM, CLA data memory, and the shared peripherals.

7.2.3 Shared Peripherals and EALLOW Protection

Refer to the device data manual for the list of peripherals connected to the bus.

Several peripheral control registers are protected from spurious 28x CPU writes by the EALLOW protection mechanism. These same registers are also protected from spurious CLA writes. The EALLOW bit in the CPU status register 1 (ST1) indicates the state of protection for the CPU. Likewise the MEALLOW bit in the CLA status register (MSTF) indicates the state of write protection for the CLA. The MEALLOW CLA instruction enables write access by the CLA to EALLOW protected registers. Likewise the MEDIS CLA instruction will disable write access. This way the CLA can enable/disable write access independent of the CPU.

The ADC offers the option to generate an early interrupt pulse at the start of a sample conversion. If this option is used to start an ADC-triggered CLA task, the user may use the intervening cycles, until the completion of the conversion, to perform preliminary calculations or loads and stores before finally reading the ADC value. The CLA pipeline activity for this scenario is shown in Section 7.5.

7.2.4 CLA Tasks and Interrupt Vectors

The CLA program code is divided up into tasks or interrupt service routines. Tasks do not have a fixed starting location or length. The CLA program memory can be divided up as desired. The CLA uses the contents of the interrupt vectors (MVECT1 to MVECT8) to determine where a task begins; tasks are terminated by the MSTOP instruction.

The CLA supports eight tasks. Task 1 has the highest priority and task 8 has the lowest priority. The Type-2 CLA offers the option of setting the lowest priority task, for example, Task 8, as a background task that, once triggered, runs continuously until the user either terminates it or resets the CLA or the device. The remaining tasks, 1 through 7, maintain their priority levels and interrupt the background task when triggered.

The background task is enabled by setting the BGEN bit in the MCTLBGRND register; this causes the hardware to disable task 8 in the MIER register. The background task derives its interrupt vector from the MVECTBGRND register instead of MVECT8.

A task can be requested by a peripheral interrupt or by software:

- **Peripheral interrupt trigger**

  Each task can be triggered by software-selectable interrupt sources. The trigger for each task is defined by writing an appropriate value to the DmaClaSrcSelRegs.CLA1TASKSRCSELx[TASKx] bit field. Each option specifies an interrupt source from a specific peripheral on the shared bus. The peripheral interrupt triggers are listed in Table 7-1.

  For example, Task 1 (MVECT1) can be set to trigger on EPWMINT1 by writing 36 to DmaClaSrcSelRegs.CLA1TASKSRCSEL1.bit.TASK1. To disable the triggering of a task by a peripheral, the user must set the DmaClaSrcSelRegs.CLA1TASKSRCSELx[TASKx] bit field to 0. It should be noted that a CLA task only triggers on a level transition (an edge) of the configured interrupt source.

Table 7-1. Configuration Options

Select Value (8-bit)	CLA Trigger Source
0	Software Trigger
1	ADCA1_INT
2	ADCA2_INT
3	ADCA3_INT
4	ADCA4_INT
5	ADCA_EVT_INT
6	ADCB1_INT
7	ADCB2_INT
8	ADCB3_INT
9	ADCB4_INT
10	ADCB_EVT_INT
11	ADCC1_INT
12	ADCC2_INT
13	ADCC3_INT
14	ADCC4_INT
15	ADCC_EVT_INT
16	Reserved
17	Reserved
18	Reserved
19	Reserved
20	Reserved
21-28	Reserved
29	XINT1
30	XINT2
31	XINT3
32	XINT4
33	XINT5
34-35	Reserved
36	EPWM1INT
37	EPWM2INT
38	EPWM3INT
39	EPWM4INT

Table 7-1. Configuration Options (continued)

Select Value (8-bit)	CLA Trigger Source
40	EPWM5INT
41	EPWM6INT
42	EPWM7INT
43	EPWM8INT
44	Reserved
45	Reserved
46	Reserved
47	Reserved
48	Reserved
49	Reserved
50	Reserved
51	Reserved
52	MCANA_EVT0
53	MCANA_EVT1
54	MCANA_EVT2
55-67	Reserved
68	TINT0
69	TINT1
70	TINT2
71	Reserved
72	Reserved
73	Reserved
74	Reserved
75	ECAP1INT
76	ECAP2INT
77	ECAP3INT
78	Reserved
79	Reserved
80	Reserved
81	Reserved
82	Reserved
83	EQEP1INT
84	EQEP2INT
85	Reserved
86	Reserved
87	Reserved
88	Reserved
89	ECAP3INT2
90	Reserved
91	Reserved
92	Reserved
93	Reserved
94	Reserved
95	SD1INT
96	SD1DRINT1

Table 7-1. Configuration Options (continued)

Select Value (8-bit)	CLA Trigger Source
97	SD1DRINT2
98	SD1DRINT3
99	SD1DRINT4
100	SD2INT
101	SD2DRINT1
102	SD2DRINT2
103	SD2DRINT3
104	SD2DRINT4
105	PMBUSAINT
106	Reserved
107	Reserved
108	Reserved
109	SPITXINTA
110	SPIRXINTA
111	SPITXINTB
112	SPIRXINTB
113	Reserved
114	Reserved
115	Reserved
116	Reserved
117	LINAINT1
118	LINAINT0
119	LINBINT1
120	LINBINT0
121	CLA1CRC_INT
122	Reserved
123	FSITXA_INT1
124	FSITXA_INT2
125	FSIRXA_INT1
126	FSIRXA_INT2
127	CLB1INT
128	CLB2INT
129	CLB3INT
130	CLB4INT
131-178	Reserved
179	HICA_INT
180-183	Reserved
184	DMA_CH1INT
185	DMA_CH2INT
186	DMA_CH3INT
187	DMA_CH4INT
188	DMA_CH5INT
189	DMA_CH6INT
190-255	Reserved

- **Software Trigger**

  CPU software can trigger tasks by writing to the MIFRC register or by the IACK instruction. Using the IACK instruction is more efficient because it does not require you to issue an EALLOW to set MIFR bits. Set the MCTL[IACKE] bit to enable the IACK feature. Each bit in the operand of the IACK instruction corresponds to a task. For example IACK #0x0001 will set bit 0 in the MIFR register to start task 1. Likewise IACK #0x0003 will set bits 0 and 1 in the MIFR register to start task 1 and task 2.
- **Background Task**

  The Type-2 CLA allows the user to use Task 8 as a background task that runs continuously until the user disables it or resets the device (or the CLA using a soft reset). The background task vector is given by the MVECTBGRND register and its operation controlled by the MCTLBGRND register; it is enabled by setting the BGEN bit to 1. The user can then kick off the task through software by writing a 1 to the BGSTART bit (TRIGEN must be 0), or through a peripheral by setting the TRIGEN bit to 1 and then setting the trigger source in the bit-field, DmaClaSrcSelRegs.CLA1TASKSRCSEL2.bit.TASK8. By default the background task is interruptible; the highest priority pending task will be executed first. When a task completes, and there aren't any pending tasks, the execution returns to the background task. The CLA keeps track of the branching point by saving the return address to the MVECTBGRNDACTIVE register, and then popping this address to the MPC when execution returns. The user may choose to make sections of the background task un-interruptible; it is possible to do this with the MSETC BGINTM assembly instruction.

  Subsequently, enabling interrupts with the MCLRC BGINTM instruction.

  The background interrupt mask bit, BGINTM, can be queried in the MSTSBGRND register. This register also provides the current status of the background task; if it is currently executing, the RUN bit is set to 1, if another trigger for the background task is received while it has already started the overflow (BGOVF) bit is set.

The CLA has its own fetch mechanism and can run and execute a task independently of the CPU. Only one task is serviced at a time; there is no nesting of tasks unless the background task in enabled, then one level of nesting is possible. The task currently running is indicated in the MIRUN register; if the background task is enabled, and running, it is reflected in the MSTSBGRND register (the RUN bit).

Interrupts that have been received but not yet serviced are indicated in the flag register (MIFR). If an interrupt request from a peripheral is received and that same task is already flagged, then the overflow flag bit is set. Overflow flags will remain set until they are cleared by the CPU. If the CLA is idle (no task is currently running) or is executing the background task, then the highest priority interrupt request that is both flagged (MIFR) and enabled (MIER) will start.

The flow is as follows:

1. The associated RUN register bit is set (MIRUN) and the flag bit (MIFR) is cleared.
2. The CLA begins execution at the location indicated by the associated interrupt vector (MVECTx). MVECT contains the absolute 16-bit address of the task in the lower 64K memory space. If a task is interrupting the background task then the current program address is stored in the MVECTBGRNDACTIVE register before execution jumps to the task; this saved address is restored to the MPC when the task completes and execution returns to the background task.
3. The CLA executes instructions until the MSTOP instruction is found. This indicates the end of the task.
4. The MIRUN bit is cleared.
5. The task-specific interrupt to the PIE is issued. This informs the main CPU that the task has completed.
6. The CLA returns to idle (or to the background task, if enabled). Once a task completes the next highest-priority pending task is automatically serviced and this sequence repeats.

7.2.5 CLA Software Interrupt to CPU

The CLA can issue a software interrupt to the C28x CPU at any point in the code through the use of the CLA1SOFTINTEN and CLA1INTFRC registers. See for a description of these registers. If a software interrupt is selected for a CLA task, then an end-of-task interrupt will not be issued to the C28x CPU when that task completes.

7.3 CLA and CPU Arbitration

Typically, CLA activity is independent of the CPU activity. Under the circumstance where the CLA or CPU attempt to concurrently access memory or a peripheral register within the same interface, an arbitration procedure will occur. This section describes this arbitration.

The arbitration follows a fixed arbitration scheme with highest priority first:

They are covered in detail in the Memory Controller Module section of the *System Control and Interrupts* chapter.

7.3.1 CLA Message RAM

Message RAMs consist of blocks:

- CLA to CPU Message RAM
- CPU to CLA Message RAM

These blocks are useful for passing data between the CLA and CPU. No opcode fetches, from either the CLA or CPU, are allowed from the message RAMs. A write protection violation is not generated if the CLA attempts to write to the CPU to CLA message RAM, but the write is ignored. The arbitration scheme for the message RAMs are the same as those for the shared memories, described in the Memory Controller Module section of the *System Control and Interrupts* chapter.

The message RAMs have the following characteristics:

- CLA to CPU Message RAM:
  The following accesses are allowed:
  - CPU reads
  - CLA data reads and writes
  - CPU debug reads and writes

  The following accesses are ignored:
  - CPU writes
- CPU to CLA Message RAM:
  The following accesses are allowed:
  - CPU reads and writes
  - CLA reads
  - CPU debug reads and writes

  The following accesses are ignored:
  - CLA writes

7.4 CLA Configuration and Debug

This section discusses the steps necessary to configure and debug the CLA.

7.4.1 Building a CLA Application

The control law accelerator can be programmed in either CLA assembly code, using the instructions described in Section 7.6, or a reduced subset of the C language. CLA assembly code resides in the same project with C28x code. The only restriction is the CLA code must be in its own assembly section. This can be easily done using the **.sect** assembly directive. This does not prevent CLA and C28x code from being linked into the same memory region in the linker command file.

System and CLA initialization are performed by the main CPU. This would typically be done in C or C++ but can also include C28x assembly code. The main CPU will also copy the CLA code to the program memory and, if needed, initialize the CLA data RAM(s). Once system initialization is complete and the application begins, the CLA will service its interrupts using the CLA assembly code (or tasks). The main CPU can perform other tasks concurrently with CLA program execution.

7.4.2 Typical CLA Initialization Sequence

A typical CLA initialization sequence is performed by the main CPU as described in this section.

1. **Copy CLA code into the CLA program RAM**

   The source for the CLA code can initially reside in the Flash or a data stream from a communications peripheral or anywhere the main CPU can access it. The debugger can also be used to load code directly to the CLA program RAM during development.
2. **Initialize CLA data RAM, if necessary**

   Populate the CLA data RAM with any required data coefficients or constants.
3. **Configure the CLA registers**

   Configure the CLA registers, but keep interrupts disabled until later (leave MIER = 0):
   - **Enable the CLA peripheral clock using the assigned PCLKCRn register**

     The peripheral clock control (PCLKCRn) registers are defined in Section 3.15.4.
   - **Populate the CLA task interrupt vectors**
     - MVECT1 to MVECT8
   - **Select the task interrupt sources**

     For each task select the interrupt source in the CLA1TASKSRCSELx register. If a task is software triggered, select no interrupt.
   - **Enable IACK to start a task from software, if desired**

     To enable the IACK instruction to start a task set the MCTL[IACKE] bit. Using the IACK instruction avoids having to set and clear the EALLOW bit.
   - **Map CLA data RAM to CLA space, if necessary**
   - **Map CLA program RAM to CLA space**
4. **Initialize the PIE vector table and registers**

   When a CLA task completes, the associated interrupt in the PIE will be flagged. The CLA overflow and underflow flags also have associated interrupts within the PIE.
5. **Enable CLA tasks/interrupts**

   Set appropriate bits in the interrupt enable register (MIER) to allow the CLA to service interrupts. It should be noted that a CLA task only triggers on a level transition (a falling edge) of the configured interrupt source. If a peripheral is enabled and an interrupt fires before the CLA is configured, then the CLA will not see the interrupt edge and will not respond. To avoid this, configure the CLA before the peripherals or clear any pending peripheral interrupts before setting bits in the MIER register.
6. **Initialize other peripherals**

   Initialize any peripherals (such as ePWM, ADC, and others) that will generate interrupt triggers for enabled CLA tasks.

   The CLA is now ready to service interrupts and the message RAMs can be used to pass data between the CPU and the CLA. Mapping of the CLA program and data RAMs typically occurs only during the initialization process. If the RAM mapping needs to be changed after initialization, the CLA interrupts must be disabled and all tasks must be completed (by checking the MIRUN register) prior to modifying the RAM ownership.

7.4.3 Debugging CLA Code

Debugging the CLA code is a simple process that occurs independently of the main CPU. .

7.4.3.1 Software Breakpoint Support (MDEBUGSTOP1)

The MDEBUGSTOP1 instruction is meant to be used as a software breakpoint; the instruction on which the execution must halt is replaced with this instruction.

The MDEBUGSTOP1 and MDEBUGSTOP instructions differ in how the CLA pipeline is treated. When halted, the MDEBUGSTOP1 instruction will flush all the instructions that have already been fetched; on a single-step or run-free command, the CLA will re-fetch the same instruction which it replaced. Table 7-2 illustrates the pipeline behavior.

Table 7-2. Pipeline Behavior of the MDEBUGSTOP1 Instruction

Cycles	F1	F2	D1	D2	R1	R2	E	W	Comments
1	i1								
2	i2	i1							
3	i3	i2	i1						
4	i4	i3	i2	i1					
5	MDEBUG STOP1	i4	i3	i2	i1				
6	i6	MDEBUG STOP1	i4	i3	i2	i1			
7	i7	i6	MDEBUG STOP1	i4	i3	i2	i1		
9	i8	Flushed (MNOP)	Flushed (MNOP)	Flushed (MNOP)	i4	i3	i2	i1	CLA halted
10	i5(MDEBUGS TOP1)	Flushed (MNOP)	Flushed (MNOP)	Flushed (MNOP)	Flushed (MNOP)	i4	i3	i2	CLA step/run
11	i6	i5(MDEBUGS TOP1)	Flushed (MNOP)	Flushed (MNOP)	Flushed (MNOP)	Flushed (MNOP)	i4	i3	CLA step/run
12	i7	i6	i5(MDEBUGS TOP1)	Flushed (MNOP)	Flushed (MNOP)	Flushed (MNOP)	Flushed (MNOP)	i4	CLA step/run
13	i8	i7	i6	i5(MDEBUGS TOP1)	Flushed (MNOP)	Flushed (MNOP)	Flushed (MNOP)	Flushed (MNOP)	CLA step/run

A software breakpoint is placed at instruction i5. The instruction, i5, is then replaced with MDEBUGSTOP1. It takes 3 cycles for the MDEBUGSTOP1 to reach the D2 phase at which point the instructions i6, i7, and i8 that were previously fetched are now flushed from the pipeline. The instruction, i5, is then re-fetched and execution continues as before.

7.4.3.2 Breakpoint Support (MDEBUGSTOP)

1. **Insert a breakpoint in CLA code**

   Insert a CLA breakpoint (MDEBUGSTOP instruction) into the code where you want the CLA to halt, then rebuild and reload the code. Because the CLA does not flush its pipeline when you single-step, the MDEBUGSTOP instruction must be inserted as part of the code. The debugger cannot insert it as needed.

   If CLA breakpoints are not enabled, then the MDEBUGSTOP will be ignored and is treated as a MNOP. The MDEBUGSTOP instruction can be placed anywhere in the CLA code as long as it is not within three instructions of a MBCNDD, MCCNDD, or MRCNDD instruction. When programming in C, the user can use the __mdebugstop() intrinsic instead; the compiler will ensure that the placement of the MDEBUSTOP instruction in the generated assembly does not violate any of the pipeline restrictions.

2. **Enable CLA breakpoints**

   Enable the CLA breakpoints in the debugger. In the Code Composer Studio™ IDE, this is done by connecting to the CLA core (or tap) from the debug perspective. Breakpoints are disabled when the core is disconnected.
3. **Start the task**

   There are three ways to start the task:
   a. The peripheral can assert an interrupt,
   b. The main CPU can execute an IACK instruction, or
   c. The user can manually write to the MIFRC register in the debugger window

   When the task starts, the CLA will execute instructions until the MDEBUGSTOP is in the D2 phase of the pipeline. At this point, the CLA will halt and the pipeline will be frozen. The MPC register will reflect the address of the MDEBUGSTOP instruction.
4. **Single-step the CLA code**

   Once halted, the user can single-step the CLA code. The behavior of a CLA single-step is different than the main C28x. When issuing a CLA single-step, the pipeline is clocked only one cycle and then again frozen. On the C28x CPU, the pipeline is flushed for each single-step.

   You can also run to the next MDEBUGSTOP or to the end of the task. If another task is pending, it will automatically start when you run to the end of the task.

Note

A CLA fetch has higher priority than CPU debug reads. For this reason, it is possible for the CLA to permanently block CPU debug accesses if the CLA is executing in a loop. This might occur when initially developing CLA code due to a bug that causes an infinite loop. To avoid locking up the main CPU, the program memory will return all 0x0000 for CPU debug reads when the CLA is running. When the CLA is halted or idle then normal CPU debug read and write access to CLA program memory can be performed.

If the CLA gets caught in an infinite loop, you can use a soft or hard reset to exit the condition. A debugger reset will also exit the condition.

There are special cases that can occur when single-stepping a task such that the program counter, MPC, reaches the MSTOP instruction at the end of the task.

- **MPC halts at or after the MSTOP with a task already pending**

  If you are single-stepping or halted in "task A" and "task B" comes in before the MPC reaches the MSTOP, then "task B" will start if you continue to step through the MSTOP instruction. Basically if "task B" is pending before the MPC reaches MSTOP in "task A" then there is no issue in "task B" starting and no special action is required.
- **MPC halts at or after the MSTOP with no task pending**

  In this case you have single-stepped or halted in "task A" and the MPC has reached the MSTOP with no tasks pending. If "task B" comes in at this point, it will be flagged in the MIFR register but it may or may not start if you continue to single-step through the MSTOP instruction of "task A."

  It depends on exactly when the new task comes in. To reliably start "task B" perform a soft reset and reconfigure the MIER bits. Once this is done, you can start single-stepping "task B."

  This case can be handled slightly differently if there is control over when "task B" comes in (for example, using the IACK instruction to start the task). In this case you have single-stepped or halted in "task A" and the MPC has reached the MSTOP with no tasks pending. Before forcing "task B," run free to force the CLA out of the debug state. Once this is done you can force "task B" and continue debugging.

5. **Disable CLA breakpoints, if desired**

   In the Code Composer Studio™ IDE, you can disable the CLA breakpoints by disconnecting the CLA core in the debug perspective. Make sure to first issue a run or reset; otherwise, the CLA will be halted and no other tasks will start.

7.4.4 CLA Illegal Opcode Behavior

If the CLA fetches an opcode that does not correspond to a legal instruction, it will behave as follows:

- The CLA will halt with the illegal opcode in the D2 phase of the pipeline as if it were a breakpoint. This will occur whether CLA breakpoints are enabled or not.
- The CLA will issue the task-specific interrupt to the PIE.
- The MIRUN bit for the task will remain set.

Further single-stepping is ignored once execution halts due to an illegal op-code. To exit this situation, issue either a soft or hard reset of the CLA as described in Section 7.4.5.

7.4.5 Resetting the CLA

There may be times when you need to reset the CLA. For example, during code debug the CLA may enter an infinite loop due to a code bug. The CLA has two types of resets: hard and soft. Both of these resets can be performed by the debugger or by the main CPU.

- **Hard Reset**

  Writing a 1 to the MCTL[HARDRESET] bit will perform a hard reset of the CLA. The behavior of a hard reset is the same as a system reset (via $\overline{\text{XRS}}$ or the debugger). In this case all CLA configuration and execution registers will be set to their default state and CLA execution will halt.
- **Soft Reset**

  Writing a 1 to the MCTL[SOFTRESET] bit performs a soft reset of the CLA. If a task is executing it will halt and the associated MIRUN bit will be cleared. All bits within the interrupt enable (MIER) register will also be cleared so that no new tasks start.

7.5 Pipeline

This section describes the CLA pipeline stages and presents cases where pipeline alignment must be considered.

7.5.1 Pipeline Overview

The CLA pipeline is very similar to the C28x pipeline with eight stages:

1. **Fetch 1 (F1):** During the F1 stage the program read address is placed on the CLA program address bus.
2. **Fetch 2 (F2):** During the F2 stage the instruction is read using the CLA program data bus.
3. **Decode 1 (D1):** During D1 the instruction is decoded.
4. **Decode 2 (D2):** Generate the data read address. Changes to MAR0 and MAR1 due to post-increment using indirect addressing takes place in the D2 phase. Conditional branch decisions are also made at this stage based on the MSTF register flags.
5. **Read 1 (R1):** Place the data read address on the CLA data-read address bus. If a memory conflict exists, the R1 stage will be stalled.
6. **Read 2 (R2):** Read the data value using the CLA data read data bus.
7. **Execute (EXE):** Execute the operation. Changes to MAR0 and MAR1 due to loading an immediate value or value from memory take place in this stage.
8. **Write (W):** Place the write address and write data on the CLA write data bus. If a memory conflict exists, the W stage will be stalled.

7.5.2 CLA Pipeline Alignment

The majority of the CLA instructions do not require any special pipeline considerations. This section lists the few operations that do require special consideration.

- **Write Followed by Read**

  In both the C28x pipeline and the CLA pipeline, the read operation occurs before the write. This means that if a read operation immediately follows a write, then the read will complete first as shown in Table 7-3. In most cases this does not cause a problem since the contents of one memory location does not depend on the state of another. For accesses to peripherals where a write to one location can affect the value in another location, the code must wait for the write to complete before issuing the read as shown in Table 7-4.

  This behavior is different for the C28x CPU. For the C28x CPU, any write followed by read to the same location is protected by what is called write-followed-by-read protection. This protection automatically stalls the pipeline so that the write will complete before the read. In addition, some peripheral frames are protected such that a C28x CPU write to one location within the frame will always complete before a read to the frame. The CLA does not have this protection mechanism. Instead the code must wait to perform the read.

Table 7-3. Write Followed by Read - Read Occurs First

Instruction	F1	F2	D1	D2	R1	R2	E	W
I1 MMOV16 @Reg1, MR3	I1							
I2 MMOV16 MR2, @Reg2	I2	I1						
		I2	I1					
			I2	I1				
				I2	I1			
					I2	I1		
						I2	I1	
							I2	I1

Table 7-4. Write Followed by Read - Write Occurs First

Instruction	F1	F2	D1	D2	R1	R2	E	W
I1 MMOV16 @Reg1, MR3	I1							
I2	I2	I1						
I3	I3	I2	I1					
I4	I4	I3	I2	I1				
I5 MMOV16 MR2, @Reg2	I5	I4	I3	I2	I1			
		I5	I4	I3	I2	I1		
			I5	I4	I3	I2	I1	
				I5	I4	I3	I2	I1
					I5	I4	I3	
						I5	I4	
							I5	

- **Delayed Conditional instructions: MBCNDD, MCCNDD, and MRCNDD**

  Referring to Example 7-1, the following applies to delayed conditional instructions:

  – **I1**

    I1 is the last instruction that can effect the CNDF flags for the branch, call or return instruction. The CNDF flags are tested in the D2 phase of the pipeline. That is, a decision is made whether to branch or not when MBCNDD, MCCNDD, or MRCNDD is in the D2 phase.

  – **I2, I3, and I4**

    The three instructions preceding MBCNDD can change MSTF flags but will have no effect on whether the MBCNDD instruction branches or not. This is because the flag modification will occur after the D2 phase of the branch, call or return instruction. These three instructions must not be a MSTOP, MDEBUGSTOP, MBCNDD, MCCNDD, or MRCNDD.

  – **I5, I6, and I7**

    The three instructions following a branch, call or return are always executed irrespective of whether the condition is true or not. These instructions must not be MSTOP, MDEBUGSTOP, MBCNDD, MCCNDD or MRCNDD.

  For a more detailed description refer to the functional description for MBCNDD, MCCNDD, and MRCNDD.

Example 7-1. Code Fragment For MBCNDD, MCCNDD, or MRCNDD

```
<Instruction 1>    ; I1 Last instruction that can affect flags for
                   ;     the branch, call or return operation
<Instruction 2>    ; I2 Cannot be stop, branch, call or return
<Instruction 3>    ; I3 Cannot be stop, branch, call or return
<Instruction 4>    ; I4 Cannot be stop, branch, call or return
<branch/call/ret>  ; MBCNDD, MCCNDD or MRCNDD
                   ; I5-I7: Three instructions after are always
                   ; executed whether the branch/call or return is
                   ; taken or not
<Instruction 5>    ; I5 Cannot be stop, branch, call or return
<Instruction 6>    ; I6 Cannot be stop, branch, call or return
<Instruction 7>    ; I7 Cannot be stop, branch, call or return
<Instruction 8>    ; I8
<Instruction 9>    ; I9
....
```

- **Stop or Halting a Task: MSTOP and MDEBUGSTOP**

  The MSTOP and MDEBUGSTOP instructions cannot be placed three instructions before or after a conditional branch, call or return instruction (MBCNDD, MCCNDD, or MRCNDD). Refer to Example 7-1. To single-step through a branch/call or return, insert the MDEBUGSTOP at least four instructions back and step from there.

- **Loading MAR0 or MAR1**

  A load of auxiliary register MAR0 or MAR1 will occur in the EXE phase of the pipeline. Any post increment of MAR0 or MAR1 using indirect addressing will occur in the D2 phase of the pipeline. Referring to Example 7-2, the following applies when loading the auxiliary registers:

  – **I1 and I2**

    The two instructions following the load instruction will use the value in MAR0 or MAR1 before the update occurs.

  – **I3**

    Loading of an auxiliary register occurs in the EXE phase while updates due to post-increment addressing occur in the D2 phase. Thus I3 cannot use the auxiliary register or there will be a conflict. In the case of a conflict, the update due to address-mode post increment will win and the auxiliary register will not be updated with #_X.

  – **I4**

    Starting with the 4th instruction MAR0 or MAR1 will have the new value.

Example 7-2. Code Fragment for Loading MAR0 or MAR1

```
; Assume MAR0 is 50 and #_X is 20

   MMOVI16 MAR0, #_X ; Load MAR0 with address of X (20)
   <Instruction 1>   ; I1 Will use the old value of MAR0 (50)
   <Instruction 2>   ; I2 Will use the old value of MAR0 (50)
   <Instruction 3>   ; I3 Cannot use MAR0
   <Instruction 4>   ; I4 Will use the new value of MAR0 (20)
   <Instruction 5>   ; I5 Will use the new value of MAR0 (20
   ....
```

7.5.2.1 ADC Early Interrupt to CLA Response

The ADC can be configured to generate an early interrupt pulse before the ADC conversion completes. If this option is used to start a CLA task, the CLA will be able to read the result as soon as the conversion result is available in the ADC result register. This combination of just-in-time sampling along with the low interrupt response of the CLA enable faster system response and higher frequency control loops. The CLA task trigger to first instruction fetch interrupt latency is 4 cycles.

Timings for ADC conversions are shown in the timing diagrams of the ADC chapter. If the ADCCLK is a divided down version of the SYSCLK, the user will have to account for the conversion time in SYSCLK cycles.

From a CLA perspective, the pipeline activity is shown in Table 7-5 for an N-cycle (SYSCLK) ADC conversion. The N-2 instruction will arrive in the R2 phase just in time to read the result register. While the prior instructions will enter the R2 phase of the pipeline too soon to read the conversion, they can be efficiently used for pre-processing calculations needed by the task.

Table 7-5. ADC to CLA Early Interrupt Response

ADC Activity	CLA Activity	F1	F2	D1	D2	R1	R2	E	W
Sample									
Sample									
...									
Sample									
$Conversion_{(Cycle\ 1)}$	Interrupt Received								
$Conversion_{(Cycle\ 2)}$	Task Startup								
$Conversion_{(Cycle\ 3)}$	Task Startup								
$Conversion_{(Cycle\ 4)}$	$I_{(Cycle\ 4)}$	$I_{(Cycle\ 4)}$							
$Conversion_{(Cycle\ 5)}$	$I_{(Cycle\ 5)}$	$I_{(Cycle\ 5)}$	$I_{(Cycle\ 4)}$						
$Conversion_{(...)}$	...	...	...	...	...	...	...		
$Conversion_{(Cycle\ N-6)}$	$I_{(Cycle\ N-6)}$	$I_{(Cycle\ N-6)}$	$I_{(Cycle\ N-7)}$	$I_{(Cycle\ N-8)}$	$I_{(Cycle\ N-9)}$	$I_{(Cycle\ N-10)}$	$I_{(Cycle\ N-11)}$		
$Conversion_{(Cycle\ N-5)}$	$I_{(Cycle\ N-5)}$	$I_{(Cycle\ N-5)}$	$I_{(Cycle\ N-6)}$	$I_{(Cycle\ N-7)}$	$I_{(Cycle\ N-8)}$	$I_{(Cycle\ N-9)}$	$I_{(Cycle\ N-10)}$		
$Conversion_{(Cycle\ N-4)}$	$I_{(Cycle\ N-4)}$	$I_{(Cycle\ N-4)}$	$I_{(Cycle\ N-5)}$	$I_{(Cycle\ N-6)}$	$I_{(Cycle\ N-7)}$	$I_{(Cycle\ N-8)}$	$I_{(Cycle\ N-9)}$		
$Conversion_{(Cycle\ N-3)}$	$I_{(Cycle\ N-3)}$	$I_{(Cycle\ N-3)}$	$I_{(Cycle\ N-4)}$	$I_{(Cycle\ N-5)}$	$I_{(Cycle\ N-6)}$	$I_{(Cycle\ N-7)}$	$I_{(Cycle\ N-8)}$		
$Conversion_{(Cycle\ N-2)}$	**Read RESULT**	**Read RESULT**	$I_{(Cycle\ N-3)}$	$I_{(Cycle\ N-4)}$	$I_{(Cycle\ N-5)}$	$I_{(Cycle\ N-6)}$	$I_{(Cycle\ N-7)}$		
$Conversion_{(Cycle\ N-1)}$			**Read RESULT**	$I_{(Cycle\ N-3)}$	$I_{(Cycle\ N-4)}$	$I_{(Cycle\ N-5)}$	$I_{(Cycle\ N-6)}$		
$Conversion_{(Cycle\ N-0)}$				**Read RESULT**	$I_{(Cycle\ N-3)}$	$I_{(Cycle\ N-4)}$	$I_{(Cycle\ N-5)}$		
Conversion Complete					**Read RESULT**	$I_{(Cycle\ N-3)}$	$I_{(Cycle\ N-4)}$		
RESULT Latched						**Read RESULT**	$I_{(Cycle\ N-3)}$		
RESULT Available							**Read RESULT**		

The ADCINTCYCLE register of the ADC can be programmed by the application to adjust the generation of the interrupt pulse to align with the ADC read operation. For example, if the first instruction in the task reads the ADC and the conversion time is N SYSCLK cycles, then the delay programmed is (N-2) - 4 = N-6.

7.5.3 Parallel Instructions

Parallel instructions are single opcodes that perform two operations in parallel. The following types of parallel instructions are available: math operation in parallel with a move operation, or two math operations in parallel. Both operations complete in a single cycle and there are no special pipeline alignment requirements.

Example 7-3. Math Operation with Parallel Load

```
;  MADDF32 || MMOV32 instruction: 32-bit floating-point add with parallel move
;  MADDF32 is a 1 cycle operation
;  MMOV32 is a 1 cycle operation
         MADDF32    MR0,  MR1, #2    ; MR0 = MR1 + 2,
      || MMOV32   MR1,  @Val         ; MR1 gets the contents of Val
                                     ; <-- MMOV32 completes here (MR1 is valid)
                                     ; <-- DDF32 completes here (MR0 is valid)
         MMPYF32 MR0, MR0, MR1       ; Any instruction, can use MR1 and/or MR0
```

Example 7-4. Multiply with Parallel Add

```
;  MMPYF32 || MADDF32 instruction: 32-bit floating-point multiply with parallel add
;  MMPYF32 is a 1 cycle operation
;  MADDF32 is a 1 cycle operation
     MMPYF32 MR0, MR1, MR3        ; MR0 = MR1 * MR3
 ||  MADDF32 MR1, MR2, MR0        ; MR1 = MR2 + MR0 (Uses value of MR0 before MMPYF32)
                                  ; <-- MMPYF32 and MADDF32 complete here (MR0 and MR1 are valid)
     MMPYF32 MR1, MR1, MR0        ; Any instruction, can use MR1 and/or MR0
```

7.6 Instruction Set

This section describes the assembly language instructions of the control law accelerator. Also described are parallel operations, conditional operations, resource constraints, and addressing modes. The instructions listed here are independent from C28x and C28x+FPU instruction sets.

7.6.1 Instruction Descriptions

This section gives detailed information on the instruction set. Each instruction may present the following information:

- Operands
- Opcode
- Description
- Exceptions
- Pipeline
- Examples
- See also

The example INSTRUCTION is shown to familiarize you with the way each instruction is described. The example describes the kind of information you will find in each part of the individual instruction description and where to obtain more information. CLA instructions follow the same format as the C28x; the source operand(s) are always on the right and the destination operand(s) are on the left.

The explanations for the syntax of the operands used in the instruction descriptions for the C28x CLA are given in Table 7-6.

Table 7-6. Operand Nomenclature

Symbol	Description
#16FHi	16-bit immediate (hex or float) value that represents the upper 16-bits of an IEEE 32-bit floating-point value. Lower 16-bits of the mantissa are assumed to be zero.
#16FHiHex	16-bit immediate hex value that represents the upper 16-bits of an IEEE 32-bit floating-point value. Lower 16-bits of the mantissa are assumed to be zero.
#16FLoHex	A 16-bit immediate hex value that represents the lower 16-bits of an IEEE 32-bit floating-point value
#32Fhex	32-bit immediate value that represents an IEEE 32-bit floating-point value
#32F	Immediate float value represented in floating-point representation
#0.0	Immediate zero
#SHIFT	Immediate value of 1 to 32 used for arithmetic and logical shifts.
addr	Opcode field indicating the addressing mode
CNDF	Condition to test the flags in the MSTF register
FLAG	Selected flags from MSTF register (OR) 8 bit mask indicating which floating-point status flags to change
MAR0	auxiliary register 0
MAR1	auxiliary register 1
MARx	Either MAR0 or MAR1
mem16	16-bit memory location accessed using direct, indirect, or offset addressing modes
mem32	32-bit memory location accessed using direct, indirect, or offset addressing modes
MRa	MR0 to MR3 registers
MRb	MR0 to MR3 registers
MRc	MR0 to MR3 registers
MRd	MR0 to MR3 registers
MRe	MR0 to MR3 registers
MRf	MR0 to MR3 registers
MSTF	CLA Floating-point Status Register
shift	Opcode field indicating the number of bits to shift.

Table 7-6. Operand Nomenclature (continued)

Symbol	Description
VALUE	Flag value of 0 or 1 for selected flag (OR) 8 bit mask indicating the flag value; 0 or 1

Each instruction has a table that gives a list of the operands and a short description. Instructions always have their destination operand(s) first followed by the source operand(s).

Table 7-7. INSTRUCTION dest, source1, source2 Short Description

	Description
dest1	Description for the 1st operand for the instruction
source1	Description for the 2nd operand for the instruction
source2	Description for the 3rd operand for the instruction
Opcode	This section shows the opcode for the instruction
Description	Detailed description of the instruction execution is described. Any constraints on the operands imposed by the processor or the assembler are discussed.
Restrictions	Any constraints on the operands or use of the instruction imposed by the processor are discussed.
Pipeline	This section describes the instruction in terms of pipeline cycles as described in Section 7.5
Example	Examples of instruction execution. If applicable, register and memory values are given before and after instruction execution. Some examples are code fragments while other examples are full tasks that assume the CLA is correctly configured and the main CPU has passed it data.
Operands	Each instruction has a table that gives a list of the operands and a short description. Instructions always have their destination operand(s) first followed by the source operand(s).

7.6.2 Addressing Modes and Encoding

The CLA uses the same address to access data and registers as the main CPU. For example if the main CPU accesses an ePWM register at address 0x00 6800, then the CLA will access it using address 0x6800. Since all CLA accessible memory and registers are within the low 64k x 16 of memory, only the low 16-bits of the address are used by the CLA.

To address the CLA data memory, message RAMs and shared peripherals, the CLA supports two addressing modes:

- Direct addressing mode: Uses the address of the variable or register directly.
- Indirect addressing with 16-bit post increment. This mode uses either XAR0 or XAR1.

The CLA does not use a data page pointer or a stack pointer. The two addressing modes are encoded as shown Table 7-8.

Table 7-8. Addressing Modes

Addressing Mode	'addr' Opcode Field Encode(1)	Description
@dir	**0000**	**Direct Addressing Mode** Example 1: MMOV32 MR1, @_VarA Example 2: MMOV32 MR1, @_EPwm1Regs.CMPA.all In this case the 'mmmm mmmm mmmm mmmm' opcode field will be populated with the 16-bit address of the variable. This is the low 16-bits of the address that you would use to access the variable using the main CPU. For example @_VarA will populate the address of the variable VarA. and @_EPwm1Regs.CMPA.all will populate the address of the CMPA register.
***MAR0[#imm16]++**	**0001**	**MAR0 Indirect Addressing with 16-bit Immediate Post Increment**
***MAR1[#imm16]++**	**0010**	**MAR1 Indirect Addressing with 16-bit Immediate Post Increment** addr = MAR0 (or MAR1) Access memory using the address stored in MAR0 (or MAR1). MAR0 (or MAR1) += #imm16 Then post increment MAR0 (or MAR1) by #imm16. Example 1: MMOV32 MR0, *MAR0[2]++ Example 2: MMOV32 MR1, *MAR1[-2]++ For a post increment of 0 the assembler will accept both *MAR0 and *MAR0[0]++. The 'mmmm mmmm mmmm mmmm' opcode field will be populated with the signed 16-bit pointer offset. For example if #imm16 is 2, then the opcode field will be 0x0002. Likewise if #imm16 is -2, then the opcode field will be 0xFFFE. If addition of the 16-bit immediate causes overflow, then the value will wrap around on a 16-bit boundary.

(1) Values not shown are reserved.

Encoding for the shift fields in the MASR32, MLSR32 and MLSL32 instructions is shown in Table 7-9.

Table 7-9. Shift Field Encoding

Shift Value	'shift' Opcode Field Encode
1	0000
2	0001
3	0010
....	....
32	1111

For instructions that use MRx (where x could be 'a' through 'f') as operands, the trailing alphabet appears in the opcode as a two-bit field. For example:

```
MMPYF32 MRa, MRb, MRc ||
MADDF32 MRd, MRe, MRf
```

whose opcode is,

```
LSW: 0000 ffee ddcc bbaa
MSW: 0111 1010 0000 0000
```

The two-bit field specifies one of four working registers according to Table 7-10.

Table 7-10. Operand Encoding

Two-Bit Field	Working Register
00	MR0
01	MR1
10	MR2
11	MR3

Table 7-11 shows the condition field encoding for conditional instructions such as MNEGF, MSWAPF, MBCNDD, MCCNDD, and MRCNDD.

Table 7-11. Condition Field Encoding

Encode(1)	CNDF	Description	MSTF Flags Tested
0000	NEQ	Not equal to zero	ZF == 0
0001	EQ	Equal to zero	ZF == 1
0010	GT	Greater than zero	ZF == 0 AND NF == 0
0011	GEQ	Greater than or equal to zero	NF == 0
0100	LT	Less than zero	NF == 1
0101	LEQ	Less than or equal to zero	ZF == 1 OR NF == 1
1010	TF	Test flag set	TF == 1
1011	NTF	Test flag not set	TF == 0
1100	LU	Latched underflow	LUF == 1
1101	LV	Latched overflow	LVF == 1
1110	UNC	Unconditional	None
1111	UNCF(2)	Unconditional with flag modification	None

(1) Values not shown are reserved.
(2) This is the default operation if no CNDF field is specified. This condition will allow the ZF and NF flags to be modified when a conditional operation is executed. All other conditions will not modify these flags.

7.6.3 Instructions

The instructions are listed alphabetically.

MABSF32 MRa, MRb

32-Bit Floating-Point Absolute Value

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1110 0010 0000
```

Description

The absolute value of MRb is loaded into MRa. Only the sign bit of the operand is modified by the MABSF32 instruction.

```
if (MRb < 0) {MRa = -MRb};
        else {MRa =  MRb};
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified as follows:

```
NF = 0;
ZF = 0;
if ( MRa(30:23) == 0) ZF = 1;
```

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ MR0, #-2.0 ; MR0 = -2.0 (0xC0000000)
MABSF32 MR0, MR0  ; MR0 = 2.0 (0x40000000), ZF = NF = 0
MMOVIZ MR0, #5.0  ; MR0 = 5.0 (0x40A00000)
MABSF32 MR0, MR0  ; MR0 = 5.0 (0x40A00000), ZF = NF = 0
MMOVIZ MR0, #0.0  ; MR0 = 0.0
MABSF32 MR0, MR0  ; MR0 = 0.0 ZF = 1, NF = 0
```

See also

MNEGF32 MRa, MRb {, CNDF}

MADD32 MRa, MRb, MRc

32-Bit Integer Add

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point destination register (MR0 to MR3)
MRc	CLA floating-point destination register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 000cc bbaa
MSW: 0111 1110 1100 0000
```

Description

32-bit integer addition of MRb and MRc.

```
MRa(31:0) = MRb(31:0) + MRc(31:0);
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified based on the integer results of the operation.

```
NF = MRa(31);
ZF = 0;
if(MRa(31:0) == 0) { ZF = 1; };
```

Pipeline

This is a single-cycle instruction.

Example

```
; Given A = (int32)1
;       B = (int32)2
;       C = (int32)-7
;
; Calculate Y2 = A + B + C
;
_Cla1Task1:
       MMOV32 MR0, @_A      ; MR0 = 1 (0x00000001)
       MMOV32 MR1, @_B      ; MR1 = 2 (0x00000002)
       MMOV32 MR2, @_C      ; MR2 = -7 (0xFFFFFFF9)
       MADD32 MR3, MR0, MR1 ; A + B
       MADD32 MR3, MR2, MR3 ; A + B + C = -4 (0xFFFFFFFC)
       MMOV32 @_y2, MR3     ; Store y2
       MSTOP                ; end of task
```

See also

MAND32 MRa, MRb, MRc
MASR32 MRa, #SHIFT
MLSL32 MRa, #SHIFT
MLSR32 MRa, #SHIFT
MOR32 MRa, MRb, MRc
MXOR32 MRa, MRb, MRc
MSUB32 MRa, MRb, MRc

MADDF32 MRa, #16FHi, MRb

32-Bit Floating-Point Addition

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
#16FHi	A 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0.
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: IIII IIII IIII IIII
MSW: 0111 0111 1100 bbaa
```

Description

Add MRb to the floating-point value represented by the immediate operand. Store the result of the addition in MRa.

#16FHi is a 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0. #16FHi is most useful for representing constants where the lowest 16-bits of the mantissa are 0. Some examples are 2.0 (0x40000000), 4.0 (0x40800000), 0.5 (0x3F000000), and -1.5 (0xBFC00000). The assembler will accept either a hex or float as the immediate value. That is, the value -1.5 can be represented as #-1.5 or #0xBFC0.

```
MRa = MRb + #16FHi:0;
```

This instruction can also be written as MADDF32 MRa, MRb, #16FHi.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MADDF32 generates an underflow condition.
- LVF = 1 if MADDF32 generates an overflow condition.

Pipeline

This is a single-cycle instruction.

Example

```
; Add to MR1 the value 2.0 in 32-bit floating-point format
; Store the result in MR0
  MADDF32 MR0, #2.0, MR1    ; MR0 = 2.0 + MR1
; Add to MR3 the value -2.5 in 32-bit floating-point format
; Store the result in MR2
  MADDF32 MR2, #-2.5, MR3   ; MR2 = -2.5 + MR3
; Add to MR3 the value 0x3FC00000 (1.5)
; Store the result in MR3
  MADDF32 MR3, #0x3FC0, MR3 ; MR3 = 1.5 + MR3
```

See also

MADDF32 MRa, MRb, #16FHi
MADDF32 MRa, MRb, MRc
MADDF32 MRd, MRe, MRf || MMOV32 MRa, mem32
MADDF32 MRd, MRe, MRf || MMOV32 mem32, MRa
MMPYF32 MRa, MRb, MRc || MADDF32 MRd, MRe, MRf

MADDF32 MRa, MRb, #16FHi

32-Bit Floating-Point Addition

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)
#16FHi	A 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0.

Opcode

```
LSW: IIII IIII IIII IIII
MSW: 0111 0111 1100 bbaa
```

Description

Add MRb to the floating-point value represented by the immediate operand. Store the result of the addition in MRa.

#16FHi is a 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0. #16FHi is most useful for representing constants where the lowest 16-bits of the mantissa are 0. Some examples are 2.0 (0x40000000), 4.0 (0x40800000), 0.5 (0x3F000000), and -1.5 (0xBFC00000). The assembler will accept either a hex or float as the immediate value. That is, the value -1.5 can be represented as #-1.5 or #0xBFC0.

```
MRa = MRb + #16FHi:0;
```

This instruction can also be written as MADDF32 MRa, #16FHi, MRb.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MADDF32 generates an underflow condition.
- LVF = 1 if MADDF32 generates an overflow condition.

Pipeline

This is a single-cycle instruction.

MADDF32 MRa, MRb, #16FHi (continued)

32-Bit Floating-Point Addition

Example 1

```
; X is an array of 32-bit floating-point values
; Find the maximum value in an array X
; and store it in Result
;
_Cla1Task1:
    MMOVI16    MAR1,#_X          ; Start address
    MUI16TOF32 MR0, @_len        ; Length of the array
    MNOP                         ; delay for MAR1 load
    MNOP                         ; delay for MAR1 load
    MMOV32     MR1, *MAR1[2]++   ; MR1 = X0
LOOP
    MMOV32     MR2, *MAR1[2]++   ; MR2 = next element
    MMAXF32    MR1,  MR2         ; MR1 = MAX(MR1, MR2)
    MADDF32    MR0, MR0, #-1.0   ; Decrement the counter
    MCMPF32    MR0 #0.0          ; Set/clear flags for MBCNDD
    MNOP
    MNOP
    MNOP
    MBCNDD LOOP, NEQ             ; Branch if not equal to zero
    MMOV32 @_Result, MR1         ; Always executed
    MNOP                         ; Always executed
    MNOP                         ; Always executed
    MSTOP                        ; End of task
```

Example 2

```
; Show the basic operation of MADDF32
;
; Add to MR1 the value 2.0 in 32-bit floating-point format
; Store the result in MR0
    MADDF32 MR0, MR1, #2.0    ; MR0 = MR1 + 2.0
; Add to MR3 the value -2.5 in 32-bit floating-point format
; Store the result in MR2
    MADDF32 MR2, MR3, #-2.5   ; MR2 = MR3 + (-2.5)
; Add to MR0 the value 0x3FC00000 (1.5)
; Store the result in MR0
   MADDF32 MR0, MR0, #0x3FC0  ; MR0 = MR0 + 1.5
```

See also

MADDF32 MRa, #16FHi, MRb
MADDF32 MRa, MRb, MRc
MADDF32 MRd, MRe, MRf || MMOV32 MRa, mem32
MADDF32 MRd, MRe, MRf || MMOV32 mem32, MRa
MMPYF32 MRa, MRb, MRc || MADDF32 MRd, MRe, MRf

MADDF32 MRa, MRb, MRc

32-Bit Floating-Point Addition

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)
MRc	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 000 0000 00cc bbaa
MSW: 0111 1100 0010 0000
```

Description

Add the contents of MRc to the contents of MRb and load the result into MRa.

```
MRa = MRb + MRc;
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MADDF32 generates an underflow condition.
- LVF = 1 if MADDF32 generates an overflow condition.

Pipeline

This is a single-cycle instruction.

Example

```
; Given M1, X1 and B1 are 32-bit floating point numbers
; Calculate Y1 = M1*X1+B1
;
_Cla1Task1:
   MMOV32 MR0,@M1       ; Load MR0 with M1
   MMOV32 MR1,@X1       ; Load MR1 with X1
   MMPYF32 MR1,MR1,MR0  ; Multiply M1*X1
|| MMOV32 MR0,@B1       ; and in parallel load MR0 with B1
   MADDF32 MR1,MR1,MR0  ; Add M*X1 to B1 and store in MR1
   MMOV32 @Y1,MR1       ; Store the result
   MSTOP                ; end of task
```

See also

MADDF32 MRa, #16FHi, MRb
MADDF32 MRa, MRb, #16FHi
MADDF32 MRd, MRe, MRf || MMOV32 MRa, mem32
MADDF32 MRd, MRe, MRf || MMOV32 mem32, MRa
MMPYF32 MRa, MRb, MRc || MADDF32 MRd, MRe, MRf

MADDF32 MRd, MRe, MRf||MMOV32 mem32, MRa

32-Bit Floating-Point Addition with Parallel Move

Operands

MRd	CLA floating-point destination register for the MADDF32 (MR0 to MR3)
MRe	CLA floating-point source register for the MADDF32 (MR0 to MR3)
MRf	CLA floating-point source register for the MADDF32 (MR0 to MR3)
mem32	32-bit memory location accessed using one of the available addressing modes. This will be the destination of the MMOV32.
MRa	CLA floating-point source register for the MMOV32 (MR0 to MR3)

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0101 ffee ddaa addr
```

Description

Perform an MADDF32 and a MMOV32 in parallel. Add MRf to the contents of MRe and store the result in MRd. In parallel move the contents of MRa to the 32-bit location mem32.

```
MRd = MRe + MRf;
[mem32] = MRa;
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MADDF32 generates an underflow condition.
- LVF = 1 if MADDF32 generates an overflow condition.

Pipeline

Both MADDF32 and MMOV32 complete in a single cycle.

Example

```
; Given A, B and C are 32-bit floating-point numbers
; Calculate Y2 = (A * B)
;           Y3 = (A * B) + C
;
_Cla1Task2:
   MMOV32   MR0, @_A         ; Load MR0 with A
   MMOV32   MR1, @_B         ; Load MR1 with B
   MMPYF32  MR1, MR1, MR0    ; Multiply A*B
|| MMOV32   MR0, @_C         ;     and in parallel load MR0 with C
   MADDF32  MR1, MR1, MR0    ; Add (A*B) to C
|| MMOV32   @_Y2, MR1        ;     and in parallel store A*B
   MMOV32   @_Y3, MR1        ; Store the A*B + C
   MSTOP                     ; end of task
```

See also

MADDF32 MRa, #16FHi, MRb
MADDF32 MRa, MRb, #16FHi
MADDF32 MRa, MRb, MRc
MMPYF32 MRa, MRb, MRc || MADDF32 MRd, MRe, MRf
MADDF32 MRd, MRe, MRf || MMOV32 MRa, mem32

MADDF32 MRd, MRe, MRf ||MMOV32 MRa, mem32

32-Bit Floating-Point Addition with Parallel Move

Operands

MRd	CLA floating-point destination register for the MADDF32 (MR0 to MR3). MRd cannot be the same register as MRa.
MRe	CLA floating-point source register for the MADDF32 (MR0 to MR3)
MRf	CLA floating-point source register for the MADDF32 (MR0 to MR3)
MRa	CLA floating-point destination register for the MMOV32 (MR0 to MR3). MRa cannot be the same register as MRd.
mem32	32-bit memory location accessed using one of the available addressing modes. This is the source for the MMOV32.

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0001 ffee ddaa addr
```

Description

Perform an MADDF32 and a MMOV32 operation in parallel. Add MRf to the contents of MRe and store the result in MRd. In parallel move the contents of the 32-bit location mem32 to MRa.

```
MRd = MRe + MRf;
MRa = [mem32];
```

Restrictions

The destination register for the MADDF32 and the MMOV32 must be unique. That is, MRa and MRd cannot be the same register.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MADDF32 generates an underflow condition.
- LVF = 1 if MADDF32 generates an overflow condition.

The MMOV32 Instruction will set the NF and ZF flags as follows:

```
NF = MRa(31);
ZF = 0;
if(MRa(30:23) == 0) { ZF = 1; NF = 0; };
```

Pipeline

The MADDF32 and the MMOV32 both complete in a single cycle.

MADDF32 MRd, MRe, MRf ||MMOV32 MRa, mem32 (continued)

32-Bit Floating-Point Addition with Parallel Move

Example 1

```
; Given A, B and C are 32-bit floating-point numbers
; Calculate Y1 = A + 4B
;           Y2 = A + C
;
_Cla1Task1:
    MMOV32 MR0, @A          ; Load MR0 with A
    MMOV32 MR1, @B          ; Load MR1 with B
    MMPYF32 MR1, MR1, #4.0 ; Multiply 4 * B
||  MMOV32 MR2, @C            and in parallel load C
    MADDF32 MR3, MR0, MR1   ; Add A + 4B
    MADDF32 MR3, MR0, MR2   ; Add A + C
||  MMOV32 @Y1, MR3         ; and in parallel store A+4B
    MMOV32 @Y2, MR3         ; store A + C MSTOP
                            ; end of task
```

Example 2

```
; Given A, B and C are 32-bit floating-point numbers
; Calculate Y3 = (A + B)
;           Y4 = (A + B) * C
;
_Cla1Task2:
     MMOV32 MR0, @A          ; Load MR0 with A
     MMOV32 MR1, @B          ; Load MR1 with B
     MADDF32 MR1, MR1, MR0   ; Add A+B
||   MMOV32 MR0, @C          ; and in parallel load MR0 with C
     MMPYF32 MR1, MR1, MR0   ; Multiply (A+B) by C
||   MMOV32 @Y3, MR1         ; and in parallel store A+B
     MMOV32 @Y4, MR1         ; Store the (A+B) * C
     MSTOP                   ; end of task
```

See also

MADDF32 MRa, #16FHi, MRb
MADDF32 MRa, MRb, #16FHi
MADDF32 MRa, MRb, MRc
MADDF32 MRd, MRe, MRf || MMOV32 mem32, MRa
MMPYF32 MRa, MRb, MRc || MADDF32 MRd, MRe, MRf

MAND32 MRa, MRb, MRc

Bitwise AND

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)
MRc	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 00cc bbaa
MSW: 0111 1100 0110 0000
```

Description

Bitwise AND of MRb with MRc.

```
MRa(31:0) = MRb(31:0) AND MRc(31:0);
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified based on the integer results of the operation.

```
NF = MRa(31);
ZF = 0;
if(MRa(31:0) == 0) { ZF = 1; }
```

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ   MR0,    #0x5555  ; MR0 = 0x5555AAAA
MMOVXI   MR0,    #0xAAAA
MMOVIZ   MR1,    #0x5432  ; MR1 = 0x5432FEDC
MMOVXI   MR1,    #0xFEDC
; 0101 AND 0101 = 0101 (5)
; 0101 AND 0100 = 0100 (4)
; 0101 AND 0011 = 0001 (1)
; 0101 AND 0010 = 0000 (0)
; 1010 AND 1111 = 1010 (A)
; 1010 AND 1110 = 1010 (A)
; 1010 AND 1101 = 1000 (8)
; 1010 AND 1100 = 1000 (8)
MAND32 MR2, MR1, MR0      ; MR3 = 0x5410AA88
```

See also

MADD32 MRa, MRb, MRc
MASR32 MRa, #SHIFT
MLSL32 MRa, #SHIFT
MLSR32 MRa, #SHIFT
MOR32 MRa, MRb, MRc
MXOR32 MRa, MRb, MRc
MSUB32 MRa, MRb, MRc

MASR32 MRa, #SHIFT

Arithmetic Shift Right

Operands

MRa	CLA floating-point source/destination register (MR0 to MR3)
#SHIFT	Number of bits to shift (1 to 32)

Opcode

```
LSW: 0000 0000 0shi ftaa
MSW: 0111 1011 0100 0000
```

Description

Arithmetic shift right of MRa by the number of bits indicated. The number of bits can be 1 to 32.

```
MARa(31:0) = Arithmetic Shift(MARa(31:0) by #SHIFT bits);
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified based on the integer results of the operation.

```
NF = MRa(31);
ZF = 0;
if(MRa(31:0) == 0) { ZF = 1; }
```

Pipeline

This is a single-cycle instruction.

Example

```
; Given m2 = (int32)32
;       x2 = (int32)64
;       b2 = (int32)-128
;
; Calculate
;       m2 = m2/2
;       x2 = x2/4
;       b2 = b2/8
;
_Cla1Task2:
   MMOV32 MR0, @_m2 ; MR0 = 32 (0x00000020)
   MMOV32 MR1, @_x2 ; MR1 = 64 (0x00000040)
   MMOV32 MR2, @_b2 ; MR2 = -128 (0xFFFFFF80)
   MASR32 MR0, #1   ; MR0 = 16 (0x00000010)
   MASR32 MR1, #2   ; MR1 = 16 (0x00000010)
   MASR32 MR2, #3   ; MR2 = -16 (0xFFFFFFF0)
   MMOV32 @_m2, MR0 ; store results
   MMOV32 @_x2, MR1
   MMOV32 @_b2, MR2
   MSTOP ; end of task
```

See also

MADD32 MRa, MRb, MRc
MAND32 MRa, MRb, MRc
MLSL32 MRa, #SHIFT
MLSR32 MRa, #SHIFT
MOR32 MRa, MRb, MRc
MXOR32 MRa, MRb, MRc
MSUB32 MRa, MRb, MRc

MBCNDD 16BitDest {, CNDF}

Branch Conditional Delayed

Operands

16BitDest	16-bit destination if condition is true
CNDF	Optional condition tested

Opcode

```
LSW: dest dest dest dest
MSW: 0111 1001 1000 cndf
```

Description

If the specified condition is true, then branch by adding the signed 16BitDest value to the MPC value. Otherwise, continue without branching. If the address overflows, it wraps around. Therefore a value of "0xFFFE" will put the MPC back to the MBCNDD instruction.

Please refer to the pipeline section for important information regarding this instruction.

```
if (CNDF == TRUE) MPC += 16BitDest;
```

CNDF is one of the following conditions:

Encode(1)	CNDF	Description	MSTF Flags Tested
0000	NEQ	Not equal to zero	ZF == 0
0001	EQ	Equal to zero	ZF == 1
0010	GT	Greater than zero	ZF == 0 AND NF == 0
0011	GEQ	Greater than or equal to zero	NF == 0
0100	LT	Less than zero	NF == 1
0101	LEQ	Less than or equal to zero	ZF == 1 OR NF == 1
1010	TF	Test flag set	TF == 1
1011	NTF	Test flag not set	TF == 0
1100	LU	Latched underflow	LUF == 1
1101	LV	Latched overflow	LVF == 1
1110	UNC	Unconditional	None
1111	UNCF(2)	Unconditional with flag modification	None

(1) Values not shown are reserved.
(2) This is the default operation, if no CNDF field is specified. This condition will allow the ZF and NF flags to be modified when a conditional operation is executed. All other conditions will not modify these flags.

Restrictions

The MBCNDD instruction is not allowed three instructions before or after a MBCNDD, MCCNDD or MRCNDD instruction. Refer to the pipeline section for more information.

Flags

This instruction does not modify flags in the MSTF register.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

MBCNDD 16BitDest {, CNDF} (continued)

Branch Conditional Delayed

Pipeline

The MBCNDD instruction by itself is a single-cycle instruction. As shown in Table 7-12 for each branch 6 instruction slots are executed; three before the branch instruction (I2-I4) and three after the branch instruction (I5-I7). The total number of cycles for a branch taken or not taken depends on the usage of these slots. That is, the number of cycles depends on how many slots are filled with a MNOP as well as which slots are filled. The effective number of cycles for a branch can, therefore, range from 1 to 7 cycles. The number of cycles for a branch taken may not be the same as for a branch not taken.

Referring to Table 7-12 and Table 7-13, the instructions before and after MBCNDD have the following properties:

- **I1**
  - I1 is the last instruction that can effect the CNDF flags for the MBCNDD instruction. The CNDF flags are tested in the D2 phase of the pipeline. That is, a decision is made whether to branch or not when MBCNDD is in the D2 phase.
  - There are no restrictions on the type of instruction for I1.
- **I2, I3 and I4**
  - The three instructions proceeding MBCNDD can change MSTF flags but will have no effect on whether the MBCNDD instruction branches or not. This is because the flag modification will occur after the D2 phase of the MBCNDD instruction.
  - These instructions must not be the following: MSTOP, MDEBUGSTOP, MBCNDD, MCCNDD or MRCNDD.
- **I5, I6 and I7**
  - The three instructions following MBCNDD are always executed irrespective of whether the branch is taken or not.
  - These instructions must not be the following: MSTOP, MDEBUGSTOP, MBCNDD, MCCNDD or MRCNDD.

```
<Instruction 1>   ; I1 Last instruction that can affect flags for
                  ; the MBCNDD operation
<Instruction 2>   ; I2 Cannot be stop, branch, call or return
<Instruction 3>   ; I3 Cannot be stop, branch, call or return
<Instruction 4>   ; I4 Cannot be stop, branch, call or return
MBCNDD _Skip, NEQ ; Branch to Skip if not eqal to zero
                  ; Three instructions after MBCNDD are always
                  ; executed whether the branch is taken or not
<Instruction 5>   ; I5 Cannot be stop, branch, call or return
<Instruction 6>   ; I6 Cannot be stop, branch, call or return
<Instruction 7>   ; I7 Cannot be stop, branch, call or return
<Instruction 8>   ; I8
<Instruction 9>   ; I9
....
_Skip:
 <Destination 1>  ; d1 Can be any instruction
 <Destination 2>  ; d2
 <Destination 3>  ; d3
....
....
MSTOP
....
```

MBCNDD 16BitDest {, CNDF} (continued)

Branch Conditional Delayed

Table 7-12. Pipeline Activity For MBCNDD, Branch Not Taken

Instruction	F1	F2	D1	D2	R1	R2	E	W
I1	I1							
I2	I2	I1						
I3	I3	I2	I1					
I4	I4	I3	I2	I1				
MBCNDD	MBCNDD	I4	I3	I2	I1			
I5	I5	MBCNDD	I4	I3	I2	I1		
I6	I6	I5	MBCNDD	I4	I3	I2	I1	
I7	I7	I6	I5	MBCNDD	I4	I3	I2	
I8	I8	I7	I6	I5	-	I4	I3	
I9	I9	I8	I7	I6	I5	-	I4	
I10	I10	I9	I8	I7	I6	I5	-	
		I10	I9	I8	I7	I6	I5	
			I10	I9	I8	I7	I6	
				I10	I9	I8	I7	
					I10	I9	I8	
						I10	I9	
							I10	

Table 7-13. Pipeline Activity For MBCNDD, Branch Taken

Instruction	F1	F2	D1	D2	R1	R2	E	W
I1	I1							
I2	I2	I1						
I3	I3	I2	I1					
I4	I4	I3	I2	I1				
MBCNDD	MBCNDD	I4	I3	I2	I1			
I5	I5	MBCNDD	I4	I3	I2	I1		
I6	I6	I5	MBCNDD	I4	I3	I2	I1	
I7	I7	I6	I5	MBCNDD	I4	I3	I2	
d1	d1	I7	I6	I5	-	I4	I3	
d2	d2	d1	I7	I6	I5	-	I4	
d3	d3	d2	d1	I7	I6	I5	-	
		d3	d2	d1	I7	I6	I5	
			d3	d2	d1	I7	I6	
				d3	d2	d1	I7	
					d3	d2	d1	
						d3	d2	
							d3	

MBCNDD 16BitDest {, CNDF} (continued)

Branch Conditional Delayed

Example 1

```
; if (State == 0.1)
; RampState = RampState || RAMPMASK
; else if (State == 0.01)
; CoastState = CoastState || COASTMASK
; else
; SteadyState = SteadyState || STEADYMASK
;
_Cla1Task1:
 MMOV32 MR0, @State
 MCMPF32 MR0, #0.1        ; Affects flags for 1st MBCNDD (A)
 MNOP
 MNOP
 MNOP
 MBCNDD Skip1, NEQ        ; (A) If State != 0.1, go to Skip1
 MNOP ; Always executed
 MNOP ; Always executed
 MNOP ; Always executed
 MMOV32 MR1, @RampState   ; Execute if (A) branch not taken
 MMOVXI MR2, #RAMPMASK    ; Execute if (A) branch not taken
 MOR32 MR1, MR2           ; Execute if (A) branch not taken
 MMOV32 @RampState, MR1   ; Execute if (A) branch not taken
 MSTOP                    ; end of task if (A) branch not taken
Skip1:
 MCMPF32 MR0,#0.01        ; Affects flags for 2nd MBCNDD (B)
 MNOP
 MNOP
 MNOP
 MBCNDD Skip2,NEQ         ; (B) If State != 0.01, go to Skip2
 MNOP ; Always executed
 MNOP ; Always executed
 MNOP ; Always executed
 MMOV32 MR1, @CoastState  ; Execute if (B) branch not taken
 MMOVXI MR2, #COASTMASK   ; Execute if (B) branch not taken
 MOR32 MR1, MR2           ; Execute if (B) branch not taken
 MMOV32 @CoastState, MR1  ; Execute if (B) branch not taken
 MSTOP
Skip2:
 MMOV32 MR3, @SteadyState ; Executed if (B) branch taken
 MMOVXI MR2, #STEADYMASK  ; Executed if (B) branch taken
 MOR32 MR3, MR2           ; Executed if (B) branch taken
 MMOV32 @SteadyState, MR3 ; Executed if (B) branch taken
 MSTOP
```

MBCNDD 16BitDest {, CNDF} (continued)

Branch Conditional Delayed

Example 2

```
; This example is the same as Example 1, except
; the code is optimized to take advantage of delay slots
;
; if (State == 0.1)
; RampState = RampState || RAMPMASK
; else if (State == 0.01)
; CoastState = CoastState || COASTMASK
; else
; SteadyState = SteadyState || STEADYMASK
;
_Cla1Task2:
 MMOV32 MR0, @State
 MCMPF32 MR0, #0.1          ; Affects flags for 1st MBCNDD (A)
 MCMPF32 MR0, #0.01         ; Check used by 2nd MBCNDD (B)
 MTESTTF EQ                 ; Store EQ flag in TF for 2nd MBCNDD (B)
 MNOP
 MBCNDD Skip1, NEQ          ; (A) If State != 0.1, go to Skip1
 MMOV32 MR1, @RampState     ; Always executed
 MMOVXI MR2, #RAMPMASK      ; Always executed
 MOR32 MR1, MR2             ; Always executed
 MMOV32 @RampState, MR1     ; Execute if (A) branch not taken
 MSTOP                      ; end of task if (A) branch not taken
Skip1:
 MMOV32 MR3, @SteadyState
 MMOVXI MR2, #STEADYMASK
 MOR32 MR3, MR2
 MBCNDD Skip2, NTF          ; (B) if State != .01, go to Skip2
 MMOV32 MR1, @CoastState    ; Always executed
 MMOVXI MR2, #COASTMASK     ; Always executed
 MOR32 MR1, MR2             ; Always executed
 MMOV32 @CoastState, MR1    ; Execute if (B) branch not taken
 MSTOP                      ; end of task if (B) branch not taken
Skip2:
 MMOV32 @SteadyState, MR3   ; Executed if (B) branch taken
 MSTOP
```

See also

MCCNDD 16BitDest, CNDF
MRCNDD CNDF

MCCNDD 16BitDest {, CNDF}

Call Conditional Delayed

Operands

16BitDest	16-bit destination if condition is true
CNDF	Optional condition to be tested

Opcode

```
LSW: dest dest dest dest
MSW: 0111 1001 1001 cndf
```

Description

If the specified condition is true, then store the return address in the RPC field of MSTF and make the call by adding the signed 16BitDest value to the MPC value. Otherwise, continue code execution without making the call. If the address overflows, it wraps around. Therefore a value of "0xFFFE" will put the MPC back to the MCCNDD instruction.

Please refer to the pipeline section for important information regarding this instruction.

```
if (CNDF == TRUE)
{
    RPC = return address;
    MPC += 16BitDest;
};
```

CNDF is one of the following conditions:

Encode[1]	CNDF	Description	MSTF Flags Tested
0000	NEQ	Not equal to zero	ZF == 0
0001	EQ	Equal to zero	ZF == 1
0010	GT	Greater than zero	ZF == 0 AND NF == 0
0011	GEQ	Greater than or equal to zero	NF == 0
0100	LT	Less than zero	NF == 1
0101	LEQ	Less than or equal to zero	ZF == 1 OR NF == 1
1010	TF	Test flag set	TF == 1
1011	NTF	Test flag not set	TF == 0
1100	LU	Latched underflow	LUF == 1
1101	LV	Latched overflow	LVF == 1
1110	UNC	Unconditional	None
1111	UNCF[2]	Unconditional with flag modification	None

(1) Values not shown are reserved.
(2) This is the default operation if no CNDF field is specified. This condition will allow the ZF and NF flags to be modified when a conditional operation is executed. All other conditions will not modify these flags.

Restrictions

The MCCNDD instruction is not allowed three instructions before or after a MBCNDD, MCCNDD, or MRCNDD instruction. Refer to the Pipeline section for more details.

Flags

This instruction does not modify flags in the MSTF register.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

MCCNDD 16BitDest {, CNDF} (continued)

Call Conditional Delayed

Pipeline

The MCCNDD instruction by itself is a single-cycle instruction. As shown in Table 7-14, for each call 6 instruction slots are executed; three before the call instruction (I2-I4) and three after the call instruction (I5-I7). The total number of cycles for a call taken or not taken depends on the usage of these slots. That is, the number of cycles depends on how many slots are filled with a MNOP as well as which slots are filled. The effective number of cycles for a call can, therefore, range from 1 to 7 cycles. The number of cycles for a call taken may not be the same as for a call not taken.

Referring to the following code fragment and the pipeline diagrams in Table 7-14 and Table 7-15, the instructions before and after MCCNDD have the following properties:

- **I1**
  - I1 is the last instruction that can effect the CNDF flags for the MCCNDD instruction. The CNDF flags are tested in the D2 phase of the pipeline. That is, a decision is made whether to branch or not when MCCNDD is in the D2 phase.
  - There are no restrictions on the type of instruction for I1.
- **I2, I3 and I4**
  - The three instructions proceeding MCCNDD can change MSTF flags but will have no effect on whether the MCCNDD instruction makes the call or not. This is because the flag modification will occur after the D2 phase of the MCCNDD instruction.
  - These instructions must not be the following: MSTOP, MDEBUGSTOP, MBCNDD, MCCNDD or MRCNDD.
- **I5, I6 and I7**
  - The three instructions following MBCNDD are always executed irrespective of whether the branch is taken or not.
  - These instructions must not be the following: MSTOP, MDEBUGSTOP, MBCNDD, MCCNDD or MRCNDD.

```
<Instruction 1>   ; I1 Last instruction that can affect flags for
                  ;    the MCCNDD operation
<Instruction 2>   ; I2 Cannot be stop, branch, call or return
<Instruction 3>   ; I3 Cannot be stop, branch, call or return
<Instruction 4>   ; I4 Cannot be stop, branch, call or return
MCCNDD _func, NEQ ; Call to func if not eqal to zero
                  ; Three instructions after MCCNDD are always
                  ; executed whether the call is taken or not
<Instruction 5>   ; I5 Cannot be stop, branch, call or return
<Instruction 6>   ; I6 Cannot be stop, branch, call or return
<Instruction 7>   ; I7 Cannot be stop, branch, call or return
<Instruction 8>   ; I8 The address of this instruction is saved
                  ;    in the RPC field of the MSTF register.
                  ;    Upon return this value is loaded into MPC
                  ;    and fetching continues from this point.
<Instruction 9>   ; I9
....
_func:
<Destination 1>   ; d1 Can be any instruction
<Destination 2>   ; d2
<Destination 3>   ; d3
<Destination 4>   ; d4 Last instruction that can affect flags for
                  ;    the MRCNDD operation
<Destination 5>   ; d5 Cannot be stop, branch, call or return
<Destination 6>   ; d6 Cannot be stop, branch, call or return
<Destination 7>   ; d7 Cannot be stop, branch, call or return
MRCNDD UNC        ; Return to <Instruction 8>, unconditional
                  ; Three instructions after MRCNDD are always
                  ; executed whether the return is taken or not
<Destination 8>   ; d8 Cannot be stop, branch, call or return
<Destination 9>   ; d9 Cannot be stop, branch, call or return
```

MCCNDD 16BitDest {, CNDF} (continued)

Call Conditional Delayed

```
<Destination 10>  ; d10 Cannot be stop, branch, call or return
<Destination 11>  ; d11
....
MSTOP
```

Table 7-14. Pipeline Activity For MCCNDD, Call Not Taken

Instruction	F1	F2	D1	D2	R1	R2	E	W
I1	I1							
I2	I2	I1						
I3	I3	I2	I1					
I4	I4	I3	I2	I1				
MCCNDD	MCCNDD	I4	I3	I2	I1			
I5	I5	MCCNDD	I4	I3	I2	I1		
I6	I6	I5	MCCNDD	I4	I3	I2	I1	
I7	I7	I6	I5	MCCNDD	I4	I3	I2	
I8	I8	I7	I6	I5	-	I4	I3	
I9	I9	I8	I7	I6	I5	-	I4	
I10	I10	I9	I8	I7	I6	I5	-	
etc		I10	I9	I8	I7	I6	I5	
....			I10	I9	I8	I7	I6	
....				I10	I9	I8	I7	
....					I10	I9	I8	
						I10	I9	
							I10	

Table 7-15. Pipeline Activity For MCCNDD, Call Taken

Instruction	F1	F2	D1	D2	R1	R2	E	W
I1	I1							
I2	I2	I1						
I3	I3	I2	I1					
I4	I4	I3	I2	I1				
MCCNDD	MCCNDD	I4	I3	I2	I1			
I5	I5	MCCNDD	I4	I3	I2	I1		
I6	I6	I5	MCCNDD	I4	I3	I2	I1	
I7 (1)	I7	I6	I5	MCCNDD	I4	I3	I2	
d1	d1	I7	I6	I5	-	I4	I3	
d2	d2	d1	I7	I6	I5	-	I4	
d3	d3	d2	d1	I7	I6	I5	-	
etc		d3	d2	d1	I7	I6	I5	
....			d3	d2	d1	I7	I6	
....				d3	d2	d1	I7	
....					d3	d2	d1	
						d3	d2	
							d3	

(1) The RPC value in the MSTF register will point to the instruction following I7 (instruction I8).

See also

MBCNDD #16BitDest, CNDF
MMOV32 mem32, MSTF
MMOV32 MSTF, mem32

MCCNDD 16BitDest {, CNDF} (continued)

Call Conditional Delayed

MRCNDD CNDF

MCLRC BGINTM

Clear Background Task Interrupt Mask

Operands

None	This instruction does not have any operands

Opcode

```
LSW: 0000 0000 0000 0000
MSW: 0111 1111 0111 0000
```

Description

This instruction will clear the background task interrupt mask (BGINTM) bit in the MSTSBGRND register, allowing any code thereafter to be interrupted by a higher priority task. This instruction clears the BGINTM bit at the end of its D2 phase.

Note: This instruction does not require the MEALLOW bit to be asserted before or deasserted after clearing BGINTM.

Flags

This instruction does not modify flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
MCLRC BGINTM       ; Allow the background task to be
                   ; interrupted by clearing the
                   ; MSTSBGRND.BGINTM bit
```

See also

MSETC BGINTM

MCMP32 MRa, MRb

32-Bit Integer Compare for Equal, Less Than or Greater Than

Operands

MRa	CLA floating-point source register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1111 0010 0000
```

Description

Set ZF and NF flags on the result of MRa - MRb where MRa and MRb are 32-bit integers. For a floating point compare refer to MCMPF32.

Note

A known hardware issue exists in the MCMP32 instruction. Signed integer comparisons using MCMP32 by itself will set the status bits in a way that is not useful for comparison when the difference between the two operands is too large, such as when the inputs have opposite sign and are near the extreme 32-bit signed values. This affects both signed and unsigned integer comparisons.

The compiler (version 18.1.5.LTS or higher) has implemented a workaround for this issue. The compiler checks the upper bits of the operands by performing a floating point comparison before proceeding to do the integer comparison or subtraction.

The compiler flag --cla_signed_compare_workaround enables this workaround.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified based on the integer results of the operation.

```
If(MRa ==    MRb) {ZF=1; NF=0;}
If(MRa >  MRb) {ZF=0; NF=0;}
If(MRa <  MRb) {ZF=0; NF=1;}
```

Pipeline

This is a single-cycle instruction.

Example

```
; Behavior of ZF and NF flags for different comparisons
;
; Given A = (int32)1
;       B = (int32)2
;       C = (int32)-7
;
    MMOV32 MR0, @_A ; MR0 = 1 (0x00000001)
    MMOV32 MR1, @_B ; MR1 = 2 (0x00000002)
    MMOV32 MR2, @_C ; MR2 = -7 (0xFFFFFFF9)
    MCMP32 MR2, MR2 ; NF = 0, ZF = 1
    MCMP32 MR0, MR1 ; NF = 1, ZF = 0
    MCMP32 MR1, MR0 ; NF = 0, ZF = 0
```

See also

MADD32 MRa, MRb, MRc

MCMP32 MRa, MRb (continued)

32-Bit Integer Compare for Equal, Less Than or Greater Than

MSUB32 MRa, MRb, MRc

MCMPF32 MRa, MRb

32-Bit Floating-Point Compare for Equal, Less Than or Greater Than

Operands

MRa	CLA floating-point source register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1101 0000 0000
```

Description

Set ZF and NF flags on the result of MRa - MRb. The MCMPF32 instruction is performed as a logical compare operation. This is possible because of the IEEE format offsetting the exponent. Basically the bigger the binary number, the bigger the floating-point value.

Special cases for inputs:

- Negative zero will be treated as positive zero.
- A denormalized value will be treated as positive zero.
- Not-a-Number (NaN) will be treated as infinity.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified as follows:

```
If(MRa == MRb)   {ZF=1; NF=0;}
If(MRa > MRb) {ZF=0; NF=0;}
If(MRa < MRb) {ZF=0; NF=1;}
```

Pipeline

This is a single-cycle instruction.

Example

```
; Behavior of ZF and NF flags for different comparisons
    MMOVIZ    MR1, #-2.0  ; MR1 = -2.0 (0xC0000000)
    MMOVIZ    MR0, #5.0   ; MR0 = 5.0 (0x40A00000)
    MCMPF32   MR1, MR0    ; ZF = 0, NF = 1
    MCMPF32   MR0, MR1    ; ZF = 0, NF = 0
    MCMPF32   MR0, MR0    ; ZF = 1, NF = 0
```

See also

MCMPF32 MRa, #16FHi
MMAXF32 MRa, #16FHi
MMAXF32 MRa, MRb
MMINF32 MRa, #16FHi
MMINF32 MRa, MRb

MCMPF32 MRa, #16FHi

32-Bit Floating-Point Compare for Equal, Less Than or Greater Than

Operands

MRa	CLA floating-point source register (MR0 to MR3)
#16FHi	A 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0.

Opcode

```
LSW: IIII IIII IIII IIII
MSW: 0111 1000 1100 00aa
```

Description

Compare the value in MRa with the floating-point value represented by the immediate operand. Set the ZF and NF flags on (MRa - #16FHi:0).

#16FHi is a 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0. This addressing mode is most useful for constants where the lowest 16-bits of the mantissa are 0. Some examples are 2.0 (0x40000000), 4.0 (0x40800000), 0.5 (0x3F000000), and -1.5 (0xBFC00000). The assembler will accept either a hex or float as the immediate value. That is, -1.5 can be represented as #-1.5 or #0xBFC0.

The MCMPF32 instruction is performed as a logical compare operation. This is possible because of the IEEE floating-point format offsets the exponent. Basically the bigger the binary number, the bigger the floating-point value.

Special cases for inputs:

- Negative zero will be treated as positive zero.
- Denormalized value will be treated as positive zero.
- Not-a-Number (NaN) will be treated as infinity.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified as follows:

```
If(MRa == #16FHi:0)   {ZF=1, NF=0;}
If(MRa > #16FHi:0) {ZF=0, NF=0;}
If(MRa < #16FHi:0) {ZF=0, NF=1;}
```

Pipeline

This is a single-cycle instruction

Example 1

```
; Behavior of ZF and NF flags for different comparisons
  MMOVIZ      MR1, #-2.0  ; MR1 = -2.0 (0xC0000000)
  MMOVIZ      MR0, #5.0   ; MR0 = 5.0 (0x40A00000)
  MCMPF32     MR1, #-2.2  ; ZF = 0, NF = 0
  MCMPF32     MR0, #6.5   ; ZF = 0, NF = 1
  MCMPF32     MR0, #5.0   ; ZF = 1, NF = 0
```

MCMPF32 MRa, #16FHi (continued)

32-Bit Floating-Point Compare for Equal, Less Than or Greater Than

Example 2

```
; X is an array of 32-bit floating-point values
; and has len elements. Find the maximum value in
; the array and store it in Result
;
; Note: MCMPF32 and MSWAPF can be replaced with MMAXF32
;
_Cla1Task1:
  MMOVI16 MAR1,#_X         ; Start address
  MUI16TOF32 MR0, @_len    ; Length of the array
  MNOP                     ; delay for MAR1 load
  MNOP                     ; delay for MAR1 load
  MMOV32 MR1, *MAR1[2]++   ; MR1 = X0
LOOP
  MMOV32 MR2, *MAR1[2]++   ; MR2 = next element
  MCMPF32 MR2, MR1         ; Compare MR2 with MR1
  MSWAPF MR1, MR2, GT      ; MR1 = MAX(MR1, MR2)
  MADDF32 MR0, MR0, #-1.0 ; Decrememt the counter
  MCMPF32 MR0 #0.0         ; Set/clear flags for MBCNDD
  MNOP
  MNOP
  MNOP
  MBCNDD LOOP, NEQ         ; Branch if not equal to zero
  MMOV32 @_Result, MR1     ; Always executed
  MNOP                     ; Always executed
  MNOP                     ; Always executed
  MSTOP                    ; End of task
```

See also

MCMPF32 MRa, MRb
MMAXF32 MRa, #16FHi
MMAXF32 MRa, MRb
MMINF32 MRa, #16FHi
MMINF32 MRa, MRb

MDEBUGSTOP

Debug Stop Task

Operands

none	This instruction does not have any operands

Opcode

```
LSW: 0000 0000 0000 0000
MSW: 0111 1111 0110 0000
```

Description

When CLA breakpoints are enabled, the MDEBUGSTOP instruction is used to halt a task so that it can be debugged. That is, MDEBUGSTOP is the CLA breakpoint. If CLA breakpoints are not enabled, the MDEBUGSTOP instruction behaves like a MNOP. Unlike the MSTOP, the MIRUN flag is not cleared and an interrupt is not issued. A single-step or run operation will continue execution of the task.

Restrictions

The MDEBUGSTOP instruction cannot be placed 3 instructions before or after a MBCNDD, MCCNDD, or MRCNDD instruction.

Flags

This instruction does not modify flags in the MSTF register.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

See also

MSTOP

MDEBUGSTOP1

Software Breakpoint

Operands

none	This instruction does not have any operands

Opcode

```
LSW: 0000 0000 0000 0000
MSW: 0111 1111 0011 0000
```

Description

The instruction at which a software breakpoint is placed will be replaced by the MDEBUGSTOP1 instruction. It will halt execution once it reaches the D2 phase in the pipeline; at that point the subsequent instructions that were fetched, after the halt, will be flushed from the pipeline. The replace instruction will be re-fetched after this and execution can continue normally (either in run or step mode).

See Section 7.4.3 for a detailed explanation of its operation.

Restrictions

The MDEBUGSTOP1 instruction cannot be placed 3 instructions before or after a MBCNDD, MCCNDD, or MRCNDD instruction.

Flags

This instruction does not modify flags in the MSTF register.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

See also

MSTOP, MDEBUGSTOP

MEALLOW

Enable CLA Write Access to EALLOW Protected Registers

Operands

none	This instruction does not have any operands

Opcode

```
LSW: 0000 0000 0000 0000
MSW: 0111 1111 1001 0000
```

Description

This instruction sets the MEALLOW bit in the CLA status register MSTF. When this bit is set, the CLA is allowed write access to EALLOW protected registers. To again protect against CLA writes to protected registers, use the MEDIS instruction.

MEALLOW and MEDIS only control CLA write access; reads are allowed even if MEALLOW has not been executed. MEALLOW and MEDIS are also independent from the main CPU's EALLOW/EDIS. This instruction does not modify the EALLOW bit in the main CPU's status register. The MEALLOW bit in MSTF only controls access for the CLA while the EALLOW bit in the ST1 register only controls access for the main CPU.

As with EALLOW, the MEALLOW bit is overridden via the JTAG port, allowing full control of register accesses during debug from Code Composer Studio.

Flags

This instruction does not modify flags in the MSTF register.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
; C header file including definition of
; the EPwm1Regs structure
;
; The ePWM TZSEL register is EALLOW protected
;
   .cdecls C,LIST,"CLAShared.h"
   ...
_Cla1Task1:
   ...
   MEALLOW                          ; Allow CLA write access
   MMOV16 @_EPwm1Regs.TZSEL.all, MR3 ; Write to TZSEL
   MEDIS                            ; Disallow CLA write access
   ...
   ...
   MSTOP
```

See also

MEDIS

MEDIS

Disable CLA Write Access to EALLOW Protected Registers

Operands

none	This instruction does not have any operands

Opcode

```
LSW: 0000 0000 0000 0000
MSW: 0111 1111 1011 0000
```

Description

This instruction clears the MEALLOW bit in the CLA status register MSTF. When this bit is clear, the CLA is not allowed write access to EALLOW-protected registers. To enable CLA writes to protected registers, use the MEALLOW instruction.

MEALLOW and MEDIS only control CLA write access; reads are allowed even if MEALLOW has not been executed. MEALLOW and MEDIS are also independent from the main CPU's EALLOW/EDIS. This instruction does not modify the EALLOW bit in the main CPU's status register. The MEALLOW bit in MSTF only controls access for the CLA while the EALLOW bit in the ST1 register only controls access for the main CPU.

As with EALLOW, the MEALLOW bit is overridden via the JTAG port, allowing full control of register accesses during debug from the Code Composer Studio™ IDE.

Flags

This instruction does not modify flags in the MSTF register.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
; C header file including definition of
; the EPwm1Regs structure
;
; The ePWM TZSEL register is EALLOW protected
;
   .cdecls C,LIST,"CLAShared.h"
   ...
_Cla1Task1:
   ...
   MEALLOW                          ; Allow CLA write access
   MMOV16 @_EPwm1Regs.TZSEL.all, MR3 ; Write to TZSEL
   MEDIS                            ; Disallow CLA write access
   ...
   ...
   MSTOP
```

See also

MEALLOW

MEINVF32 MRa, MRb

32-Bit Floating-Point Reciprocal Approximation

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1111 0000 0000
```

Description

This operation generates an estimate of 1/X in 32-bit floating-point format accurate to approximately 8 bits. This value can be used in a Newton-Raphson algorithm to get a more accurate answer. That is:

```
Ye = Estimate(1/X);
Ye = Ye*(2.0 - Ye*X);
Ye = Ye*(2.0 - Ye*X);
```

After two iterations of the Newton-Raphson algorithm, you will get an exact answer accurate to the 32-bit floating-point format. On each iteration the mantissa bit accuracy approximately doubles. The MEINVF32 operation will not generate a negative zero, DeNorm or NaN value.

```
MRa = Estimate of 1/MRb;
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MEINVF32 generates an underflow condition.
- LVF = 1 if MEINVF32 generates an overflow condition.

Pipeline

This is a single-cycle instruction.

Example

```
; Calculate Num/Den using a Newton-Raphson algorithum for 1/Den
; Ye = Estimate(1/X)
; Ye = Ye*(2.0 - Ye*X)
; Ye = Ye*(2.0 - Ye*X)
;
_Cla1Task1:
    MMOV32 MR1, @_Den      ; MR1 = Den
    MEINVF32 MR2, MR1      ; MR2 = Ye = Estimate(1/Den)
    MMPYF32 MR3, MR2, MR1  ; MR3 = Ye*Den
    MSUBF32 MR3, #2.0, MR3 ; MR3 = 2.0 - Ye*Den
    MMPYF32 MR2, MR2, MR3  ; MR2 = Ye = Ye*(2.0 - Ye*Den)
    MMPYF32 MR3, MR2, MR1  ; MR3 = Ye*Den
 || MMOV32 MR0, @_Num      ; MR0 = Num
    MSUBF32 MR3, #2.0, MR3 ; MR3 = 2.0 - Ye*Den
    MMPYF32 MR2, MR2, MR3  ; MR2 = Ye = Ye*(2.0 - Ye*Den)
 || MMOV32 MR1, @_Den      ; Reload Den To Set Sign
    MNEGF32 MR0, MR0, EQ   ; if(Den == 0.0) Change Sign Of Num
    MMPYF32 MR0, MR2, MR0  ; MR0 = Y = Ye*Num
    MMOV32 @_Dest, MR0     ; Store result
    MSTOP                  ; end of task
```

See also

MEISQRTF32 MRa, MRb

MEISQRTF32 MRa, MRb

32-Bit Floating-Point Square-Root Reciprocal Approximation

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1110 0100 0000
```

Description

This operation generates an estimate of 1/sqrt(X) in 32-bit floating-point format accurate to approximately 8 bits. This value can be used in a Newton-Raphson algorithm to get a more accurate answer. That is:

```
Ye = Estimate(1/sqrt(X));
Ye = Ye*(1.5 - Ye*Ye*X/2.0);
Ye = Ye*(1.5 - Ye*Ye*X/2.0);
```

After 2 iterations of the Newton-Raphson algorithm, you will get an exact answer accurate to the 32-bit floating-point format. On each iteration the mantissa bit accuracy approximately doubles. The MEISQRTF32 operation will not generate a negative zero, DeNorm or NaN value.

```
MRa = Estimate of 1/sqrt (MRb);
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MEISQRTF32 generates an underflow condition.
- LVF = 1 if MEISQRTF32 generates an overflow condition.

Pipeline

This is a single-cycle instruction.

Example

```
; Y = sqrt(X)
; Ye = Estimate(1/sqrt(X));
; Ye = Ye*(1.5 - Ye*Ye*X*0.5)
; Ye = Ye*(1.5 - Ye*Ye*X*0.5)
; Y = X*Ye
;
_Cla1Task3:
    MMOV32 MR0, @_x          ; MR0 = X
    MEISQRTF32 MR1, MR0      ; MR1 = Ye = Estimate(1/sqrt(X))
    MMOV32 MR1, @_x, EQ      ; if(X == 0.0) Ye = 0.0
    MMPYF32 MR3, MR0, #0.5   ; MR3 = X*0.5
    MMPYF32 MR2, MR1, MR3    ; MR2 = Ye*X*0.5
    MMPYF32 MR2, MR1, MR2    ; MR2 = Ye*Ye*X*0.5
    MSUBF32 MR2, #1.5, MR2   ; MR2 = 1.5 - Ye*Ye*X*0.5
    MMPYF32 MR1, MR1, MR2    ; MR1 = Ye = Ye*(1.5 - Ye*Ye*X*0.5)
    MMPYF32 MR2, MR1, MR3    ; MR2 = Ye*X*0.5
    MMPYF32 MR2, MR1, MR2    ; MR2 = Ye*Ye*X*0.5
    MSUBF32 MR2, #1.5, MR2   ; MR2 = 1.5 - Ye*Ye*X*0.5
    MMPYF32 MR1, MR1, MR2    ; MR1 = Ye = Ye*(1.5 - Ye*Ye*X*0.5)
    MMPYF32 MR0, MR1, MR0    ; MR0 = Y = Ye*X
    MMOV32 @_y, MR0          ; Store Y = sqrt(X)
    MSTOP                    ; end of task
```

MEISQRTF32 MRa, MRb (continued)

32-Bit Floating-Point Square-Root Reciprocal Approximation

See also MEINVF32 MRa, MRb

MF32TOI16 MRa, MRb

Convert 32-Bit Floating-Point Value to 16-Bit Integer

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1101 1110 0000
```

Description

Convert a 32-bit floating point value in MRb to a 16-bit integer and truncate. The result will be stored in MRa.

```
MRa(15:0) = F32TOI16(MRb);
MRa(31:16) = sign extension of MRa(15);
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ      MR0, #5.0   ; MR0        = 5.0 (0x40A00000)
MF32TOI16   MR1, MR0    ; MR1(15:0)  = MF32TOI16(MR0) = 0x0005
                        ; MR1(31:16) = Sign extension of MR1(15) = 0x0000
MMOVIZ      MR2, #-5.0  ; MR2        = -5.0 (0xC0A00000)
MF32TOI16   MR3, MR2    ; MR3(15:0)  = MF32TOI16(MR2) = -5 (0xFFFB)
                        ; MR3(31:16) = Sign extension of MR3(15) = 0xFFFF
```

See also

MF32TOI16R MRa, MRb
MF32TOUI16 MRa, MRb
MF32TOUI16R MRa, MRb
MI16TOF32 MRa, MRb
MI16TOF32 MRa, mem16
MUI16TOF32 MRa, mem16
MUI16TOF32 MRa, MRb

MF32TOI16R MRa, MRb

Convert 32-Bit Floating-Point Value to 16-Bit Integer and Round

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1110 0110 0000
```

Description

Convert the 32-bit floating point value in MRb to a 16-bit integer and round to the nearest even value. The result is stored in MRa.

```
MRa(15:0) = F32TOI16round(MRb);
MRa(31:16) = sign extension of MRa(15);
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ MR0, #0x3FD9    ; MR0(31:16) = 0x3FD9
MMOVXI MR0, #0x999A    ; MR0(15:0) = 0x999A
                       ; MR0 = 1.7 (0x3FD9999A)
MF32TOI16R MR1, MR0    ; MR1(15:0) = MF32TOI16round (MR0) = 2 (0x0002)
                       ; MR1(31:16) = Sign extension of MR1(15) = 0x0000
MMOVF32 MR2, #-1.7     ; MR2 = -1.7 (0xBFD9999A)
MF32TOI16R MR3, MR2    ; MR3(15:0) = MF32TOI16round (MR2) = -2 (0xFFFE)
                       ; MR3(31:16) = Sign extension of MR2(15) = 0xFFFF
```

See also

MF32TOI16 MRa, MRb
MF32TOUI16 MRa, MRb
MF32TOUI16R MRa, MRb
MI16TOF32 MRa, MRb
MI16TOF32 MRa, mem16
MUI16TOF32 MRa, mem16
MUI16TOF32 MRa, MRb

MF32TOI32 MRa, MRb

Convert 32-Bit Floating-Point Value to 32-Bit Integer

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1101 0110 0000
```

Description

Convert the 32-bit floating-point value in MRb to a 32-bit integer value and truncate. Store the result in MRa.

```
MRa = F32TOI32(MRb);
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example 1

```
MMOVF32    MR2, #11204005.0   ; MR2 = 11204005.0 (0x4B2AF5A5)
MF32TOI32  MR3, MR2           ; MR3 = MF32TOI32(MR2) = 11204005 (0x00AAF5A5)
MMOVF32    MR0, #-11204005.0  ; MR0 = -11204005.0 (0xCB2AF5A5)
MF32TOI32  MR1, MR0           ; MR1 = MF32TOI32(MR0) = -11204005 (0xFF550A5B)
```

Example 2

```
; Given X, M and B are IQ24 numbers:
; X = IQ24(+2.5) = 0x02800000
; M = IQ24(+1.5) = 0x01800000
; B = IQ24(-0.5) = 0xFF800000
;
; Calculate Y = X * M + B
;
; Convert M, X and B from IQ24 to float
;
_Cla1Task2:
   MI32TOF32 MR0, @_M          ; MR0 = 0x4BC00000
   MI32TOF32 MR1, @_X          ; MR1 = 0x4C200000
   MI32TOF32 MR2, @_B          ; MR2 = 0xCB000000
   MMPYF32   MR0, MR0, #0x3380 ; M = 1/(1*2^24) * iqm = 1.5 (0x3FC00000)
   MMPYF32   MR1, MR1, #0x3380 ; X = 1/(1*2^24) * iqx = 2.5 (0x40200000)
   MMPYF32   MR2, MR2, #0x3380 ; B = 1/(1*2^24) * iqb = -.5 (0xBF000000)
   MMPYF32   MR3, MR0, MR1     ; M*X
   MADDF32   MR2, MR2, MR3     ; Y=MX+B = 3.25 (0x40500000)
; Convert Y from float32 to IQ24
   MMPYF32 MR2, MR2, #0x4B80   ; Y * 1*2^24
   MF32TOI32 MR2, MR2          ; IQ24(Y) = 0x03400000
   MMOV32 @_Y, MR2             ; store result
   MSTOP                       ; end of task
```

See also

MF32TOUI32 MRa, MRb
MI32TOF32 MRa, MRb
MI32TOF32 MRa, mem32
MUI32TOF32 MRa, MRb
MUI32TOF32 MRa, mem32

MF32TOUI16 MRa, MRb

Convert 32-Bit Floating-Point Value to 16-bit Unsigned Integer

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1110 1010 0000
```

Description

Convert the 32-bit floating point value in MRb to an unsigned 16-bit integer value and truncate to zero. The result will be stored in MRa. To instead round the integer to the nearest even value use the MF32TOUI16R instruction.

```
MRa(15:0) = F32TOUI16(MRb);
MRa(31:16) = 0x0000;
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ      MR0, #9.0     ; MR0 = 9.0 (0x41100000)
MF32TOUI16  MR1, MR0      ; MR1(15:0) = MF32TOUI16(MR0) = 9 (0x0009)
                          ; MR1(31:16) = 0x0000
MMOVIZ      MR2, #-9.0    ; MR2 = -9.0 (0xC1100000)
MF32TOUI16  MR3, MR2      ; MR3(15:0) = MF32TOUI16(MR2) = 0 (0x0000)
                          ; MR3(31:16) = 0x0000
```

See also

MF32TOI16 MRa, MRb
MF32TOUI16 MRa, MRb
MF32TOUI16R MRa, MRb
MI16TOF32 MRa, MRb
MI16TOF32 MRa, mem16
MUI16TOF32 MRa, mem16
MUI16TOF32 MRa, MRb

MF32TOUI16R MRa, MRb

Convert 32-Bit Floating-Point Value to 16-bit Unsigned Integer and Round

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1110 1100 0000
```

Description

Convert the 32-bit floating-point value in MRb to an unsigned 16-bit integer and round to the closest even value. The result will be stored in MRa. To instead truncate the converted value, use the MF32TOUI16 instruction.

```
MRa(15:0)  = MF32TOUI16round(MRb);
MRa(31:16) = 0x0000;
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ       MR0, #0x412C    ; MR0 = 0x412C
MMOVXI       MR0, #0xCCCD    ; MR0 = 0xCCCD ; MR0 = 10.8 (0x412CCCCD)
MF32TOUI16R  MR1, MR0        ; MR1(15:0) = MF32TOUI16round(MR0) = 11 (0x000B)
                             ; MR1(31:16) = 0x0000
MMOVF32      MR2, #-10.8     ; MR2 = -10.8 (0x0xC12CCCCD)
MF32TOUI16R  MR3, MR2        ; MR3(15:0) = MF32TOUI16round(MR2) = 0 (0x0000)
                             ; MR3(31:16) = 0x0000
```

See also

MF32TOI16 MRa, MRb
MF32TOI16R MRa, MRb
MF32TOUI16 MRa, MRb
MI16TOF32 MRa, MRb
MI16TOF32 MRa, mem16
MUI16TOF32 MRa, mem16
MUI16TOF32 MRa, MRb

MF32TOUI32 MRa, MRb

Convert 32-Bit Floating-Point Value to 32-Bit Unsigned Integer

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1101 1010 0000
```

Description

Convert the 32-bit floating-point value in MRb to an unsigned 32-bit integer and store the result in MRa.

```
MRa = F32TOUI32(MRb);
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ      MR0, #12.5   ; MR0 = 12.5 (0x41480000)
MF32TOUI32  MR0, MR0     ; MR0 = MF32TOUI32 (MR0) = 12 (0x0000000C)
MMOVIZ      MR1, #-6.5   ; MR1 = -6.5 (0xC0D00000)
MF32TOUI32  MR2, MR1     ; MR2 = MF32TOUI32 (MR1) = 0.0 (0x00000000)
```

See also

MF32TOI32 MRa, MRb
MI32TOF32 MRa, MRb
MI32TOF32 MRa, mem32
MUI32TOF32 MRa, MRb
MUI32TOF32 MRa, mem32

MFRACF32 MRa, MRb

Fractional Portion of a 32-Bit Floating-Point Value

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1110 0000 0000
```

Description Returns in MRa the fractional portion of the 32-bit floating-point value in MRb

Flags This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline This is a single-cycle instruction.

Example

```
MMOVIZ    MR2, #19.625 ; MR2 = 19.625 (0x419D0000)
MFRACF32  MR3, MR2     ; MR3 = MFRACF32(MR2) = 0.625 (0x3F200000)0)
```

MI16TOF32 MRa, MRb

Convert 16-Bit Integer to 32-Bit Floating-Point Value

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1110 1000 0000
```

Description

Convert the 16-bit signed integer in MRb to a 32-bit floating point value and store the result in MRa.

```
MRa = MI16TOF32(MRb);
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ      MR0, #0x0000    ; MR0(31:16) = 0.0 (0x0000)
MMOVXI      MR0, #0x0004    ; MR0(15:0) = 4.0 (0x0004)
MI16TOF32   MR1, MR0        ; MR1 = MI16TOF32 (MR0) = 4.0 (0x40800000)
MMOVIZ      MR2, #0x0000    ; MR2(31:16) = 0.0 (0x0000)
MMOVXI      MR2, #0xFFFC    ; MR2(15:0) = -4.0 (0xFFFC)
MI16TOF32   MR3, MR2        ; MR3 = MI16TOF32 (MR2) = -4.0 (0xC0800000)
MSTOP
```

See also

MF32TOI16 MRa, MRb
MF32TOI16R MRa, MRb
MF32TOUI16 MRa, MRb
MF32TOUI16R MRa, MRb
MI16TOF32 MRa, mem16
MUI16TOF32 MRa, mem16
MUI16TOF32 MRa, MRb

MI16TOF32 MRa, mem16

Convert 16-Bit Integer to 32-Bit Floating-Point Value

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
mem16	16-bit source memory location to be converted

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0111 0101 00aa addr
```

Description

Convert the 16-bit signed integer indicated by the mem16 pointer to a 32-bit floating-point value and store the result in MRa.

```
MRa = MI16TOF32[mem16];
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction:

Example

```
; Assume A = 4 (0x0004)
;        B = -4 (0xFFFC)
   MI16TOF32 MR0, @_A ; MR0 = MI16TOF32(A) = 4.0 (0x40800000)
   MI16TOF32 MR1, @_B ; MR1 = MI16TOF32(B) = -4.0 (0xC0800000
```

See also

MF32TOI16 MRa, MRb
MF32TOI16R MRa, MRb
MF32TOUI16 MRa, MRb
MF32TOUI16R MRa, MRb
MI16TOF32 MRa, MRb
MUI16TOF32 MRa, mem16
MUI16TOF32 MRa, MRb

MI32TOF32 MRa, mem32

Convert 32-Bit Integer to 32-Bit Floating-Point Value

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
mem32	32-bit memory source for the MMOV32 operation.

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0111 0100 01aa addr
```

Description

Convert the 32-bit signed integer indicated by mem32 to a 32-bit floating point value and store the result in MRa.

```
MRa = MI32TOF32[mem32];
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
; Given X, M and B are IQ24 numbers:
; X = IQ24(+2.5) = 0x02800000
; M = IQ24(+1.5) = 0x01800000
; B = IQ24(-0.5) = 0xFF800000
;
; Calculate Y = X * M + B
;
; Convert M, X and B from IQ24 to float
;
_Cla1Task3:
   MI32TOF32 MR0, @_M          ; MR0 = 0x4BC00000
   MI32TOF32 MR1, @_X          ; MR1 = 0x4C200000
   MI32TOF32 MR2, @_B          ; MR2 = 0xCB000000
   MMPYF32 MR0, MR0, #0x3380   ; M = 1/(1*2^24) * iqm = 1.5 (0x3FC00000)
   MMPYF32 MR1, MR1, #0x3380   ; X = 1/(1*2^24) * iqx = 2.5 (0x40200000)
   MMPYF32 MR2, MR2, #0x3380   ; B = 1/(1*2^24) * iqb = -.5 (0xBF000000)
   MMPYF32 MR3, MR0, MR1       ; M*X
   MADDF32 MR2, MR2, MR3       ; Y=MX+B = 3.25 (0x40500000)
; Convert Y from float32 to IQ24
   MMPYF32 MR2, MR2, #0x4B80   ; Y * 1*2^24
   MF32TOI32 MR2, MR2          ; IQ24(Y) = 0x03400000
   MMOV32 @_Y, MR2             ; store result
   MSTOP                       ; end of task
```

See also

MF32TOI32 MRa, MRb
MF32TOUI32 MRa, MRb
MI32TOF32 MRa, MRb
MUI32TOF32 MRa, MRb
MUI32TOF32 MRa, mem32

MI32TOF32 MRa, MRb

Convert 32-Bit Integer to 32-Bit Floating-Point Value

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1101 1000 0000
```

Description

Convert the signed 32-bit integer in MRb to a 32-bit floating-point value and store the result in MRa.

```
MRa = MI32TOF32(MRb);
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ    MR2, #0x1111 ; MR2(31:16) = 4369 (0x1111)
MMOVXI    MR2, #0x1111 ; MR2(15:0) = 4369 (0x1111)
                       ; MR2 = +286331153 (0x11111111)
MI32TOF32 MR3, MR2     ; MR3 = MI32TOF32 (MR2) = 286331153.0 (0x4D888888)
```

See also

MF32TOI32 MRa, MRb
MF32TOUI32 MRa, MRb
MI32TOF32 MRa, mem32
MUI32TOF32 MRa, MRb
MUI32TOF32 MRa, mem32

MLSL32 MRa, #SHIFT

Logical Shift Left

Operands

MRa	CLA floating-point source/destination register (MR0 to MR3)
#SHIFT	Number of bits to shift (1 to 32)

Opcode

```
LSW: 0000 0000 0shi ftaa
MSW: 0111 1011 1100 0000
```

Description

Logical shift left of MRa by the number of bits indicated. The number of bits can be 1 to 32.

```
MARa(31:0) = Logical Shift Left(MARa(31:0) by #SHIFT bits);
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified based on the integer results of the operation.

```
NF = MRa(31);
ZF = 0;
if(MRa(31:0) == 0) { ZF = 1; }
```

Pipeline

This is a single-cycle instruction.

Example

```
; Given m2 = (int32)32
;       x2 = (int32)64
;       b2 = (int32)-128
;
; Calculate:
;       m2 = m2*2
;       x2 = x2*4
;       b2 = b2*8
;
_Cla1Task3:
   MMOV32 MR0, @_m2    ; MR0 = 32 (0x00000020)
   MMOV32 MR1, @_x2    ; MR1 = 64 (0x00000040)
   MMOV32 MR2, @_b2    ; MR2 = -128 (0xFFFFFF80)
   MLSL32 MR0, #1      ; MR0 = 64 (0x00000040)
   MLSL32 MR1, #2      ; MR1 = 256 (0x00000100)
   MLSL32 MR2, #3      ; MR2 = -1024 (0xFFFFFC00)
   MMOV32 @_m2, MR0    ; Store results
   MMOV32 @_x2, MR1
   MMOV32 @_b2, MR2
   MSTOP               ; end of task
```

See also

MADD32 MRa, MRb, MRc
MASR32 MRa, #SHIFT
MAND32 MRa, MRb, MRc
MLSR32 MRa, #SHIFT
MOR32 MRa, MRb, MRc
MXOR32 MRa, MRb, MRc
MSUB32 MRa, MRb, MRc

MLSR32 MRa, #SHIFT

Logical Shift Right

Operands

MRa	CLA floating-point source/destination register (MR0 to MR3)
#SHIFT	Number of bits to shift (1 to 32)

Opcode

```
LSW: 0000 0000 0shi ftaa
MSW: 0111 1011 1000 0000
```

Description

Logical shift right of MRa by the number of bits indicated. The number of bits can be 1 to 32. Unlike the arithmetic shift (MASR32), the logical shift does not preserve the number's sign bit. Every bit in the operand is moved the specified number of bit positions, and the vacant bit-positions are filled in with zeros

```
MARa(31:0) = Logical Shift Right(MARa(31:0) by #SHIFT bits);
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified based on the integer results of the operation.

```
NF = MRa(31);
ZF = 0;
if(MRa(31:0) == 0) { ZF = 1;}
```

Pipeline

This is a single-cycle instruction.

Example

```
; Illustrate the difference between MASR32 and MLSR32
MMOVIZ MR0, #0xAAAA    ; MR0 = 0xAAAA5555
MMOVXI MR0, #0x5555
MMOV32 MR1, MR0        ; MR1 = 0xAAAA5555
MMOV32 MR2, MR0        ; MR2 = 0xAAAA5555
MASR32 MR1, #1         ; MR1 = 0xD5552AAA
MLSR32 MR2, #1         ; MR2 = 0x55552AAA
MASR32 MR1, #1         ; MR1 = 0xEAAA9555
MLSR32 MR2, #1         ; MR2 = 0x2AAA9555
MASR32 MR1, #6         ; MR1 = 0xFFAAAA55
MLSR32 MR2, #6         ; MR2 = 0x00AAAA55
```

See also

MADD32 MRa, MRb, MRc
MASR32 MRa, #SHIFT
MAND32 MRa, MRb, MRc
MLSL32 MRa, #SHIFT
MOR32 MRa, MRb, MRc
MXOR32 MRa, MRb, MRc
MSUB32 MRa, MRb, MRc

MMACF32 MR3, MR2, MRd, MRe, MRf ||MMOV32 MRa, mem32

32-Bit Floating-Point Multiply and Accumulate with Parallel Move

Operands

MR3	floating-point destination/source register MR3 for the add operation
MR2	CLA floating-point source register MR2 for the add operation
MRd	CLA floating-point destination register (MR0 to MR3) for the multiply operation MRd cannot be the same register as MRa
MRe	CLA floating-point source register (MR0 to MR3) for the multiply operation
MRf	CLA floating-point source register (MR0 to MR3) for the multiply operation
MRa	CLA floating-point destination register for the MMOV32 operation (MR0 to MR3). MRa cannot be MR3 or the same register as MRd.
mem32	32-bit source for the MMOV32 operation

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0011 ffee ddaa addr
```

Description

Multiply and accumulate the contents of floating-point registers and move from register to memory. The destination register for the MMOV32 cannot be the same as the destination registers for the MMACF32.

```
MR3 = MR3 + MR2;
MRd = MRe * MRf;
MRa = [mem32];
```

Restrictions

The destination registers for the MMACF32 and the MMOV32 must be unique. That is, MRa cannot be MR3 and MRa cannot be the same register as MRd.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MMACF32 (add or multiply) generates an underflow condition.
- LVF = 1 if MMACF32 (add or multiply) generates an overflow condition.

MMOV32 sets the NF and ZF flags as follows:

```
NF = MRa(31);
ZF = 0;
if(MRa(30:23) == 0) { ZF = 1; NF = 0; }
```

Pipeline

MMACF32 and MMOV32 complete in a single cycle.

MMACF32 MR3, MR2, MRd, MRe, MRf ||MMOV32 MRa, mem32 (continued)

32-Bit Floating-Point Multiply and Accumulate with Parallel Move

Example 1

```
; Perform 5 multiply and accumulate operations:
;
; X and Y are 32-bit floating point arrays
;
; 1st multiply: A = X0 * Y0
; 2nd multiply: B = X1 * Y1
; 3rd multiply: C = X2 * Y2
; 4th multiply: D = X3 * Y3
; 5th multiply: E = X3 * Y3
;
; Result = A + B + C + D + E
;
_Cla1Task1:
   MMOVI16 MAR0, #_X               ; MAR0 points to X array
   MMOVI16 MAR1, #_Y               ; MAR1 points to Y array
   MNOP                            ; Delay for MAR0, MAR1 load
   MNOP                            ; Delay for MAR0, MAR1 load
                                   ; <-- MAR0 valid
   MMOV32 MR0, *MAR0[2]++          ; MR0 = X0, MAR0 += 2
                                   ; <-- MAR1 valid
   MMOV32 MR1, *MAR1[2]++          ; MR1 = Y0, MAR1 += 2
   MMPYF32 MR2, MR0, MR1           ; MR2 = A = X0 * Y0
|| MMOV32 MR0, *MAR0[2]++          ; In parallel MR0 = X1, MAR0 += 2
   MMOV32 MR1, *MAR1[2]++          ; MR1 = Y1, MAR1 += 2
   MMPYF32 MR3, MR0, MR1           ; MR3 = B = X1 * Y1
|| MMOV32 MR0, *MAR0[2]++          ; In parallel MR0 = X2, MAR0 += 2
   MMOV32 MR1, *MAR1[2]++          ; MR1 = Y2, MAR2 += 2
   MMACF32 MR3, MR2, MR2, MR0, MR1 ; MR3 = A + B, MR2 = C = X2 * Y2
|| MMOV32 MR0, *MAR0[2]++          ; In parallel MR0 = X3
   MMOV32 MR1, *MAR1[2]++          ; MR1 = Y3 M
   MACF32 MR3, MR2, MR2, MR0, MR1  ; MR3 = (A + B) + C, MR2 = D = X3 * Y3
|| MMOV32 MR0, *MAR0               ; In parallel MR0 = X4
   MMOV32 MR1, *MAR1               ; MR1 = Y4
   MMPYF32 MR2, MR0, MR1           ; MR2 = E = X4 * Y4
|| MADDF32 MR3, MR3, MR2           ; in parallel MR3 = (A + B + C) + D
   MADDF32 MR3, MR3, MR2           ; MR3 = (A + B + C + D) + E
   MMOV32 @_Result, MR3            ; Store the result
   MSTOP                           ; end of task
```

Example 2

```
; sum = X0*B0 + X1*B1 + X2*B2 + Y1*A1 + Y2*B2
;
;       X2 = X1
;       X1 = X0
;       Y2 = Y1 ; Y1 = sum
;
_ClaTask2:
     MMOV32     MR0, @_B2       ; MR0 = B2
     MMOV32     MR1, @_X2       ; MR1 = X2
     MMPYF32    MR2, MR1, MR0   ; MR2 = X2*B2
  || MMOV32     MR0, @_B1       ; MR0 = B1
     MMOVD32    MR1, @_X1       ; MR1 = X1, X2 = X1
     MMPYF32    MR3, MR1, MR0   ; MR3 = X1*B1
  || MMOV32     MR0, @_B0       ; MR0 = B0
     MMOVD32    MR1, @_X0       ; MR1 = X0, X1 = X0
;  MR3 = X1*B1 + X2*B2, MR2 = X0*B0
;  MR0 = A2
     MMACF32 MR3, MR2, MR2, MR1, MR0
  || MMOV32 MR0, @_A2 M

     MOV32 MR1, @_Y2            ; MR1 = Y2
;  MR3 = X0*B0 + X1*B1 + X2*B2, MR2 = Y2*A2
;  MR0 = A1
     MMACF32 MR3, MR2, MR2, MR1, MR0
  || MMOV32 MR0, @_A1
     MMOVD32 MR1,@_Y1           ; MR1 = Y1, Y2 = Y1
     MADDF32 MR3, MR3, MR2      ; MR3 = Y2*A2 + X0*B0 + X1*B1 + X2*B2
  || MMPYF32 MR2, MR1, MR0      ; MR2 = Y1*A1
     MADDF32 MR3, MR3, MR2      ; MR3 = Y1*A1 + Y2*A2 + X0*B0 + X1*B1 + X2*B2
```

MMACF32 MR3, MR2, MRd, MRe, MRf ||MMOV32 MRa, mem32 (continued)

32-Bit Floating-Point Multiply and Accumulate with Parallel Move

```
    MMOV32 @_Y1, MR3          ; Y1 = MR3
    MSTOP                     ; end of task
```

See also MMPYF32 MRa, MRb, MRc || MADDF32 MRd, MRe, MRf

MMAXF32 MRa, MRb

32-Bit Floating-Point Maximum

Operands

MRa	CLA floating-point source/destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1101 0010 0000
```

Description

```
if(MRa < MRb) MRa = MRb;
```

Special cases for the output from the MMAXF32 operation:

- NaN output will be converted to infinity
- A denormalized output will be converted to positive zero.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The ZF and NF flags are configured on the result of the operation, not the result stored in the destination register.

```
if(MRa == MRb)  {ZF=1; NF=0;}
if(MRa > MRb) {ZF=0; NF=0;}
if(MRa < MRb) {ZF=0; NF=1;}
```

Pipeline

This is a single-cycle instruction.

Example 1

```
MMOVIZ   MR0, #5.0  ; MR0 =  5.0  (0x40A00000)
MMOVIZ   MR1, #-2.0 ; MR1 = -2.0  (0xC0000000)
MMOVIZ   MR2, #-1.5 ; MR2 = -1.5  (0xBFC00000)
MMAXF32   MR2, MR1  ; MR2 = -1.5, ZF = NF = 0
MMAXF32   MR1, MR2  ; MR1 = -1.5, ZF = 0, NF = 1
MMAXF32   MR2, MR0  ; MR2 =  5.0, ZF = 0, NF = 1
MAXF32   MR0, MR2   ; MR2 =  5.0, ZF = 1, NF = 0
```

MMAXF32 MRa, MRb (continued)

32-Bit Floating-Point Maximum

Example 2

```
; X is an array of 32-bit floating-point values
; Find the maximum value in an array X
; and store it in Result
;
_Cla1Task1:
  MMOVI16    MAR1,#_X          ; Start address
  MUI16TOF32 MR0, @_len        ; Length of the array
  MNOP                         ; delay for MAR1 load
  MNOP                         ; delay for MAR1 load
  MMOV32     MR1, *MAR1[2]++   ; MR1 = X0
LOOP
  MMOV32     MR2, *MAR1[2]++   ; MR2 = next element
  MMAXF32    MR1, MR2          ; MR1 = MAX(MR1, MR2)
  MADDF32    MR0, MR0, #-1.0   ; Decrememt the counter
  MCMPF32    MR0 #0.0          ; Set/clear flags for MBCNDD
  MNOP
  MNOP
  MNOP
  MBCNDD     LOOP, NEQ         ; Branch if not equal to zero
  MMOV32     @_Result, MR1     ; Always executed
  MNOP                         ; Always executed
  MNOP                         ; Always executed
  MSTOP                        ; End of task
```

See also

MCMPF32 MRa, MRb
MCMPF32 MRa, #16FHi
MMAXF32 MRa, #16FHi
MMINF32 MRa, MRb
MMINF32 MRa, #16FHi

MMAXF32 MRa, #16FHi

32-Bit Floating-Point Maximum

Operands

MRa	CLA floating-point source/destination register (MR0 to MR3)
#16FHi	A 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0.

Opcode

```
LSW: IIII IIII IIII IIII
MSW: 0111 1001 0000 00aa
```

Description

Compare MRa with the floating-point value represented by the immediate operand. If the immediate value is larger, then load it into MRa.

```
if(MRa < #16FHi:0) MRa = #16FHi:0;
```

#16FHi is a 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0. This addressing mode is most useful for constants where the lowest 16-bits of the mantissa are 0. Some examples are 2.0 (0x40000000), 4.0 (0x40800000), 0.5 (0x3F000000), and -1.5 (0xBFC00000). The assembler will accept either a hex or float as the immediate value. That is, -1.5 can be represented as #-1.5 or #0xBFC0.

Special cases for the output from the MMAXF32 operation:

- NaN output will be converted to infinity
- A denormalized output will be converted to positive zero.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The ZF and NF flags are configured on the result of the operation, not the result stored in the destination register.

```
if(MRa == #16FHi:0)  {ZF=1; NF=0;}
if(MRa > #16FHi:0) {ZF=0; NF=0;}
if(MRa < #16FHi:0) {ZF=0; NF=1;}
```

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ    MR0, #5.0  ; MR0 =  5.0  (0x40A00000)
MMOVIZ    MR1, #4.0  ; MR1 =  4.0  (0x40800000)
MMOVIZ    MR2, #-1.5 ; MR2 = -1.5  (0xBFC00000)
MMAXF32   MR0, #5.5  ; MR0 =  5.5, ZF = 0, NF = 1
MMAXF32   MR1, #2.5  ; MR1 =  4.0, ZF = 0, NF = 0
MMAXF32   MR2, #-1.0 ; MR2 = -1.0, ZF = 0, NF = 1
MMAXF32   MR2, #-1.0 ; MR2 = -1.5, ZF = 1, NF = 0
```

See also

MMAXF32 MRa, MRb
MMINF32 MRa, MRb
MMINF32 MRa, #16FHi

MMINF32 MRa, MRb

32-Bit Floating-Point Minimum

Operands

MRa	CLA floating-point source/destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1101 0100 0000
```

Description

```
if(MRa > MRb) MRa = MRb;
```

Special cases for the output from the MMINF32 operation:

- NaN output will be converted to infinity
- A denormalized output will be converted to positive zero.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The ZF and NF flags are configured on the result of the operation, not the result stored in the destination register.

```
if(MRa == MRb)  {ZF=1; NF=0;}
if(MRa > MRb) {ZF=0; NF=0;}
if(MRa < MRb) {ZF=0; NF=1;}
```

Pipeline

This is a single-cycle instruction.

Example 1

```
MMOVIZ MR0, #5.0  ; MR0 = 5.0 (0x40A00000)
MMOVIZ MR1, #4.0  ; MR1 = 4.0 (0x40800000)
MMOVIZ MR2, #-1.5 ; MR2 = -1.5 (0xBFC00000)
MMINF32 MR0, MR1  ; MR0 = 4.0, ZF = 0, NF = 0
MMINF32 MR1, MR2  ; MR1 = -1.5, ZF = 0, NF = 0
MMINF32 MR2, MR1  ; MR2 = -1.5, ZF = 1, NF = 0
MMINF32 MR1, MR0  ; MR2 = -1.5, ZF = 0, NF = 1
```

MMINF32 MRa, MRb (continued)

32-Bit Floating-Point Minimum

Example 2

```
;
; X is an array of 32-bit floating-point values
; Find the minimum value in an array X
; and store it in Result
;
_Cla1Task1:
MMOVI16    MAR1,#_X          ; Start address
MUI16TOF32 MR0, @_len        ; Length of the array
MNOP                         ; delay for MAR1 load
MNOP                         ; delay for MAR1 load
MMOV32     MR1, *MAR1[2]++   ; MR1 = X0
LOOP
MMOV32     MR2, *MAR1[2]++   ; MR2 = next element
MMINF32    MR1, MR2          ; MR1 = MAX(MR1, MR2)
MADDF32    MR0, MR0, #-1.0   ; Decrememt the counter
MCMPF32    MR0 #0.0          ; Set/clear flags for MBCNDD
MNOP
MNOP
MNOP
MBCNDD     LOOP, NEQ         ; Branch if not equal to zero
MMOV32     @_Result, MR1     ; Always executed
MNOP                         ; Always executed
MNOP                         ; Always executed
MSTOP                        ; End of task
```

See also

MMAXF32 MRa, MRb
MMAXF32 MRa, #16FHi
MMINF32 MRa, #16FHi

MMINF32 MRa, #16FHi

32-Bit Floating-Point Minimum

Operands

MRa	floating-point source/destination register (MR0 to MR3)
#16FHi	A 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0.

Opcode

```
LSW: IIII IIII IIII IIII
MSW: 0111 1001 0100 00aa
```

Description

Compare MRa with the floating-point value represented by the immediate operand. If the immediate value is smaller, then load it into MRa.

```
if(MRa > #16FHi:0) MRa = #16FHi:0;
```

#16FHi is a 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0. This addressing mode is most useful for constants where the lowest 16-bits of the mantissa are 0. Some examples are 2.0 (0x40000000), 4.0 (0x40800000), 0.5 (0x3F000000), and -1.5 (0xBFC00000). The assembler will accept either a hex or float as the immediate value. That is, -1.5 can be represented as #-1.5 or #0xBFC0.

Special cases for the output from the MMINF32 operation:

- NaN output will be converted to infinity
- A denormalized output will be converted to positive zero.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The ZF and NF flags are configured on the result of the operation, not the result stored in the destination register.

```
if(MRa == #16FHi:0)   {ZF=1; NF=0;}
if(MRa > #16FHi:0) {ZF=0; NF=0;}
if(MRa < #16FHi:0) {ZF=0; NF=1;}
```

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ   MR0, #5.0   ; MR0 =  5.0  (0x40A00000)
MMOVIZ   MR1, #4.0   ; MR1 =  4.0  (0x40800000)
MMOVIZ   MR2, #-1.5  ; MR2 = -1.5  (0xBFC00000)
MMINF32  MR0, #5.5   ; MR0 =  5.0, ZF = 0, NF = 1
MMINF32  MR1, #2.5   ; MR1 =  2.5, ZF = 0, NF = 0
MMINF32  MR2, #-1.0  ; MR2 = -1.5, ZF = 0, NF = 1
MMINF32  MR2, #-1.5  ; MR2 = -1.5, ZF = 1, NF = 0
```

See also

MMAXF32 MRa, #16FHi
MMAXF32 MRa, MRb
MMINF32 MRa, MRb

MMOV16 MARx, MRa, #16I

Load the Auxiliary Register with MRa + 16-bit Immediate Value

Operands

MARx	Auxiliary register MAR0 or MAR1
MRa	CLA Floating-point register (MR0 to MR3)
#16I	16-bit immediate value

Opcode

```
LSW: IIII IIII IIII IIII (opcode of MMOV16 MAR0, MRa, #16I)
MSW: 0111 1111 1101 00AA
LSW: IIII IIII IIII IIII (opcode of MMOV16 MAR1, MRa, #16I)
MSW: 0111 1111 1111 00AA
```

Description

Load the auxiliary register, MAR0 or MAR1, with MRa(15:0) + 16-bit immediate value. Refer to the pipeline section for important information regarding this instruction.

```
MARx = MRa(15:0) + #16I;
```

Flags

This instruction does not modify flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction. The load of MAR0 or MAR1 will occur in the EXE phase of the pipeline. Any post increment of MAR0 or MAR1 using indirect addressing will occur in the D2 phase of the pipeline. Therefore the following applies when loading the auxiliary registers:

- **I1 and I2**

  The two instructions following MMOV16 will use MAR0/MAR1 before the update occurs. Thus these two instructions will use the old value of MAR0 or MAR1.
- **I3**

  Loading of an auxiliary register occurs in the EXE phase while updates due to post-increment addressing occur in the D2 phase. Thus I3 cannot use the auxiliary register or there will be a conflict. In the case of a conflict, the update due to address-mode post increment will win and the auxiliary register will not be updated with #_X.
- **I4**

  Starting with the 4th instruction MAR0 or MAR1 will be the new value loaded with MMOVI16.

```
; Assume MAR0 is 50, MR0 is 10, and #_X is 20
  MMOV16 MAR0, MR0, #_X        ; Load MAR0 with address of X (20) + MR0 (10)
  <Instruction 1>         ; I1 Will use the old value of MAR0 (50)
  <Instruction 2>         ; I2 Will use the old value of MAR0 (50)
  <Instruction 3>         ; I3 Cannot use MAR0
  <Instruction 4>         ; I4 Will use the new value of MAR0 (30)
  <Instruction 5>         ; I5
```

MMOV16 MARx, MRa, #16I (continued)

Load the Auxiliary Register with MRa + 16-bit Immediate Value

Table 7-16. Pipeline Activity For MMOV16 MARx, MRa , #16I

Instruction	F1	F2	D1	D2	R1	R2	E	W
MMOV16 MAR0, MR0, #_X	MMOV16							
I1	I1	MMOV16						
I2	I2	I1	MMOV16					
I3	I3	I2	I1	MMOV16				
I4	I4	I3	I2	I1	MMOV16			
I5	I5	I4	I3	I2	I1	MMOV16		
I6	I6	I5	I4	I3	I2	I1	MMOV16	

Example 1

```
; Calculate an offset into a sin/cos table
;
_Cla1Task1:
    MMOV32 MR0,@_rad                ; MR0 = rad
    MMOV32 MR1,@_TABLE_SIZEDivTwoPi ; MR1 = TABLE_SIZE/(2*Pi)
    MMPYF32 MR1,MR0,MR1             ; MR1 = rad* TABLE_SIZE/(2*Pi)
||  MMOV32 MR2,@_TABLE_MASK         ; MR2 = TABLE_MASK
    MF32TOI32 MR3,MR1               ; MR3 = K=int(rad*TABLE_SIZE/(2*Pi))
    MAND32 MR3,MR3,MR2              ; MR3 = K & TABLE_MASK
    MLSL32 MR3,#1                   ; MR3 = K * 2
    MMOV16 MAR0,MR3,#_Cos0          ; MAR0 K*2+addr of table.Cos0
    MFRACF32 MR1,MR1                ; I1
    MMOV32 MR0,@_TwoPiDivTABLE_SIZE ; I2
    MMPYF32 MR1,MR1,MR0             ; I3
||  MMOV32 MR0,@_Coef3
    MMOV32 MR2,*MAR0[#-64]++        ; MR2 = *MAR0, MAR0 += (-64)
    ...
    ...
    MSTOP ; end of task
```

MMOV16 MARx, MRa, #16I (continued)

Load the Auxiliary Register with MRa + 16-bit Immediate Value

Example 2

```
; This task logs the last NUM_DATA_POINTS
; ADCRESULT1 values in the array VoltageCLA
;
; When the last element in the array has been
; filled, the task will go back to the
; the first element.
;
; Before starting the ADC conversions, force
; Task 8 to initialize the ConversionCount to zero
;
; The ADC is set to sample (acquire) for 15 SYSCLK cycles
; or 75ns. After the capacitor has captured the analog
; value, the ADC will trigger this task early.
; It takes 10.5 ADCCLKs to complete a conversion,
; the ADCCLK being SYSCLK/4
;   T_sys = 1/200MHz = 5ns
;   T_adc = 4*T_sys  = 20ns
; The ADC will take 10.5 * 4 or 42 SYSCLK cycles to complete
; a conversion. The ADC result register may be read on the
; 36th instruction after the task begins.
;
_Cla1Task2:
    .asg       0, N
    .loop
    MNOP                                    ;I1 - I28 Wait till I36 to read
result
    .eval      N + 1, N
    .break     N = 28
    .endloop
    MMOVZ16    MR0, @_ConversionCount           ;I29 Current Conversion
    MMOV16     MAR1, MR0, #_VoltageCLA          ;I30 Next array location
    MUI16TOF32 MR0, MR0                         ;I31 Convert count to float32
    MADDF32    MR0, MR0, #1.0                   ;I32 Add 1 to conversion count
    MCMPF32    MR0, #NUM_DATA_POINTS.0          ;I33 Compare count to max
    MF32TOUI16 MR0, MR0                         ;I34 Convert count to Uint16
    MNOP                                        ;I35 Wait till I36 to read
result
    MMOVZ16    MR2, @_AdcaResultRegs.ADCRESULT1 ;I36 Read ADCRESULT1
    MMOV16     *MAR1, MR2                       ; Store ADCRESULT1
    MBCNDD     _RestartCount, GEQ               ; If count >= NUM_DATA_POINTS
    MMOVIZ     MR1, #0.0                        ; Always executed: MR1=0
    MNOP
    MNOP
    MMOV16     @_ConversionCount, MR0           ; If branch not taken
    MSTOP                                       ; store current count
_RestartCount
    MMOV16     @_ConversionCount, MR1           ; If branch taken, restart
count
    MSTOP                                       ; end of task
; This task initializes the ConversionCount
; to zero
;
_Cla1Task8:
    MMOVIZ     MR0, #0.0
    MMOV16     @_ConversionCount, MR0
    MSTOP
_ClaT8End:
```

MMOV16 MARx, mem16

Load MAR1 with 16-bit Value

Operands

MARx	CLA auxiliary register MAR0 or MAR1
mem16	16-bit destination memory accessed using one of the available addressing modes

Opcode

```
LSW: mmmm mmmm mmmm mmmm (Opcode for MMOV16 MAR0, mem16)
MSW: 0111 0110 0000 addr
LSW: mmmm mmmm mmmm mmmm (Opcode for MMOV16 MAR1, mem16)
MSW: 0111 0110 0100 addr
```

Description

Load MAR0 or MAR1 with the 16-bit value pointed to by mem16. Refer to the pipeline section for important information regarding this instruction.

```
MAR1 = [mem16];
```

Flags

No flags MSTF flags are affected.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction. The load of MAR0 or MAR1 will occur in the EXE phase of the pipeline. Any post increment of MAR0 or MAR1 using indirect addressing will occur in the D2 phase of the pipeline. Therefore the following applies when loading the auxiliary registers:

- **I1 and I2**

  The two instructions following MMOV16 will use MAR0/MAR1 before the update occurs. Thus these two instructions will use the old value of MAR0 or MAR1.
- **I3**

  Loading of an auxiliary register occurs in the EXE phase while updates due to post-increment addressing occur in the D2 phase. Thus I3 cannot use the auxiliary register or there will be a conflict. In the case of a conflict, the update due to address-mode post increment will win send the auxiliary register will not be updated with #_X.
- **I4**

  Starting with the 4th instruction MAR0 or MAR1 will be the new value loaded with MMOV16.

```
; Assume MAR0 is 50 and @_X is 20
MMOV16 MAR0, @_X    ; Load MAR0 with the contents of X (20)
<Instruction 1>     ; I1 Will use the old value of MAR0 (50)
<Instruction 2>     ; I2 Will use the old value of MAR0 (50)
<Instruction 3>     ; I3 Cannot use MAR0
<Instruction 4>     ; I4 Will use the new value of MAR0 (20)
<Instruction 5>     ; I5
....
```

MMOV16 MARx, mem16 (continued)

Load MAR1 with 16-bit Value

Table 7-17. Pipeline Activity For MMOV16 MAR0/MAR1, mem16

Instruction	F1	F2	D1	D2	R1	R2	E	W
MMOV16 MAR0, @_X	MMOV16							
I1	I1	MMOV16						
I2	I2	I1	MMOV16					
I3	I3	I2	I1	MMOV16				
I4	I4	I3	I2	I1	MMOV16			
I5	I5	I4	I3	I2	I1	MMOV16		
I6	I6	I5	I4	I3	I2	I1	MMOV16	

Example

```
; This task logs the last NUM_DATA_POINTS
; ADCRESULT1 values in the array VoltageCLA
;
; When the last element in the array has been
; filled, the task will go back to the
; the first element.
;
; Before starting the ADC conversions, force
; Task 8 to initialize the ConversionCount to zero
;
; The ADC is set to sample (acquire) for 15 SYSCLK cycles
; or 75ns. After the capacitor has captured the analog
; value, the ADC will trigger this task early.
; It takes 10.5 ADCCLKs to complete a conversion,
; the ADCCLK being SYSCLK/4
;   T_sys = 1/200MHz = 5ns
;   T_adc = 4*T_sys  = 20ns
; The ADC will take 10.5 * 4 or 42 SYSCLK cycles to complete
; a conversion. The ADC result register may be read on the
; 36th instruction after the task begins.
;
_Cla1Task2:
    .asg       0, N
    .loop
    MNOP
;I1 - I28 Wait till I36 to read result
    .eval      N + 1, N
    .break     N = 28
    .endloop
    MMOVZ16   MR0, @_ConversionCount            ;I29 Current Conversion
    MMOV16    MAR1, MR0, #_VoltageCLA           ;I30 Next array location
    MUI16TOF32 MR0, MR0                         ;I31 Convert count to float32
    MADDF32   MR0, MR0, #1.0                    ;I32 Add 1 to conversion count
    MCMPF32   MR0, #NUM_DATA_POINTS.0           ;I33 Compare count to max
    MF32TOUI16 MR0, MR0                         ;I34 Convert count to Uint16
    MNOP                                        ;I35 Wait till I36 to read
result
    MMOVZ16   MR2, @_AdcaResultRegs.ADCRESULT1  ;I36 Read ADCRESULT1
    MMOV16    *MAR1, MR2                        ; Store ADCRESULT1
    MBCNDD    _RestartCount, GEQ                ; If count >= NUM_DATA_POINTS
    MMOVIZ    MR1, #0.0                         ; Always executed: MR1=0
    MNOP
    MNOP
    MMOV16    @_ConversionCount, MR0            ; If branch not taken
    MSTOP                                       ; store current count
_RestartCount
    MMOV16    @_ConversionCount, MR1            ; If branch taken, restart
count
    MSTOP                                       ; end of task
; This task initializes the ConversionCount
; to zero
;
_Cla1Task8:
```

MMOV16 MARx, mem16 (continued)

Load MAR1 with 16-bit Value

```
    MMOVIZ    MR0, #0.0
    MMOV16    @_ConversionCount, MR0
    MSTOP
_ClaT8End:
```

MMOV16 mem16, MARx

Move 16-Bit Auxiliary Register Contents to Memory

Operands

mem16	16-bit destination memory accessed using one of the available addressing modes
MARx	CLA auxiliary register MAR0 or MAR1

Opcode

```
LSW: mmmm mmmm mmmm mmmm (Opcode for MMOV16 mem16, MAR0)
MSW: 0111 0110 1000 addr
LSW: mmmm mmmm mmmm mmmm (Opcode for MMOV16 mem16, MAR1)
MSW: 0111 0110 1100 addr
```

Description

Store the contents of MAR0 or MAR1 in the 16-bit memory location pointed to by mem16.

```
[mem16] = MAR0;
```

Flags

No flags MSTF flags are affected.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

MMOV16 mem16, MRa

Move 16-Bit Floating-Point Register Contents to Memory

Operands

mem16	16-bit destination memory accessed using one of the available addressing modes
MRa	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0111 0101 11aa addr
```

Description

Move 16-bit value from the lower 16-bits of the floating-point register (MRa(15:0)) to the location pointed to by mem16.

```
[mem16] = MRa(15:0);
```

Flags

No flags MSTF flags are affected.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
; This task logs the last NUM_DATA_POINTS
; ADCRESULT1 values in the array VoltageCLA
;
; When the last element in the array has been
; filled, the task will go back to the
; the first element.
;
; Before starting the ADC conversions, force
; Task 8 to initialize the ConversionCount to zero
;
; The ADC is set to sample (acquire) for 15 SYSCLK cycles
; or 75ns. After the capacitor has captured the analog
; value, the ADC will trigger this task early.
; It takes 10.5 ADCCLKs to complete a conversion,
; the ADCCLK being SYSCLK/4
;   T_sys = 1/200MHz = 5ns
;   T_adc = 4*T_sys  = 20ns
; The ADC will take 10.5 * 4 or 42 SYSCLK cycles to complete
; a conversion. The ADC result register may be read on the
; 36th instruction after the task begins.
;
_Cla1Task2:
    .asg       0, N
    .loop
    MNOP
;I1 - I28 Wait till I36 to read result
    .eval      N + 1, N
    .break     N = 28
    .endloop
    MMOVZ16   MR0, @_ConversionCount            ;I29 Current Conversion
    MMOV16    MAR1, MR0, #_VoltageCLA           ;I30 Next array location
    MUI16TOF32 MR0, MR0                         ;I31 Convert count to float32
    MADDF32   MR0, MR0, #1.0                    ;I32 Add 1 to conversion count
    MCMPF32   MR0, #NUM_DATA_POINTS.0           ;I33 Compare count to max
    MF32TOUI16 MR0, MR0                         ;I34 Convert count to Uint16
    MNOP                                        ;I35 Wait till I36 to read
result
    MMOVZ16   MR2, @_AdcaResultRegs.ADCRESULT1  ;I36 Read ADCRESULT1
    MMOV16    *MAR1, MR2                        ; Store ADCRESULT1
    MBCNDD    _RestartCount, GEQ                ; If count >= NUM_DATA_POINTS
    MMOVIZ    MR1, #0.0                         ; Always executed: MR1=0
```

MMOV16 mem16, MRa (continued)

Move 16-Bit Floating-Point Register Contents to Memory

```
    MNOP
    MNOP
    MMOV16    @_ConversionCount, MR0            ; If branch not taken
    MSTOP                                       ; store current count
_RestartCount
    MMOV16    @_ConversionCount, MR1            ; If branch taken, restart
count
    MSTOP                                       ; end of task
; This task initializes the ConversionCount
; to zero
;
_Cla1Task8:
    MMOVIZ    MR0, #0.0
    MMOV16    @_ConversionCount, MR0
    MSTOP
_ClaT8End:
```

See also

MMOVIZ MRa, #16FHi
MMOVXI MRa, #16FLoHex

MMOV32 mem32, MRa

Move 32-Bit Floating-Point Register Contents to Memory

Operands

MRa	floating-point register (MR0 to MR3)
mem32	32-bit destination memory accessed using one of the available addressing modes

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0111 0100 11aa addr
```

Description

Move from MRa to 32-bit memory location indicated by mem32.

```
[mem32] = MRa;
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

No flags affected.

Pipeline

This is a single-cycle instruction.

Example

```
; Perform 5 multiply and accumulate operations:
;
; X and Y are 32-bit floating point arrays;
; 1st multiply: A = X0 * Y0
; 2nd multiply: B = X1 * Y1
; 3rd multiply: C = X2 * Y2
; 4th multiply: D = X3 * Y3
; 5th multiply: E = X3 * Y3;
; Result = A + B + C + D + E
;
_Cla1Task1:
   MMOVI16       MAR0, #_X               ; MAR0 points to X array
   MMOVI16       MAR1, #_Y               ; MAR1 points to Y array
   MNOP                                  ; Delay for MAR0, MAR1 load
   MNOP                                  ; Delay for MAR0, MAR1 load
                                         ; <-- MAR0 valid
   MMOV32        MR0, *MAR0[2]++         ; MR0 = X0, MAR0 += 2
                                         ; <-- MAR1 valid
   MMOV32        MR1, *MAR1[2]++         ; MR1 = Y0, MAR1 += 2
   MMPYF32       MR2,  MR0, MR1          ; MR2 = A = X0 * Y0
|| MMOV32        MR0, *MAR0[2]++         ; In parallel MR0 = X1, MAR0 += 2
   MMOV32        MR1, *MAR1[2]++         ; MR1 = Y1, MAR1 += 2
   MMPYF32       MR3, MR0, MR1           ; MR3 = B = X1 * Y1
|| MMOV32        MR0, *MAR0[2]++         ; In parallel MR0 = X2, MAR0 += 2
   MMOV32        MR1, *MAR1[2]++         ; MR1 = Y2, MAR2 += 2
   MMACF32       MR3, MR2, MR2, MR0, MR1 ; MR3 = A + B, MR2 = C = X2 * Y2
|| MMOV32        MR0, *MAR0[2]++         ; In parallel MR0 = X3
   MMOV32        MR1, *MAR1[2]++         ; MR1 = Y3
   MMACF32       MR3, MR2, MR2, MR0, MR1 ; MR3 = (A + B) + C, MR2 = D = X3 * Y3
|| MMOV32        MR0, *MAR0              ; In parallel MR0 = X4
   MMOV32        MR1, *MAR1              ; MR1 = Y4
   MMPYF32       MR2, MR0, MR1           ; MR2 = E = X4 * Y4
|| MADDF32       MR3, MR3, MR2           ; in parallel MR3 = (A + B + C) + D
   MADDF32       MR3, MR3, MR2           ; MR3 = (A + B + C + D) + E
   MMOV32        @_Result, MR3           ; Store the result MSTOP ; end of task
```

See also

MMOV32 mem32, MSTF

MMOV32 mem32, MSTF

Move 32-Bit MSTF Register to Memory

Operands

MSTF	floating-point status register
mem32	32-bit destination memory

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0111 0111 0100 addr
```

Description

Copy the CLA's floating-point status register, MSTF, to memory.

```
[mem32] = MSTF;
```

Flags

This instruction does not modify flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

One of the uses of this instruction is to save off the return PC (RPC) prior to calling a function. The decision to jump to a function is made when the MCCNDD is in the decode2 (D2) phase of the pipeline; the RPC is also updated in this phase. The actual jump occurs three cycles later when MCCNDD enters its execution (E) phase. The user must, therefore, save the old RPC before MCCNDD updates it in the D2 phase, that is, it must save MSTF three instructions prior to the function call.

Example

The following example illustrates the pipeline flow for the context save (of the flags and RPC) prior to a function call. The first column in the comments shows the pipeline stages for the MMOV32 instruction while the second column pertains to the MCCNDD instruction.

```
 MMOV32 @_temp, MSTF ; D2|   |
MNOP                 ; R1|F1| MCCNDD is fetched
MNOP                 ; R2|F2|
MNOP                 ; E |D1|
MCCNDD _bar, UNC     ; W |D2| old RPC written to memory,
                     ;   |  | RPC updated with MPC+1
MNOP                 ;   |R1|
MNOP                 ;   |R2|
MNOP                 ;   |E | execution branches to _bar
```

See also

MMOV32 mem32, MRa

MMOV32 MRa, mem32 {, CNDF}

Conditional 32-Bit Move

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
mem32	32-bit memory location accessed using one of the available addressing modes
CNDF	optional condition.

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0111 00cn dfaa addr
```

Description

If the condition is true, then move the 32-bit value referenced by mem32 to the floating-point register indicated by MRa.

```
if (CNDF == TRUE) MRa = [mem32];
```

CNDF is one of the following conditions:

Encode(1)	CNDF	Description	MSTF Flags Tested
0000	NEQ	Not equal to zero	ZF == 0
0001	EQ	Equal to zero	ZF == 1
0010	GT	Greater than zero	ZF == 0 AND NF == 0
0011	GEQ	Greater than or equal to zero	NF == 0
0100	LT	Less than zero	NF == 1
0101	LEQ	Less than or equal to zero	ZF == 1 OR NF == 1
1010	TF	Test flag set	TF == 1
1011	NTF	Test flag not set	TF == 0
1100	LU	Latched underflow	LUF == 1
1101	LV	Latched overflow	LVF == 1
1110	UNC	Unconditional	None
1111	UNCF(2)	Unconditional with flag modification	None

(1) Values not shown are reserved.
(2) This is the default operation if no CNDF field is specified. This condition will allow the ZF and NF flags to be modified when a conditional operation is executed. All other conditions will not modify these flags.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

```
if(CNDF == UNCF)
{
  NF = MRa(31);
  ZF = 0;
  if(MRa(30:23) == 0) { ZF = 1; NF = 0; }
}
  else No flags modified;
```

Pipeline

This is a single-cycle instruction.

MMOV32 MRa, mem32 {, CNDF} (continued)

Conditional 32-Bit Move

Example

```
; Given A, B, X, M1 and M2 are 32-bit floating-point
; numbers
;
; if(A == B) calculate Y = X*M1
; if(A! = B) calculate Y = X*M2
;
_Cla1Task5:
    MMOV32    MR0, @_A
    MMOV32    MR1, @_B
    MCMPF32   MR0, MR1
    MMOV32    MR2, @_M1, EQ  ; if A == B, MR2 = M1
                             ;     Y = M1*X
    MMOV32    MR2, @_M2, NEQ ; if A! = B, MR2 = M2
                             ;     Y = M2*X
    MMOV32    MR3, @_X
    MMPYF32   MR3, MR2, MR3  ; Calculate Y
    MMOV32    @_Y, MR3       ; Store Y
    MSTOP                    ; end of task
```

See also

MMOV32 MRa, MRb {, CNDF}
MMOVD32 MRa, mem32

MMOV32 MRa, MRb {, CNDF}

Conditional 32-Bit Move

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)
CNDF	optional condition.

Opcode

```
LSW: 0000 0000 cndf bbaa
MSW: 0111 1010 1100 0000
```

Description

If the condition is true, then move the 32-bit value in MRb to the floating-point register indicated by MRa.

```
if (CNDF == TRUE) MRa = MRb;
```

CNDF is one of the following conditions:

Encode(1)	CNDF	Description	MSTF Flags Tested
0000	NEQ	Not equal to zero	ZF == 0
0001	EQ	Equal to zero	ZF == 1
0010	GT	Greater than zero	ZF == 0 AND NF == 0
0011	GEQ	Greater than or equal to zero	NF == 0
0100	LT	Less than zero	NF == 1
0101	LEQ	Less than or equal to zero	ZF == 1 OR NF == 1
1010	TF	Test flag set	TF == 1
1011	NTF	Test flag not set	TF == 0
1100	LU	Latched underflow	LUF == 1
1101	LV	Latched overflow	LVF == 1
1110	UNC	Unconditional	None
1111	UNCF(2)	Unconditional with flag modification	None

(1) Values not shown are reserved.
(2) This is the default operation if no CNDF field is specified. This condition will allow the ZF, and NF flags to be modified when a conditional operation is executed. All other conditions will not modify these flags.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

```
if(CNDF == UNCF)
 {
   NF = MRa(31); ZF = 0;
   if(MRa(30:23) == 0) {ZF = 1; NF = 0;}
}
else No flags modified;
```

Pipeline

This is a single-cycle instruction.

MMOV32 MRa, MRb {, CNDF} (continued)

Conditional 32-Bit Move

Example

```
; Given: X = 8.0
;        Y = 7.0
;        A = 2.0
;        B = 5.0
; _ClaTask1
    MMOV32   MR3, @_X      ; MR3 = X = 8.0
    MMOV32   MR0, @_Y      ; MR0 = Y = 7.0
    MMAXF32  MR3, MR0      ; ZF = 0, NF = 0, MR3 = 8.0
    MMOV32   MR1, @_A, GT  ; true, MR1 = A = 2.0
    MMOV32   MR1, @_B, LT  ; false, does not load MR1
    MMOV32   MR2, MR1, GT  ; true, MR2 = MR1 = 2.0
    MMOV32   MR2, MR0, LT  ; false, does not load MR2
    MSTOP
```

See also MMOV32 MRa, mem32 {,CNDF}

MMOV32 MSTF, mem32

Move 32-Bit Value from Memory to the MSTF Register

Operands

MSTF	CLA status register
mem32	32-bit source memory location

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0111 0111 0000 addr
```

Description

Move from memory to the CLA's status register MSTF. This instruction is most useful when nesting function calls (via MCCNDD).

```
MSTF = [mem32];
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	Yes	Yes	Yes	Yes	Yes

Loading the status register will overwrite all flags and the RPC field. The MEALLOW field is not affected.

Pipeline

This is a single-cycle instruction.

See also

MMOV32 mem32, MSTF

MMOVD32 MRa, mem32

Move 32-Bit Value from Memory with Data Copy

Operands

MRa	CLA floating-point register (MR0 to MR3)
mem32	32-bit memory location accessed using one of the available addressing modes

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0111 0100 00aa addr
```

Description

Move the 32-bit value referenced by mem32 to the floating-point register indicated by MRa.

```
MRa = [mem32];
[mem32+2] = [mem32];
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

```
NF = MRa(31);
ZF = 0;
if(MRa(30:23) == 0){ ZF = 1; NF = 0; }
```

Pipeline

This is a single-cycle instruction.

Example

```
; sum = X0*B0 + X1*B1 + X2*B2 + Y1*A1 + Y2*B2
;
;       X2 = X1
;       X1 = X0
;       Y2 = Y1
;       Y1 = sum
;
_Cla1Task2:
    MMOV32 MR0, @_B2        ; MR0 = B2
    MMOV32 MR1, @_X2        ; MR1 = X2
    MMPYF32 MR2, MR1, MR0   ; MR2 = X2*B2
||  MMOV32 MR0, @_B1        ; MR0 = B1
    MMOVD32 MR1, @_X1       ; MR1 = X1, X2 = X1
    MMPYF32 MR3, MR1, MR0   ; MR3 = X1*B1
||  MMOV32 MR0, @_B0        ; MR0 = B0
    MMOVD32 MR1, @_X0       ; MR1 = X0, X1 = X0
; MR3 = X1*B1 + X2*B2, MR2 = X0*B0
; MR0 = A2
    MMACF32 MR3, MR2, MR2, MR1, MR0
||  MMOV32 MR0, @_A2

    MMOV32 MR1, @_Y2        ; MR1 = Y2
; MR3 = X0*B0 + X1*B1 + X2*B2, MR2 = Y2*A2
; MR0 = A1
    MMACF32 MR3, MR2, MR2, MR1, MR0
||  MMOV32 MR0, @_A1
    MMOVD32 MR1,@_Y1        ; MR1 = Y1, Y2 = Y1
    MADDF32 MR3, MR3, MR2   ; MR3 = Y2*A2 + X0*B0 + X1*B1 + X2*B2
||  MMPYF32 MR2, MR1, MR0   ; MR2 = Y1*A1
    MADDF32 MR3, MR3, MR2   ; MR3 = Y1*A1 + Y2*A2 + X0*B0 + X1*B1 + X2*B2
    MMOV32 @_Y1, MR3        ; Y1 = MR3
    MSTOP                   ; end of task
```

MMOVD32 MRa, mem32 (continued)

Move 32-Bit Value from Memory with Data Copy

See also MMOV32 MRa, mem32 {,CNDF}

MMOVF32 MRa, #32F

Load the 32-Bits of a 32-Bit Floating-Point Register

Operands

This instruction is an alias for MMOVIZ and MMOVXI instructions. The second operand is translated by the assembler such that the instruction becomes:

```
MMOVIZ MRa, #16FHiHex MMOVXI MRa, #16FLoHex
```

MRa	CLA floating-point destination register (MR0 to MR3)
#32F	immediate float value represented in floating-point representation

Opcode

```
LSW: IIII IIII IIII IIII (opcode of MMOVIZ MRa, #16FHiHex)
MSW: 0111 1000 0100 00aa
LSW: IIII IIII IIII IIII (opcode of MMOVXI MRa, #16FLoHex)
MSW: 0111 1000 1000 00aa
```

Description

Note: This instruction accepts the immediate operand only in floating-point representation. To specify the immediate value as a hex value (IEEE 32-bit floating- point format) use the MOVI32 MRa, #32FHex instruction.

Load the 32-bits of MRa with the immediate float value represented by #32F.

#32F is a float value represented in floating-point representation. The assembler will only accept a float value represented in floating-point representation. That is, 3.0 can only be represented as #3.0. #0x40400000 will result in an error.

```
MRa = #32F;
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

Depending on #32FH, this instruction takes one or two cycles. If all of the lower 16-bits of the IEEE 32-bit floating-point format of #32F are zeros, then the assembler will convert MMOVF32 into only MMOVIZ instruction. If the lower 16-bits of the IEEE 32-bit floating-point format of #32F are not zeros, then the assembler will convert MMOVF32 into MMOVIZ and MMOVXI instructions.

Example

```
MMOVF32 MR1, #3.0     ; MR1 = 3.0 (0x40400000)
                      ; Assembler converts this instruction as
                      ; MMOVIZ MR1, #0x4040
MMOVF32 MR2, #0.0     ; MR2 = 0.0 (0x00000000)
                      ; Assembler converts this instruction as
                      ; MMOVIZ MR2, #0x0
MMOVF32 MR3, #12.265  ; MR3 = 12.625 (0x41443D71)
                      ; Assembler converts this instruction as
                      ; MMOVIZ MR3, #0x4144
                      ; MMOVXI MR3, #0x3D71
```

See also

MMOVIZ MRa, #16FHi
MMOVXI MRa, #16FLoHex
MMOVI32 MRa, #32FHex

MMOVI16 MARx, #16I

Load the Auxiliary Register with the 16-Bit Immediate Value

Operands

MARx	Auxiliary register MAR0 or MAR1
#16I	16-bit immediate value

Opcode

```
LSW: IIII IIII IIII IIII (opcode of MMOVI16 MAR0, #16I)
MSW: 0111 1111 1100 0000
LSW: IIII IIII IIII IIII (opcode of MMOVI16 MAR1, #16I)
MSW: 0111 1111 1110 0000
```

Description

Load the auxiliary register, MAR0 or MAR1, with a 16-bit immediate value. Refer to the pipeline section for important information regarding this instruction.

```
MARx = #16I;
```

Flags

This instruction does not modify flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction. The immediate load of MAR0 or MAR1 will occur in the EXE phase of the pipeline. Any post increment of MAR0 or MAR1 using indirect addressing will occur in the D2 phase of the pipeline. Therefore the following applies when loading the auxiliary registers:

- **I1 and I2**

  The two instructions following MMOVI16 will use MAR0/MAR1 before the update occurs. Thus these two instructions will use the old value of MAR0 or MAR1.
- **I3**

  Loading of an auxiliary register occurs in the EXE phase while updates due to post-increment addressing occur in the D2 phase. Thus I3 cannot use the auxiliary register or there will be a conflict. In the case of a conflict, the update due to address-mode post increment will win snd the auxiliary register will not be updated with #_X.
- **I4**

  Starting with the 4th instruction MAR0 or MAR1 will be the new value loaded with MMOVI16.

```
;  Assume MAR0 is 50 and #_X is 20
MMOVI16 MAR0, #_X             ; Load MAR0 with address of X (20)
<Instruction 1>          ; I1 Will use the old value of MAR0 (50)
<Instruction 2>          ; I2 Will use the old value of MAR0 (50)
<Instruction 3>          ; I3 Cannot use MAR0
<Instruction 4>          ; I4 Will use the new value of MAR0 (20)
<Instruction 5>          ; I5
....
```

MMOVI16 MARx, #16I (continued)

Load the Auxiliary Register with the 16-Bit Immediate Value

Table 7-18. Pipeline Activity For MMOVI16 MAR0/MAR1, #16I

Instruction	F1	F2	D1	D2	R1	R2	E	W
MMOVI16 MAR0, #_X	MMOVI16							
I1	I1	MMOVI16						
I2	I2	I1	MMOVI16					
I3	I3	I2	I1	MMOVI16				
I4	I4	I3	I2	I1	MMOVI16			
I5	I5	I4	I3	I2	I1	MMOVI16		
I6	I6	I5	I4	I3	I2	I1	MMOVI16	

MMOVI32 MRa, #32FHex

Load the 32-Bits of a 32-Bit Floating-Point Register with the Immediate

Operands

MRa	floating-point register (MR0 to MR3)
#32FHex	A 32-bit immediate value that represents an IEEE 32-bit floating-point value.

This instruction is an alias for MMOVIZ and MMOVXI instructions. The second operand is translated by the assembler such that the instruction becomes:

```
MMOVIZ MRa, #16FHiHex
MMOVXI MRa, #16FLoHex
```

Opcode

```
LSW: IIII IIII IIII IIII (opcode of MMOVIZ MRa, #16FHiHex)
MSW: 0111 1000 0100 00aa
LSW: IIII IIII IIII IIII (opcode of MMOVXI MRa, #16FLoHex)
MSW: 0111 1000 1000 00aa
```

Description

Note: This instruction only accepts a hex value as the immediate operand. To specify the immediate value with a floating-point representation use the MMOVF32 MRa, #32F instruction.

Load the 32-bits of MRa with the immediate 32-bit hex value represented by #32Fhex.

#32Fhex is a 32-bit immediate hex value that represents the IEEE 32-bit floating-point value of a floating-point number. The assembler will only accept a hex immediate value. That is, 3.0 can only be represented as #0x40400000. #3.0 will result in an error.

```
MRa = #32FHex;
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

Depending on #32FHex, this instruction takes one or two cycles. If all of the lower 16-bits of #32FHex are zeros, then assembler will convert MOVI32 to the MMOVIZ instruction. If the lower 16-bits of #32FHex are not zeros, then assembler will convert MOVI32 to a MMOVIZ and a MMOVXI instruction.

Example

```
MOVI32   MR1, #0x40400000 ; MR1 = 0x40400000
                          ; Assembler converts this instruction as
                          ; MMOVIZ MR1, #0x4040
MOVI32  MR2, #0x00000000  ; MR2 = 0x00000000
                          ; Assembler converts this instruction as
                          ; MMOVIZ MR2, #0x0
MOVI32  MR3, #0x40004001  ; MR3 = 0x40004001
                          ; Assembler converts this instruction as
                          ; MMOVIZ MR3, #0x4000
                          ; MMOVXI MR3, #0x4001
MOVI32  MR0, #0x00004040  ; MR0 = 0x00004040
                          ; Assembler converts this instruction as
                          ; MMOVIZ MR0, #0x0000
                          ; MMOVXI MR0, #0x4040
```

MMOVI32 MRa, #32FHex (continued)

Load the 32-Bits of a 32-Bit Floating-Point Register with the Immediate

See also

MMOVIZ MRa, #16FHi
MMOVXI MRa, #16FLoHex
MMOVF32 MRa, #32F

MMOVIZ MRa, #16FHi

Load the Upper 16-Bits of a 32-Bit Floating-Point Register

Operands

MRa	floating-point register (MR0 to MR3)
#16FHi	A 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0.

Opcode

```
LSW: IIII IIII IIII IIII
MSW: 0111 1000 0100 00aa
```

Description

Load the upper 16-bits of MRa with the immediate value #16FHi and clear the low 16-bits of MRa.

#16FHiHex is a 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0. The assembler will only accept a decimal or hex immediate value. That is, -1.5 can be represented as #-1.5 or #0xBFC0.

By itself, MMOVIZ is useful for loading a floating-point register with a constant in which the lowest 16-bits of the mantissa are 0. Some examples are 2.0 (0x40000000), 4.0 (0x40800000), 0.5 (0x3F000000), and -1.5 (0xBFC00000). If a constant requires all 32-bits of a floating-point register to be iniitalized, then use MMOVIZ along with the MMOVXI instruction.

```
MRa(31:16) = #16FHi;
MRa(15:0)  = 0;
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
; Load MR0 and MR1 with -1.5 (0xBFC00000)
     MMOVIZ    MR0, #0xBFC0      ; MR0 = 0xBFC00000 (1.5)
     MMOVIZ    MR1, #-1.5        ; MR1 = -1.5 (0xBFC00000)
; Load MR2 with pi = 3.141593 (0x40490FDB)
     MMOVIZ    MR2, #0x4049      ; MR2 = 0x40490000
     MMOVXI    MR2, #0x0FDB      ; MR2 = 0x40490FDB
```

See also

MMOVF32 MRa, #32F
MMOVI32 MRa, #32FHex
MMOVXI MRa, #16FLoHex

MMOVZ16 MRa, mem16

Load MRx With 16-bit Value

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
mem16	16-bit source memory location

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0111 0101 10aa addr
```

Description

Move the 16-bit value referenced by mem16 to the floating-point register indicated by MRa.

```
MRa(31:16) = 0;
MRa(15:0) = [mem16];
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified based on the integer results of the operation.

```
NF = 0;
if (MRa(31:0)== 0) { ZF = 1; }
```

Pipeline

This is a single-cycle instruction.

MMOVXI MRa, #16FLoHex

Move Immediate to the Low 16-Bits of a Floating-Point Register

Operands

MRa	CLA floating-point register (MR0 to MR3)
#16FLoHex	A 16-bit immediate hex value that represents the lower 16-bits of an IEEE 32-bit floating-point value. The upper 16-bits will not be modified.

Opcode

```
LSW: IIII IIII IIII IIII
MSW: 0111 1000 1000 00aa
```

Description

Load the low 16-bits of MRa with the immediate value #16FLoHex. #16FLoHex represents the lower 16-bits of an IEEE 32-bit floating-point value. The upper 16-bits of MRa will not be modified. MMOVXI can be combined with the MMOVIZ instruction to initialize all 32-bits of a MRa register.

```
MRa(15:0) = #16FLoHex;
MRa(31:16) = Unchanged;
```

Flags

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
; Load MR0 with pi = 3.141593 (0x40490FDB)
    MMOVIZ      MR0,#0x4049   ; MR0 = 0x40490000
    MMOVXI      MR0,#0x0FDB   ; MR0 = 0x40490FDB
```

See also

MMOVIZ MRa, #16FHi

MMPYF32 MRa, MRb, MRc

32-Bit Floating-Point Multiply

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)
MRc	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 00cc bbaa
MSW: 0111 1100 0000 0000
```

Description

Multiply the contents of two floating-point registers.

```
MRa = MRb * MRc;
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MMPYF32 generates an underflow condition.
- LVF = 1 if MMPYF32 generates an overflow condition.

Pipeline

This is a single-cycle instruction.

Example

```
; Calculate Num/Den using a Newton-Raphson algorithum for 1/Den
; Ye = Estimate(1/X)
; Ye = Ye*(2.0 - Ye*X)
; Ye = Ye*(2.0 - Ye*X)
;
_Cla1Task1:
    MMOV32     MR1, @_Den      ; MR1 = Den
    MEINVF32   MR2, MR1        ; MR2 = Ye = Estimate(1/Den)
    MMPYF32    MR3, MR2, MR1   ; MR3 = Ye*Den
    MSUBF32    MR3, #2.0, MR3  ; MR3 = 2.0 - Ye*Den
    MMPYF32    MR2, MR2, MR3   ; MR2 = Ye = Ye*(2.0 - Ye*Den)
    MMPYF32    MR3, MR2, MR1   ; MR3 = Ye*Den
||  MMOV32     MR0, @_Num      ; MR0 = Num
    MSUBF32    MR3, #2.0, MR3  ; MR3 = 2.0 - Ye*Den
    MMPYF32    MR2, MR2, MR3   ; MR2 = Ye = Ye*(2.0 - Ye*Den)
||  MMOV32     MR1, @_Den      ; Reload Den To Set Sign
    MNEGF32    MR0, MR0, EQ    ; if(Den == 0.0) Change Sign Of Num
    MMPYF32    MR0, MR2, MR0   ; MR0 = Y = Ye*Num
    MMOV32     @_Dest, MR0     ; Store result
    MSTOP                      ; end of task
```

See also

MMPYF32 MRa, #16FHi, MRb
MMPYF32 MRa, MRb, MRc || MADDF32 MRd, MRe, MRf
MMPYF32 MRd, MRe, MRf || MMOV32 MRa, mem32
MMPYF32 MRd, MRe, MRf || MMOV32 mem32, MRa
MMPYF32 MRa, MRb, MRc || MSUBF32 MRd, MRe, MRf
MMACF32 MR3, MR2, MRd, MRe, MRf || MMOV32 MRa, mem32

MMPYF32 MRa, #16FHi, MRb

32-Bit Floating-Point Multiply

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
#16FHi	A 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0.
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: IIII IIII IIII IIII
MSW: 0111 0111 1000 bbaa
```

Description

Multiply MRb with the floating-point value represented by the immediate operand. Store the result of the addition in MRa.

#16FHi is a 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0. #16FHi is most useful for representing constants where the lowest 16-bits of the mantissa are 0. Some examples are 2.0 (0x40000000), 4.0 (0x40800000), 0.5 (0x3F000000), and -1.5 (0xBFC00000). The assembler will accept either a hex or float as the immediate value. That is, the value -1.5 can be represented as #-1.5 or #0xBFC0.

```
MRa = MRb * #16FHi:0;
```

This instruction can also be written as MMPYF32 MRa, MRb, #16FHi.

Flags

This instruction modifies the following flags in the MSTF register:.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MMPYF32 generates an underflow condition.
- LVF = 1 if MMPYF32 generates an overflow condition.

Pipeline

This is a single-cycle instruction.

Example 1

```
; Same as example 2 but #16FHi is represented in float
    MMOVIZ    MR3, #2.0       ; MR3 = 2.0 (0x40000000)
    MMPYF32   MR0, #3.0, MR3 ; MR0 = 3.0 * MR3 = 6.0 (0x40C00000)
    MMOV32    @_X, MR0        ; Save the result in variable X
```

Example 2

```
; Same as example 1 but #16FHi is represented in Hex
    MMOVIZ    MR3, #2.0          ; MR3 = 2.0 (0x40000000)
    MMPYF32   MR0, #0x4040, MR3 ; MR0 = 0x4040 * MR3 = 6.0 (0x40C00000)
    MMOV32    @_X, MR0           ; Save the result in variable X
```

MMPYF32 MRa, #16FHi, MRb (continued)

32-Bit Floating-Point Multiply

Example 3

```
; Given X, M and B are IQ24 numbers:
; X = IQ24(+2.5) = 0x02800000
; M = IQ24(+1.5) = 0x01800000
; B = IQ24(-0.5) = 0xFF800000
;
; Calculate Y = X * M + B
;
;
_Cla1Task2:
;
; Convert M, X and B from IQ24 to float
     MI32TOF32   MR0, @_M          ; MR0 = 0x4BC00000
     MI32TOF32   MR1, @_X          ; MR1 = 0x4C200000
     MI32TOF32   MR2, @_B          ; MR2 = 0xCB000000
     MMPYF32     MR0, MR0, #0x3380 ; M = 1/(1*2^24) * iqm = 1.5 (0x3FC00000)
     MMPYF32     MR1, MR1, #0x3380 ; X = 1/(1*2^24) * iqx = 2.5 (0x40200000)
     MMPYF32     MR2, MR2, #0x3380 ; B = 1/(1*2^24) * iqb = -.5 (0xBF000000)
     MMPYF32     MR3, MR0, MR1     ; M*X
     MADDF32     MR2, MR2, MR3     ; Y=MX+B = 3.25 (0x40500000)
; Convert Y from float32 to IQ24
     MMPYF32     MR2, MR2, #0x4B80 ; Y * 1*2^24
     MF32TOI32   MR2, MR2          ; IQ24(Y) = 0x03400000
     MMOV32 @_Y, MR2               ; store result
     MSTOP                         ; end of task
```

See also

MMPYF32 MRa, MRb, #16FHi
MMPYF32 MRa, MRb, MRc
MMPYF32 MRa, MRb, MRc || MADDF32 MRd, MRe, MRf

MMPYF32 MRa, MRb, #16FHi

32-Bit Floating-Point Multiply

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)
#16FHi	A 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0.

Opcode

```
LSW: IIII IIII IIII IIII
MSW: 0111 0111 1000 bbaa
```

Description

Multiply MRb with the floating-point value represented by the immediate operand. Store the result of the addition in MRa.

#16FHi is a 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0. #16FHi is most useful for representing constants where the lowest 16-bits of the mantissa are 0. Some examples are 2.0 (0x40000000), 4.0 (0x40800000), 0.5 (0x3F000000), and -1.5 (0xBFC00000). The assembler will accept either a hex or float as the immediate value. That is, the value -1.5 can be represented as #-1.5 or #0xBFC0.

```
MRa = MRb * #16FHi:0;
```

This instruction can also be writen as MMPYF32 MRa, #16FHi, MRb.

Flags

This instruction modifies the following flags in the MSTF register:.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MMPYF32 generates an underflow condition.
- LVF = 1 if MMPYF32 generates an overflow condition.

Pipeline

This is a single-cycle instruction.

Example 1

```
;Same as example 2 but #16FHi is represented in float
    MMOVIZ    MR3, #2.0        ; MR3 = 2.0 (0x40000000)
    MMPYF32   MR0, MR3, #3.0   ; MR0 = MR3 * 3.0 = 6.0 (0x40C00000)
    MMOV32    @_X, MR0         ; Save the result in variable X
```

Example 2

```
;Same as example 1 but #16FHi is represented in Hex
    MMOVIZ    MR3, #2.0           ; MR3 = 2.0 (0x40000000)
    MMPYF32   MR0, MR3, #0x4040   ; MR0 = MR3 * 0x4040 = 6.0 (0x40C00000)
    MMOV32    @_X, MR0            ; Save the result in variable X
```

MMPYF32 MRa, MRb, #16FHi (continued)

32-Bit Floating-Point Multiply

Example 3

```
; Given X, M and B are IQ24 numbers:
; X = IQ24(+2.5) = 0x02800000
; M = IQ24(+1.5) = 0x01800000
; B = IQ24(-0.5) = 0xFF800000
;
; Calculate Y = X * M + B
;
_Cla1Task2:
;
; Convert M, X and B from IQ24 to float
     MI32TOF32   MR0, @_M          ; MR0 = 0x4BC00000
     MI32TOF32   MR1, @_X          ; MR1 = 0x4C200000
     MI32TOF32   MR2, @_B          ; MR2 = 0xCB000000
     MMPYF32     MR0, #0x3380, MR0 ; M = 1/(1*2^24) * iqm = 1.5 (0x3FC00000)
     MMPYF32     MR1, #0x3380, MR1 ; X = 1/(1*2^24) * iqx = 2.5 (0x40200000)
     MMPYF32     MR2, #0x3380, MR2 ; B = 1/(1*2^24) * iqb = -.5 (0xBF000000)
     MMPYF32     MR3, MR0, MR1     ; M*X
     MADDF32     MR2, MR2, MR3     ; Y=MX+B = 3.25 (0x40500000)
; Convert Y from float32 to IQ24
     MMPYF32     MR2, #0x4B80, MR2 ; Y * 1*2^24
     MF32TOI32   MR2, MR2          ; IQ24(Y) = 0x03400000
     MMOV32      @_Y, MR2          ; store result
     MSTOP                         ; end of task
```

See also

MMPYF32 MRa, #16FHi, MRb
MMPYF32 MRa, MRb, MRc

MMPYF32 MRa, MRb, MRc||MADDF32 MRd, MRe, MRf

32-Bit Floating-Point Multiply with Parallel Add

Operands

MRa	CLA floating-point destination register for MMPYF32 (MR0 to MR3) MRa cannot be the same register as MRd
MRb	CLA floating-point source register for MMPYF32 (MR0 to MR3)
MRc	CLA floating-point source register for MMPYF32 (MR0 to MR3)
MRd	CLA floating-point destination register for MADDF32 (MR0 to MR3) MRd cannot be the same register as MRa
MRe	CLA floating-point source register for MADDF32 (MR0 to MR3)
MRf	CLA floating-point source register for MADDF32 (MR0 to MR3)

Opcode

```
LSW: 0000 ffee ddcc bbaa
MSW: 0111 1010 0000 0000
```

Description

Multiply the contents of two floating-point registers with parallel addition of two registers.

```
MRa = MRb * MRc;
MRd = MRe + MRf;
```

Restrictions

The destination register for the MMPYF32 and the MADDF32 must be unique. That is, MRa cannot be the same register as MRd.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MMPYF32 or MADDF32 generates an underflow condition.
- LVF = 1 if MMPYF32 or MADDF32 generates an overflow condition.

Pipeline

Both MMPYF32 and MADDF32 complete in a single cycle.

MMPYF32 MRa, MRb, MRc||MADDF32 MRd, MRe, MRf (continued)

32-Bit Floating-Point Multiply with Parallel Add

Example

```
; Perform 5 multiply and accumulate operations:
;
; X and Y are 32-bit floating point arrays
;
; 1st multiply: A = X0 * Y0
; 2nd multiply: B = X1 * Y1
; 3rd multiply: C = X2 * Y2
; 4th multiply: D = X3 * Y3
; 5th multiply: E = X3 * Y3
;
; Result = A + B + C + D + E
;
_Cla1Task1:
    MMOVI16    MAR0, #_X               ; MAR0 points to X array
    MMOVI16    MAR1, #_Y               ; MAR1 points to Y array
    MNOP                               ; Delay for MAR0, MAR1 load
    MNOP                               ; Delay for MAR0, MAR1 load
                                       ; <-- MAR0 valid
    MMOV32     MR0, *MAR0[2]++         ; MR0 = X0, MAR0 += 2
                                       ; <-- MAR1 valid
    MMOV32     MR1, *MAR1[2]++         ; MR1 = Y0, MAR1 += 2
    MMPYF32    MR2, MR0, MR1           ; MR2 = A = X0 * Y0
||  MMOV32     MR0, *MAR0[2]++         ; In parallel MR0 = X1, MAR0 += 2
    MMOV32     MR1, *MAR1[2]++         ; MR1 = Y1, MAR1 += 2
    MMPYF32    MR3, MR0, MR1           ; MR3 = B = X1 * Y1
||  MMOV32     MR0, *MAR0[2]++         ; In parallel MR0 = X2, MAR0 += 2
    MMOV32     MR1, *MAR1[2]++         ; MR1 = Y2, MAR2 += 2
    MMACF32    MR3, MR2, MR2, MR0, MR1 ; MR3 = A + B, MR2 = C = X2 * Y2
||  MMOV32     MR0, *MAR0[2]++         ; In parallel MR0 = X3
    MMOV32     MR1, *MAR1[2]++         ; MR1 = Y3
    MMACF32    MR3, MR2, MR2, MR0, MR1 ; MR3 = (A + B) + C, MR2 = D = X3 * Y3
||  MMOV32     MR0, *MAR0              ; In parallel MR0 = X4
    MMOV32     MR1, *MAR1              ; MR1 = Y4
    MMPYF32    MR2, MR0, MR1           ; MR2 = E = X4 * Y4
||  MADDF32    MR3, MR3, MR2           ; in parallel MR3 = (A + B + C) + D

    MADDF32    MR3, MR3, MR2           ; MR3 = (A + B + C + D) + E
    MMOV32     @_Result, MR3           ; Store the result
    MSTOP                              ; end of task
```

See also MMACF32 MR3, MR2, MRd, MRe, MRf || MMOV32 MRa, mem32

MMPYF32 MRd, MRe, MRf ||MMOV32 MRa, mem32

32-Bit Floating-Point Multiply with Parallel Move

Operands

MRd	CLA floating-point destination register for MMPYF32 (MR0 to MR3) MRd cannot be the same register as MRa
MRe	CLA floating-point source register for MMPYF32 (MR0 to MR3)
MRf	CLA floating-point source register for MMPYF32 (MR0 to MR3)
MRa	CLA floating-point destination register for MMOV32 (MR0 to MR3) MRa cannot be the same register as MRd
mem32	32-bit memory location accessed using one of the available addressing modes. This will be the source of MMOV32.

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0000 ffee ddaa addr
```

Description

Multiply the contents of two floating-point registers and load another.

```
MRd = MRe * MRf;
MRa = [mem32];
```

Restrictions

The destination register for the MMPYF32 and the MMOV32 must be unique. That is, MRa cannot be the same register as MRd.

Flags

This instruction modifies the following flags in the MSTF register:.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MMPYF32 generates an underflow condition.
- LVF = 1 if MMPYF32 generates an overflow condition.

The MMOV32 Instruction will set the NF and ZF flags as follows:

```
NF = MRa(31);
ZF = 0;
if(MRa(30:23) == 0) { ZF = 1; NF = 0; }
```

Pipeline

Both MMPYF32 and MMOV32 complete in a single cycle.

Example 1

```
; Given M1, X1 and B1 are 32-bit floating point
; Calculate Y1 = M1*X1+B1
;
_Cla1Task1:
    MMOV32     MR0, @M1         ; Load MR0 with M1
    MMOV32     MR1, @X1         ; Load MR1 with X1
    MMPYF32    MR1, MR1, MR0    ; Multiply M1*X1
||  MMOV32     MR0, @B1         ; and in parallel load MR0 with B1
    MADDF32    MR1, MR1, MR0    ; Add M*X1 to B1 and store in MR1
    MMOV32     @Y1, MR1         ; Store the result
    MSTOP                       ; end of task
```

MMPYF32 MRd, MRe, MRf ||MMOV32 MRa, mem32 (continued)

32-Bit Floating-Point Multiply with Parallel Move

Example 2

```
; Given A, B and C are 32-bit floating-point numbers
; Calculate Y2 = (A * B)
;           Y3 = (A * B) * C
;
_Cla1Task2:
    MMOV32    MR0, @A         ; Load MR0 with A
    MMOV32    MR1, @B         ; Load MR1 with B
    MMPYF32   MR1, MR1, MR0   ; Multiply A*B
||  MMOV32    MR0, @C         ; and in parallel load MR0 with C
    MMPYF32   MR1, MR1, MR0   ; Multiply (A*B) by C
||  MMOV32    @Y2, MR1        ; and in parallel store A*B
    MMOV32    @Y3, MR1        ; Store the result
    MSTOP                     ; end of task
```

See also

MMPYF32 MRd, MRe, MRf || MMOV32 mem32, MRa
MMACF32 MR3, MR2, MRd, MRe, MRf || MMOV32 MRa, mem32

MMPYF32 MRd, MRe, MRf ||MMOV32 mem32, MRa

32-Bit Floating-Point Multiply with Parallel Move

Operands

MRd	CLA floating-point destination register for MMPYF32 (MR0 to MR3)
MRe	CLA floating-point source register for MMPYF32 (MR0 to MR3)
MRf	CLA floating-point source register for MMPYF32 (MR0 to MR3)
mem32	32-bit memory location accessed using one of the available addressing modes. This will be the destination of MMOV32.
MRa	CLA floating-point source register for MMOV32 (MR0 to MR3)

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0100 ffee ddaa addr
```

Description

Multiply the contents of two floating-point registers and move from memory to register.

```
MRd = MRe * MRf;
[mem32] = MRa;
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MMPYF32 generates an underflow condition.
- LVF = 1 if MMPYF32 generates an overflow condition.

Pipeline

MMPYF32 and MMOV32 both complete in a single cycle.

Example

```
; Given A, B and C are 32-bit floating-point numbers
; Calculate Y2 = (A * B)
;           Y3 = (A * B) * C
;
_Cla1Task2:
     MMOV32    MR0, @A         ; Load MR0 with A
     MMOV32    MR1, @B         ; Load MR1 with B
     MMPYF32   MR1, MR1, MR0   ; Multiply A*B
||   MMOV32    MR0, @C         ; and in parallel load MR0 with C
     MMPYF32   MR1, MR1, MR0   ; Multiply (A*B) by C
||   MMOV32    @Y2, MR1        ; and in parallel store A*B
     MMOV32    @Y3, MR1        ; Store the result
     MSTOP                     ; end of task
```

See also

MMPYF32 MRd, MRe, MRf || MMOV32 MRa, mem32
MMACF32 MR3, MR2, MRd, MRe, MRf || MMOV32 MRa, mem32

MMPYF32 MRa, MRb, MRc ||MSUBF32 MRd, MRe, MRf

32-Bit Floating-Point Multiply with Parallel Subtract

Operands

MRa	CLA floating-point destination register for MMPYF32 (MR0 to MR3) MRa cannot be the same register as MRd
MRb	CLA floating-point source register for MMPYF32 (MR0 to MR3)
MRc	CLA floating-point source register for MMPYF32 (MR0 to MR3)
MRd	CLA floating-point destination register for MSUBF32 (MR0 to MR3) MRd cannot be the same register as MRa
MRe	CLA floating-point source register for MSUBF32 (MR0 to MR3)
MRf	CLA floating-point source register for MSUBF32 (MR0 to MR3)

Opcode

```
LSW: 0000 ffee ddcc bbaa
MSW: 0111 1010 0100 0000
```

Description

Multiply the contents of two floating-point registers with parallel subtraction of two registers.

```
MRa = MRb * MRc;
MRd = MRe - MRf;
```

Restrictions

The destination register for the MMPYF32 and the MSUBF32 must be unique. That is, MRa cannot be the same register as MRd.

Flags

This instruction modifies the following flags in the MSTF register:.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MMPYF32 or MSUBF32 generates an underflow condition.
- LVF = 1 if MMPYF32 or MSUBF32 generates an overflow condition.

Pipeline

MMPYF32 and MSUBF32 both complete in a single cycle.

Example

```
; Given A, B and C are 32-bit floating-point numbers
; Calculate Y2 = (A * B)
;           Y3 = (A - B)
;
_Cla1Task2:
     MMOV32   MR0, @A          ; Load MR0 with A
     MMOV32   MR1, @B          ; Load MR1 with B
     MMPYF32  MR2, MR0, MR1    ; Multiply (A*B)
||   MSUBF32  MR3, MR0, MR1    ; and in parallel Sub (A-B)
     MMOV32   @Y2, MR2         ; Store A*B
     MMOV32   @Y3, MR3         ; Store A-B
     MSTOP                     ; end of task
```

See also

MSUBF32 MRa, MRb, MRc
MSUBF32 MRd, MRe, MRf || MMOV32 MRa, mem32
MSUBF32 MRd, MRe, MRf || MMOV32 mem32, MRa

MNEGF32 MRa, MRb{, CNDF}

Conditional Negation

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)
CNDF	condition tested

Opcode

```
LSW: 0000 0000 cndf bbaa
MSW: 0111 1010 1000 0000
```

Description

```
if (CNDF == true) {MRa = - MRb; }
else {MRa = MRb; }
```

CNDF is one of the following conditions:

Encode(1)	CNDF	Description	MSTF Flags Tested
0000	NEQ	Not equal to zero	ZF == 0
0001	EQ	Equal to zero	ZF == 1
0010	GT	Greater than zero	ZF == 0 AND NF == 0
0011	GEQ	Greater than or equal to zero	NF == 0
0100	LT	Less than zero	NF == 1
0101	LEQ	Less than or equal to zero	ZF == 1 OR NF == 1
1010	TF	Test flag set	TF == 1
1011	NTF	Test flag not set	TF == 0
1100	LU	Latched underflow	LUF == 1
1101	LV	Latched overflow	LVF == 1
1110	UNC	Unconditional	None
1111	UNCF(2)	Unconditional with flag modification	None

(1) Values not shown are reserved.
(2) This is the default operation if no CNDF field is specified. This condition will allow the ZF, and NF flags to be modified when a conditional operation is executed. All other conditions will not modify these flags.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

Pipeline

This is a single-cycle instruction.

MNEGF32 MRa, MRb{, CNDF} (continued)

Conditional Negation

Example 1

```
; Show the basic operation of MNEGF32
;
     MMOVIZ    MR0, #5.0       ; MR0 = 5.0 (0x40A00000)
     MMOVIZ    MR1, #4.0       ; MR1 = 4.0 (0x40800000)
     MMOVIZ    MR2, #-1.5      ; MR2 = -1.5 (0xBFC00000)
     MMPYF32   MR3, MR1, MR2   ; MR3 = -6.0
     MMPYF32   MR0, MR0, MR1   ; MR0 = 20.0
     MMOVIZ    MR1, #0.0
     MCMPF32   MR3, MR1        ; NF = 1
     MNEGF32   MR3, MR3, LT    ; if NF = 1, MR3 = 6.0
     MCMPF32   MR0, MR1        ; NF = 0
     MNEGF32   MR0, MR0, GEQ   ; if NF = 0, MR0 = -20.0
```

Example 2

```
; Calculate Num/Den using a Newton-Raphson algorithum for 1/Den
; Ye = Estimate(1/X)
; Ye = Ye*(2.0 - Ye*X)
; Ye = Ye*(2.0 - Ye*X)
;
_Cla1Task1:
    MMOV32    MR1, @_Den      ; MR1 = Den
    MEINVF32  MR2, MR1        ; MR2 = Ye = Estimate(1/Den)
    MMPYF32   MR3, MR2, MR1   ; MR3 = Ye*Den
    MSUBF32   MR3, #2.0, MR3  ; MR3 = 2.0 - Ye*Den
    MMPYF32   MR2, MR2, MR3   ; MR2 = Ye = Ye*(2.0 - Ye*Den)
    MMPYF32   MR3, MR2, MR1   ; MR3 = Ye*Den
||  MMOV32    MR0, @_Num      ; MR0 = Num
    MSUBF32   MR3, #2.0, MR3  ; MR3 = 2.0 - Ye*Den
    MMPYF32   MR2, MR2, MR3   ; MR2 = Ye = Ye*(2.0 - Ye*Den)
||  MMOV32    MR1, @_Den      ; Reload Den To Set Sign
    MNEGF32   MR0, MR0, EQ    ; if(Den == 0.0) Change Sign Of Num
    MMPYF32   MR0, MR2, MR0   ; MR0 = Y = Ye*Num
    MMOV32    @_Dest, MR0     ; Store result
    MSTOP                     ; end of task
```

See also MABSF32 MRa, MRb

MNOP

No Operation

Operands

none	This instruction does not have any operands

Opcode

```
LSW: 0000 0000 0000 0000
MSW: 0111 1111 1010 0000
```

Description

Do nothing. This instruction is used to fill required pipeline delay slots when other instructions are not available to fill the slots.

Flags

This instruction does not modify flags in the MSTF register.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
; X is an array of 32-bit floating-point values
; Find the maximum value in an array X
; and store it in Result
;
_Cla1Task1:
    MMOVI16    MAR1,#_X          ; Start address
    MUI16TOF32 MR0, @_len        ; Length of the array
    MNOP                         ; delay for MAR1 load
    MNOP                         ; delay for MAR1 load
    MMOV32     MR1, *MAR1[2]++   ; MR1 = X0
LOOP
    MMOV32     MR2, *MAR1[2]++   ; MR2 = next element
    MMAXF32    MR1, MR2          ; MR1 = MAX(MR1, MR2)
    MADDF32    MR0, MR0, #-1.0   ; Decrememt the counter
    MCMPF32    MR0 #0.0          ; Set/clear flags for MBCNDD
    MNOP                         ; Too late to affect MBCNDD
    MNOP                         ; Too late to affect MBCNDD
    MNOP                         ; Too late to affect MBCNDD
    MBCNDD    LOOP, NEQ          ; Branch if not equal to zero
    MMOV32    @_Result, MR1      ; Always executed
    MNOP                         ; Pad to seperate MBCNDD and MSTOP
    MNOP                         ; Pad to seperate MBCNDD and MSTOP
    MSTOP                        ; End of task
```

MOR32 MRa, MRb, MRc

Bitwise OR

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)
MRc	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 00cc bbaa
MSW: 0111 1100 1000 0000
```

Description

Bitwise OR of MRb with MRc.

```
MARa(31:0) = MARb(31:0) OR MRc(31:0);
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified based on the integer results of the operation.

```
NF = MRa(31);
ZF = 0;
if(MRa(31:0) == 0) { ZF = 1; }
```

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ    MR0, #0x5555 ; MR0 = 0x5555AAAA
MMOVXI    MR0, #0xAAAA
MMOVIZ    MR1, #0x5432 ; MR1 = 0x5432FEDC
MMOVXI    MR1, #0xFEDC
                       ; 0101 OR 0101 = 0101 (5)
                       ; 0101 OR 0100 = 0101 (5)
                       ; 0101 OR 0011 = 0111 (7)
                       ; 0101 OR 0010 = 0111 (7)
                       ; 1010 OR 1111 = 1111 (F)
                       ; 1010 OR 1110 = 1110 (E)
                       ; 1010 OR 1101 = 1111 (F)
                       ; 1010 OR 1100 = 1110 (E)
MOR32 MR2, MR1, MR0    ; MR3 = 0x5555FEFE
```

See also

MAND32 MRa, MRb, MRc
MXOR32 MRa, MRb, MRc

MRCNDD {CNDF}

Return Conditional Delayed

Operands

CNDF	optional condition.

Opcode

```
LSW: 0000 0000 0000 0000
MSW: 0111 1001 1010 cndf
```

Description

If the specified condition is true, then the RPC field of MSTF is loaded into MPC and fetching continues from that location. Otherwise program fetches will continue without the return.

Please refer to the pipeline section for important information regarding this instruction.

```
if (CNDF == TRUE) MPC = RPC;
```

CNDF is one of the following conditions:

Encode[1]	CNDF	Description	MSTF Flags Tested
0000	NEQ	Not equal to zero	ZF == 0
0001	EQ	Equal to zero	ZF == 1
0010	GT	Greater than zero	ZF == 0 AND NF == 0
0011	GEQ	Greater than or equal to zero	NF == 0
0100	LT	Less than zero	NF == 1
0101	LEQ	Less than or equal to zero	ZF == 1 OR NF == 1
1010	TF	Test flag set	TF == 1
1011	NTF	Test flag not set	TF == 0
1100	LU	Latched underflow	LUF == 1
1101	LV	Latched overflow	LVF == 1
1110	UNC	Unconditional	None
1111	UNCF[2]	Unconditional with flag modification	None

(1) Values not shown are reserved.
(2) This is the default operation if no CNDF field is specified. This condition will allow the ZF and NF flags to be modified when a conditional operation is executed. All other conditions will not modify these flags.

Flags

This instruction does not modify flags in the MSTF register.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

The MRCNDD instruction by itself is a single-cycle instruction. As shown in Table 7-19, for each return 6 instruction slots are executed; three before the return instruction (d5-d7) and three after the return instruction (d8-d10). The total number of cycles for a return taken or not taken depends on the usage of these slots. That is, the number of cycles depends on how many slots are filled with a MNOP as well as which slots are filled. The effective number of cycles for a return can, therefore, range from 1 to 7 cycles. The number of cycles for a return taken may not be the same as for a return not taken.

MRCNDD {CNDF} (continued)

Return Conditional Delayed

Referring to the following code fragment and the pipeline diagrams in Table 7-19 and Table 7-20, the instructions before and after MRCNDD have the following properties:

```
;
;
<Instruction 1>     ; I1 Last instruction that can affect flags for
                    ; the MCCNDD operation
<Instruction 2>     ; I2 Cannot be stop, branch, call or return
<Instruction 3>     ; I3 Cannot be stop, branch, call or return
<Instruction 4>     ; I4 Cannot be stop, branch, call or return
MCCNDD _func, NEQ   ; Call to func if not eqal to zero
                    ; Three instructions after MCCNDD are always
                    ; executed whether the call is taken or not
<Instruction 5>     ; I5 Cannot be stop, branch, call or return
<Instruction 6>     ; I6 Cannot be stop, branch, call or return
<Instruction 7>     ; I7 Cannot be stop, branch, call or return
<Instruction 8>     ; I8 The address of this instruction is saved
                    ; in the RPC field of the MSTF register.
                    ; Upon return this value is loaded into MPC
                    ; and fetching continues from this point.
<Instruction 9>     ; I9
<Instruction 10>    ; I10
....
....
_func:
<Destination 1>     ; d1 Can be any instruction
<Destination 2>     ; d2
<Destination 3>     ; d3
<Destination 4>     ; d4 Last instruction that can affect flags for
                    ; the MRCNDD operation
<Destination 5>     ; d5 Cannot be stop, branch, call or return
<Destination 6>     ; d6 Cannot be stop, branch, call or return
<Destination 7>     ; d7 Cannot be stop, branch, call or return
MRCNDD NEQ          ; Return to <Instruction 8> if not equal to zero
                    ; Three instructions after MRCNDD are always
                    ; executed whether the return is taken or not
<Destination 8>     ; d8 Cannot be stop, branch, call or return
<Destination 9>     ; d9 Cannot be stop, branch, call or return
<Destination 10>    ; d10 Cannot be stop, branch, call or return
<Destination 11>    ; d11
<Destination 12>    ; d12
....
....
MSTOP
....
```

- **d4**
  - d4 is the last instruction that can effect the CNDF flags for the MRCNDD instruction. The CNDF flags are tested in the D2 phase of the pipeline. That is, a decision is made whether to return or not when MRCNDD is in the D2 phase.
  - There are no restrictions on the type of instruction for d4.
- **d5, d6 and d7**
  - The three instructions proceeding MRCNDD can change MSTF flags but will have no effect on whether the MRCNDD instruction makes the return or not. This is because the flag modification will occur after the D2 phase of the MRCNDD instruction.
  - These instructions must not be the following: MSTOP, MDEBUGSTOP, MBCNDD, MCCNDD or MRCNDD.
- **d8, d9 and d10**
  - The three instructions following MRCNDD are always executed irrespective of whether the return is taken or not.
  - These instructions must not be the following: MSTOP, MDEBUGSTOP, MBCNDD, MCCNDD or MRCNDD.

MRCNDD {CNDF} (continued)

Return Conditional Delayed

Table 7-19. Pipeline Activity For MRCNDD, Return Not Taken

Instruction	F1	F2	D1	D2	R1	R2	E	W
d4	d4	d3	d2	d1	I7	I6	I5	
d5	d5	d4	d3	d2	d1	I7	I6	
d6	d6	d5	d4	d3	d2	d1	i7	
d7	d7	d6	d5	d4	d3	d2	d1	
MRCNDD	MRCNDD	d7	d6	d5	d4	d3	d2	
d8	d8	MRCNDD	d7	d6	d5	d4	d3	
d9	d9	d8	MRCNDD	d7	d6	d5	d4	
d10	d10	d9	d8	MRCNDD	d7	d6	d5	
d11	d11	d10	d9	d8	-	d7	d6	
d12	d12	d11	d10	d9	d8	-	d7	
etc....	....	d12	d11	d10	d9	d8	-	
....	....	....	d12	d11	d10	d9	d8	
....	....	....	....	d12	d11	d10	d9	
					d12	d11	d10	
						d12	d11	
							d12	

Table 7-20. Pipeline Activity For MRCNDD, Return Taken

Instruction	F1	F2	D1	D2	R1	R2	E	W
d4	d4	d3	d2	d1	I7	I6	I5	
d5	d5	d4	d3	d2	d1	I7	I6	
d6	d6	d5	d4	d3	d2	d1	i7	
d7	d7	d6	d5	d4	d3	d2	d1	
MRCNDD	MRCNDD	d7	d6	d5	d4	d3	d2	
d8	d8	MRCNDD	d7	d6	d5	d4	d3	
d9	d9	d8	MRCNDD	d7	d6	d5	d4	
d10	d10	d9	d8	MRCNDD	d7	d6	d5	
I8	I8	d10	d9	d8	-	d7	d6	
I9	I9	I8	d10	d9	d8	-	d7	
I10	I10	I9	I8	d10	d9	d8	-	
etc....	....	I10	I9	I8	d10	d9	d8	
....	....		I10	I9	I8	d10	d9	
....	....			I10	I9	I8	d10	
					I10	I9	I8	
						I10	I9	
							I10	

See also

MBCNDD #16BitDest, CNDF
MCCNDD 16BitDest, CNDF
MMOV32 mem32, MSTF
MMOV32 MSTF, mem32

MSETC BGINTM

Set Background Task Interrupt Mask

Operands

none	This instruction does not have any operands

Opcode

```
LSW: 0000 0000 0000 0000
MSW: 0111 1111 0101 0000
```

Description

This instruction will set the background task interrupt mask (BGINTM) bit in the MSTSBGRND register, making any code thereafter un-interruptible. No other higher priority task will be able to interrupt the background task until the BGINTM is cleared. This instruction sets the BGINTM bit at the end of its D2 phase.

Note: This instruction does not require the MEALLOW bit to be asserted before, or de-asserted after, setting BGINTM.

Flags

This instruction does not modify the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
MSETC BGINTM          ; Set the MSTSBGRND.BGINTM bit
                      ; to prevent any other tasks from
                      ; interrupting the background task
```

See also

MCLRC BGINTM

MSETFLG FLAG, VALUE

Set or Clear Selected Floating-Point Status Flags

Operands

FLAG	8-bit mask indicating which floating-point status flags to change.
VALUE	8-bit mask indicating the flag value: 0 or 1.

Opcode

```
LSW: FFFF FFFF VVVV VVVV
MSW: 0111 1001 1100 0000
```

Description

The MSETFLG instruction is used to set or clear selected floating-point status flags in the MSTF register. The FLAG field is an 11-bit value that indicates which flags will be changed. That is, if a FLAG bit is set to 1 it indicates that flag will be changed; all other flags will not be modified. The bit mapping of the FLAG field is:

9	8	7	6	5	4	3	2	1	0
RNDF32	Reserved		TF	Reserved		ZF	NF	LUF	LVF

The VALUE field indicates the value the flag should be set to: 0 or 1.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	Yes	Yes	Yes	Yes	Yes

Any flag can be modified by this instruction. The MEALLOW and RPC fields cannot be modified with this instruction.

Pipeline

This is a single-cycle instruction.

Example

To make it easier and legible, the assembler accepts a FLAG=VALUE syntax for the MSTFLG operation as shown below:

```
MSETFLG RNDF32=0, TF=0, NF=1; FLAG = 11000100; VALUE = 00XXX1XX;
```

See also

MMOV32 mem32, MSTF
MMOV32 MSTF, mem32

MSTOP

Stop Task

Operands

none	This instruction does not have any operands

Opcode

```
LSW: 0000 0000 0000 0000
MSW: 0111 1111 1000 0000
```

Description

The MSTOP instruction must be placed to indicate the end of each task. In addition, placing MSTOP in unused memory locations within the CLA program RAM can be useful for debugging and preventing run away CLA code. When MSTOP enters the D2 phase of the pipeline, the MIRUN flag for the task is cleared and the associated interrupt is flagged in the PIE vector table.

There are three special cases that can occur when single-stepping a task such that the MPC reaches the MSTOP instruction.

1. If you are single-stepping or halted in "task A" and "task B" comes in before the MPC reaches the MSTOP, then "task B" will start if you continue to step through the MSTOP instruction. Basically if "task B" is pending before the MPC reaches MSTOP in "task A" then there is no issue in "task B" starting and no special action is required.
2. In this case you have single-stepped or halted in "task A" and the MPC has reached the MSTOP with no tasks pending. If "task B" comes in at this point, it will be flagged in the MIFR register but it may or may not start if you continue to single-step through the MSTOP instruction of "task A". It depends on exactly when the new task comes in. To reliably start "task B" perform a soft reset and reconfigure the MIER bits. Once this is done, you can start single-stepping "task B".
3. Case 2 can be handled slightly differently if there is control over when "task B" comes in (for example using the IACK instruction to start the task). In this case you have single-stepped or halted in "task A" and the MPC has reached the MSTOP with no tasks pending. Before forcing "task B", run free to force the CLA out of the debug state. Once this is done you can force "task B" and continue debugging.

Restrictions

The MSTOP instruction cannot be placed 3 instructions before or after a MBCNDD, MCCNDD, or MRCNDD instruction.

Flags

This instruction does not modify flags in the MSTF register.

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

MSTOP (continued)

Stop Task

Pipeline

This is a single-cycle instruction. Table 7-21 shows the pipeline behavior of the MSTOP instruction. The MSTOP instruction cannot be placed with 3 instructions of a MBCNDD, MCCNDD, or MRCNDD instruction.

Table 7-21. Pipeline Activity For MSTOP

Instruction	F1	F2	D1	D2	R1	R2	E	W
I1	I1							
I2	I2	I1						
I3	I3	I2	I1					
MSTOP	MSTOP	I3	I2	I1				
I4	I4	MSTOP	I3	I2	I1			
I5	I5	I4	MSTOP	I3	I2	I1		
I6	I6	I5	I4	MSTOP	I3	I2	I1	
New Task Arbitrated and Prioritized	-	-	-	-	-	I3	I2	
New Task Arbitrated and Prioritized	-	-	-	-	-	-	I3	
I1	I1	-	-	-	-	-	-	
I2	I2	I1	-	-	-	-	-	
I3	I3	I2	I1	-	-	-	-	
I4	I4	I3	I2	I1	-	-	-	
I5	I5	I4	I3	I2	I1	-	-	
I6	I6	I5	I4	I3	I2	I1	-	
I7	I7	I6	I5	I4	I3	I2	I1	
etc								

Example

```
; Given A = (int32)1
;       B = (int32)2
;       C = (int32)-7
;
; Calculate Y2 = A - B - C
_Cla1Task3:
   MMOV32   MR0, @_A      ; MR0 = 1 (0x00000001)
   MMOV32   MR1, @_B      ; MR1 = 2 (0x00000002)
   MMOV32   MR2, @_C      ; MR2 = -7 (0xFFFFFFF9)
   MSUB32   MR3, MR0, MR1 ; A + B
   MSUB32   MR3, MR3, MR2 ; A + B + C = 6 (0x0000006)
   MMOV32   @_y2, MR3     ; Store y2
   MSTOP                  ; End of task
```

See also

MDEBUGSTOP

MSUB32 MRa, MRb, MRc

32-Bit Integer Subtraction

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point destination register (MR0 to MR3)
MRc	CLA floating-point destination register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 00cc bbaa
MSW: 0111 1100 1110 0000
```

Description

32-bit integer addition of MRb and MRc.

```
MARa(31:0) = MARb(31:0) - MRc(31:0);
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified as follows:

```
NF = MRa(31);
ZF = 0;
if(MRa(31:0) == 0) { ZF = 1; }
```

Pipeline

This is a single-cycle instruction.

Example

```
; Given A = (int32)1
;       B = (int32)2
;       C = (int32)-7
;
;
Calculate Y2 = A - B - C
;
_Cla1Task3:
   MMOV32   MR0, @_A          ; MR0 = 1 (0x00000001)
   MMOV32   MR1, @_B          ; MR1 = 2 (0x00000002)
   MMOV32   MR2, @_C          ; MR2 = -7 (0xFFFFFFF9)
   MSUB32   MR3, MR0, MR1     ; A + B
   MSUB32   MR3, MR3, MR2     ; A + B + C = 6 (0x0000006)
   MMOV32   @_y2, MR3         ; Store y2
   MSTOP                      ; End of task
```

See also

MADD32 MRa, MRb, MRc
MAND32 MRa, MRb, MRc
MASR32 MRa, #SHIFT
MLSL32 MRa, #SHIFT
MLSR32 MRa, #SHIFT
MOR32 MRa, MRb, MRc
MXOR32 MRa, MRb, MRc

MSUBF32 MRa, MRb, MRc

32-Bit Floating-Point Subtraction

Operands

MRa	CLA floating-point destination register (MR0 to R1)
MRb	CLA floating-point source register (MR0 to R1)
MRc	CLA floating-point source register (MR0 to R1)

Opcode

```
LSW: 0000 0000 00cc bbaa
MSW: 0111 1100 0100 0000
```

Description

Subtract the contents of two floating-point registers

```
MRa = MRb - MRc;
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MSUBF32 generates an underflow condition.
- LVF = 1 if MSUBF32 generates an overflow condition.

Pipeline

This is a single-cycle instruction.

Example

```
; Given A, B and C are 32-bit floating-point numbers
; Calculate Y2 = A + B - C
;
_Cla1Task5:
   MMOV32    MR0, @_A      ; Load MR0 with A
   MMOV32    MR1, @_B      ; Load MR1 with B
   MADDF32   MR0, MR1, MR0 ; Add A + B
|| MMOV32    MR1, @_C      ;   and in parallel load C
   MSUBF32   MR0, MR0, MR1 ; Subtract C from (A + B)
   MMOV32    @Y,  MR0      ; (A+B) - C
   MSTOP                   ; end of task
```

See also

MSUBF32 MRa, #16FHi, MRb
MSUBF32 MRd, MRe, MRf || MMOV32 MRa, mem32
MSUBF32 MRd, MRe, MRf || MMOV32 mem32, MRa
MMPYF32 MRa, MRb, MRc || MSUBF32 MRd, MRe, MRf

MSUBF32 MRa, #16FHi, MRb

32-Bit Floating-Point Subtraction

Operands

MRa	CLA floating-point destination register (MR0 to R1)
#16FHi	A 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0.
MRb	CLA floating-point source register (MR0 to R1)

Opcode

```
LSW: IIII IIII IIII IIII
MSW: 0111 1000 0000 baaa
```

Description

Subtract MRb from the floating-point value represented by the immediate operand. Store the result of the addition in MRa.

#16FHi is a 16-bit immediate value that represents the upper 16-bits of an IEEE 32-bit floating-point value. The low 16-bits of the mantissa are assumed to be all 0. #16FHi is most useful for representing constants where the lowest 16-bits of the mantissa are 0. Some examples are 2.0 (0x40000000), 4.0 (0x40800000), 0.5 (0x3F000000), and -1.5 (0xBFC00000). The assembler will accept either a hex or float as the immediate value. That is, the value -1.5 can be represented as #-1.5 or #0xBFC0.

```
MRa = #16FHi:0 - MRb;
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MSUBF32 generates an underflow condition.
- LVF = 1 if MSUBF32 generates an overflow condition.

Pipeline

This is a single-cycle instruction.

Example

```
; Y = sqrt(X)
; Ye = Estimate(1/sqrt(X));
; Ye = Ye*(1.5 - Ye*Ye*X*0.5)
; Ye = Ye*(1.5 - Ye*Ye*X*0.5)
; Y = X*Ye
;
_Cla1Task3:
   MMOV32     MR0, @_x           ; MR0 = X
   MEISQRTF32 MR1, MR0           ; MR1 = Ye = Estimate(1/sqrt(X))
   MMOV32     MR1, @_x, EQ       ; if(X == 0.0) Ye = 0.0
   MMPYF32    MR3, MR0, #0.5     ; MR3 = X*0.5
   MMPYF32    MR2, MR1, MR3      ; MR2 = Ye*X*0.5
   MMPYF32    MR2, MR1, MR2      ; MR2 = Ye*Ye*X*0.5
   MSUBF32    MR2, #1.5, MR2     ; MR2 = 1.5 - Ye*Ye*X*0.5
   MMPYF32    MR1, MR1, MR2      ; MR1 = Ye = Ye*(1.5 - Ye*Ye*X*0.5)
   MMPYF32    MR2, MR1, MR3      ; MR2 = Ye*X*0.5
   MMPYF32    MR2, MR1, MR2      ; MR2 = Ye*Ye*X*0.5
   MSUBF32    MR2, #1.5, MR2     ; MR2 = 1.5 - Ye*Ye*X*0.5
   MMPYF32    MR1, MR1, MR2      ; MR1 = Ye = Ye*(1.5 - Ye*Ye*X*0.5)
   MMPYF32    MR0, MR1, MR0      ; MR0 = Y = Ye*X
   MMOV32     @_y, MR0           ; Store Y = sqrt(X)
   MSTOP                         ; end of task
```

MSUBF32 MRa, #16FHi, MRb (continued)

32-Bit Floating-Point Subtraction

See also

MSUBF32 MRa, MRb, MRc
MSUBF32 MRd, MRe, MRf || MMOV32 MRa, mem32
MSUBF32 MRd, MRe, MRf || MMOV32 mem32, MRa
MMPYF32 MRa, MRb, MRc || MSUBF32 MRd, MRe, MRf

MSUBF32 MRd, MRe, MRf ||MMOV32 MRa, mem32

32-Bit Floating-Point Subtraction with Parallel Move

Operands

MRd	CLA floating-point destination register (MR0 to MR3) for the MSUBF32 operation MRd cannot be the same register as MRa
MRe	CLA floating-point source register (MR0 to MR3) for the MSUBF32 operation
MRf	CLA floating-point source register (MR0 to MR3) for the MSUBF32 operation
MRa	CLA floating-point destination register (MR0 to MR3) for the MMOV32 operation MRa cannot be the same register as MRd
mem32	32-bit memory location accessed using one of the available addressing modes. Source for the MMOV32 operation.

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0010 ffee ddaa addr
```

Description

Subtract the contents of two floating-point registers and move from memory to a floating-point register.

```
MRd = MRe - MRf;
MRa = [mem32];
```

Restrictions

The destination register for the MSUBF32 and the MMOV32 must be unique. That is, MRa cannot be the same register as MRd.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MSUBF32 generates an underflow condition.
- LVF = 1 if MSUBF32 generates an overflow condition.

The MMOV32 Instruction will set the NF and ZF flags.

Pipeline

Both MSUBF32 and MMOV32 complete in a single cycle.

Example

```
NF = MRa(31);
ZF = 0;
if(MRa(30:23) == 0) { ZF = 1; NF = 0; }
```

See also

MSUBF32 MRa, MRb, MRc
MSUBF32 MRa, #16FHi, MRb
MMPYF32 MRa, MRb, MRc || MSUBF32 MRd, MRe, MRf

MSUBF32 MRd, MRe, MRf ||MMOV32 mem32, MRa

32-Bit Floating-Point Subtraction with Parallel Move

Operands

MRd	CLA floating-point destination register (MR0 to MR3) for the MSUBF32 operation
MRe	CLA floating-point source register (MR0 to MR3) for the MSUBF32 operation
MRf	CLA floating-point source register (MR0 to MR3) for the MSUBF32 operation
mem32	32-bit destination memory location for the MMOV32 operation
MRa	CLA floating-point source register (MR0 to MR3) for the MMOV32 operation

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0110 ffee ddaa addr
```

Description

Subtract the contents of two floating-point registers and move from a floating-point register to memory.

```
MRd = MRe - MRf;
[mem32] = MRa;
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	Yes	Yes

The MSTF register flags are modified as follows:

- LUF = 1 if MSUBF32 generates an underflow condition.
- LVF = 1 if MSUBF32 generates an overflow condition.

Pipeline

Both MSUBF32 and MMOV32 complete in a single cycle.

See also

MSUBF32 MRa, MRb, MRc
MSUBF32 MRa, #16FHi, MRb
MSUBF32 MRd, MRe, MRf || MMOV32 MRa, mem32
MMPYF32 MRa, MRb, MRc || MSUBF32 MRd, MRe, MRf

MSWAPF MRa, MRb {, CNDF}

Conditional Swap

Operands

MRa	CLA floating-point register (MR0 to MR3)
MRb	CLA floating-point register (MR0 to MR3)
CNDF	Optional condition tested based on the MSTF flags

Opcode

```
LSW: 0000 0000 CNDF bbaa
MSW: 0111 1011 0000 0000
```

Description

Conditional swap of MRa and MRb.

```
if (CNDF == true) swap MRa and MRb;
```

CNDF is one of the following conditions:

Encode[1]	CNDF	Description	MSTF Flags Tested
0000	NEQ	Not equal to zero	ZF == 0
0001	EQ	Equal to zero	ZF == 1
0010	GT	Greater than zero	ZF == 0 AND NF == 0
0011	GEQ	Greater than or equal to zero	NF == 0
0100	LT	Less than zero	NF == 1
0101	LEQ	Less than or equal to zero	ZF == 1 OR NF == 1
1010	TF	Test flag set	TF == 1
1011	NTF	Test flag not set	TF == 0
1100	LU	Latched underflow	LUF == 1
1101	LV	Latched overflow	LVF == 1
1110	UNC	Unconditional	None
1111	UNCF[2]	Unconditional with flag modification	None

(1) Values not shown are reserved.
(2) This is the default operation if no CNDF field is specified. This condition will allow the ZF and NF flags to be modified when a conditional operation is executed. All other conditions will not modify these flags.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

No flags affected

Pipeline

This is a single-cycle instruction.

MSWAPF MRa, MRb {, CNDF} (continued)

Conditional Swap

Example

```
; X is an array of 32-bit floating-point values
; and has len elements. Find the maximum value in
; the array and store it in Result
;
; Note: MCMPF32 and MSWAPF can be replaced by MMAXF32
;
_Cla1Task1:
    MMOVI16    MAR1,#_X          ; Start address
    MUI16TOF32 MR0, @_len        ; Length of the array
    MNOP                         ; delay for MAR1 load
    MNOP                         ; delay for MAR1 load
    MMOV32     MR1, *MAR1[2]++   ; MR1 = X0
LOOP
    MMOV32     MR2, *MAR1[2]++   ; MR2 = next element
    MCMPF32    MR2, MR1          ; Compare MR2 with MR1
    MSWAPF     MR1, MR2, GT      ; MR1 = MAX(MR1, MR2)
    MADDF32    MR0, MR0, #-1.0   ; Decrememt the counter
    MCMPF32    MR0 #0.0          ; Set/clear flags for MBCNDD
    MNOP
    MNOP
    MNOP
    MBCNDD     LOOP, NEQ         ; Branch if not equal to zero
    MMOV32     @_Result, MR1     ; Always executed
    MNOP                         ; Always executed
    MNOP                         ; Always executed
    MSTOP                        ; End of task
```

MTESTTF CNDF

Test MSTF Register Flag Condition

Operands

CNDF	condition to test based on MSTF flags

Opcode

```
LSW: 0000 0000 0000 cndf
MSW: 0111 1111 0100 0000
```

Description

Test the CLA floating-point condition and if true, set the MSTF[TF] flag. If the condition is false, clear the MSTF[TF] flag. This is useful for temporarily storing a condition for later use.

```
if (CNDF == true) TF = 1;
else TF = 0;
```

CNDF is one of the following conditions:

Encode(1)	CNDF	Description	MSTF Flags Tested
0000	NEQ	Not equal to zero	ZF == 0
0001	EQ	Equal to zero	ZF == 1
0010	GT	Greater than zero	ZF == 0 AND NF == 0
0011	GEQ	Greater than or equal to zero	NF == 0
0100	LT	Less than zero	NF == 1
0101	LEQ	Less than or equal to zero	ZF == 1 OR NF == 1
1010	TF	Test flag set	TF == 1
1011	NTF	Test flag not set	TF == 0
1100	LU	Latched underflow	LUF == 1
1101	LV	Latched overflow	LVF == 1
1110	UNC	Unconditional	None
1111	UNCF(2)	Unconditional with flag modification	None

(1) Values not shown are reserved.
(2) This is the default operation if no CNDF field is specified. This condition will allow the ZF and NF flags to be modified when a conditional operation is executed. All other conditions will not modify these flags.

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	Yes	No	No	No	No

```
TF = 0;
if (CNDF == true) TF = 1;
```

Note: If (CNDF == UNC or UNCF), the TF flag will be set to 1.

Pipeline

This is a single-cycle instruction.

MTESTTF CNDF (continued)

Test MSTF Register Flag Condition

Example

```
; if (State == 0.1)
;    RampState = RampState || RAMPMASK
; else if (State == 0.01)
;    CoastState = CoastState || COASTMASK
; else
;    SteadyState = SteadyState || STEADYMASK
;
_Cla1Task2:
   MMOV32    MR0, @_State
   MCMPF32   MR0, #0.1         ; Affects flags for 1st MBCNDD (A)
   MCMPF32   MR0, #0.01        ; Check used by 2nd MBCNDD (B)
   MTESTTF   EQ                ; Store EQ flag in TF for 2nd MBCNDD (B)
   MNOP
   MBCNDD    _Skip1, NEQ       ; (A) If State != 0.1, go to Skip1
   MMOV32    MR1, @_RampState  ; Always executed
   MMOVXI    MR2, #RAMPMASK    ; Always executed
   MOR32     MR1, MR2          ; Always executed
   MMOV32    @_RampState, MR1  ; Execute if (A) branch not taken
   MSTOP                       ; end of task if (A) branch not taken
_Skip1:
   MMOV32    MR3, @_SteadyState
   MMOVXI    MR2, #STEADYMASK
   MOR32     MR3, MR2
   MBCNDD    _Skip2, NTF       ; (B) if State != .01, go to Skip2
   MMOV32    MR1, @_CoastState ; Always executed
   MMOVXI    MR2, #COASTMASK   ; Always executed
   MOR32     MR1, MR2          ; Always executed
   MMOV32    @_CoastState, MR1 ; Execute if (B) branch not taken
   MSTOP                       ; end of task if (B) branch not taken
_Skip2:
   MMOV32 @_SteadyState, MR3   ; Executed if (B) branch taken
   MSTOP
```

MUI16TOF32 MRa, mem16

Convert Unsigned 16-Bit Integer to 32-Bit Floating-Point Value

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
mem16	16-bit source memory location

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0111 0101 01aa addr
```

Description

When converting F32 to I16/UI16 data format, the MF32TOI16/UI16 operation truncates to zero while the MF32TOI16R/UI16R operation will round to nearest (even) value.

```
MRa = UI16TOF32[mem16];
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

See also

MF32TOI16 MRa, MRb
MF32TOI16R MRa, MRb
MF32TOUI16 MRa, MRb
MF32TOUI16R MRa, MRb
MI16TOF32 MRa, MRb
MI16TOF32 MRa, mem16
MUI16TOF32 MRa, MRb

MUI16TOF32 MRa, MRb

Convert Unsigned 16-Bit Integer to 32-Bit Floating-Point Value

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1110 1110 0000
```

Description

Convert an unsigned 16-bit integer to a 32-bit floating-point value. When converting float32 to I16/UI16 data format, the MF32TOI16/UI16 operation truncates to zero while the MF32TOI16R/UI16R operation will round to nearest (even) value.

```
MRa = UI16TOF32[MRb];
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
MMOVXI MR1, #0x800F ; MR1(15:0) = 32783 (0x800F)
MUI16TOF32 MR0, MR1 ; MR0 = UI16TOF32 (MR1(15:0))
                    ; = 32783.0 (0x47000F00)
```

See also

MF32TOI16 MRa, MRb
MF32TOI16R MRa, MRb
MF32TOUI16 MRa, MRb
MF32TOUI16R MRa, MRb
MI16TOF32 MRa, MRb
MI16TOF32 MRa, mem16
MUI16TOF32 MRa, mem16

MUI32TOF32 MRa, mem32

Convert Unsigned 32-Bit Integer to 32-Bit Floating-Point Value

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
mem32	32-bit memory location accessed using one of the available addressing modes

Opcode

```
LSW: mmmm mmmm mmmm mmmm
MSW: 0111 0100 10aa addr
```

Description

```
MRa = UI32TOF32[mem32];
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
; Given x2, m2 and b2 are Uint32 numbers:
;
; x2 = Uint32(2) = 0x00000002
; m2 = Uint32(1) = 0x00000001
; b2 = Uint32(3) = 0x00000003
;
; Calculate y2 = x2 * m2 + b2
;
_Cla1Task1:
   MUI32TOF32  MR0, @_m2      ; MR0 = 1.0 (0x3F800000)
   MUI32TOF32  MR1, @_x2      ; MR1 = 2.0 (0x40000000)
   MUI32TOF32  MR2, @_b2      ; MR2 = 3.0 (0x40400000)
   MMPYF32     MR3, MR0, MR1  ; M*X
   MADDF32     MR3, MR2, MR3  ; Y=MX+B = 5.0 (0x40A00000)
   MF32TOUI32  MR3, MR3       ; Y = Uint32(5.0) = 0x00000005
   MMOV32      @_y2, MR3      ; store result
   MSTOP                      ; end of task
```

See also

MF32TOI32 MRa, MRb
MF32TOUI32 MRa, MRb
MI32TOF32 MRa, mem32
MI32TOF32 MRa, MRb
MUI32TOF32 MRa, MRb

MUI32TOF32 MRa, MRb

Convert Unsigned 32-Bit Integer to 32-Bit Floating-Point Value

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 0000 bbaa
MSW: 0111 1101 1100 0000
```

Description

```
MRa = UI32TOF32 [MRb];
```

Flags

This instruction does not affect any flags:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	No	No	No	No

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ     MR3, #0x8000 ; MR3(31:16) = 0x8000
MMOVXI     MR3, #0x1111 ; MR3(15:0) = 0x1111
                        ; MR3 = 2147488017
MUI32TOF32 MR3, MR3     ; MR3 = MUI32TOF32 (MR3) = 2147488017.0 (0x4F000011)
```

See also

MF32TOI32 MRa, MRb
MF32TOUI32 MRa, MRb
MI32TOF32 MRa, mem32
MI32TOF32 MRa, MRb
MUI32TOF32 MRa, mem32

MXOR32 MRa, MRb, MRc

Bitwise Exclusive Or

Operands

MRa	CLA floating-point destination register (MR0 to MR3)
MRb	CLA floating-point source register (MR0 to MR3)
MRc	CLA floating-point source register (MR0 to MR3)

Opcode

```
LSW: 0000 0000 00cc bbaa
MSW: 0111 1100 1010 0000
```

Description

Bitwise XOR of MRb with MRc.

```
MARa(31:0) = MARb(31:0) XOR MRc(31:0);
```

Flags

This instruction modifies the following flags in the MSTF register:

Flag	TF	ZF	NF	LUF	LVF
Modified	No	Yes	Yes	No	No

The MSTF register flags are modified based on the integer results of the operation.

```
NF = MRa(31);
ZF = 0;
if(MRa(31:0) == 0) { ZF = 1; }
```

Pipeline

This is a single-cycle instruction.

Example

```
MMOVIZ MR0, #0x5555   ; MR0 = 0x5555AAAA
  MMOVXI MR0, #0xAAAA
  MMOVIZ MR1, #0x5432   ; MR1 = 0x5432FEDC
  MMOVXI MR1, #0xFEDC
  ; 0101 XOR 0101 = 0000 (0)
  ; 0101 XOR 0100 = 0001 (1)
  ; 0101 XOR 0011 = 0110 (6)
  ; 0101 XOR 0010 = 0111 (7)
  ; 1010 XOR 1111 = 0101 (5)
  ; 1010 XOR 1110 = 0100 (4)
  ; 1010 XOR 1101 = 0111 (7)
  ; 1010 XOR 1100 = 0110 (6)
  MXOR32 MR2, MR1, MR0  ; MR3 = 0x01675476
```

See also

MAND32 MRa, MRb, MRc
MOR32 MRa, MRb, MRc

7.7 Software

7.7.1 CLA Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/cla

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

7.7.1.1 CLA $arcsine(x)$ Using a Lookup Table (cla_asin_cpu01)

FILE: cla_ex1_asin.c

In this example, Task 1 of the CLA will calculate the arcsine of an input argument in the range (-1.0 to 1.0) using a lookup table.

Note that since this example does not use background CLA task, the compile flag cla_background_task is turned off for this project. Set this flag as on to enable background CLA task. The option is available in Project Properties -> C2000 Build -> C2000 Compiler -> Advanced Options -> Runtime Model Options.

Memory Allocation

- CLA1 Math Tables (RAMLS1)
  - CLAasinTable - Lookup table
- CLA1 to CPU Message RAM
  - fResult - Result of the lookup algorithm
- CPU to CLA1 Message RAM
  - fVal - Sample input to the lookup algorithm

Watch Variables

- fVal - Argument to task 1
- fResult - Result of $arcsin(fVal)$

7.7.1.2 CLA $arctangent(x)$ Using a Lookup Table (cla_atan_cpu01)

FILE: cla_ex2_atan.c

In this example, Task 1 of the CLA will calculate the arctangent of an input argument using a lookup table.

Note that since this example does not use background CLA task, the compile flag cla_background_task is turned off for this project. Set this flag as on to enable background CLA task. The option is available in Project Properties -> C2000 Build -> C2000 Compiler -> Advanced Options -> Runtime Model Options.

Memory Allocation

- CLA1 Math Tables (RAMLS1)
  - CLAatan2Table - Lookup table
- CLA1 to CPU Message RAM
  - fResult - Result of the lookup algorithm
- CPU to CLA1 Message RAM
  - fNum - Numerator of sample input
  - fDen - Denominator of sample input

Watch Variables

- fVal - Argument to task 1
- fResult - Result of $arctan(fVal)$

7.7.1.3 CLA Background Nesting Task

FILE: cla_ex3_background_nesting_task.c

This example configures CLA task 1 to be triggered by EPWM1 running at 2 Hz (period = 0.5s). A background task is configured to be triggered by CPU timer running at .5 Hz (period = 2s). CLA task 1 toggles LED1 at the start and end of the task and the background task toggles LED2 at the start and end of the task. Background task will be preempted by Task1 and hence LED1 will be toggling even while LED2 is ON.

Note that the compile flag cla_background_task is turned on in this project. Enabling background task adds additional context save/restore cycles during task switching thus increasing the overall trigger-to-task latency. If the application does not use the background CLA task, it is recommended to turn this flag off for better performance. The option is available in Project Properties -> C2000 Build -> C2000 Compiler -> Advanced Options -> Runtime Model Options.

External Connections

- None

Watch Variables

- None

7.7.1.4 Controlling PWM Output Using CLA

FILE: cla_ex4_pwm_control.c

This example showcases how to update PWM signal output using CLA. EPWM1 is configured to generate complementary signals on both of its channels of fixed frequency 100 kHz. EPWM4 is configured to trigger a periodic CLA control task of frequency 10 kHz. The CLA task implements a very simple logic to vary the duty of the EPWM1 outputs by increasing it by 0.1 in every iteration and maintaining it in the range of 0.1-0.9. For actual use cases, the control logic could be modified to much more complex depending upon the application. The other CLA task (CLA task 8) is triggered by software at beginning to initialize the CLA global variables

External Connections

- Observe GPIO0 (EPWM1A) on oscilloscope
- Observe GPIO1 (EPWM1B) on oscilloscope

Watch Variables

- duty

7.7.1.5 Just-in-time ADC Sampling with CLA

FILE: cla_ex5_adc_just_in_time.c

This example showcases how to utilize early-interrupt feature of ADC in combination with the low interrupt response of CLA to enable faster system response and achieve high frequency control loops. EPWM1 is configured to generate a PWM output signal of frequency 1 MHz and this is also used to trigger the ADC sampling at each cycle. ADCA is configured to sample the input on Channel 0 and to generate the early interrupt at the end of S/H + offset cycles. This interrupt is used to trigger the CLA control task. The CLA task implements the control logic to update the duty of the PWM output based on reading the ADC sample data just-in-time, that is, as soon as the ADC results gets latched.The early interrupt feature and low interrupt latency of CLA allows to do some pre-processing as well before reading the ADC data and still completes updating the PWM output before the next interrupts comes in, that is, data read and PWM update is done within a 1 MHz cycle. For illustration purposes, 3-point moving average filter is used to simulate some processing and few steps of the filtering code are done before reading the ADC result which we consider as pre-processing code. The ADC interrupt offset is programmed based on the cycles consumed by the pre-processing code.

The calculation for interrupt offset value is:

ADC acquisition cycles programmed = 10 SYSCLKS

Conversion time for 12-bit data = 10.5 ADCCLKS = N = 42 SYSCLKS

CLA task trigger to first instruction in Fetch delay = 4

Let the interrupt offset value be 'x'

The code inside CLA control task before ADC read takes:

Setting up profiling gpio: 3 cycles

Pre-processing: 13 cycles

Total = 3 + 13 = 16 cycles

In order to read just-in-time, the total delay before reading ADC should be (N-2) cycles = 40 (that is, x + 4 + 16 = 40; where x = 20)

NOTE: The optimization is off for this project and the cycles quoted above corresponds to that case.

GPIO2 is used for profiling purposes. GPIO2 is set at the beginning of CLA task 1 and is reset at the end of the task. Thus ON time of GPIO2 indicates the CLA activity. In order to validate the example functionality , observe the GPIO0 (PWM output) and GPIO2 (profiling GPIO) on CRO. The cycles difference between the rising edge of the GPIO0 and GPIO2 indicate the total delay from the time of ADC trigger to setting up of profiling GPIO inside CLA task which should be around 44 cycles (367 ns) based on the above calculation.

External Connections

- Provide constant DC input on ADCA0 for quick validation.
  - GND -> Should observe PWM output duty = 0.1
  - 3.3V -> Should observe PWM output duty = 0.9
  - Can also provide analog input in range 0 - 3.3V up to fs / 10 = 100 kHz for observing continuous duty variations
- Observe GPIO0 on oscilloscope
- Observe GPIO2 on oscilloscope

Watch Variables

- None

7.7.1.6 Optimal Offloading of Control Algorithms to CLA

FILE: cla_ex6_cpu_offloading.c

This example showcases how to optimally offload the control algorithms from CPU to CLA in order to meet the system requirements. In this example, two control loops are simulated, the faster one (loop1) running at 200 kHz and the slower one (loop2) running at 20 kHz. Loop1 senses the first parameter at ADCA Channel 0, runs the PI controller to achieve the target and contributes to the duty of EPWM1A output with 80% weightage. Loop2 senses the second parameter at ADCB Channel 2, runs the PI controller and contributes to the duty of EPWM1A output with 20% weightage. It is important to note that since these are just software simulated control loops but there is no actual physical process involved and hence updating the duty is not going to have any affect on sampled inputs. ADCA is configured to oversample the first parameter using SOCs 0-3 to suppress the noise and similarly ADCB is used to oversample the second parameter. EPWM4 and EPWM5 are configured to trigger the ADCA and ADCB sampling at loop1 and loop2 frequencies respectively. Once the conversion of all 4 SOCs complete, a CPU ISR or a CLA task is triggered based on the user-configuration. There is also a background task running in the main loop which disables the entire system including PWM output and the control loops when "system_OFF" is set to 1. The system gets enabled again once "system_OFF" is restored back to 0. By default system_OFF is set to 0 but it's value can be updated dynamically by adding it to expression window and writing to it. DCL library is included in the project to make use of optimal PI controllers used in both the loops. User-configurable pre-defined symbol "run_loop1_cla" has been added to the project options in order to specify whether to run the loop1 on C28x or CLA. GPIO2 and GPIO3 are used to profile the execution of loop1 and loop2.

For run_loop1_cla == 0, that is, both loops running on CPU -> Loop1 Utilization = ~77.5% (measured using profiling GPIO2) -> Loop2 Utilization = ~6% (measured using profiling GPIO3) -> Background task in a while loop -> Total CPU utilization is greater than Utilization bound (UB) Hence the system is non-schedulable, lower priority task (Loop2) execution never completes (no toggling observed on GPIO3) and also background task never gets chance to execute.

For run_loop1_cla == 1, that is, high-frequency control loop (loop1) is offloaded to CLA while loop2 runs on CPU -> Loop1 Utilization (CLA) = ~73% -> Loop2 Utilization (CPU)= ~6% -> Total CPU utilization has come down to just ~6%. Hence, the system is perfectly schedulable, no miss happens for any of the loops and offloading of loop1 to CLA saves CPU bandwidth to execute background tasks as well.

For quick inspection of the example functionality, constant DC HIGH/LOW inputs can be provided to the analog channels instead of varying analog voltages. The target value for both the loops are set as some intermediate value, that is, 3500 corresponds to ~2.8V. Now since the sensed inputs are constant and not same as target so the controller outputs will get saturated soon to either 1 or 0. Thus the "duty" variable can take only fixed values based on the equations used in the loops. In fact, the duty output would be very intuitive, for instance if both inputs are LOW (GND), the controller will try to produce the maximum duty as the target is higher than sensed value hence the duty should be 1.0 (0.2 + 0.8) but will get saturated to 0.9 (the maximum value defined). Similarly if both inputs are made HIGH, the duty will be 0.1 (the minimum saturation value defined). The final duty table is shown below :

External Connections

- Observe GPIO2 (Loop1 Profiling) on oscilloscope
- Observe GPIO3 (Loop2 Profiling) on oscilloscope
- Observe GPIO0 (EPWM1A Output) on oscilloscope
- Provide constant HIGH(3.3V)/LOW(0V) on both ADCA Ch0 and ADCB Ch2 for quick validation, the following duty value should be observable at EPWM1A for various combinations if the system is perfectly schedulable, that is, both loops get a chance to execute properly:- A0 B2 duty GND GND 0.9 3.3V GND 0.2 GND 3.3V 0.8 3.3V 3.3V 0.1
- Note: The optimization is OFF for this project and all the profiling data quoted above corresponds to this case.

7.7.1.7 Handling Shared Resources Across C28x and CLA

FILE: cla_ex7_shared_resource_handling.c

This example showcases how to handle shared resource challenges across C28x and CLA. As the peripherals are shared between CLA and the CPU, overlapping read-modify-write to the registers by them can lead to data race conditions ultimately leading to data violation or incorrect functionality. In this example, CPU ISR and CLA tasks runs independently. CPU ISR gets triggered by EPWM4 @10 kHz and toggles the EPWM1B output via software by controlling CSFB bits of AQCSFRC. CLA task gets triggered by EPWM5 @100 kHz and toggles the EPWM1A output via software by controlling CSFA bits of AQCSFRC. Thus in this process both CPU and CLA do read-modify -write to AQCSFRC register independently at different frequencies so there is chance of race condition and updates due to one of them can get lost/. overwritten. This can be clearly observed by updating "phase_shift_ON" to 0U and probing the EPWM1A and 1B outputs on a scope.

This is a standard critical section problem and can be handled by software handshaking mechanism like mutex. But most of the real-time control applications are time-sensitive and cannot afford addition software overhead hence this example suggests an alternative hardware based technique to avoid shared resource conflicts between CPU and CLA. The phase shifting mechanism of the EPWM modules is utilized to schedule the CLA task and CPU ISR as desired. EPWM4 generates a synchronous pulse every ZERO event and provides a phase shift of 20 cycles to EPWM5. This way both CLA task and C28x ISR runs at original frequencies, that is, 100 kHz and 10 kHz but CLA task leads with a phase offset of 20 cycles with respect to CPU ISR. Hence, concurrent read-modify-writes to AQCSFRC never happens and the EPWM1A and EPWM1B outputs behave as desired, that is, consistent 50 kHz PWM output on EPWM1A and 5 kHz PWM output on EPWM1B with a duty ~50% on both should be generated. In order to utilize this phase shifting mechanism in this example, make sure "phase_shift_ON" is set to 1.

External Connections

- Observe GPIO0 (EPWM1A Output) on oscilloscope
- Observe GPIO1 (EPWM1B Output) on oscilloscope
- Observe GPIO2 (CLA Task Profiling) on oscilloscope
- Observe GPIO3 (CPU ISR Profiling) on oscilloscope

Note: The phase offset value can easily be configured by updating the TBPHS register to schedule the CLA task and C28x ISR as desired depending upon the application need so as to avoid overlapping register writes by CPU and CLA.

Note: The optimization is on and set to O2 for the project and all the results quoted correspond to this case.

7.8 CLA Registers

This section describes the Control Law Accelerator registers.

7.8.1 CLA Base Address Table

Table 7-22. CLA Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
Cla1OnlyRegs	CLA_ONLY_REGS	CLA1_ONLY_BASE	0x0000_0C00	-	-	-	YES	-
Cla1SoftIntRegs	CLA_SOFTINT_REGS	CLA1_SOFTINT_BASE	0x0000_0CE0	-	-	-	YES	-
Cla1Regs	CLA_REGS	CLA1_BASE	0x0000_1400	YES	-	-	-	-

7.8.2 CLA_ONLY_REGS Registers

Table 7-23 lists the memory-mapped registers for the CLA_ONLY_REGS registers. All register offset addresses not listed in Table 7-23 should be considered as reserved locations and the register contents should not be modified.

Table 7-23. CLA_ONLY_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
80h	_MVECTBGRNDACTIVE	Active register for MVECTBGRND.	EALLOW	Go
C0h	_MPSACTL	CLA PSA Control Register	EALLOW	Go
C2h	_MPSA1	CLA PSA1 Register	EALLOW	Go
C4h	_MPSA2	CLA PSA2 Register	EALLOW	Go
E0h	SOFTINTEN	CLA Software Interrupt Enable Register		Go
E2h	SOFTINTFRC	CLA Software Interrupt Force Register		Go

Complex bit access types are encoded to fit into small table cells. Table 7-24 shows the codes that are used for access types in this section.

Table 7-24. CLA_ONLY_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

7.8.2.1 _MVECTBGRNDACTIVE Register (Offset = 80h) [Reset = 0h]

_MVECTBGRNDACTIVE is shown in Figure 7-2 and described in Table 7-25.

Return to the Summary Table.

Gives the current interrupted MPC value of the background task, if the background task was running and interrupted, or reflects the MVECTBGRND value, if MCTLBGRND.BGSTART bit is 0.

Figure 7-2. _MVECTBGRNDACTIVE Register

15	14	13	12	11	10	9	8
i16							
R-0h							
7	6	5	4	3	2	1	0
i16							
R-0h							

Table 7-25. _MVECTBGRNDACTIVE Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	i16	R	0h	Gives the current interrupted MPC value of the background task, if the background task was running and interrupted, or reflects the MVECTBGRND value, if MCTLBGRND.BGSTART bit is 0. Reset type: SYSRSn

7.8.2.2 _MPSACTL Register (Offset = C0h) [Reset = 0h]

_MPSACTL is shown in Figure 7-3 and described in Table 7-26.

Return to the Summary Table.

PSA Control Register

Figure 7-3. _MPSACTL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
MPSA2CFG		MPSA2CLEAR	MPSA1CLEAR	MDWDBCYC	MDWDBSTART	MPABCYC	MPABSTART
R/W-0h		R-0/W1S-0h	R-0/W1S-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 7-26. _MPSACTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-6	MPSA2CFG	R/W	0h	CLA PSA2 Polynomial Configuration Bits: These bits configure the type of polynomial used for PSA2. The polynomials chosen are commonly used in the industry: Mode Polynomial Type 0,0 PSA 0,1 CRC32 1,0 CRC16 1,1 CRC16-CCITT Note: [1] Polynomial configuration should be performed when PSA2 is stopped. Reset type: SYSRSn
5	MPSA2CLEAR	R-0/W1S	0h	CLA PSA2 Clear Bit: Writing of "1" will clear contents of PSA2 register. Writes of "0" are ignored. Always reads back a "0" Note: Clearing operation should be performed when PSA2 is stopped. Reset type: SYSRSn
4	MPSA1CLEAR	R-0/W1S	0h	CLA PSA1 Clear Bit: Writing of "1" will clear contents of PSA1 register. Writes of "0" are ignored. Always reads back a "0" Note: Clearing operation should be performed when PSA1 is stopped. Reset type: SYSRSn
3	MDWDBCYC	R/W	0h	CLA Data Write Data Bus PSA2 Cycle or Event Based Bit: 0 PSA2 calculated on every cycle 1 PSA2 calculated on every bus event Reset type: SYSRSn
2	MDWDBSTART	R/W	0h	CLA Data Write Data Bus PSA2 Start/Stop Bit: 0 PSA2 stopped 1 PSA2 start Reset type: SYSRSn
1	MPABCYC	R/W	0h	CLA Program Address Bus PSA1 Cycle/Event Based Bit: 0 PSA1 calculated on every cycle 1 PSA1 calculated on every bus event Reset type: SYSRSn
0	MPABSTART	R/W	0h	CLA Program Address Bus PSA1 Start/Stop Bit: 0 PSA1 stopped 1 PSA1 start Reset type: SYSRSn

7.8.2.3 _MPSA1 Register (Offset = C2h) [Reset = 0h]

_MPSA1 is shown in Figure 7-4 and described in Table 7-27.

Return to the Summary Table.

PSA1 Register

Figure 7-4. _MPSA1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
i32																															
R/W-0h																															

Table 7-27. _MPSA1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	i32	R/W	0h	PSA1 Value: Reading this register gives the current PSA1 value. The value can be read at any time. Writes to this register are allowed to initialize the PSA1 to a known value. Writes to this register should only be made when PSA1 is stopped. Register value is cleared to zero by reset or by writing to the MPSA1CLEAR bit in the MPSACTL register. Reset type: SYSRSn

7.8.2.4 _MPSA2 Register (Offset = C4h) [Reset = 0h]

_MPSA2 is shown in Figure 7-5 and described in Table 7-28.

Return to the Summary Table.

PSA2 Register

Figure 7-5. _MPSA2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
i32																															
R/W-0h																															

Table 7-28. _MPSA2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	i32	R/W	0h	PSA2 Value: Reading this register gives the current PSA2 value. The value can be read at any time. Writes to this register are allowed to initialize the PSA2 to a known value. Writes to this register should only be made when PSA2 is stopped. Register value is cleared to zero by reset or by writing to the MPSA2CLEAR bit in the MPSACTL register. Reset type: SYSRSn

7.8.2.5 SOFTINTEN Register (Offset = E0h) [Reset = 0h]

SOFTINTEN is shown in Figure 7-6 and described in Table 7-29.

Return to the Summary Table.

Enables the ability to generate CLA task interrupt from within the task, by writing to SOFTINTFRC register. SOFTINTFRC register can only be written from CLA.

Figure 7-6. SOFTINTEN Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
TASK8	TASK7	TASK6	TASK5	TASK4	TASK3	TASK2	TASK1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 7-29. SOFTINTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	TASK8	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
6	TASK7	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
5	TASK6	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
4	TASK5	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
3	TASK4	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
2	TASK3	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
1	TASK2	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn

Table 7-29. SOFTINTEN Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	TASK1	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn

7.8.2.6 SOFTINTFRC Register (Offset = E2h) [Reset = 0h]

SOFTINTFRC is shown in Figure 7-7 and described in Table 7-30.

Return to the Summary Table.

Writing a value of 1 in a bit will generate the corresponding task interrupt.

Figure 7-7. SOFTINTFRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
TASK8	TASK7	TASK6	TASK5	TASK4	TASK3	TASK2	TASK1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 7-30. SOFTINTFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	TASK8	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
6	TASK7	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
5	TASK6	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
4	TASK5	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
3	TASK4	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
2	TASK3	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
1	TASK2	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
0	TASK1	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn

7.8.3 CLA_SOFTINT_REGS Registers

Table 7-31 lists the memory-mapped registers for the CLA_SOFTINT_REGS registers. All register offset addresses not listed in Table 7-31 should be considered as reserved locations and the register contents should not be modified.

Table 7-31. CLA_SOFTINT_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	SOFTINTEN	CLA Software Interrupt Enable Register		Go
2h	SOFTINTFRC	CLA Software Interrupt Force Register		Go

Complex bit access types are encoded to fit into small table cells. Table 7-32 shows the codes that are used for access types in this section.

Table 7-32. CLA_SOFTINT_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

7.8.3.1 SOFTINTEN Register (Offset = 0h) [Reset = 0h]

SOFTINTEN is shown in Figure 7-8 and described in Table 7-33.

Return to the Summary Table.

Enables the ability to generate CLA task interrupt from within the task, by writing to SOFTINTFRC register. SOFTINTFRC register can only be written from CLA.

Figure 7-8. SOFTINTEN Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
TASK8	TASK7	TASK6	TASK5	TASK4	TASK3	TASK2	TASK1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 7-33. SOFTINTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	TASK8	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
6	TASK7	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
5	TASK6	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
4	TASK5	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
3	TASK4	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
2	TASK3	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
1	TASK2	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn

Table 7-33. SOFTINTEN Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	TASK1	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn

7.8.3.2 SOFTINTFRC Register (Offset = 2h) [Reset = 0h]

SOFTINTFRC is shown in Figure 7-9 and described in Table 7-34.

Return to the Summary Table.

Writing a value of 1 in a bit will generate the corresponding task interrupt.This register is only accessible by the CLA (not the CPU).

Figure 7-9. SOFTINTFRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
TASK8	TASK7	TASK6	TASK5	TASK4	TASK3	TASK2	TASK1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 7-34. SOFTINTFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	TASK8	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
6	TASK7	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
5	TASK6	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
4	TASK5	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
3	TASK4	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
2	TASK3	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
1	TASK2	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn
0	TASK1	R-0/W1S	0h	Write of '1' will generate a CLA software interrupt to the CPU for the corresponding task. Write of '0' has no effect. Reset type: SYSRSn

7.8.4 CLA_REGS Registers

Table 7-35 lists the memory-mapped registers for the CLA_REGS registers. All register offset addresses not listed in Table 7-35 should be considered as reserved locations and the register contents should not be modified.

Table 7-35. CLA_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	MVECT1	Task Interrupt Vector	EALLOW	Go
1h	MVECT2	Task Interrupt Vector	EALLOW	Go
2h	MVECT3	Task Interrupt Vector	EALLOW	Go
3h	MVECT4	Task Interrupt Vector	EALLOW	Go
4h	MVECT5	Task Interrupt Vector	EALLOW	Go
5h	MVECT6	Task Interrupt Vector	EALLOW	Go
6h	MVECT7	Task Interrupt Vector	EALLOW	Go
7h	MVECT8	Task Interrupt Vector	EALLOW	Go
10h	MCTL	Control Register	EALLOW	Go
1Bh	_MVECTBGRNDACTIVE	Active register for MVECTBGRND.	EALLOW	Go
1Ch	SOFTINTEN	CLA Software Interrupt Enable Register		Go
1Dh	_MSTSBGRND	Status register for the back ground task.	EALLOW	Go
1Eh	_MCTLBGRND	Control register for the back ground task.	EALLOW	Go
1Fh	_MVECTBGRND	Vector for the back ground task.	EALLOW	Go
20h	MIFR	Interrupt Flag Register	EALLOW	Go
21h	MIOVF	Interrupt Overflow Flag Register	EALLOW	Go
22h	MIFRC	Interrupt Force Register	EALLOW	Go
23h	MICLR	Interrupt Flag Clear Register	EALLOW	Go
24h	MICLROVF	Interrupt Overflow Flag Clear Register	EALLOW	Go
25h	MIER	Interrupt Enable Register	EALLOW	Go
26h	MIRUN	Interrupt Run Status Register	EALLOW	Go
28h	_MPC	CLA Program Counter		Go
2Ah	_MAR0	CLA Auxiliary Register 0		Go
2Bh	_MAR1	CLA Auxiliary Register 1		Go
2Eh	_MSTF	CLA Floating-Point Status Register		Go
30h	_MR0	CLA Floating-Point Result Register 0		Go
34h	_MR1	CLA Floating-Point Result Register 1		Go
38h	_MR2	CLA Floating-Point Result Register 2		Go
3Ch	_MR3	CLA Floating-Point Result Register 3		Go
42h	_MPSACTL	CLA PSA Control Register	EALLOW	Go
44h	_MPSA1	CLA PSA1 Register	EALLOW	Go
46h	_MPSA2	CLA PSA2 Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 7-36 shows the codes that are used for access types in this section.

Table 7-36. CLA_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		

Table 7-36. CLA_REGS Access Type Codes (continued)

Access Type	Code	Description
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

7.8.4.1 MVECT1 Register (Offset = 0h) [Reset = 0h]

MVECT1 is shown in Figure 7-10 and described in Table 7-37.

Return to the Summary Table.

Each CLA interrupt has its own interrupt vector (MVECT1 to MVECT8). This interrupt vector points to the first instruction of the associated task. When a task begins, the CLA will start fetching instructions at the location indicated by the appropriate MVECT register .

Figure 7-10. MVECT1 Register

15	14	13	12	11	10	9	8
MVECT							
R/W-0h							
7	6	5	4	3	2	1	0
MVECT							
R/W-0h							

Table 7-37. MVECT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	MVECT	R/W	0h	MPC Start Address: These bits specify the start address for the given interrupt (task). The address range of the CLA with a 16-bit MVECT is 64Kx16 words or 32K CLA instructions. There is one MVECT register per interrupt (task). Interrupt 1 uses MVECT1, interrupt 2 uses MVECT2 and so forth. Note: While the CLA is running or executing a task, the CPU can change the MVECT values.. Reset type: SYSRSn

7.8.4.2 MVECT2 Register (Offset = 1h) [Reset = 0h]

MVECT2 is shown in Figure 7-11 and described in Table 7-38.

Return to the Summary Table.

Each CLA interrupt has its own interrupt vector (MVECT1 to MVECT8). This interrupt vector points to the first instruction of the associated task. When a task begins, the CLA will start fetching instructions at the location indicated by the appropriate MVECT register .

Figure 7-11. MVECT2 Register

15	14	13	12	11	10	9	8
MVECT							
R/W-0h							
7	6	5	4	3	2	1	0
MVECT							
R/W-0h							

Table 7-38. MVECT2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	MVECT	R/W	0h	MPC Start Address: These bits specify the start address for the given interrupt (task). The address range of the CLA with a 16-bit MVECT is 64Kx16 words or 32K CLA instructions. There is one MVECT register per interrupt (task). Interrupt 1 uses MVECT1, interrupt 2 uses MVECT2 and so forth. Note: While the CLA is running or executing a task, the CPU can change the MVECT values.. Reset type: SYSRSn

7.8.4.3 MVECT3 Register (Offset = 2h) [Reset = 0h]

MVECT3 is shown in Figure 7-12 and described in Table 7-39.

Return to the Summary Table.

Each CLA interrupt has its own interrupt vector (MVECT1 to MVECT8). This interrupt vector points to the first instruction of the associated task. When a task begins, the CLA will start fetching instructions at the location indicated by the appropriate MVECT register .

Figure 7-12. MVECT3 Register

15	14	13	12	11	10	9	8
MVECT							
R/W-0h							
7	6	5	4	3	2	1	0
MVECT							
R/W-0h							

Table 7-39. MVECT3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	MVECT	R/W	0h	MPC Start Address: These bits specify the start address for the given interrupt (task). The address range of the CLA with a 16-bit MVECT is 64Kx16 words or 32K CLA instructions. There is one MVECT register per interrupt (task). Interrupt 1 uses MVECT1, interrupt 2 uses MVECT2 and so forth. Note: While the CLA is running or executing a task, the CPU can change the MVECT values.. Reset type: SYSRSn

7.8.4.4 MVECT4 Register (Offset = 3h) [Reset = 0h]

MVECT4 is shown in Figure 7-13 and described in Table 7-40.

Return to the Summary Table.

Each CLA interrupt has its own interrupt vector (MVECT1 to MVECT8). This interrupt vector points to the first instruction of the associated task. When a task begins, the CLA will start fetching instructions at the location indicated by the appropriate MVECT register .

Figure 7-13. MVECT4 Register

15	14	13	12	11	10	9	8
MVECT							
R/W-0h							
7	6	5	4	3	2	1	0
MVECT							
R/W-0h							

Table 7-40. MVECT4 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	MVECT	R/W	0h	MPC Start Address: These bits specify the start address for the given interrupt (task). The address range of the CLA with a 16-bit MVECT is 64Kx16 words or 32K CLA instructions. There is one MVECT register per interrupt (task). Interrupt 1 uses MVECT1, interrupt 2 uses MVECT2 and so forth. Note: While the CLA is running or executing a task, the CPU can change the MVECT values.. Reset type: SYSRSn

7.8.4.5 MVECT5 Register (Offset = 4h) [Reset = 0h]

MVECT5 is shown in Figure 7-14 and described in Table 7-41.

Return to the Summary Table.

Each CLA interrupt has its own interrupt vector (MVECT1 to MVECT8). This interrupt vector points to the first instruction of the associated task. When a task begins, the CLA will start fetching instructions at the location indicated by the appropriate MVECT register .

Figure 7-14. MVECT5 Register

15	14	13	12	11	10	9	8
MVECT							
R/W-0h							
7	6	5	4	3	2	1	0
MVECT							
R/W-0h							

Table 7-41. MVECT5 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	MVECT	R/W	0h	MPC Start Address: These bits specify the start address for the given interrupt (task). The address range of the CLA with a 16-bit MVECT is 64Kx16 words or 32K CLA instructions. There is one MVECT register per interrupt (task). Interrupt 1 uses MVECT1, interrupt 2 uses MVECT2 and so forth. Note: While the CLA is running or executing a task, the CPU can change the MVECT values.. Reset type: SYSRSn

7.8.4.6 MVECT6 Register (Offset = 5h) [Reset = 0h]

MVECT6 is shown in Figure 7-15 and described in Table 7-42.

Return to the Summary Table.

Each CLA interrupt has its own interrupt vector (MVECT1 to MVECT8). This interrupt vector points to the first instruction of the associated task. When a task begins, the CLA will start fetching instructions at the location indicated by the appropriate MVECT register .

Figure 7-15. MVECT6 Register

15	14	13	12	11	10	9	8
MVECT							
R/W-0h							
7	6	5	4	3	2	1	0
MVECT							
R/W-0h							

Table 7-42. MVECT6 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	MVECT	R/W	0h	MPC Start Address: These bits specify the start address for the given interrupt (task). The address range of the CLA with a 16-bit MVECT is 64Kx16 words or 32K CLA instructions. There is one MVECT register per interrupt (task). Interrupt 1 uses MVECT1, interrupt 2 uses MVECT2 and so forth. Note: While the CLA is running or executing a task, the CPU can change the MVECT values.. Reset type: SYSRSn

7.8.4.7 MVECT7 Register (Offset = 6h) [Reset = 0h]

MVECT7 is shown in Figure 7-16 and described in Table 7-43.

Return to the Summary Table.

Each CLA interrupt has its own interrupt vector (MVECT1 to MVECT8). This interrupt vector points to the first instruction of the associated task. When a task begins, the CLA will start fetching instructions at the location indicated by the appropriate MVECT register .

Figure 7-16. MVECT7 Register

15	14	13	12	11	10	9	8
MVECT							
R/W-0h							
7	6	5	4	3	2	1	0
MVECT							
R/W-0h							

Table 7-43. MVECT7 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	MVECT	R/W	0h	MPC Start Address: These bits specify the start address for the given interrupt (task). The address range of the CLA with a 16-bit MVECT is 64Kx16 words or 32K CLA instructions. There is one MVECT register per interrupt (task). Interrupt 1 uses MVECT1, interrupt 2 uses MVECT2 and so forth. Note: While the CLA is running or executing a task, the CPU can change the MVECT values.. Reset type: SYSRSn

7.8.4.8 MVECT8 Register (Offset = 7h) [Reset = 0h]

MVECT8 is shown in Figure 7-17 and described in Table 7-44.

Return to the Summary Table.

Each CLA interrupt has its own interrupt vector (MVECT1 to MVECT8). This interrupt vector points to the first instruction of the associated task. When a task begins, the CLA will start fetching instructions at the location indicated by the appropriate MVECT register .

Figure 7-17. MVECT8 Register

15	14	13	12	11	10	9	8
MVECT							
R/W-0h							
7	6	5	4	3	2	1	0
MVECT							
R/W-0h							

Table 7-44. MVECT8 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	MVECT	R/W	0h	MPC Start Address: These bits specify the start address for the given interrupt (task). The address range of the CLA with a 16-bit MVECT is 64Kx16 words or 32K CLA instructions. There is one MVECT register per interrupt (task). Interrupt 1 uses MVECT1, interrupt 2 uses MVECT2 and so forth. Note: While the CLA is running or executing a task, the CPU can change the MVECT values.. Reset type: SYSRSn

7.8.4.9 MCTL Register (Offset = 10h) [Reset = 0h]

MCTL is shown in Figure 7-18 and described in Table 7-45.

Return to the Summary Table.

Control Register

Figure 7-18. MCTL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					IACKE	SOFTRESET	HARDRESET
R-0h					R/W-0h	R-0/W1S-0h	R-0/W1S-0h

Table 7-45. MCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-3	RESERVED	R	0h	Reserved
2	IACKE	R/W	0h	IACK Operation Enable Bit: Writing a "1" to this bit will enable the IACK operation for setting the MIFR bits in the same manner as the MIFRC register (write of "1" will set respective MIFR bit). At reset, this feature is disabled. This feature enables the C28 CPU to efficiently trigger a task. Note: IACK operation should ignore EALLOW status of C28 core when accessing the MIFRC register. Reset type: SYSRSn 0h (R/W) = The CLA ignores the IACK instruction. (default) 1h (R/W) = Enable the main CPU to use the IACK #16bit instruction to set MIFR bits in the same manner as writing to the MIFRC register. Each bit in the operand, #16bit, corresponds to a bit in the MIFRC register. Using IACK has the advantage of not having to first set the EALLOW bit. This allows the main CPU to efficiently trigger a CLA task through software. Examples IACK #0x0001 Write a 1 to MIFRC bit 0 to force task 1 IACK #0x0003 Write a 1 to MIFRC bit 0 and 1 to force task 1 and task 2
1	SOFTRESET	R-0/W1S	0h	Soft Reset Bit: Writing a "1" to this bit will stop a current task, clear the RUN flag and also clear all bits in the MIER register. Writes of "0" are ignored and reads always return a "0". Note: After issuing SOFTRESET command, user should wait at least 1 clock cycle before attempting to write to MIER register. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 are ignored. 1h (R/W) = Writing a 1 will cause a soft reset of the CLA. This will stop the current task, clear the MIRUN flag and clear all bits in the MIER register. After a soft reset you must wait at least 1 SYSCLKOUT cycle before reconfiguring the MIER bits. If these two operations are done back-to-back then the MIER bits will not get set.
0	HARDRESET	R-0/W1S	0h	Hard Reset Bit: Writing a "1" to this bit will cause a HARD reset on the CLA. The behavior of a HARD reset is the same as a system reset SYSRSn on the CLA. Writes of "0" are ignored and reads always return a "0". Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 are ignored. 1h (R/W) = Writing a 1 will cause a hard reset of the CLA. This will set all CLA registers to their default state.

7.8.4.10 _MVECTBGRNDACTIVE Register (Offset = 1Bh) [Reset = 0h]

_MVECTBGRNDACTIVE is shown in Figure 7-19 and described in Table 7-46.

Return to the Summary Table.

Gives the current interrupted MPC value of the background task, if the background task was running and interrupted, or reflects the MVECTBGRND value, if MCTLBGRND.BGSTART bit is 0.

Figure 7-19. _MVECTBGRNDACTIVE Register

15	14	13	12	11	10	9	8
i16							
R-0h							
7	6	5	4	3	2	1	0
i16							
R-0h							

Table 7-46. _MVECTBGRNDACTIVE Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	i16	R	0h	Gives the current interrupted MPC value of the background task, if the background task was running and interrupted, or reflects the MVECTBGRND value, if MCTLBGRND.BGSTART bit is 0. Reset type: SYSRSn

7.8.4.11 SOFTINTEN Register (Offset = 1Ch) [Reset = 0h]

SOFTINTEN is shown in Figure 7-20 and described in Table 7-47.

Return to the Summary Table.

Enables the ability to generate CLA task interrupt from within the task, by writing to SOFTINTFRC register. SOFTINTFRC register can only be written from CLA. Only reads are allowed from CPU. Writes are not allowed from CPU.

Figure 7-20. SOFTINTEN Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
TASK8	TASK7	TASK6	TASK5	TASK4	TASK3	TASK2	TASK1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 7-47. SOFTINTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	TASK8	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
6	TASK7	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
5	TASK6	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
4	TASK5	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
3	TASK4	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
2	TASK3	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn
1	TASK2	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn

Table 7-47. SOFTINTEN Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	TASK1	R/W	0h	0: End-Of-Task Interrupt is fired for the respective task 1: Enable Software Interrupt for the respective task. End-of-Task interrupt is not sent to CPU in this case. Note: SOFTINTEN register is read only in the CPU memory map. Reset type: SYSRSn

7.8.4.12 _MSTSBGRND Register (Offset = 1Dh) [Reset = 0h]

_MSTSBGRND is shown in Figure 7-21 and described in Table 7-48.

Return to the Summary Table.

Status bits for the backgrondtask.

Figure 7-21. _MSTSBGRND Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					BGOVF	_BGINTM	RUN
R-0h					R/W1C-0h	R-0h	R-0h

Table 7-48. _MSTSBGRND Register Field Descriptions

Bit	Field	Type	Reset	Description
15-3	RESERVED	R	0h	Reserved
2	BGOVF	R/W1C	0h	Value of 1 indicates a hardware trigger (which is enabled) occurred while the MCTLBGRND.BGSTART bit is set. Writing a value of 1 to this bit clears the BGOVF bit. Write of 0 has no effect, Value of 0 indicates the background task trigger did not result in a overflow. Reset type: SYSRSn
1	_BGINTM	R	0h	Value of 1 indicates that backgroiund task will not be interrupted. This bit is set when MSETC _BGINTM bit is executed. Value of 0 indicates that background task can be interrupted. Reset type: SYSRSn
0	RUN	R	0h	Value of 1 indicates that background task is running. Value of 0 indicates that background task is not running. Reset type: SYSRSn

7.8.4.13 _MCTLBGRND Register (Offset = 1Eh) [Reset = 0h]

_MCTLBGRND is shown in Figure 7-22 and described in Table 7-49.

Return to the Summary Table.

Holds the configuration bits to start the background task, enable hardware trigger.

Figure 7-22. _MCTLBGRND Register

15	14	13	12	11	10	9	8
BGEN	RESERVED						
R/W-0h	R-0h						
7	6	5	4	3	2	1	0
RESERVED						TRIGEN	BGSTART
R-0h						R/W-0h	R/W1S-0h

Table 7-49. _MCTLBGRND Register Field Descriptions

Bit	Field	Type	Reset	Description
15	BGEN	R/W	0h	0 Background task is disabled, BGSTART will not be set either in a hardware trigger or by writing 1 to BGSTART bit. 1 Background task is enabled and MIER[INT8] will be cleared, preventing task 8 from triggering. Reset type: SYSRSn
14-2	RESERVED	R	0h	Reserved
1	TRIGEN	R/W	0h	Hardware trigger enable for the background task. 1 Hardware trigger is enabled. 0 Hardware trigger is disabled. Note: Trigger source for the background task will be the same as that for task 8 Reset type: SYSRSn
0	BGSTART	R/W1S	0h	Value of 1 will start the background task, provided there are no other pending tasks. - Value of 0 has no effect if the background task has not started. - This bit is also set by hardware, if MCTLBGRND.TRIGEN = 1 and a hardware trigger occurs. - This bit is cleared by hardware when a MSTOP instruction occurs in the background task - If the background task is running and this bit is cleared, it will not have any effect on the task execution. Reset type: SYSRSn

7.8.4.14 _MVECTBGRND Register (Offset = 1Fh) [Reset = 0h]

_MVECTBGRND is shown in Figure 7-23 and described in Table 7-50.

Return to the Summary Table.

These bits specify the start address for the background task . The value in this register is forced into the MPC register when the background task starts.

Figure 7-23. _MVECTBGRND Register

15	14	13	12	11	10	9	8
i16							
R/W-0h							
7	6	5	4	3	2	1	0
i16							
R/W-0h							

Table 7-50. _MVECTBGRND Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	i16	R/W	0h	MPC Start Address: These bits specify the start address for the background task . The value in this register is forced into the MPC register, when the background task starts. Reset type: SYSRSn

7.8.4.15 MIFR Register (Offset = 20h) [Reset = 0h]

MIFR is shown in Figure 7-24 and described in Table 7-51.

Return to the Summary Table.

Each bit in the interrupt flag register corresponds to a CLA task. The corresponding bit is automatically set when the task request is received from the peripheral interrupt. The bit can also be set by the main CPU writing to the MIFRC register or using the IACK instruction to start the task. To use the IACK instruction to begin a task first enable this feature in the MCTL register. If the bit is already set when a new peripheral interrupt is received, then the corresponding overflow bit will be set in the MIOVF register.
The corresponding MIFR bit is automatically cleared when the task begins execution. This will occur if the interrupt is enabled in the MIER register and no other higher priority task is pending. The bits can also be cleared manually by writing to the MICLR register. Writes to the MIFR register are ignored.

Figure 7-24. MIFR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
INT8	INT7	INT6	INT5	INT4	INT3	INT2	INT1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 7-51. MIFR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	INT8	R	0h	These bits, when set to "1", indicate a valid peripheral interrupt has been latched by the CLA. Writes to this register are ignored. The IFR flag bit is automatically cleared if the respective interrupt is enabled in the MIER register and the respective task starts running. If a new peripheral interrupt attempts to set the bit to "1" while on the same cycle the task tries to clear it, then the peripheral interrupt will have priority. The IFR flag bits can also be set and cleared by the MIFRC and MICLR registers. If the MIFRC register is trying to set the respective bit while a new task tries to clear it, then the MIFRC event has priority. If the MICLR register is trying to clear the respective bit and a peripheral interrupt occurs on the same cycle, then the peripheral interrupt has priority. The respective overflow flag in the MIOVF register will not be set under this condition. Reset type: SYSRSn 0h (R/W) = TASK_FLAG_DISABLE Task 8 interrupt is currently not flagged (default) 1h (R/W) = TASK_FLAG_ENABLE Task 8 interrupt has been received and is pending execution

Table 7-51. MIFR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	INT7	R	0h	These bits, when set to "1", indicate a valid peripheral interrupt has been latched by the CLA. Writes to this register are ignored. The IFR flag bit is automatically cleared if the respective interrupt is enabled in the MIER register and the respective task starts running. If a new peripheral interrupt attempts to set the bit to "1" while on the same cycle the task tries to clear it, then the peripheral interrupt will have priority. The IFR flag bits can also be set and cleared by the MIFRC and MICLR registers. If the MIFRC register is trying to set the respective bit while a new task tries to clear it, then the MIFRC event has priority. If the MICLR register is trying to clear the respective bit and a peripheral interrupt occurs on the same cycle, then the peripheral interrupt has priority. The respective overflow flag in the MIOVF register will not be set under this condition. Reset type: SYSRSn 0h (R/W) = TASK_FLAG_DISABLE Task 7 interrupt is currently not flagged (default) 1h (R/W) = TASK_FLAG_ENABLE Task 7 interrupt has been received and is pending execution
5	INT6	R	0h	These bits, when set to "1", indicate a valid peripheral interrupt has been latched by the CLA. Writes to this register are ignored. The IFR flag bit is automatically cleared if the respective interrupt is enabled in the MIER register and the respective task starts running. If a new peripheral interrupt attempts to set the bit to "1" while on the same cycle the task tries to clear it, then the peripheral interrupt will have priority. The IFR flag bits can also be set and cleared by the MIFRC and MICLR registers. If the MIFRC register is trying to set the respective bit while a new task tries to clear it, then the MIFRC event has priority. If the MICLR register is trying to clear the respective bit and a peripheral interrupt occurs on the same cycle, then the peripheral interrupt has priority. The respective overflow flag in the MIOVF register will not be set under this condition. Reset type: SYSRSn 0h (R/W) = TASK_FLAG_DISABLE Task 6 interrupt is currently not flagged (default) 1h (R/W) = TASK_FLAG_ENABLE Task 6 interrupt has been received and is pending execution
4	INT5	R	0h	These bits, when set to "1", indicate a valid peripheral interrupt has been latched by the CLA. Writes to this register are ignored. The IFR flag bit is automatically cleared if the respective interrupt is enabled in the MIER register and the respective task starts running. If a new peripheral interrupt attempts to set the bit to "1" while on the same cycle the task tries to clear it, then the peripheral interrupt will have priority. The IFR flag bits can also be set and cleared by the MIFRC and MICLR registers. If the MIFRC register is trying to set the respective bit while a new task tries to clear it, then the MIFRC event has priority. If the MICLR register is trying to clear the respective bit and a peripheral interrupt occurs on the same cycle, then the peripheral interrupt has priority. The respective overflow flag in the MIOVF register will not be set under this condition. Reset type: SYSRSn 0h (R/W) = TASK_FLAG_DISABLE Task 5 interrupt is currently not flagged (default) 1h (R/W) = TASK_FLAG_ENABLE Task 5 interrupt has been received and is pending execution

Table 7-51. MIFR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	INT4	R	0h	These bits, when set to "1", indicate a valid peripheral interrupt has been latched by the CLA. Writes to this register are ignored. The IFR flag bit is automatically cleared if the respective interrupt is enabled in the MIER register and the respective task starts running. If a new peripheral interrupt attempts to set the bit to "1" while on the same cycle the task tries to clear it, then the peripheral interrupt will have priority. The IFR flag bits can also be set and cleared by the MIFRC and MICLR registers. If the MIFRC register is trying to set the respective bit while a new task tries to clear it, then the MIFRC event has priority. If the MICLR register is trying to clear the respective bit and a peripheral interrupt occurs on the same cycle, then the peripheral interrupt has priority. The respective overflow flag in the MIOVF register will not be set under this condition. Reset type: SYSRSn 0h (R/W) = TASK_FLAG_DISABLE Task 4 interrupt is currently not flagged (default) 1h (R/W) = TASK_FLAG_ENABLE Task 4 interrupt has been received and is pending execution
2	INT3	R	0h	These bits, when set to "1", indicate a valid peripheral interrupt has been latched by the CLA. Writes to this register are ignored. The IFR flag bit is automatically cleared if the respective interrupt is enabled in the MIER register and the respective task starts running. If a new peripheral interrupt attempts to set the bit to "1" while on the same cycle the task tries to clear it, then the peripheral interrupt will have priority. The IFR flag bits can also be set and cleared by the MIFRC and MICLR registers. If the MIFRC register is trying to set the respective bit while a new task tries to clear it, then the MIFRC event has priority. If the MICLR register is trying to clear the respective bit and a peripheral interrupt occurs on the same cycle, then the peripheral interrupt has priority. The respective overflow flag in the MIOVF register will not be set under this condition. Reset type: SYSRSn 0h (R/W) = TASK_FLAG_DISABLE Task 3 interrupt is currently not flagged (default) 1h (R/W) = TASK_FLAG_ENABLE Task 3 interrupt has been received and is pending execution
1	INT2	R	0h	These bits, when set to "1", indicate a valid peripheral interrupt has been latched by the CLA. Writes to this register are ignored. The IFR flag bit is automatically cleared if the respective interrupt is enabled in the MIER register and the respective task starts running. If a new peripheral interrupt attempts to set the bit to "1" while on the same cycle the task tries to clear it, then the peripheral interrupt will have priority. The IFR flag bits can also be set and cleared by the MIFRC and MICLR registers. If the MIFRC register is trying to set the respective bit while a new task tries to clear it, then the MIFRC event has priority. If the MICLR register is trying to clear the respective bit and a peripheral interrupt occurs on the same cycle, then the peripheral interrupt has priority. The respective overflow flag in the MIOVF register will not be set under this condition. Reset type: SYSRSn 0h (R/W) = TASK_FLAG_DISABLE Task 2 interrupt is currently not flagged (default) 1h (R/W) = TASK_FLAG_ENABLE Task 2 interrupt has been received and is pending execution

Table 7-51. MIFR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INT1	R	0h	These bits, when set to "1", indicate a valid peripheral interrupt has been latched by the CLA. Writes to this register are ignored. The IFR flag bit is automatically cleared if the respective interrupt is enabled in the MIER register and the respective task starts running. If a new peripheral interrupt attempts to set the bit to "1" while on the same cycle the task tries to clear it, then the peripheral interrupt will have priority. The IFR flag bits can also be set and cleared by the MIFRC and MICLR registers. If the MIFRC register is trying to set the respective bit while a new task tries to clear it, then the MIFRC event has priority. If the MICLR register is trying to clear the respective bit and a peripheral interrupt occurs on the same cycle, then the peripheral interrupt has priority. The respective overflow flag in the MIOVF register will not be set under this condition. Reset type: SYSRSn 0h (R/W) = TASK_FLAG_DISABLE Task 1 interrupt is currently not flagged (default) 1h (R/W) = TASK_FLAG_ENABLE Task 1 interrupt has been received and is pending execution

7.8.4.16 MIOVF Register (Offset = 21h) [Reset = 0h]

MIOVF is shown in Figure 7-25 and described in Table 7-52.

Return to the Summary Table.

Each bit in the overflow flag register corresponds to a CLA task. The bit is set when an interrupt overflow event has occurred for the specific task. An overflow event occurs when the MIFR register bit is already set when a new interrupt is received from a peripheral source. The MIOVF bits are only affected by peripheral interrupt events. They do not respond to a task request by the main CPU IACK instruction or by directly setting MIFR bits. The overflow flag will remain latched and can only be cleared by writing to the overflow flag clear (MICLROVF) register. Writes to the MIOVF register are ignored.

Figure 7-25. MIOVF Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
INT8	INT7	INT6	INT5	INT4	INT3	INT2	INT1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 7-52. MIOVF Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	INT8	R	0h	These bits, when set to "1", indicate an interrupt overflow event occurred. Such an event occurs when the IFR bit is already set. An overflow event remains latched and respective bits can only be cleared by writing to the MICLROVF register. If the MIFR bit is being cleared by a new task on the same cycle as a new peripheral interrupt occurs, the overflow flag will not be affected and the respective MIFR bit will be set. If the MIOVF bit is being cleared by the MICLROVF register on the same cycle as the overflow bit is being set by hardware, then the hardware will have priority. Notes: [1] The MIOVF bits are only affected by peripheral interrupt events. Forcing an interrupt using the MIFRC or IACK operation will not set the overflow flag even if the MIFR bit is set. Reset type: SYSRSn 0h (R/W) = A task 8 interrupt overflow has not occurred (default) 1h (R/W) = A task 8 interrupt overflow has occurred
6	INT7	R	0h	These bits, when set to "1", indicate an interrupt overflow event occurred. Such an event occurs when the IFR bit is already set. An overflow event remains latched and respective bits can only be cleared by writing to the MICLROVF register. If the MIFR bit is being cleared by a new task on the same cycle as a new peripheral interrupt occurs, the overflow flag will not be affected and the respective MIFR bit will be set. If the MIOVF bit is being cleared by the MICLROVF register on the same cycle as the overflow bit is being set by hardware, then the hardware will have priority. Notes: [1] The MIOVF bits are only affected by peripheral interrupt events. Forcing an interrupt using the MIFRC or IACK operation will not set the overflow flag even if the MIFR bit is set. Reset type: SYSRSn 0h (R/W) = A task 7 interrupt overflow has not occurred (default) 1h (R/W) = A task 7 interrupt overflow has occurred

Table 7-52. MIOVF Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	INT6	R	0h	These bits, when set to "1", indicate an interrupt overflow event occurred. Such an event occurs when the IFR bit is already set. An overflow event remains latched and respective bits can only be cleared by writing to the MICLROVF register. If the MIFR bit is being cleared by a new task on the same cycle as a new peripheral interrupt occurs, the overflow flag will not be affected and the respective MIFR bit will be set. If the MIOVF bit is being cleared by the MICLROVF register on the same cycle as the overflow bit is being set by hardware, then the hardware will have priority. Notes: [1] The MIOVF bits are only affected by peripheral interrupt events. Forcing an interrupt using the MIFRC or IACK operation will not set the overflow flag even if the MIFR bit is set. Reset type: SYSRSn 0h (R/W) = A task 6 interrupt overflow has not occurred (default) 1h (R/W) = A task 6 interrupt overflow has occurred
4	INT5	R	0h	These bits, when set to "1", indicate an interrupt overflow event occurred. Such an event occurs when the IFR bit is already set. An overflow event remains latched and respective bits can only be cleared by writing to the MICLROVF register. If the MIFR bit is being cleared by a new task on the same cycle as a new peripheral interrupt occurs, the overflow flag will not be affected and the respective MIFR bit will be set. If the MIOVF bit is being cleared by the MICLROVF register on the same cycle as the overflow bit is being set by hardware, then the hardware will have priority. Notes: [1] The MIOVF bits are only affected by peripheral interrupt events. Forcing an interrupt using the MIFRC or IACK operation will not set the overflow flag even if the MIFR bit is set. Reset type: SYSRSn 0h (R/W) = A task 5 interrupt overflow has not occurred (default) 1h (R/W) = A task 5 interrupt overflow has occurred
3	INT4	R	0h	These bits, when set to "1", indicate an interrupt overflow event occurred. Such an event occurs when the IFR bit is already set. An overflow event remains latched and respective bits can only be cleared by writing to the MICLROVF register. If the MIFR bit is being cleared by a new task on the same cycle as a new peripheral interrupt occurs, the overflow flag will not be affected and the respective MIFR bit will be set. If the MIOVF bit is being cleared by the MICLROVF register on the same cycle as the overflow bit is being set by hardware, then the hardware will have priority. Notes: [1] The MIOVF bits are only affected by peripheral interrupt events. Forcing an interrupt using the MIFRC or IACK operation will not set the overflow flag even if the MIFR bit is set. Reset type: SYSRSn 0h (R/W) = A task 4 interrupt overflow has not occurred (default) 1h (R/W) = A task 4 interrupt overflow has occurred
2	INT3	R	0h	These bits, when set to "1", indicate an interrupt overflow event occurred. Such an event occurs when the IFR bit is already set. An overflow event remains latched and respective bits can only be cleared by writing to the MICLROVF register. If the MIFR bit is being cleared by a new task on the same cycle as a new peripheral interrupt occurs, the overflow flag will not be affected and the respective MIFR bit will be set. If the MIOVF bit is being cleared by the MICLROVF register on the same cycle as the overflow bit is being set by hardware, then the hardware will have priority. Notes: [1] The MIOVF bits are only affected by peripheral interrupt events. Forcing an interrupt using the MIFRC or IACK operation will not set the overflow flag even if the MIFR bit is set. Reset type: SYSRSn 0h (R/W) = A task 3 interrupt overflow has not occurred (default) 1h (R/W) = A task 3 interrupt overflow has occurred

Table 7-52. MIOVF Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	INT2	R	0h	These bits, when set to "1", indicate an interrupt overflow event occurred. Such an event occurs when the IFR bit is already set. An overflow event remains latched and respective bits can only be cleared by writing to the MICLROVF register. If the MIFR bit is being cleared by a new task on the same cycle as a new peripheral interrupt occurs, the overflow flag will not be affected and the respective MIFR bit will be set. If the MIOVF bit is being cleared by the MICLROVF register on the same cycle as the overflow bit is being set by hardware, then the hardware will have priority. Notes: [1] The MIOVF bits are only affected by peripheral interrupt events. Forcing an interrupt using the MIFRC or IACK operation will not set the overflow flag even if the MIFR bit is set. Reset type: SYSRSn 0h (R/W) = A task 2 interrupt overflow has not occurred (default) 1h (R/W) = A task 2 interrupt overflow has occurred
0	INT1	R	0h	These bits, when set to "1", indicate an interrupt overflow event occurred. Such an event occurs when the IFR bit is already set. An overflow event remains latched and respective bits can only be cleared by writing to the MICLROVF register. If the MIFR bit is being cleared by a new task on the same cycle as a new peripheral interrupt occurs, the overflow flag will not be affected and the respective MIFR bit will be set. If the MIOVF bit is being cleared by the MICLROVF register on the same cycle as the overflow bit is being set by hardware, then the hardware will have priority. Notes: [1] The MIOVF bits are only affected by peripheral interrupt events. Forcing an interrupt using the MIFRC or IACK operation will not set the overflow flag even if the MIFR bit is set. Reset type: SYSRSn 0h (R/W) = A task 1 interrupt overflow has not occurred (default) 1h (R/W) = A task 1 interrupt overflow has occurred

7.8.4.17 MIFRC Register (Offset = 22h) [Reset = 0h]

MIFRC is shown in Figure 7-26 and described in Table 7-53.

Return to the Summary Table.

The interrupt force register can be used by the main CPU to start tasks through software. Writing a 1 to a MIFRC bit will set the corresponding bit in the MIFR register. Writes of 0 are ignored and reads always return 0. The IACK #16bit operation can also be used to start tasks and has the same effect as the MIFRC register. To enable IACK to set MIFR bits you must first set the MCTL[IACKE] bit. Using IACK has the advantage of not having to first set the EALLOW bit. This allows the main CPU to efficiently trigger CLA tasks through software.

Figure 7-26. MIFRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
INT8	INT7	INT6	INT5	INT4	INT3	INT2	INT1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 7-53. MIFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	INT8	R-0/W1S	0h	Writing a "1" to any of the bits will set the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to force the task 8 interrupt
6	INT7	R-0/W1S	0h	Writing a "1" to any of the bits will set the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to force the task 7 interrupt
5	INT6	R-0/W1S	0h	Writing a "1" to any of the bits will set the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to force the task 6 interrupt
4	INT5	R-0/W1S	0h	Writing a "1" to any of the bits will set the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to force the task 5 interrupt

Table 7-53. MIFRC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	INT4	R-0/W1S	0h	Writing a "1" to any of the bits will set the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to force the task 4 interrupt
2	INT3	R-0/W1S	0h	Writing a "1" to any of the bits will set the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to force the task 3 interrupt
1	INT2	R-0/W1S	0h	Writing a "1" to any of the bits will set the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to force the task 2 interrupt
0	INT1	R-0/W1S	0h	Writing a "1" to any of the bits will set the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to force the task 1 interrupt

7.8.4.18 MICLR Register (Offset = 23h) [Reset = 0h]

MICLR is shown in Figure 7-27 and described in Table 7-54.

Return to the Summary Table.

Normally bits in the MIFR register are automatically cleared when a task begins. The interrupt flag clear register can be used to instead manually clear bits in the interrupt flag (MIFR) register. Writing a 1 to a MICLR bit will clear the corresponding bit in the MIFR register. Writes of 0 are ignored and reads always return 0.

Figure 7-27. MICLR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
INT8	INT7	INT6	INT5	INT4	INT3	INT2	INT1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 7-54. MICLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	INT8	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 8 interrupt flag
6	INT7	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 7 interrupt flag
5	INT6	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 6 interrupt flag
4	INT5	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 5 interrupt flag

Table 7-54. MICLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	INT4	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 4 interrupt flag
2	INT3	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 3 interrupt flag
1	INT2	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 2 interrupt flag
0	INT1	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIFR bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIFR register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 1 interrupt flag

7.8.4.19 MICLROVF Register (Offset = 24h) [Reset = 0h]

MICLROVF is shown in Figure 7-28 and described in Table 7-55.

Return to the Summary Table.

Overflow flag bits in the MIOVF register are latched until manually cleared using the MICLROVF register. Writing a 1 to a MICLROVF bit will clear the corresponding bit in the MIOVF register. Writes of 0 are ignored and reads always return 0.

Figure 7-28. MICLROVF Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
INT8	INT7	INT6	INT5	INT4	INT3	INT2	INT1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 7-55. MICLROVF Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	INT8	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIOVF bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIOVF register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 8 interrupt overflow flag
6	INT7	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIOVF bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIOVF register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 7 interrupt overflow flag
5	INT6	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIOVF bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIOVF register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 6 interrupt overflow flag
4	INT5	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIOVF bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIOVF register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 5 interrupt overflow flag
3	INT4	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIOVF bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIOVF register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 4 interrupt overflow flag

Table 7-55. MICLROVF Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	INT3	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIOVF bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIOVF register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 3 interrupt overflow flag
1	INT2	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIOVF bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIOVF register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 2 interrupt overflow flag
0	INT1	R-0/W1S	0h	Writing a "1" to any of the bits will clear the corresponding MIOVF bit. Writes of "0" are ignored. Reads always return 0. Notes: [1] Refer to MIOVF register description for handling of boundary conditions. Reset type: SYSRSn 0h (R/W) = This bit always reads back 0 and writes of 0 have no effect 1h (R/W) = Write a 1 to clear the task 1 interrupt overflow flag

7.8.4.20 MIER Register (Offset = 25h) [Reset = 0h]

MIER is shown in Figure 7-29 and described in Table 7-56.

Return to the Summary Table.

Setting the bits in the interrupt enable register (MIER) allow an incoming interrupt or main CPU software to start the corresponding CLA task. Writing a 0 will block the task, but the interrupt request will still be latched in the flag register (MIFLG). Setting the MIER register bit to 0 while the corresponding task is executing will have no effect on the task. The task will continue to run until it hits the MSTOP instruction. When a soft reset is issued, the MIER bits are cleared. There should always be at least a 1 SYSCLKOUT delay between issuing the soft reset and reconfiguring the MIER bits.

Figure 7-29. MIER Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
INT8	INT7	INT6	INT5	INT4	INT3	INT2	INT1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 7-56. MIER Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	INT8	R/W	0h	Setting any of the bits to "1" enables the corresponding interrupt from triggering a corresponding CLA task. Writing a "0" blocks the interrupt, but the interrupt can still be latched by the MIFR register. When an interrupt is enabled and the corresponding MIFR bit is set to "1", the CLA will start executing the corresponding task and automatically clear the corresponding MIFR bit. Interrupts are be serviced in normal priority order. Notes: [1] If a task is currently executing and the corresponding MIER bit is cleared to "0", it will have no effect on the task. The task will run until it hits the STOP instruction. Reset type: SYSRSn 0h (R/W) = TASK_INT_DISABLE Task 8 interrupt is disabled (default) 1h (R/W) = TASK_INT_ENABLE Task 8 interrupt is enabled
6	INT7	R/W	0h	Setting any of the bits to "1" enables the corresponding interrupt from triggering a corresponding CLA task. Writing a "0" blocks the interrupt, but the interrupt can still be latched by the MIFR register. When an interrupt is enabled and the corresponding MIFR bit is set to "1", the CLA will start executing the corresponding task and automatically clear the corresponding MIFR bit. Interrupts are be serviced in normal priority order. Notes: [1] If a task is currently executing and the corresponding MIER bit is cleared to "0", it will have no effect on the task. The task will run until it hits the STOP instruction. Reset type: SYSRSn 0h (R/W) = TASK_INT_DISABLE Task 7 interrupt is disabled (default) 1h (R/W) = TASK_INT_ENABLE Task 7 interrupt is enabled

Table 7-56. MIER Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	INT6	R/W	0h	Setting any of the bits to "1" enables the corresponding interrupt from triggering a corresponding CLA task. Writing a "0" blocks the interrupt, but the interrupt can still be latched by the MIFR register. When an interrupt is enabled and the corresponding MIFR bit is set to "1", the CLA will start executing the corresponding task and automatically clear the corresponding MIFR bit. Interrupts are be serviced in normal priority order. Notes: [1] If a task is currently executing and the corresponding MIER bit is cleared to "0", it will have no effect on the task. The task will run until it hits the STOP instruction. Reset type: SYSRSn 0h (R/W) = TASK_INT_DISABLE Task 6 interrupt is disabled (default) 1h (R/W) = TASK_INT_ENABLE Task 6 interrupt is enabled
4	INT5	R/W	0h	Setting any of the bits to "1" enables the corresponding interrupt from triggering a corresponding CLA task. Writing a "0" blocks the interrupt, but the interrupt can still be latched by the MIFR register. When an interrupt is enabled and the corresponding MIFR bit is set to "1", the CLA will start executing the corresponding task and automatically clear the corresponding MIFR bit. Interrupts are be serviced in normal priority order. Notes: [1] If a task is currently executing and the corresponding MIER bit is cleared to "0", it will have no effect on the task. The task will run until it hits the STOP instruction. Reset type: SYSRSn 0h (R/W) = TASK_INT_DISABLE Task 5 interrupt is disabled (default) 1h (R/W) = TASK_INT_ENABLE Task 5 interrupt is enabled
3	INT4	R/W	0h	Setting any of the bits to "1" enables the corresponding interrupt from triggering a corresponding CLA task. Writing a "0" blocks the interrupt, but the interrupt can still be latched by the MIFR register. When an interrupt is enabled and the corresponding MIFR bit is set to "1", the CLA will start executing the corresponding task and automatically clear the corresponding MIFR bit. Interrupts are be serviced in normal priority order. Notes: [1] If a task is currently executing and the corresponding MIER bit is cleared to "0", it will have no effect on the task. The task will run until it hits the STOP instruction. Reset type: SYSRSn 0h (R/W) = TASK_INT_DISABLE Task 4 interrupt is disabled (default) 1h (R/W) = TASK_INT_ENABLE Task 4 interrupt is enabled
2	INT3	R/W	0h	Setting any of the bits to "1" enables the corresponding interrupt from triggering a corresponding CLA task. Writing a "0" blocks the interrupt, but the interrupt can still be latched by the MIFR register. When an interrupt is enabled and the corresponding MIFR bit is set to "1", the CLA will start executing the corresponding task and automatically clear the corresponding MIFR bit. Interrupts are be serviced in normal priority order. Notes: [1] If a task is currently executing and the corresponding MIER bit is cleared to "0", it will have no effect on the task. The task will run until it hits the STOP instruction. Reset type: SYSRSn 0h (R/W) = TASK_INT_DISABLE Task 3 interrupt is disabled (default) 1h (R/W) = TASK_INT_ENABLE Task 3 interrupt is enabled

Table 7-56. MIER Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	INT2	R/W	0h	Setting any of the bits to "1" enables the corresponding interrupt from triggering a corresponding CLA task. Writing a "0" blocks the interrupt, but the interrupt can still be latched by the MIFR register. When an interrupt is enabled and the corresponding MIFR bit is set to "1", the CLA will start executing the corresponding task and automatically clear the corresponding MIFR bit. Interrupts are be serviced in normal priority order. Notes: [1] If a task is currently executing and the corresponding MIER bit is cleared to "0", it will have no effect on the task. The task will run until it hits the STOP instruction. Reset type: SYSRSn 0h (R/W) = TASK_INT_DISABLE Task 2 interrupt is disabled (default) 1h (R/W) = TASK_INT_ENABLE Task 2 interrupt is enabled
0	INT1	R/W	0h	Setting any of the bits to "1" enables the corresponding interrupt from triggering a corresponding CLA task. Writing a "0" blocks the interrupt, but the interrupt can still be latched by the MIFR register. When an interrupt is enabled and the corresponding MIFR bit is set to "1", the CLA will start executing the corresponding task and automatically clear the corresponding MIFR bit. Interrupts are be serviced in normal priority order. Notes: [1] If a task is currently executing and the corresponding MIER bit is cleared to "0", it will have no effect on the task. The task will run until it hits the STOP instruction. Reset type: SYSRSn 0h (R/W) = TASK_INT_DISABLE Task 1 interrupt is disabled (default) 1h (R/W) = TASK_INT_ENABLE Task 1 interrupt is enabled

7.8.4.21 MIRUN Register (Offset = 26h) [Reset = 0h]

MIRUN is shown in Figure 7-30 and described in Table 7-57.

Return to the Summary Table.

The interrupt run status register (MIRUN) indicates which task is currently executing. Only one MIRUN bit will ever be set to a 1 at any given time. The bit is automatically cleared when the task competes and the respective interrupt is fed to the peripheral interrupt expansion (PIE) block of the device. This lets the main CPU know when a task has completed. The main CPU can stop a currently running task by writing to the MCTL[SOFTRESET] bit. This will clear the MIRUN flag and stop the task. In this case no interrupt will be generated to the PIE.

Figure 7-30. MIRUN Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
INT8	INT7	INT6	INT5	INT4	INT3	INT2	INT1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 7-57. MIRUN Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	INT8	R	0h	These bits indicate which task is currently active. Only one bit can be set to "1" at any one time. The bit is automatically cleared to "0" when the task completes and the respective CLAINTxn line is toggled to indicate task completion. The CLAINTxn interrupt line can be fed to the PIE of the CPU so the CPU knows when a task has completed. A currently running task can be stopped by a SOFTRESET. The RUN flag is cleared, the task is stopped, but no CLAINTxn interrupt is generated. Reset type: SYSRSn 0h (R/W) = Task 8 is not executing (default) 1h (R/W) = Task 8 is executing
6	INT7	R	0h	These bits indicate which task is currently active. Only one bit can be set to "1" at any one time. The bit is automatically cleared to "0" when the task completes and the respective CLAINTxn line is toggled to indicate task completion. The CLAINTxn interrupt line can be fed to the PIE of the CPU so the CPU knows when a task has completed. A currently running task can be stopped by a SOFTRESET. The RUN flag is cleared, the task is stopped, but no CLAINTxn interrupt is generated. Reset type: SYSRSn 0h (R/W) = Task 7 is not executing (default) 1h (R/W) = Task 7 is executing
5	INT6	R	0h	These bits indicate which task is currently active. Only one bit can be set to "1" at any one time. The bit is automatically cleared to "0" when the task completes and the respective CLAINTxn line is toggled to indicate task completion. The CLAINTxn interrupt line can be fed to the PIE of the CPU so the CPU knows when a task has completed. A currently running task can be stopped by a SOFTRESET. The RUN flag is cleared, the task is stopped, but no CLAINTxn interrupt is generated. Reset type: SYSRSn 0h (R/W) = Task 6 is not executing (default) 1h (R/W) = Task 6 is executing

Table 7-57. MIRUN Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	INT5	R	0h	These bits indicate which task is currently active. Only one bit can be set to "1" at any one time. The bit is automatically cleared to "0" when the task completes and the respective CLAINTxn line is toggled to indicate task completion. The CLAINTxn interrupt line can be fed to the PIE of the CPU so the CPU knows when a task has completed. A currently running task can be stopped by a SOFTRESET. The RUN flag is cleared, the task is stopped, but no CLAINTxn interrupt is generated. Reset type: SYSRSn 0h (R/W) = Task 5 is not executing (default) 1h (R/W) = Task 5 is executing
3	INT4	R	0h	These bits indicate which task is currently active. Only one bit can be set to "1" at any one time. The bit is automatically cleared to "0" when the task completes and the respective CLAINTxn line is toggled to indicate task completion. The CLAINTxn interrupt line can be fed to the PIE of the CPU so the CPU knows when a task has completed. A currently running task can be stopped by a SOFTRESET. The RUN flag is cleared, the task is stopped, but no CLAINTxn interrupt is generated. Reset type: SYSRSn 0h (R/W) = Task 4 is not executing (default) 1h (R/W) = Task 4 is executing
2	INT3	R	0h	These bits indicate which task is currently active. Only one bit can be set to "1" at any one time. The bit is automatically cleared to "0" when the task completes and the respective CLAINTxn line is toggled to indicate task completion. The CLAINTxn interrupt line can be fed to the PIE of the CPU so the CPU knows when a task has completed. A currently running task can be stopped by a SOFTRESET. The RUN flag is cleared, the task is stopped, but no CLAINTxn interrupt is generated. Reset type: SYSRSn 0h (R/W) = Task 3 is not executing (default) 1h (R/W) = Task 3 is executing
1	INT2	R	0h	These bits indicate which task is currently active. Only one bit can be set to "1" at any one time. The bit is automatically cleared to "0" when the task completes and the respective CLAINTxn line is toggled to indicate task completion. The CLAINTxn interrupt line can be fed to the PIE of the CPU so the CPU knows when a task has completed. A currently running task can be stopped by a SOFTRESET. The RUN flag is cleared, the task is stopped, but no CLAINTxn interrupt is generated. Reset type: SYSRSn 0h (R/W) = Task 2 is not executing (default) 1h (R/W) = Task 2 is executing
0	INT1	R	0h	These bits indicate which task is currently active. Only one bit can be set to "1" at any one time. The bit is automatically cleared to "0" when the task completes and the respective CLAINTxn line is toggled to indicate task completion. The CLAINTxn interrupt line can be fed to the PIE of the CPU so the CPU knows when a task has completed. A currently running task can be stopped by a SOFTRESET. The RUN flag is cleared, the task is stopped, but no CLAINTxn interrupt is generated. Reset type: SYSRSn 0h (R/W) = Task 1 is not executing (default) 1h (R/W) = Task 1 is executing

7.8.4.22 _MPC Register (Offset = 28h) [Reset = 0h]

_MPC is shown in Figure 7-31 and described in Table 7-58.

Return to the Summary Table.

CLA Program Counter

Figure 7-31. _MPC Register

15	14	13	12	11	10	9	8
_MPC							
R-0h							
7	6	5	4	3	2	1	0
_MPC							
R-0h							

Table 7-58. _MPC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	_MPC	R	0h	Program Counter: The PC value is initialized by the appropriate MVECTx register when an interrupt (task) is serviced. The MPC register address 16-bits and not 32-bits. Hence the address range of the CLA with a 16-bit MPC is 64Kx16 words or 32K CLA instructions. Notes: [1] To be consistent with C28 core implementation, the PC value points to the instruction in D2 stage of pipeline. [2] After a STOP operation, and with no other task pending, the PC will remain pointing to the STOP operation. Reset type: SYSRSn

7.8.4.23 _MAR0 Register (Offset = 2Ah) [Reset = 0h]

_MAR0 is shown in Figure 7-32 and described in Table 7-59.

Return to the Summary Table.

CLA Auxiliary Register 0

Figure 7-32. _MAR0 Register

15	14	13	12	11	10	9	8
_MAR0							
R-0h							
7	6	5	4	3	2	1	0
_MAR0							
R-0h							

Table 7-59. _MAR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	_MAR0	R	0h	CLA Auxillary Register 0 Reset type: SYSRSn

7.8.4.24 _MAR1 Register (Offset = 2Bh) [Reset = 0h]

_MAR1 is shown in Figure 7-33 and described in Table 7-60.

Return to the Summary Table.

CLA Auxiliary Register 1

Figure 7-33. _MAR1 Register

15	14	13	12	11	10	9	8
_MAR1							
R-0h							
7	6	5	4	3	2	1	0
_MAR1							
R-0h							

Table 7-60. _MAR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	_MAR1	R	0h	CLA Auxillary Register 1 Reset type: SYSRSn

7.8.4.25 _MSTF Register (Offset = 2Eh) [Reset = 0h]

_MSTF is shown in Figure 7-34 and described in Table 7-61.

Return to the Summary Table.

The CLA status register (MSTF) reflects the results of different operations. These are the basic rules for the flags:
- Zero and negative flags are cleared or set based on:
- floating-point moves to registers
- the result of compare, minimum, maximum, negative and absolute value operations
- the integer result of operations such as MMOV16, MAND32, MOR32, MXOR32, MCMP32, MASR32, MLSR32
- Overflow and underflow flags are set by floating-point math instructions such as multiply, add, subtract and 1/x. These flags may also be connected to the peripheral interrupt expansion (PIE) block on your device. This can be useful for debugging underflow and overflow conditions within an application.

Figure 7-34. _MSTF Register

31	30	29	28	27	26	25	24
RESERVED				_RPC			
R-0h				R-0h			
23	22	21	20	19	18	17	16
_RPC							
R-0h							
15	14	13	12	11	10	9	8
_RPC				MEALLOW	RESERVED	RNDF32	RESERVED
R-0h				R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	TF	RESERVED		ZF	NF	LUF	LVF
R-0h	R-0h	R-0h		R-0h	R-0h	R-0h	R-0h

Table 7-61. _MSTF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-12	_RPC	R	0h	Return program counter The _RPC is used to save and restore the MPC address by the MCCNDD and MRCNDD operations Reset type: SYSRSn
11	MEALLOW	R	0h	MEALLOW Status This bit enables and disables CLA write access to EALLOW protected registers This is independent of the state of the EALLOW bit in the main CPU status register This status bit can be saved and restored by the MMOV32 STF, mem32 instruction Reset type: SYSRSn 0h (R/W) = The CLA cannot write to EALLOW protected registers. This bit is cleared by the CLA instruction, MEDIS. 1h (R/W) = The CLA is allowed to write to EALLOW protected registers. This bit is set by the CLA instruction, MEALLOW.
10	RESERVED	R	0h	Reserved
9	RNDF32	R	0h	Round 32-bit Floating-Point Mode Use the MSETFLG and MMOV32 MSTF, mem32 instructions to change the rounding mode Reset type: SYSRSn 0h (R/W) = If this bit is zero, the MMPYF32, MADDF32 and MSUBF32 instructions will round to zero (truncate). 1h (R/W) = If this bit is one, the MMPYF32, MADDF32 and MSUBF32 instructions will round to the nearest even value.

Table 7-61. _MSTF Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8-7	RESERVED	R	0h	Reserved
6	TF	R	0h	Test Flag The MTESTTF instruction can modify this flag based on the condition tested The MSETFLG and MMOV32 MSTF, mem32 instructions can also be used to modify this flag Reset type: SYSRSn 0h (R/W) = The condition tested with the MTESTTF instruction is false. 1h (R/W) = The condition tested with the MTESTTF instruction is true.
5-4	RESERVED	R	0h	Reserved
3	ZF	R	0h	Zero Flag - Instructions that modify this flag based on the floating-point value stored in the destination register: MMOV32, MMOVD32, MABSF32, MNEGF32 - Instructions that modify this flag based on the floating-point result of the operation: MCMPF32, MMAXF32, and MMINF32 - Instructions that modify this flag based on the integer result of the operation: MMOV16, MAND32, MOR32, MXOR32, MCMP32, MASR32, MLSR32 and MLSL32 The MSETFLG and MMOV32 MSTF, mem32 instructions can also be used to modify this flag Reset type: SYSRSn 0h (R/W) = The value is not zero 1h (R/W) = The value is zero
2	NF	R	0h	Negative Flag - Instructions that modify this flag based on the floating-point value stored in the destination register: MMOV32, MMOVD32, MABSF32, MNEGF32 - Instructions that modify this flag based on the floating-point result of the operation: MCMPF32, MMAXF32, and MMINF32 - Instructions that modify this flag based on the integer result of the operation: MMOV16, MAND32, MOR32, MXOR32, MCMP32, MASR32, MLSR32 and MLSL32 The MSETFLG and MMOV32 MSTF, mem32 instructions can also be used to modify this flag Reset type: SYSRSn 0h (R/W) = The value is not negative 1h (R/W) = The value is negative
1	LUF	R	0h	Latched Underflow Flag The following instructions will set this flag to 1 if an underflow occurs: MMPYF32, MADDF32, MSUBF32, MMACF32, MEINVF32, MEISQRTF32 The MSETFLG and MMOV32 MSTF, mem32 instructions can also be used to modify this flag Reset type: SYSRSn 0h (R/W) = An underflow condition has not been latched 1h (R/W) = An underflow condition has been latched

Table 7-61. _MSTF Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	LVF	R	0h	Latched Overflow Flag The following instructions will set this flag to 1 if an overflow occurs: MMPYF32, MADDF32, MSUBF32, MMACF32, MEINVF32, MEISQRTF32 The MSETFLG and MMOV32 MSTF, mem32 instructions can also be used to modify this flag Reset type: SYSRSn 0h (R/W) = An overflow condition has not been latched 1h (R/W) = An overflow condition has been latched

7.8.4.26 _MR0 Register (Offset = 30h) [Reset = 0h]

_MR0 is shown in Figure 7-35 and described in Table 7-62.

Return to the Summary Table.

CLA Floating-Point Result Register 0

Figure 7-35. _MR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
i32																															
R-0h																															

Table 7-62. _MR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	i32	R	0h	CLA Result Register 0 Reset type: SYSRSn

7.8.4.27 _MR1 Register (Offset = 34h) [Reset = 0h]

_MR1 is shown in Figure 7-36 and described in Table 7-63.

Return to the Summary Table.

CLA Floating-Point Result Register 1

Figure 7-36. _MR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
i32																															
R-0h																															

Table 7-63. _MR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	i32	R	0h	CLA Result Register 1 Reset type: SYSRSn

7.8.4.28 _MR2 Register (Offset = 38h) [Reset = 0h]

_MR2 is shown in Figure 7-37 and described in Table 7-64.

Return to the Summary Table.

CLA Floating-Point Result Register 2

Figure 7-37. _MR2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
i32																															
R-0h																															

Table 7-64. _MR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	i32	R	0h	CLA Result Register 2 Reset type: SYSRSn

7.8.4.29 _MR3 Register (Offset = 3Ch) [Reset = 0h]

_MR3 is shown in Figure 7-38 and described in Table 7-65.

Return to the Summary Table.

CLA Floating-Point Result Register 3

Figure 7-38. _MR3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
i32																															
R-0h																															

Table 7-65. _MR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	i32	R	0h	CLA Result Register 3 Reset type: SYSRSn

7.8.4.30 _MPSACTL Register (Offset = 42h) [Reset = 0h]

_MPSACTL is shown in Figure 7-39 and described in Table 7-66.

Return to the Summary Table.

PSA Control Register

Figure 7-39. _MPSACTL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
MPSA2CFG		MPSA2CLEAR	MPSA1CLEAR	MDWDBCYC	MDWDBSTART	MPABCYC	MPABSTART
R/W-0h		R-0/W1S-0h	R-0/W1S-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 7-66. _MPSACTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-6	MPSA2CFG	R/W	0h	CLA PSA2 Polynomial Configuration Bits: These bits configure the type of polynomial used for PSA2. The polynomials chosen are commonly used in the industry: Mode Polynomial Type 0,0 PSA 0,1 CRC32 1,0 CRC16 1,1 CRC16-CCITT Note: [1] Polynomial configuration should be performed when PSA2 is stopped. Reset type: SYSRSn
5	MPSA2CLEAR	R-0/W1S	0h	CLA PSA2 Clear Bit: Writing of "1" will clear contents of PSA2 register. Writes of "0" are ignored. Always reads back a "0" Note: Clearing operation should be performed when PSA2 is stopped. Reset type: SYSRSn
4	MPSA1CLEAR	R-0/W1S	0h	CLA PSA1 Clear Bit: Writing of "1" will clear contents of PSA1 register. Writes of "0" are ignored. Always reads back a "0" Note: Clearing operation should be performed when PSA1 is stopped. Reset type: SYSRSn
3	MDWDBCYC	R/W	0h	CLA Data Write Data Bus PSA2 Cycle or Event Based Bit: 0 PSA2 calculated on every cycle 1 PSA2 calculated on every bus event Reset type: SYSRSn
2	MDWDBSTART	R/W	0h	CLA Data Write Data Bus PSA2 Start/Stop Bit: 0 PSA2 stopped 1 PSA2 start Reset type: SYSRSn
1	MPABCYC	R/W	0h	CLA Program Address Bus PSA1 Cycle/Event Based Bit: 0 PSA1 calculated on every cycle 1 PSA1 calculated on every bus event Reset type: SYSRSn
0	MPABSTART	R/W	0h	CLA Program Address Bus PSA1 Start/Stop Bit: 0 PSA1 stopped 1 PSA1 start Reset type: SYSRSn

7.8.4.31 _MPSA1 Register (Offset = 44h) [Reset = 0h]

_MPSA1 is shown in Figure 7-40 and described in Table 7-67.

Return to the Summary Table.

PSA1 Register

Figure 7-40. _MPSA1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
i32																															
R/W-0h																															

Table 7-67. _MPSA1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	i32	R/W	0h	PSA1 Value: Reading this register gives the current PSA1 value. The value can be read at any time. Writes to this register are allowed to initialize the PSA1 to a known value. Writes to this register should only be made when PSA1 is stopped. Register value is cleared to zero by reset or by writing to the MPSA1CLEAR bit in the MPSACTL register. Reset type: SYSRSn

7.8.4.32 _MPSA2 Register (Offset = 46h) [Reset = 0h]

_MPSA2 is shown in Figure 7-41 and described in Table 7-68.

Return to the Summary Table.

PSA2 Register

Figure 7-41. _MPSA2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
i32																															
R/W-0h																															

Table 7-68. _MPSA2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	i32	R/W	0h	PSA2 Value: Reading this register gives the current PSA2 value. The value can be read at any time. Writes to this register are allowed to initialize the PSA2 to a known value. Writes to this register should only be made when PSA2 is stopped. Register value is cleared to zero by reset or by writing to the MPSA2CLEAR bit in the MPSACTL register. Reset type: SYSRSn

7.8.5 CLA Registers to Driverlib Functions

Table 7-69. CLA Registers to Driverlib Functions

File	Driverlib Function
MVECT1	
cla.h	CLA_mapTaskVector
MVECT2	
-	See MVECT1
MVECT3	
-	See MVECT1
MVECT4	
-	See MVECT1
MVECT5	
-	See MVECT1
MVECT6	
-	See MVECT1
MVECT7	
-	See MVECT1
MVECT8	
-	See MVECT1
MCTL	
cla.h	CLA_performHardReset
cla.h	CLA_performSoftReset
cla.h	CLA_enableIACK
cla.h	CLA_disableIACK
cla.h	CLA_enableBackgroundTask
cla.h	CLA_disableBackgroundTask
cla.h	CLA_startBackgroundTask
cla.h	CLA_enableHardwareTrigger

Table 7-69. CLA Registers to Driverlib Functions (continued)

File	Driverlib Function
cla.h	CLA_disableHardwareTrigger
MVECTBGRNDACTIVE	
cla.h	CLA_getBackgroundActiveVector
SOFTINTEN	
cla.h	CLA_enableSoftwareInterrupt
cla.h	CLA_disableSoftwareInterrupt
MSTSBGRND	
cla.h	CLA_getBackgroundTaskStatus
MCTLBGRND	
cla.h	CLA_enableBackgroundTask
cla.h	CLA_disableBackgroundTask
cla.h	CLA_startBackgroundTask
cla.h	CLA_enableHardwareTrigger
cla.h	CLA_disableHardwareTrigger
MVECTBGRND	
cla.h	CLA_getBackgroundActiveVector
cla.h	CLA_mapBackgroundTaskVector
MIFR	
cla.h	CLA_getPendingTaskFlag
cla.h	CLA_getAllPendingTaskFlags
cla.h	CLA_forceTasks
MIOVF	
cla.h	CLA_getTaskOverflowFlag
cla.h	CLA_getAllTaskOverflowFlags
MIFRC	
cla.h	CLA_forceTasks
MICLR	
cla.h	CLA_clearTaskFlags
MICLROVF	
-	
MIER	
cla.h	CLA_enableTasks
cla.h	CLA_disableTasks
MIRUN	
cla.h	CLA_getTaskRunStatus
cla.h	CLA_getAllTaskRunStatus
MPC	
-	
MAR0	
-	
MAR1	
-	
MSTF	
-	
MR0	

Table 7-69. CLA Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
MR1	
-	
MR2	
-	
MR3	
-	
MPSACTL	
-	
MPSA1	
-	
MPSA2	
-	
MVECTBGRNDACTIVE	
cla.h	CLA_getBackgroundActiveVector
MPSACTL	
-	
MPSA1	
-	
MPSA2	
-	
SOFTINTEN	
cla.h	CLA_enableSoftwareInterrupt
cla.h	CLA_disableSoftwareInterrupt
SOFTINTFRC	
cla.h	CLA_forceSoftwareInterrupt

Chapter 8

Dual-Clock Comparator (DCC)

This chapter describes the Dual-Clock Comparator (DCC) module.

8.1 Introduction

The dual-clock comparator module is used for evaluating and monitoring the clock input based on a second clock, which can be a more accurate and reliable version. This instrumentation is used to detect faults in clock source or clock structures, thereby enhancing the system's safety metrics.

8.1.1 Features

The main features of each of the DCC modules are:

- Allows the application to ensure that a fixed ratio is maintained between frequencies of two clock signals.
- Supports the definition of a programmable tolerance window in terms of the number of reference clock cycles.
- Supports continuous monitoring without requiring application intervention.
- Supports a single-sequence mode for spot measurements.
- Allows the selection of a clock source for each of the counters, resulting in several specific use cases.

8.1.2 Block Diagram

Figure 8-1 shows the main concept of the DCC module.

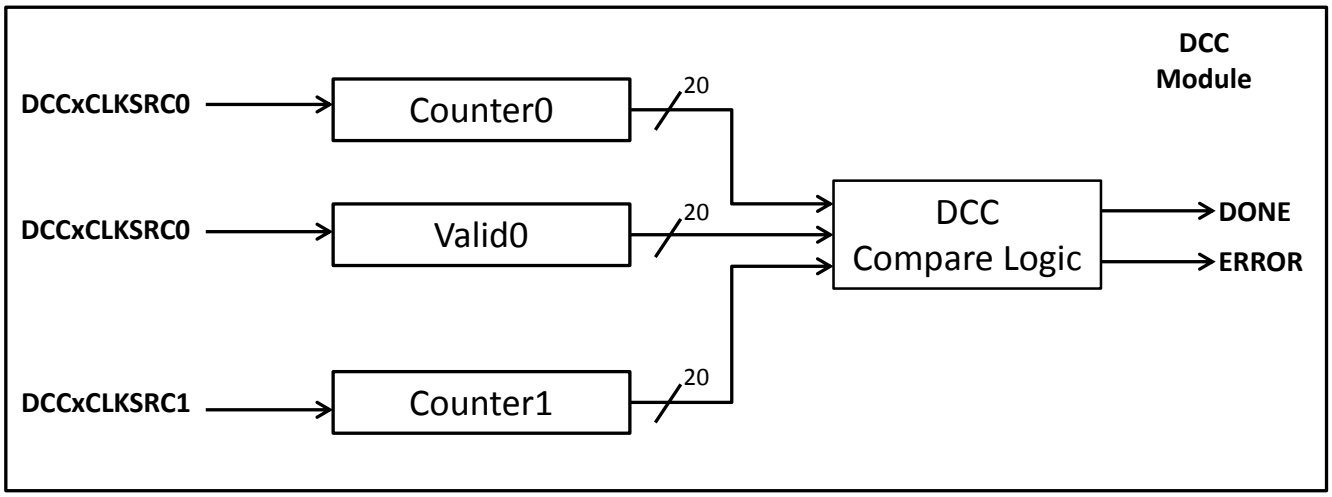


Figure 8-1. DCC Operation

8.2 Module Operation

As shown in Figure 8-1, DCC contains three counters – Counter0, Valid0 and Counter1. Initially, all counters are loaded with their user-defined, pre-load value. Counter0 and Counter1 start decrementing once the DCC is enabled at rates determined by the frequencies of Clock0 and Clock1, respectively. When Counter0 equals 0 (expires), the Valid0 counter decrements at a rate determined by Clock0. If Counter1 decrements to 0 in the valid window, then no error is generated and Clock1 is considered to be good within allowable tolerance as configured by the user.

8.2.1 Configuring DCC Counters

Counter0 and Counter1 are configured based on the ratio between the frequencies of Clock0 and Clock1 (Fclk1×Counter0 = Fclk0×Counter1). The Valid0 counter provides tolerance and is configured based on the error in DCC. Since Clock0 and Clock1 are asynchronous, the start and stop of the counters do not occur synchronously. Hence, while configuring the counters, two different sources of errors must be accounted for:

- DCC Errors due to the asynchronous timing of Clock0 and Clock1: this depends on the frequency of Clock0 and Clock1:
  - If Fclk1 > Fclk0, then Async. Error (in Clock0 cycles) = 2 + 2×(max(Fsysclk/Fclk0))
  - If Fclk1 < Fclk0, then Async. Error (in Clock0 cycles) = 2×(Fclk0/Fclk1) + 2×(max(Fsysclk/Fclk0))
  - If Fclk1 is unkown, then Async. Error (in Clock0 cycles) = 2 + 2×(Fsysclk/Fclk0)
- Digitization Error = 8 Clock0 cycles

DCC Error (in Clock0 Cycles) = Async. Error + Digitization Error

DCC error shows up as a frequency error for clock under measurement. This error is DCC induced and does not represent error in frequency of clock under measurement. The application needs to take this into consideration while configuring the counters and decide the desirable tolerance for DCC error that defines the window of measurement. To illustrate:

Window (in Clock0 Cycles) = (DCC Error) / (0.01×Tolerance)

For example, if DCC Error is 10 and the tolerance desired is +/-0.1%, then:

Window (in Clock0 Cycles) = 10/(0.01×0.1) = 10000

Based on above formula for Window, if the desired tolerance is low, then the counter values are large and increase the window of measurement. This means that counter values for a tolerance of 0.1% are larger than that of 0.2%. So, based on the application defined tolerance, you can define the window of measurement in terms of Clock0 cycles.

Clock under measurement may have allowed frequency error. If this error is expected, then it should also be accounted while configuring counters. For example, if measuring INTOSC1/2 frequency using external crystal as a reference clock, allowable tolerance of INTOSC1/2 (for example, +/-1%) should be accounted and factored in the counter configuration. The formula is:

Frequency Error Allowed (in Clock0 Cycles) = Window × (Allowable Frequency Tolerance (in %) / 100)

Total Error (in Clock0 Cycles) = DCC Error + Frequency Error Allowed

The following equations are used to configure counter values:

Counter0 (DCCCNTSEED0) = Window - Total Error

Valid0 (DCCVALIDSEED0) = 2 × Total Error

Counter1 (DCCCNTSEED1) = Window × (Fclk1/Fclk0)

Note

Counter1 is a 20-bit counter, so maximum possible value cannot exceed 1048575. If value does exceed, then you need to increase the desired Tolerance for DCC error, so that Window of measurement is lowered. The following formula should be used to compute minimum tolerance possible:

Tolerance (%) = (100 × DCC Error × (Fclk1/Fclk0)) / 1048575

8.2.2 Single-Shot Measurement Mode

The DCC module can be programmed to count down one time by enabling the single-shot mode. In this mode, the DCC stops operating when the down counter0 and the valid counter0 reach 0.

At the end of one sequence of counting down in this single-shot mode, the DCC gets disabled automatically, which prevents further counting. This mode is typically used for spot checking the frequency of a signal.

Example-1: Validating PLLRAWCLK frequency

A practical example of the usage is to validate the PLL output clock frequency using the XTAL as the reference clock. Assume XTAL is 10 MHz, PLL output frequency is 100 MHz, SYSCLK is 100 MHz, allowable Frequency Tolerance is 0.1%, and DCC Tolerance required is 0.1%. The measurement sequence proceeds as follows:

- Set Clock0 source for Counter0 and Valid0 as XTAL, and Clock1 source for Counter1 as PLL output clock.
- Based on the equations defined in Section 8.2.1, calculated seed values for Counters will be Counter0 = 29940; Valid0 = 120; Counter1 = 300000
- Once the DCC is enabled, the counters Counter0 and Counter1 both start counting down from their seed values.
- When Counter0 reaches zero, it automatically triggers the Valid0 counter.
- When Valid0 reaches zero and Counter1 is not zero, an ERROR status flag is set and a "DCC error" is sent to the PIE. Counter1 is frozen so that it stops counting down any further. The application can enable an interrupt to be generated from the PIE whenever this DCC error is indicated. Refer to Section 3.5.5 to know the channel mapping of DCC Interrupt.
- The application then needs to clear the ERROR status flag and restart the DCC module so that it is ready for the next spot measurement.

If there is no error generated at the end of the sequence, then the DONE status flag is set and a DONE interrupt is generated. The application must clear the DONE flag before restarting the DCC.

Error Conditions:

An error condition is generated by any one of the following:

1. Counter1 counts down to 0 before Counter0 reaches 0. This means that Clock1 is faster than expected, or Clock0 is slower than expected. It includes the case when Clock0 is stuck at 1 or 0.
2. Counter1 does not reach 0 even when Counter0 and Valid0 have both reached 0. This means that Clock1 is slower than expected. It includes the case when Clock1 is stuck at 1 or 0.

Any error freezes the counters from counting. An application may then read out the counter values to help determine what caused the error.

Example-2: Measuring AUXCLKIN frequency

Another example of single-shot mode is to measure the frequency of AUXCLKIN (unknown frequency) using INTOSC1 (10 MHz) as the reference clock and SYSCLK is 10 MHz. The measurement sequence proceeds as follows:

- Set Clock0 source for Counter0 and Valid0 as INTOSC1 (10 MHz), and Clock1 source for Counter1 as AUXCLKIN.
- Now configure counter values using equations in Section 8.2.1. For tolerance=+/-0.1%, Total Error=10 clock0 cycles; Window=10000 clock0 cycles; Counter0=9990; Valid0=20. Since Clock1 frequency (Fclk1) is unknown, Counter1 value should be set to the max value, that is, 1048575 (0xFFFFF).
- Once the DCC is enabled, the counters Counter0 and Counter1 both start counting down from their seed values.
- Since Counter1 is set to max value that is, 1048575 it will not expire when Counter0 and Valid0 has expired. This will generate an error which is expected and the application should ignore this error and use Counter1 values to compute the frequency of Clock1 (Fclk1).
- Knowing the frequency of Clock0 (INTOSC1) that is, Fclk0=10 MHz, and using Equation 1, frequency of AUXCLKIN that is, Fclk1, can be measured:

$$Fclk1 = \frac{Fclk0 \times (1048575 - Meas.Counter1)}{(Counter0 + Valid0)} = \frac{10 \times (1048575 - Meas.Counter1)}{(9990 + 20)} \quad (1)$$

8.2.3 Continuous Monitoring Mode

In this mode, the DCC is used by the application to ensure that two clock signals maintain the correct frequency ratio. Suppose the application wants to ensure that the PLL output signal always maintains a fixed frequency relationship with the XTAL.

- In this case, the application can use the XTAL as the Clock0 signal (for Counter0 and Valid0) and the PLL output as the Clock1 (for Counter1).
- The seed values of Counter0, Valid0 and Counter1 are selected based on the equations defined in Section 8.2.1 such that if the actual frequencies of Clock0 and Clock1 are equal to their expected frequencies, then the Counter1 will reach zero during the count down of the Valid0 counter.
- If the Counter1 reaches zero during the count down of the Valid0 counter, then all the counters (Counter0, Valid0, Counter1) are reloaded with their initial seed values.
- This sequence of counting down and checking then continues as long as there is no error, or until the DCC module is disabled.
- The counters must get reloaded if the application resets and restarts the DCC module.

Error Conditions:

An error condition is generated by any one of the following:

1. Counter1 counts down to 0 before Counter0 reaches 0. This means that Clock1 is faster than expected, or Clock0 is slower than expected. It includes the case when Clock0 is stuck at 1 or 0.
2. Counter1 does not reach 0 even when Counter0 and Valid0 have both reached 0. This means that Clock1 is slower than expected. It includes the case when Clock1 is stuck at 1 or 0.

Any error freezes the counters from counting. An application may then read out the counter values to help determine what caused the error.

8.2.4 Error Conditions

While operating in continuous mode, the counters get reloaded with the seed values and continue counting down under the following conditions:

- The module is reset or restarted by the application, OR
- Counter0, Valid 0 and Counter1 all reach 0 without any error.

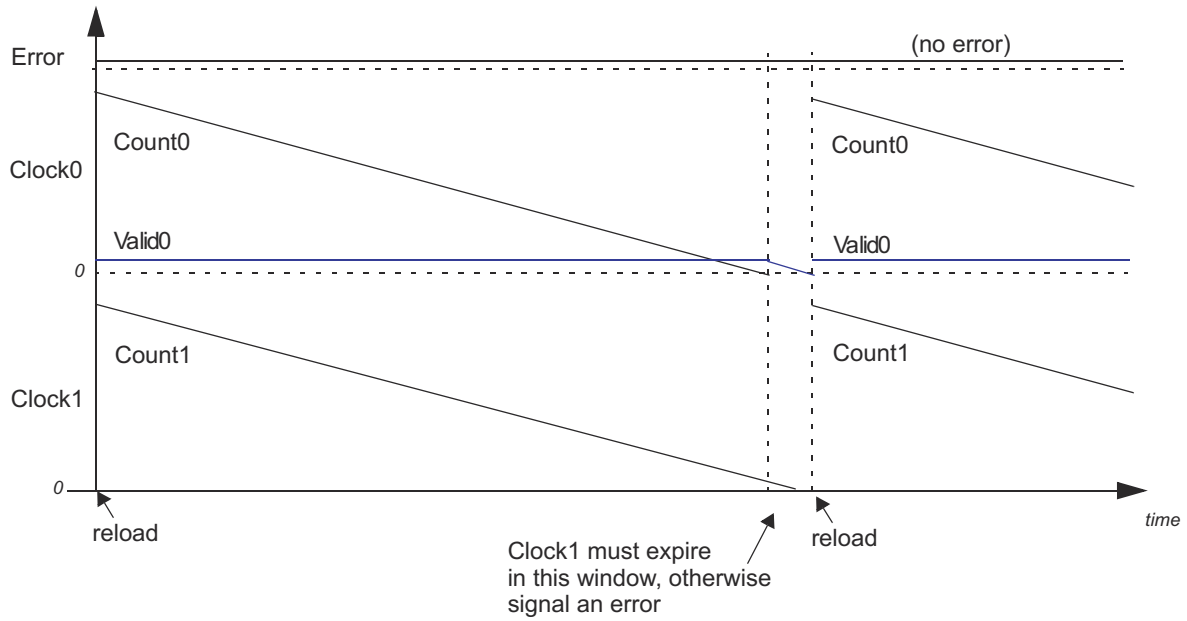


Figure 8-2. Counter Relationship

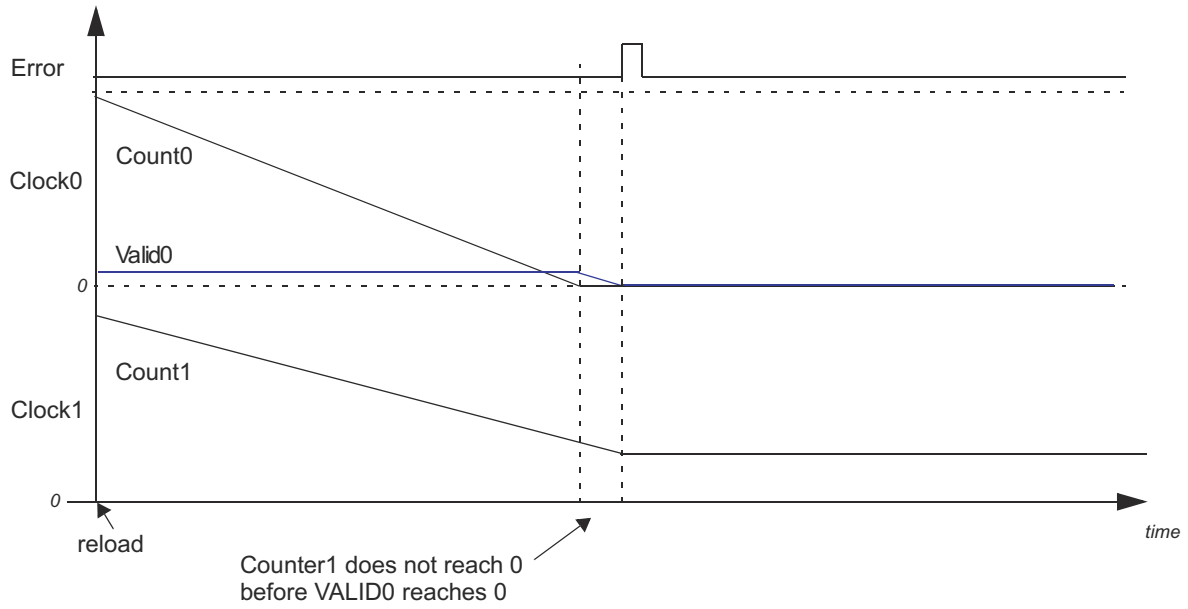


Figure 8-3. Clock1 Slower Than Clock0 - Results in an Error and Stops Counting

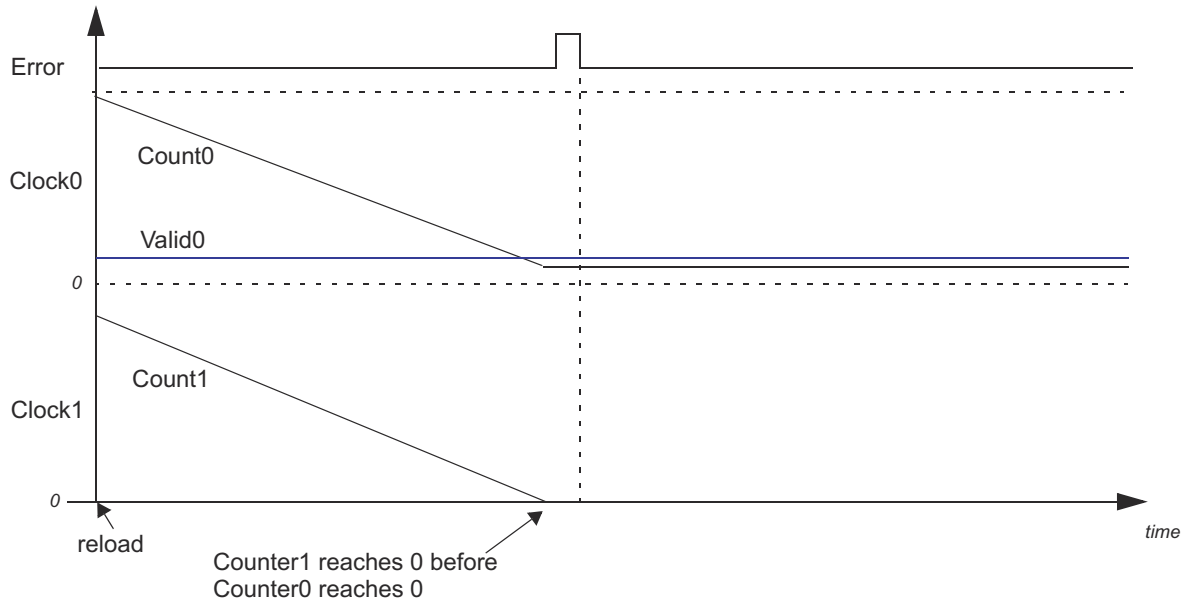


Figure 8-4. Clock1 Faster Than Clock0 - Results in an Error and Stops Counting

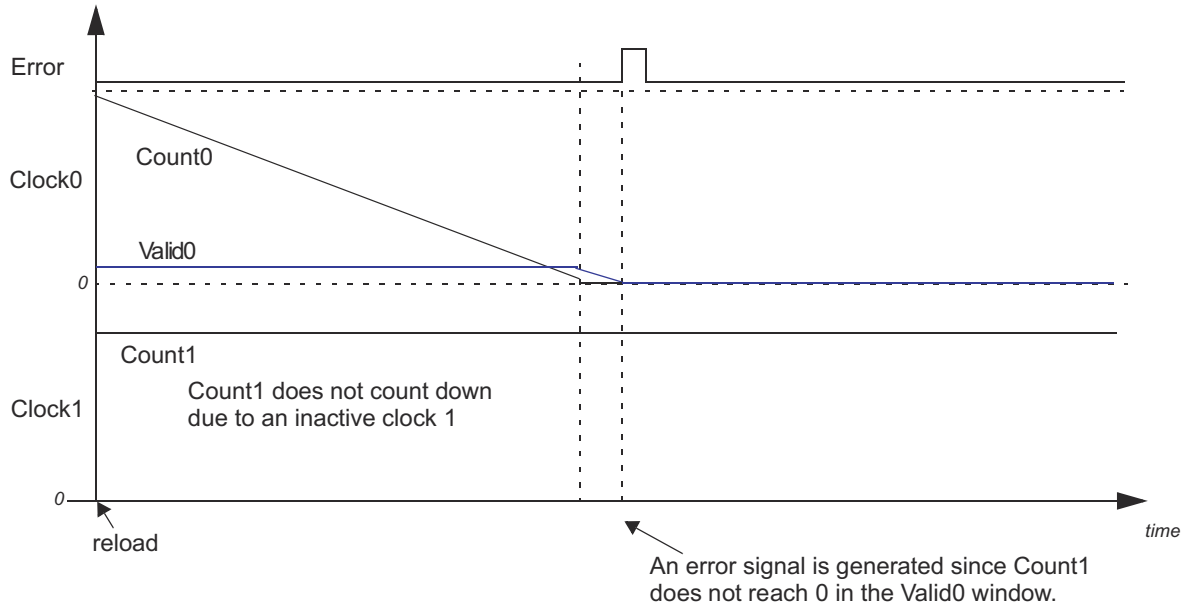


Figure 8-5. Clock1 Not Present - Results in an Error and Stops Counting

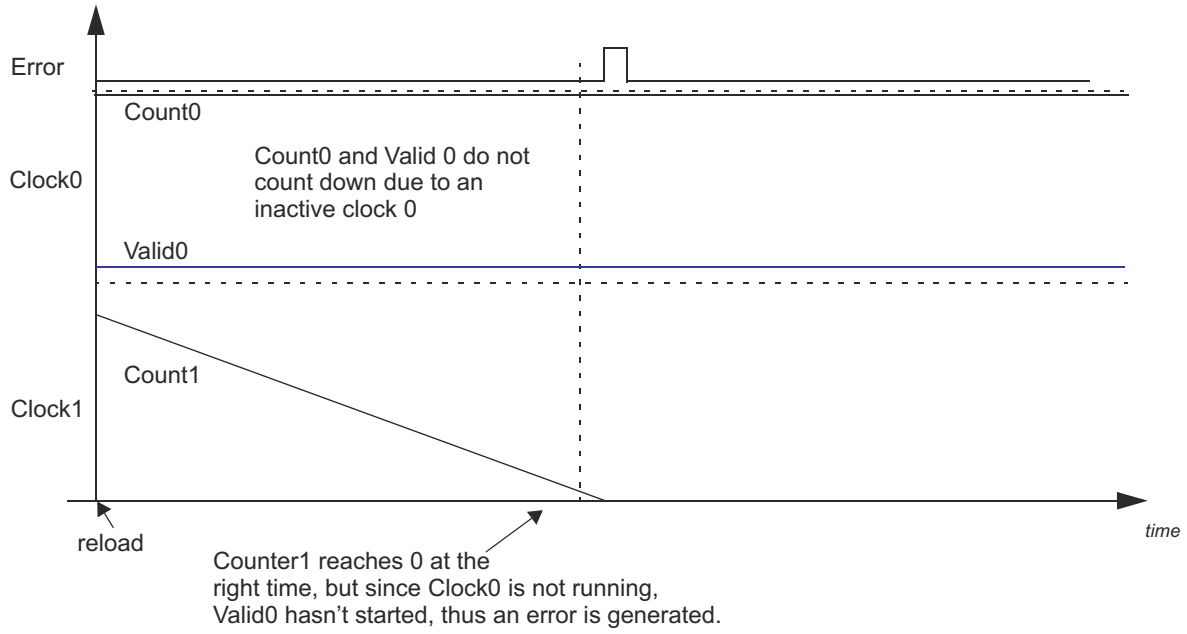


Figure 8-6. Clock0 Not Present - Results in an Error and Stops Counting

8.3 Interrupts

DCC generates an interrupt on either of two events:

- DCC finishes counting and all the counters expire within defined window indicating DONE operation, provided DCCGCTRL.DONENA=1.
- DCC finishes counting with error where counters do not expire in defined window. This indicates an ERROR event, and sets an interrupt provided DCCGCTRL.ERRENA=1.

Interrupt due to DONE or ERROR event is ORed and is flagged as DCCx Interrupt. Refer to determine the Interrupt channel mapping. Application ISR needs to check the status flag inside DCCSTATUS register to determine whether it is due to ERROR or DONE.

DCC Error interrupt can also configured as a Non Maskable Interrupt (NMI) by enabling CLKFAILCFG.DCCx_ERROR_EN flag.

8.4 Software

8.4.1 DCC Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/dcc

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

8.4.1.1 DCC Single Shot Clock Verification

FILE: dcc_ex1_single_shot_verification.c

This program uses the XTAL clock as a reference clock to verify the frequency of the PLLRAW clock.

The Dual-Clock Comparator Module 0 is used for the clock verification. The clocksource0 is the reference clock (Fclk0 = 20 MHz) and the clocksource1 is the clock that needs to be verified (Fclk1 = 120 MHz). Seed is the value that gets loaded into the Counter.

ExternalConnections

- None

Watch Variables

- *status/result* - Status of the PLLRAW clock verification

8.4.1.2 DCC Single Shot Clock Measurement

FILE: dcc_ex2_single_shot_measurement.c

This program demonstrates Single Shot measurement of the INTOSC2 clock post trim using XTAL as the reference clock.

The Dual-Clock Comparator Module 0 is used for the clock measurement. The clocksource0 is the reference clock (Fclk0 = 20 MHz) and the clocksource1 is the clock that needs to be measured (Fclk1 = 10 MHz). Since the frequency of the clock1 needs to be measured an initial seed is set to the max value of the counter.

External Connections

- None

Watch Variables

- *result* - Status if the INTOSC2 clock measurement completed successfully.
- *meas_freq1* - measured clock frequency, in this case for INTOSC2.

8.4.1.3 DCC Continuous Clock Monitoring

FILE: dcc_ex3_continuous_monitoring_of_clock.c

This program demonstrates continuous monitoring of PLL Clock in the system using INTOSC2 as the reference clock. This would trigger an interrupt on any error, causing the decrement/ reload of counters to stop.

The Dual-Clock Comparator Module 0 is used for the clock monitoring. The clocksource0 is the reference clock (Fclk0 = 10 MHz) and the clocksource1 is the clock that needs to be monitored (Fclk1 = 120 MHz). The clock0 and clock1 seed are set to achieve a window of 340 µs. Seed is the value that gets loaded into the Counter. For the sake of demo a slight variance is given to clock1 seed value to generate an error on continuous monitoring.

Note: When running in Flash configuration, it is good to do a reset and restart after loading the example to remove any stale flags/states.

External Connections

- None

Watch Variables

- *status/result* - Status of the PLLRAW clock monitoring
- *cnt0* - Counter0 Value measure when error is generated
- *cnt1* - Counter1 Value measure when error is generated
- *valid* - Valid0 Value measure when error is generated

8.4.1.4 DCC Detection of Clock Failure

FILE: dcc_ex4_clock_fail_detect.c

This program demonstrates clock failure detection on continuous monitoring of the PLL Clock in the system using XTAL as the osc clock source. Once the oscillator clock fails, it would trigger a DCC error interrupt, causing the decrement/ reload of counters to stop. In this examples, the clock failure is simulated by turning off the XTAL oscillator. Once the ISR is serviced, the osc source is changed to INTOSC1 and the PLL is turned off.

The Dual-Clock Comparator Module 0 is used for the clock monitoring. The clocksource0 is the reference clock (Fclk0 = 20 MHz) and the clocksource1 is the clock that needs to be monitored (Fclk1 = 120 MHz). Seed is the value that gets loaded into the Counter.

In the current example, the XTAL is expected to be a Resonator running in Crystal mode which is later switched off to simulate the clock failure. If an SE Crystal is used, you will need to physically disconnect the clock on the board.

Note: When running in Flash configuration, it is good to do a reset and restart after loading the example to remove any stale flags/states.

External Connections

- None

Watch Variables

- *status/result* - Status of the clock failure detection

8.5 DCC Registers

This section describes the Dual Clock Comparator Registers.

Note

DCC is used by Boot ROM, hence the register values could be different than HW reset value. User needs to make sure to configure the values of these registers to the desired value before enabling DCC.

8.5.1 DCC Base Address Table

Table 8-1. DCC Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
Dcc0Regs	DCC_REGS	DCC0_BASE	0x0005_E700	YES	-	-	-	YES
Dcc1Regs	DCC_REGS	DCC1_BASE	0x0005_E740	YES	-	-	-	YES

8.5.2 DCC_REGS Registers

Table 8-2 lists the memory-mapped registers for the DCC_REGS registers. All register offset addresses not listed in Table 8-2 should be considered as reserved locations and the register contents should not be modified.

Table 8-2. DCC_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	DCCGCTRL	Starts / stops the counters. Clears the error signal.		Go
8h	DCCCNTSEED0	Seed value for the counter attached to Clock Source 0.		Go
Ch	DCCVALIDSEED0	Seed value for the timeout counter attached to Clock Source 0.		Go
10h	DCCCNTSEED1	Seed value for the counter attached to Clock Source 1.		Go
14h	DCCSTATUS	Specifies the status of the DCC Module.		Go
18h	DCCCNT0	Value of the counter attached to Clock Source 0.		Go
1Ch	DCCVALID0	Value of the valid counter attached to Clock Source 0.		Go
20h	DCCCNT1	Value of the counter attached to Clock Source 1.		Go
24h	DCCCLKSRC1	Selects the clock source for Counter 1.		Go
28h	DCCCLKSRC0	Selects the clock source for Counter 0.		Go

Complex bit access types are encoded to fit into small table cells. Table 8-3 shows the codes that are used for access types in this section.

Table 8-3. DCC_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
R-1	R -1	Read Returns 1s
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

8.5.2.1 DCCGCTRL Register (Offset = 0h) [Reset = 5555h]

DCCGCTRL is shown in Figure 8-7 and described in Table 8-4.

Return to the Summary Table.

Starts / stops the counters. Clears the error signal.

Figure 8-7. DCCGCTRL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DONEENA				SINGLESHOT				ERRENA				DCCENA			
R/W-5h				R/W-5h				R/W-5h				R/W-5h			

Table 8-4. DCCGCTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-12	DONEENA	R/W	5h	DONE Enable Enables/disables the done interrupt signal, but has no effect on the done status flag in DCCSTAT register. 0101 The done signal is disabled Others The done signal is enabled Reset type: SYSRSn
11-8	SINGLESHOT	R/W	5h	Single-Shot Enable Enables/disables repetitive operation of the DCC. 1010: Stop counting when COUNTER0 and VALID0 both reach zero 1011: Reserved Others: Continuously repeat (until error) Reset type: SYSRSn
7-4	ERRENA	R/W	5h	Error Enable Enables/disables the error signal. 0101 The error signal is disabled Others The error signal is enabled Reset type: SYSRSn
3-0	DCCENA	R/W	5h	DCC Enable Starts and stops the operation of the DCC. 0101 Counters are stopped Others Counters are running Reset type: SYSRSn

8.5.2.2 DCCCNTSEED0 Register (Offset = 8h) [Reset = 0h]

DCCCNTSEED0 is shown in Figure 8-8 and described in Table 8-5.

Return to the Summary Table.

Seed value for the counter attached to Clock Source 0.

Figure 8-8. DCCCNTSEED0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												COUNTSEED0																			
R-0h												R/W-0h																			

Table 8-5. DCCCNTSEED0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19-0	COUNTSEED0	R/W	0h	Seed Value for Counter 0 Contains the seed value that gets loaded into Counter 0 (Clock Source 0). NOTE: Operating the DCC with '0' in the COUNTSEED0 register will result in undefined operation. Reset type: SYSRSn

8.5.2.3 DCCVALIDSEED0 Register (Offset = Ch) [Reset = 0h]

DCCVALIDSEED0 is shown in Figure 8-9 and described in Table 8-6.

Return to the Summary Table.

Seed value for the timeout counter attached to Clock Source 0.

Figure 8-9. DCCVALIDSEED0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																VALIDSEED															
R-0h																R/W-0h															

Table 8-6. DCCVALIDSEED0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	VALIDSEED	R/W	0h	Seed Value for Valid Duration Counter 0 Contains the seed value that gets loaded into the valid duration counter for Clock Source 0. NOTE: Operating the DCC with '0' in the VALIDSEED0 register will result in undefined operation. VALID0 defines a window in which COUNT1 expires. This window is meant to be at least four cycles wide. Do not program a value less than '4' into the VALID0 register. Reset type: SYSRSn

8.5.2.4 DCCCNTSEED1 Register (Offset = 10h) [Reset = 0h]

DCCCNTSEED1 is shown in Figure 8-10 and described in Table 8-7.

Return to the Summary Table.

Seed value for the counter attached to Clock Source 1.

Figure 8-10. DCCCNTSEED1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												COUNTSEED1																			
R-0h												R/W-0h																			

Table 8-7. DCCCNTSEED1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19-0	COUNTSEED1	R/W	0h	Seed Value for Counter 1 Contains the seed value that gets loaded into Counter 1 (Clock Source 1). NOTE: Operating the DCC with '0' in the COUNTSEED1 register will result in undefined operation. Reset type: SYSRSn

8.5.2.5 DCCSTATUS Register (Offset = 14h) [Reset = 0h]

DCCSTATUS is shown in Figure 8-11 and described in Table 8-8.

Return to the Summary Table.

Specifies the status of the DCC Module.

Figure 8-11. DCCSTATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						DONE	ERR
R-0h						R/W-0h	R/W-0h

Table 8-8. DCCSTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	DONE	R/W	0h	Single-Shot Done Flag Indicates when single-shot mode is complete without error. Writing a '1' to this bit clears the flag. 0 Single-shot mode has not completed. 1 Single-shot mode has completed. Reset type: SYSRSn
0	ERR	R/W	0h	Error Flag Indicates whether or not an error has occurred. Writing a '1' to this bit clears the flag. 0 No errors have occurred. 1 An error has occurred. Reset type: SYSRSn

8.5.2.6 DCCCNT0 Register (Offset = 18h) [Reset = 0h]

DCCCNT0 is shown in Figure 8-12 and described in Table 8-9.

Return to the Summary Table.

Value of the counter attached to Clock Source 0.

Figure 8-12. DCCCNT0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												COUNT0																			
R-0h												R-0h																			

Table 8-9. DCCCNT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19-0	COUNT0	R	0h	Current Value of Counter 0 Reset type: SYSRSn

8.5.2.7 DCCVALID0 Register (Offset = 1Ch) [Reset = 0h]

DCCVALID0 is shown in Figure 8-13 and described in Table 8-10.

Return to the Summary Table.

Value of the valid counter attached to Clock Source 0.

Figure 8-13. DCCVALID0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																VALID0															
R-0h																R-0h															

Table 8-10. DCCVALID0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	VALID0	R	0h	Current Value of Valid 0 Reset type: SYSRSn

8.5.2.8 DCCCNT1 Register (Offset = 20h) [Reset = 0h]

DCCCNT1 is shown in Figure 8-14 and described in Table 8-11.

Return to the Summary Table.

Value of the counter attached to Clock Source 1.

Figure 8-14. DCCCNT1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												COUNT1																			
R-0h												R-0h																			

Table 8-11. DCCCNT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19-0	COUNT1	R	0h	Current Value of Counter 1 Reset type: SYSRSn

8.5.2.9 DCCCLKSRC1 Register (Offset = 24h) [Reset = 0h]

DCCCLKSRC1 is shown in Figure 8-15 and described in Table 8-12.

Return to the Summary Table.

Selects the clock source for Counter 1.

Figure 8-15. DCCCLKSRC1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY				RESERVED							CLKSRC1				
R-0/W-0h				R-0h							R/W-0h				

Table 8-12. DCCCLKSRC1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-12	KEY	R-0/W	0h	Enables or Disables Clock Source Write for COUNT1 1010 The CLKSRC field selects the clock source for COUNT1. Others: Previous values retained new writes on register fields has no impact. Reset type: SYSRSn
11-5	RESERVED	R	0h	Reserved
4-0	CLKSRC1	R/W	0h	Clock Source Select for Counter 1 Specifies the clock source for COUNT1, when the KEY field enables this feature. 00000: PLLRAWCLK 00010: INTOSC1 00011: INTOSC2 00110: CPU1SYSCLK 01001: INPUTXBAR (Output15 of the input-xbar) 01010: AUXCLKIN 01011: EPWMCLK 01100: LSPCLK 01101: ADCCLK 01110: WDCLK 01111: CAN0BITCLK Note: All other values are "RSVD" Reset type: SYSRSn

8.5.2.10 DCCCLKSRC0 Register (Offset = 28h) [Reset = 0h]

DCCCLKSRC0 is shown in Figure 8-16 and described in Table 8-13.

Return to the Summary Table.

Selects the clock source for Counter 0.

Figure 8-16. DCCCLKSRC0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY				RESERVED								CLKSRC0			
R-0/W-0h				R-0h								R/W-0h			

Table 8-13. DCCCLKSRC0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-12	KEY	R-0/W	0h	Enables or Disables Clock Source Write for COUNT0 1010: The CLKSRC0 field written with key gets updated to with new selection to clock COUNT0. Others: Previous values retained new writes on register fields has no impact. Reset type: SYSRSn
11-4	RESERVED	R	0h	Reserved
3-0	CLKSRC0	R/W	0h	Clock Source Select for Counter 0 Specifies the clock source for COUNT0, when the KEY field enables this feature. 0000: XTAL/X1 0001: INTOSC1 0010: INTOSC2 0101: CPU1SYSCLK 1100: INPUTXBAR (Output16 of the input-xbar) Note: All other values are Reserved Reset type: SYSRSn

8.5.3 DCC Registers to Driverlib Functions

Table 8-14. DCC Registers to Driverlib Functions

File	Driverlib Function
DCCGCTRL	
dcc.h	DCC_enableModule
dcc.h	DCC_disableModule
dcc.h	DCC_enableErrorSignal
dcc.h	DCC_enableDoneSignal
dcc.h	DCC_disableErrorSignal
dcc.h	DCC_disableDoneSignal
dcc.h	DCC_enableSingleShotMode
dcc.h	DCC_disableSingleShotMode
DCCCNTSEED0	
dcc.h	DCC_setCounterSeeds
DCCVALIDSEED0	
dcc.h	DCC_setCounterSeeds
DCCCNTSEED1	

Table 8-14. DCC Registers to Driverlib Functions (continued)

File	Driverlib Function
dcc.h	DCC_setCounterSeeds
DCCSTATUS	
dcc.h	DCC_getErrorStatus
dcc.h	DCC_getSingleShotStatus
dcc.h	DCC_clearErrorFlag
dcc.h	DCC_clearDoneFlag
sysctl.c	SysCtl_isPLLValid
DCCCNT0	
dcc.h	DCC_getCounter0Value
DCCVALID0	
dcc.h	DCC_getValidCounter0Value
DCCCNT1	
dcc.h	DCC_getCounter1Value
DCCCLKSRC1	
dcc.h	DCC_setCounter1ClkSource
dcc.h	DCC_getCounter1ClkSource
DCCCLKSRC0	
dcc.h	DCC_setCounter0ClkSource
dcc.h	DCC_getCounter0ClkSource

This page intentionally left blank.

Chapter 9

Background CRC-32 (BGCRC)

The Background CRC (BGCRC) module that helps to identify memory faults and corruption, is discussed in this chapter.

9.1 Introduction

The Background CRC (BGCRC) module computes a CRC-32 on a configurable block of memory. It accomplishes this by fetching the specified block of memory during idle cycles (when the CPU, CLA, or DMA is not accessing the memory block). The calculated CRC-32 value is compared against a golden CRC-32 value to indicate a pass or fail. In essence, the BGCRC helps identify memory faults and corruption. There are two BGCRC modules, CPU_CRC and CLA_CRC. The two BGCRC modules differ only in the memories they test.

9.1.1 Features

The BGCRC module has the following features:

- One cycle CRC-32 computation on 32 bits of data
- No CPU bandwidth impact for zero wait state memory
- Minimal CPU bandwidth impact for non-zero wait state memory
- Dual operation modes (CRC-32 mode and scrub mode)
- Watchdog timer to time CRC-32 completion
- Ability to pause and resume CRC-32 computation

9.1.2 Block Diagram

Figure 9-1 shows the BGCRC block diagram.

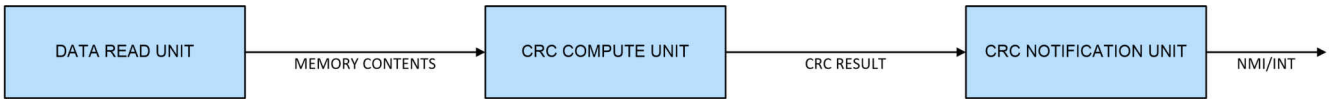


Figure 9-1. BGCRC Block Diagram

9.1.3 Memory Wait States and Memory Map

Figure 9-2 shows the memory map of the BGCRC module. M0, M1, MSGRAM, LS[x]RAM and GS[x]RAM are all zero wait-state memories. BGCRC will access these memories with minimal impact on normal program operation. For instance, if a BGCRC access is being made to a zero wait-state memory in the current cycle, the earliest the operating program can make access to the same memory location will be in the next cycle. Similarly for the non-zero wait state memories SECROM, DATAROM and BOOTROM etc, the worst case delay for functional access after a BGCRC access will be the wait state amount.

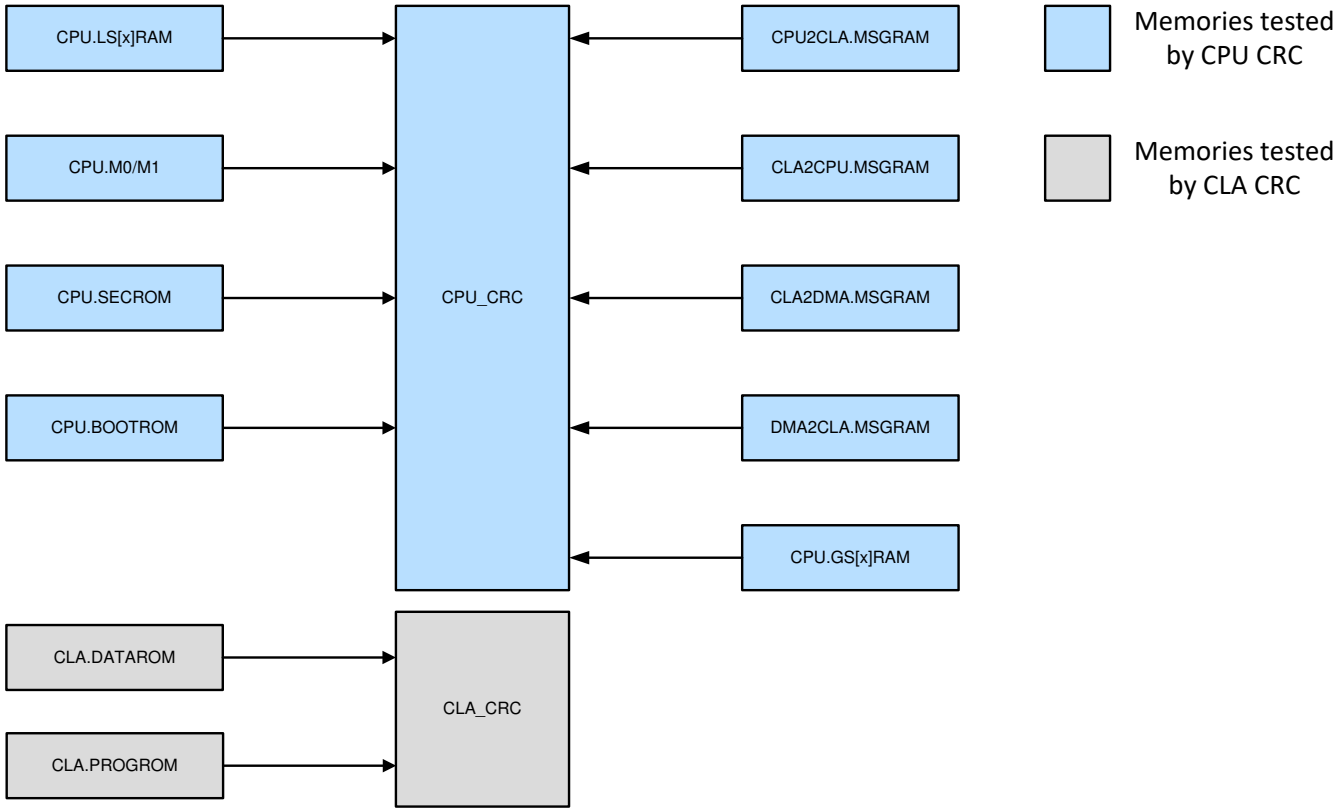


Figure 9-2. BGCRC Memory Map

9.2 Functional Description

The CRC-32 calculation of the BGCRC module can be kicked off by configuring the register BGCRC_EN.START to 1010. Once started, the module will perform the background memory checks without CPU or CLA intervention. However, on completion of the CRC-32 calculation or in the event of a failure, the CPU or CLA will be notified through an interrupt and task, respectively.

As highlighted in the overview, there are two BGCRC modules. CPU_CRC that is configurable by the CPU can only generate an interrupt. CLA_CRC however, can be configured by both the CPU and CLA and can generate an interrupt to the CPU or task for the CLA.

9.2.1 Data Read Unit

Once the CRC-32 calculation is started, the BGCRC module will continuously read data from memory as a background process. These reads happen during the idle times (when the CPU, CLA or DMA is not accessing the memory block) and so will not impact functional access. The data read unit will only read data if there is no pending functional access. The data read unit begins operation by reading a block of data BGCRC_CTRL2.BLOCK_SIZE from address BGCRC_START_ADDR. Note that BGCRC_START_ADDR must be 0x80 word aligned. For a non-0x80 word aligned BGCRC_START_ADDR, the LSB bits will be zeroed out to get a 0x80 word aligned BGCRC_START_ADDR. For instance, if the programmed BGCRC_START_ADDR = 0x1AF3, the internal 0x80 word aligned start address will be 0x1A80.

When the data read unit reads a block of data, ECC and parity will be checked. Any ECC or parity errors that occur during the read will be indicated by setting the respective NMI and generating an interrupt if configured as so. The BGCRC module however, will not write back the corrected memory contents on the occurrence of a correctable ECC error. Writing back the corrected values should be handled by software.

9.2.2 CRC-32 Compute Unit

After the data read unit reads a block of data, it feeds this data to the CRC-32 compute unit. This unit computes the CRC-32 using the standard polynomial 0x04C11DB7 (x32 + x26 + x23 + x22 + x16 + x12 + x11 + x10 + x8 + x7 + x5 + x4 + x2 + x + 1). The CRC-32 unit retrieves 32-bit of data at a time from the block of data to compute the polynomial. This computation takes 1 cycle. For instance the CRC-32 calculation for 1KB block of data will be 256 cycles, 1 cycle for each of the 256 32-bit chunks. The initial value for the CRC-32 computation can be configured using BGCRC_SEED.

After the CRC-32 calculation is complete for a data block, the final result is loaded into BGCRC_RESULT. Note that BGCRC_RESULT will only contain the final calculation for the whole data block; intermediate 32-bit calculations will not update BGCRC_RESULT. The value in BGCRC_RESULT will be compared against the value in BGCRC_GOLDEN by hardware and the NMI/Interrupt flags will be set accordingly by the CRC notification unit.

Once CRC-32 calculation is commenced, it can be halted by setting BGCRC_CTRL2.TEST_HALT to 1010. Clearing this bit resumes CRC-32 calculation from the halt point.

9.2.3 CRC Notification Unit

After the CRC-32 compute unit completes the CRC-32 calculation for a block of data or fails to complete the calculation within BGCRC_WD_MIN and BGCRC_WD_MAX, if configured, it will send out a NMI/Interrupt on the occurrence of a pass or fail. In addition, during a data read by the data read unit, if an ECC or parity error occurs, the CRC notification unit can send out a NMI/Interrupt. In the case of an ECC or parity error, the BGCRC will stop operation and BGCRC_CURR_ADDR will contain the memory address which caused the ECC or parity error. The contents of BGCRC_CURR_ADDR in addition to the NMI/Interrupt flags can be used to debug a BGCRC failure.

9.2.3.1 CPU Interrupt, CLA Task and NMI

The BGCRC module has configurable Interrupt and NMI lines. NMIs are enabled by default but can be disabled by writing 0xA to BGCRC_CTRL1.NMIDIS register. Conversely, all Interrupts are disabled by default but can be enabled by writing 0x7E to BGCRC_INTEN register. When an error occurs in the BGCRC, it can be configured to generate an NMI or Interrupt. Since NMIs are enabled by default, all BGCRC errors will cause an NMI. Figure 9-3 and Figure 9-4 show the NMI and Interrupt lines, respectively.

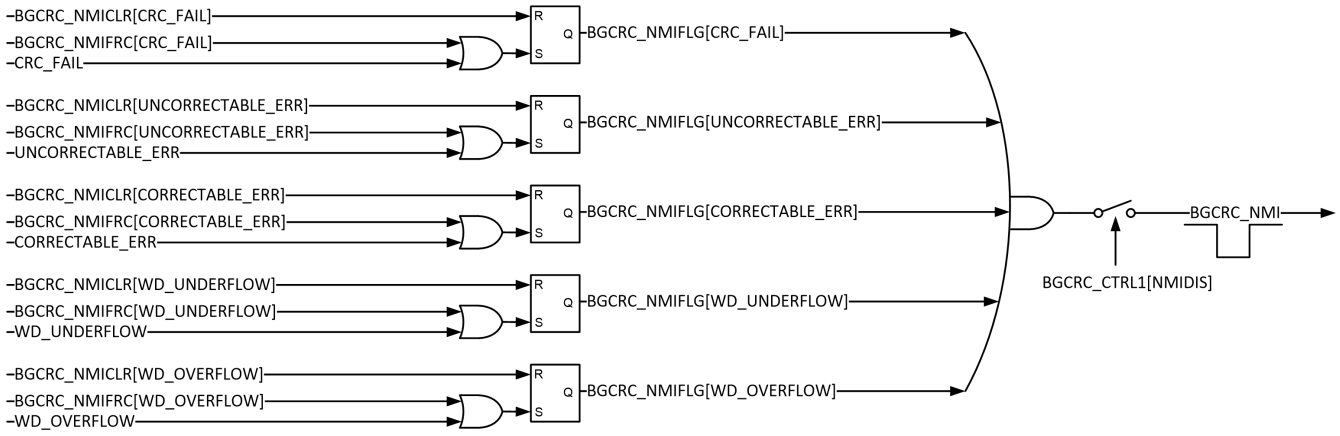


Figure 9-3. BGCRC NMI

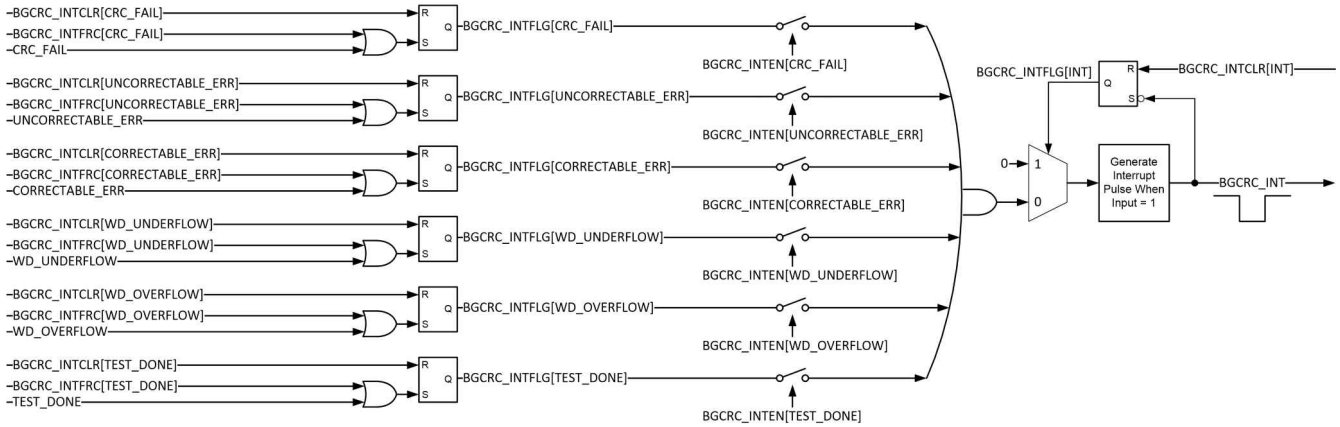


Figure 9-4. BGCRC Interrupt

9.2.4 Operating Modes

BGCRC module supports two modes of operation: CRC mode and scrub mode. The mode of operation can be configured by clearing or setting the register BGCRC_CTRL2.SCRUB_MODE.

9.2.4.1 CRC Mode

In CRC mode, the BGCRC module operates as explained in Section 9.2.2. CRC-32 calculation is performed on a block of data and the result compared against a golden value. ECC and parity errors are checked in this mode. Result mismatch, ECC or parity error can trigger an NMI/Interrupt.

9.2.4.2 Scrub Mode

In scrub mode, the CRC-32 result is not compared against the golden value and BGCRC_RESULT register is not updated. ECC and parity errors are also checked in this mode. However unlike CRC mode, NMI/Interrupt will only be from ECC or parity error. In scrub mode, Parity and ECC bits of the memory block need to be initialized by the CPU and/or CLA.

9.2.5 BGCRC Watchdog

The BGCRC module has an embedded windowed watchdog which is used as a diagnostic to check memory test completion within the expected time window. This can protect against hardware defects which can cause the memory check not to complete in the allotted time which may not be caught by the system watchdog as the error may be due to the CLA or DMA having continuous access. Windowing also helps detect additional failure modes in the watchdog operation, for example, stuck watchdog.

The BGCRC watchdog is enabled by default and starts when the BGCRC module begins reading from memory. The watchdog can be disabled using the register BGCRC_WD_CFG.WDDIS. The BGCRC watchdog counter is a 32-bit counter with its value reflected in BGCRC_WD_CNT register. The lower and upper window settings are configured using BGCRC_WD_MIN and BGCRC_WD_MAX respectively. BGCRC_WD_MIN and BGCRC_WD_MAX need to be configured before the test is started and shouldn't be changed while the BGCRC is operating. If configured, an NMI or Interrupt will be triggered if the memory test fails to complete within the configured time window. The counter stops on completion of the CRC-32 check done, CRC-32 check failure and ECC/Parity errors. The counter is reset when the next memory check begins.

The BGCRC watchdog can be halted by configuring the BGCRC_WD_CFG.WDDIS register. After the watchdog resumes from being halted, the counter starts counting from the previous count unless a new memory check operation is initiated. The counter is not halted when CRC-32 computation halts but by default will halt during a debug halt. The behavior of the watchdog during emulation can be changed by configuring the appropriate BGCRC registers. In addition, due to the changing nature of memory contents during emulation, it's not recommended to run BGCRC during emulation. CRC-32 computation will continue during a watchdog failure and software needs to address this condition.

9.2.6 Hardware and Software Faults Protection

The configuration registers are protected using a lock and commit configuration. Each of the configuration registers can be individually locked and committed. The register once locked, can no longer be updated until the lock is removed. However if the register lock is committed, no further writes is permitted until the device is reset. It is recommended to lock the registers after configuration to protect against corruption due to software faults. In addition, registers critical to the module functionality and fault detection are implemented using multi-bit fields to protect against hardware faults.

9.3 Application of the BGCRC

This section contains use case scenarios of how the BGCRC module can be configured to test a block of memory. However, the use case scenarios presented are for reference only and do not cover all the possible application scenarios. These use case scenarios could be used as a starting point to configure the BGCRC module for specific applications.

9.3.1 Software Configuration

The configuration registers for the BGCRC can be split into three groups (see Table 9-1):

- **CFG1:** Registers that determine the operating mode and configured at the beginning.
- **CFG2:** Registers that need to be updated during kickoff of a new test.
- **CFG3:** Registers used for test and error management.

CFG1 registers are expected to be locked and committed after initial configuration. It is recommended to lock the CFG2 and CFG3 registers after configuration. Figure 9-5 shows the BGCRC execution sequence.

Table 9-1. BGCRC Register Groups

CFG1 - One Time Configuration Registers	CFG2 - Periodic Configuration Registers	CFG3 - Registers Used for Test and Error Management
BGCRC_CTRL1	BGCRC_EN	BGCRC_NMICLR
BGCRC_WD_CFG	BGCRC_CTRL2	BGCRC_INTCLR
BGCRC_INTEN	BGCRC_START_ADDR	BGCRC_NMIFRC
BGCRC_SEED	BGCRC_GOLDEN	BGCRC_INTFRC
	BGCRC_WD_MIN	
	BGCRC_WD_MAX	

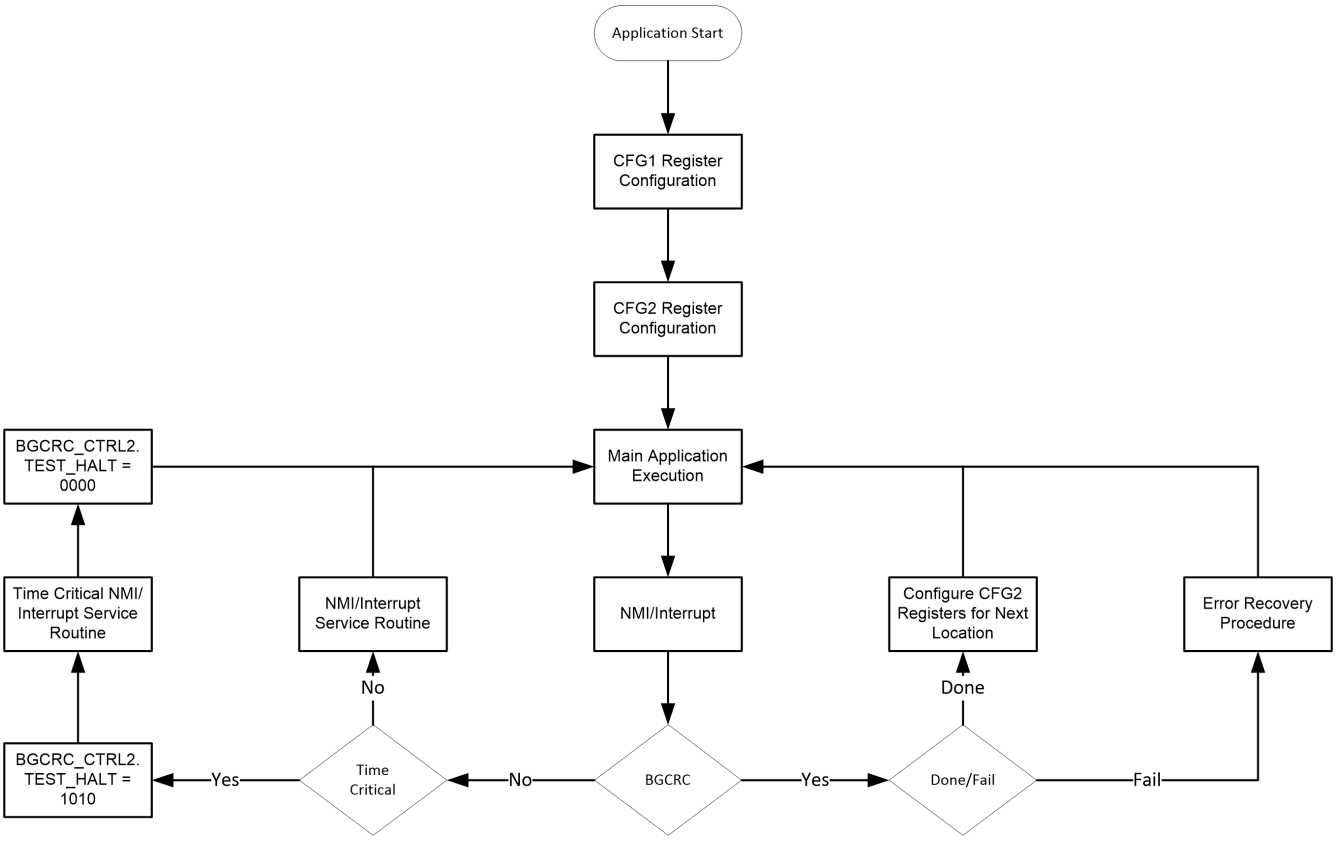


Figure 9-5. BGCRC Execution Sequence Flow

9.3.2 Decision on Error Response Severity

The error sources for BGCRC are:

- Test completion before BGCRC_WD_MIN.
- Test completed after BGCRC_WD_MAX.
- Mismatch between the calculated CRC and golden CRC.
- Uncorrectable error during data read (single bit error for parity memory or double bit error for ECC memory).
- Correctable error during memory read (single bit error for ECC memory).

Error severity can be chosen as either NMI or Interrupt. By default, error severity is set to NMI. It is not possible to configure error response for each individual error source. The response can be chosen as either Interrupt or NMI for all error sources. When error severity is chosen as NMI, ERROR_STS pin will be asserted during an error. CPU and CLA should be assigned the same error severity.

9.3.3 Decision of Controller for CLA_CRC

CPU_CRC can only be kicked off by the CPU. However, CLA_CRC can be kicked off by the CPU or CLA. If BGCRC error severity is chosen as NMI, it is possible to handle the background test using the CLA and error response with the CPU. Once the controller for BGCRC is chosen, it is possible to prevent accesses from other controllers by using the system level access protection configuration.

9.3.4 Execution of Time Critical Code from Wait-Stated Memories

BGCRC access to functionally wait-stated memories will also be wait-stated by the same number. Since it is impossible to predict the next functional access, any ongoing BGCRC access will have to complete before functional access is granted. To mitigate delay in functional access, the BGCRC can be halted when time-critical code which accesses the wait-stated memories is in progress. When BGCRC execution is halted, the BGCRC watchdog will not be halted. This is consistent with the safety requirement to complete the background test in a predictable window irrespective of the user code. In such scenarios, it is recommended to adjust the upper BGCRC watchdog window limit to account for the halt duration during functional access. However, if required, the BGCRC watchdog can be disabled.

9.3.5 BGCRC Execution

Two notes about BGCRC execution are as follows:

- BGCRC task can run as a background task once kicked off.
- BGCRC task does not impact functional execution for zero-wait stated memories. For memories with higher wait-states, the BGCRC engine can be halted to make functional execution predictable.

Figure 9-6 shows a few examples of BGCRC execution sequences.

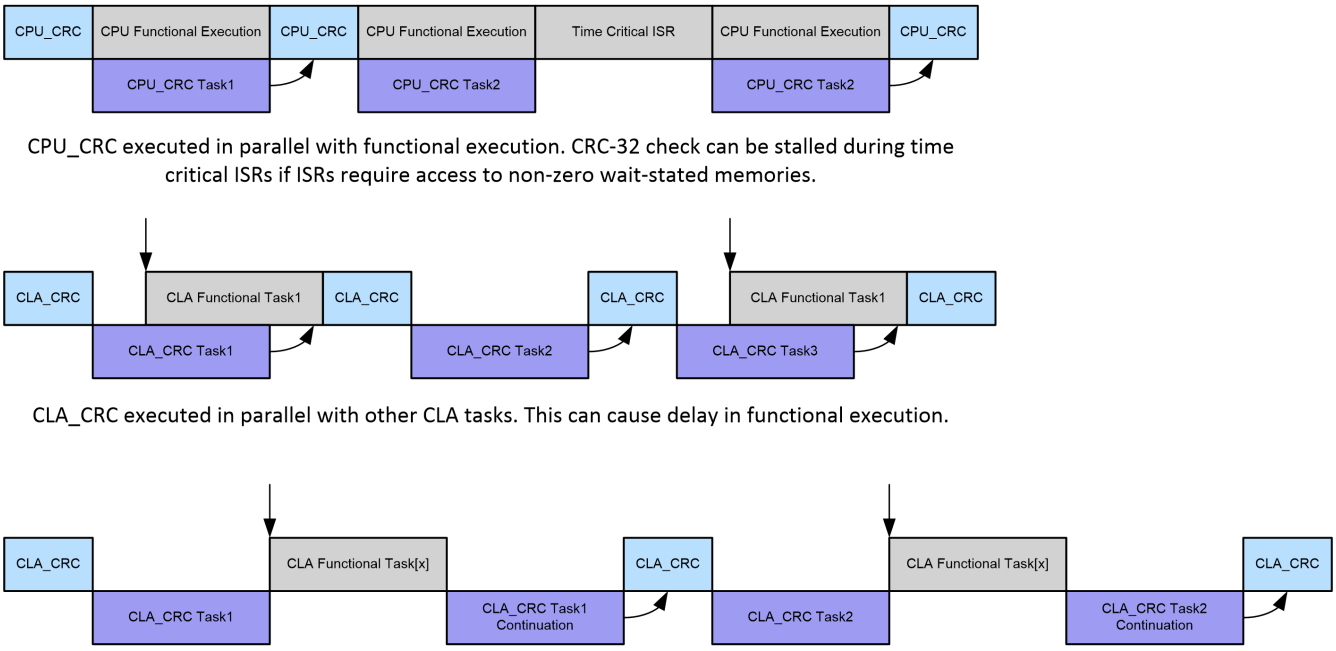


Figure 9-6. BGCRC Execution Sequence Example

9.3.6 Debug/Error Response for BGCRC Errors

The BGCRC error severity can be decoded by reading the BGCRC_INTFLG or BGCRC_NMIFLG. The errors can be:

- **WD_OVERFLOW/WD_UNDERFLOW:** The test did not complete in the programmed window. System timing needs to be checked to understand reason for non-completion of the test. BGCRC_CURR_ADDR and BGCRC_WD_CNT indicate how far the test progressed.
- **UNCORRECTABLE_ERR:** BGCRC_CURR_ADDR will contain the address of the memory location causing the error. This error is due to single bit failure for Parity SRAM or double bit failure for ECC SRAM. The memory location can be reloaded (if possible, for cases where code is copied from Flash) to see if the problem resolves. If the problem persists, it could be a permanent defect.
- **CORRECTABLE_ERR:** This error is due to single bit failure for ECC SRAM. If the problem location is accessed again (execution from the location for execute only memory or reading the location in other cases), the expectation is that the single bit error will be corrected. If the single bit error is not corrected, this could be an indication of a permanent defect.
- **CRC_FAIL:** This indicates a failure in the computation of the CRC-32 value. This error does not occur in scrub mode. For SRAMs with protection, in the absence of code bugs, this error is less likely since the error will most often manifest as a correctable/uncorrectable error. Code bugs can cause failure if the code inadvertently writes to a wrong address thus causing a CRC-32 error.

9.3.7 BGCRC Golden CRC-32 Value Computation

The C28x is a little-endian, 16-bit word addressable CPU. Therefore, the 32-bit value of 0x12345678 stored at address 0x100 is stored in the C28x memory as shown in Table 9-2.

Table 9-2. Data Address Location Example 1

Address	0x100	0x101
Data	0x5678	0x1234

The BGCRC order of byte calculations of the above example is 0x78, 0x56, 0x34, 0x12 and yields 0x6A330D2D. The 32-bit polynomial 0x04C11DB7 is used with an initialization vector of 0x00000000. The code snippet in Figure 9-7 shows the effective bit processing. Processing for all 32-bits within a word occurs in a single cycle within the BGCRC hardware.

```
seed = 0x0UL; // Initialize With Seed
poly = 0x04C11DB7UL: // 32-Bit Polynomial
crc32 = seed;

for(i=0; i<dataSize; i++)
{
        byteSwappedData = ((data[i] & 0x000000FF) << 24) |
                          ((data[i] & 0x0000FF00) <<  8) |
                          ((data[i] & 0x00FF0000) >>  8) |
                          ((data[i] & 0xFF000000) >> 24));

        crc32 = byteSwappedData ^ crc32;

        for(j=0; j<32; j++)
        {
                if(crc32 & 0x80000000) crc32 = (crc32 << 1) ^ poly;
                else crc32 = crc32 << 1;

                crc32 = crc32 & 0xFFFFFFFF;
        }
}
```

Figure 9-7. BGCRC Golden Value

A second example (Table 9-3) with two 32-bit words, 0x12345678 and 0x9ABCDEF0 at address 0x100 and 0x102 successively, would calculate the bytes in the order 0x78, 0x56, 0x34, 0x12, 0xDE, 0xBC, and 0x9A and yield 0x7E0B4164.

Table 9-3. Data Address Location Example 2

Address	0x100	0x101	0x102	0x103
Data	0x5678	0x1234	0xDEF0	0x9ABC

All data input to the BGCRC must align to a 32-bit boundary, both in the starting address and the size. It is possible to include 16-bit data within the span of data; however, when the data is read by the BGCRC, it will always assume 32-bits and conform to the above calculation order. For example, if two 16-bit words (0xA0B1 and 0xC2D3) were placed in between the previous two 32-bit words (Table 9-4), the calculations would be performed in byte order 0x78, 0x56, 0x34, 0x12, 0xB1, 0xA0, 0xD3, 0xC2, 0xF0, 0xDE, 0xBC and 0x9A and yield 0x2AEFD987.

Table 9-4. Data Address Location Example 3

Address	0x100	0x101	0x102	0x103	0x104	0x105
Data	0x5678	0x1234	0xA0B1	0xC2D3	0xDEF0	0x9ABC

9.4 Software

9.4.1 BGCRC Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/bgcrc

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

9.4.1.1 BGCRC CPU Interrupt Example

FILE: bgcrc_ex1_cpuinterrupt.c

This example demonstrates how to configure and trigger BGCRC from the CPU. BGCRC module is configured for 1 KB of GS0 RAM which is programmed with a known data. The pre-computed CRC value is used as the golden CRC value. Interrupt is generated once the computation is done and checks if no error flags are raised Calculation uses the 32-bit polynomial 0x04C11DB7 and seed value 0x00000000.

External Connections

- None.

Watch Variables

- pass - This should be 1.
- runStatus - BGCRC running status. This will be BGCRC_ACTIVE if the module is running, BGCRC_IDLE if the module is idle

9.4.1.2 BGCRC Example with Watchdog and Lock

FILE: bgcrc_ex2_cpubgcrc_basic.c

This example demonstrates how to configure and trigger BGCRC from the CPU. It also showcases how to configure the CRC watchdog and lock the registers after configuring the module. The watchdog is used as a diagnostic to check memory test completion within the expected time window. An error signal is generated if the test does not complete in the specified time window.

The module is configured for 1kB of GS0 RAM which is programmed with random data. The golden CRC value for comparison is computed using software method. Interrupt is generated once the computation is done and checks if no error flags are raised. The NMI is enabled and is triggered if an error is detected.

External Connections

- None.

Watch Variables

- pass
- bgcrcDone

9.4.1.3 CLA-BGCRC Example in CRC Mode

FILE: bgcrc_ex3_clabgcrc_crcmode.c

This example demonstrates how to configure and trigger CLABGCRC from the CPU. It also showcases how to configure the CRC watchdog and lock the registers after configuring the module. The watchdog is used as a diagnostic to check memory test completion within the expected time window. An error signal is generated if the test does not complete in the specified time window.

The module is configured for 1kB of CLA ROM memory. The golden CRC value for comparison is computed using software method. Interrupt is generated once the computation is done and checks if no error flags are raised. The NMI is enabled and is triggered if an error is detected.

External Connections

- None.

Watch Variables

- pass
- bgcrcDone

9.4.1.4 CLA-BGCRC Example in Scrub Mode

FILE: bgcrc_ex4_clabgcrc_scrubmode.c

This example demonstrates how to configure and trigger CLA-BGCRC in Scrub mode. In Scrub mode, CRC of data is not compared with the golden CRC. Error check is done using the ECC/Parity logic. It also showcases how to configure the CRC watchdog and lock the registers after configuring the module. The watchdog is used as a diagnostic to check memory test completion within the expected time window. An error signal is generated if the test does not complete in the specified time window.

The module is configured for 256 bytes of CLA ROM memory. Interrupt is generated once the computation is done and checks if no error flags are raised. The NMI is enabled and is triggered if an error is detected.

External Connections

- None.

Watch Variables

- pass
- bgcrcDone

9.5 BGCRC Registers

This section describes the Background CRC registers.

9.5.1 BGCRC Base Address Table

Table 9-5. BGCRC Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
BgcrcCpuRegs	BGCRC_REGS	BGCRC_CPU_BASE	0x0000_6340	YES	-	-	-	YES
BgcrcCla1Regs	BGCRC_REGS	BGCRC_CLA1_BASE	0x0000_6380	YES	-	-	YES	YES

9.5.2 BGCRC_REGS Registers

Table 9-6 lists the memory-mapped registers for the BGCRC_REGS registers. All register offset addresses not listed in Table 9-6 should be considered as reserved locations and the register contents should not be modified.

Table 9-6. BGCRC_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	BGCRC_EN	BGCRC Enable	EALLOW	Go
2h	BGCRC_CTRL1	BGCRC Control register 1	EALLOW	Go
4h	BGCRC_CTRL2	BGCRC Control register 2	EALLOW	Go
6h	BGCRC_START_ADDR	Start address for the BGCRC check	EALLOW	Go
8h	BGCRC_SEED	Seed for CRC calculation	EALLOW	Go
Eh	BGCRC_GOLDEN	Golden CRC to be compared against	EALLOW	Go
10h	BGCRC_RESULT	CRC calculated		Go
12h	BGCRC_CURR_ADDR	Current address regsiter		Go
1Ch	BGCRC_WD_CFG	BGCRC windowed watchdog configuration	EALLOW	Go
1Eh	BGCRC_WD_MIN	BGCRC windowed watchdog min value	EALLOW	Go
20h	BGCRC_WD_MAX	BGCRC windowed watchdog max value	EALLOW	Go
22h	BGCRC_WD_CNT	BGCRC windowed watchdog count		Go
2Ah	BGCRC_NMIFLG	BGCRC NMI flag register		Go
2Ch	BGCRC_NMICLR	BGCRC NMI flag clear register	EALLOW	Go
2Eh	BGCRC_NMIFRC	BGCRC NMI flag force register	EALLOW	Go
34h	BGCRC_INTEN	Interrupt enable	EALLOW	Go
36h	BGCRC_INTFLG	Interrupt flag		Go
38h	BGCRC_INTCLR	Interrupt flag clear	EALLOW	Go
3Ah	BGCRC_INTFRC	Interrupt flag force	EALLOW	Go
3Ch	BGCRC_LOCK	BGCRC register map lockconfiguration	EALLOW	Go
3Eh	BGCRC_COMMIT	BGCRC register map commit configuration	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 9-7 shows the codes that are used for access types in this section.

Table 9-7. BGCRC_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 9-7. BGCRC_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

9.5.2.1 BGCRC_EN Register (Offset = 0h) [Reset = 0h]

BGCRC_EN is shown in Figure 9-8 and described in Table 9-8.

Return to the Summary Table.

BGCRC Enable

Figure 9-8. BGCRC_EN Register

31	30	29	28	27	26	25	24
RUN_STS	RESERVED						
R-0h	R-0h						
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				START			
R-0h				R-0/W-0h			

Table 9-8. BGCRC_EN Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RUN_STS	R	0h	Status bit: 0 : CRC module is IDLE 1 : CRC module is Active This bit will remain set during BGCRC_CTRL2.TEST_HALT = 1 Reset type: CPUx.SYSRSn
30-16	RESERVED	R	0h	Reserved
15-4	RESERVED	R	0h	Reserved
3-0	START	R-0/W	0h	Start Bit: "1010": Kick-off CRC calculations "any other value": ignored Notes: Setting this anytime during the CRC caluclation will reset and re-start the CRC calculation. BGCRC_WD_CNT regsters will be reset CRCEN.START = "1010". BGCRC_INTFLG, BGCRC_NMIFLG will not be impacted by this configuration. Reset type: CPUx.SYSRSn

9.5.2.2 BGCRC_CTRL1 Register (Offset = 2h) [Reset = 0h]

BGCRC_CTRL1 is shown in Figure 9-9 and described in Table 9-9.

Return to the Summary Table.

BGCRC Control register 1

Figure 9-9. BGCRC_CTRL1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				NMIDIS			
R-0h				R/W-0h			
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			FREE_SOFT	RESERVED			
R-0h			R/W-0h	R-0h			

Table 9-9. BGCRC_CTRL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19-16	NMIDIS	R/W	0h	1010 : NMI is disabled Any other value : NMI is enabled. Reset type: CPUx.SYSRSn
15-5	RESERVED	R	0h	Reserved
4	FREE_SOFT	R/W	0h	Emulation control bit : This bit controls behaviour of CRC calculation during emulation 0 : Soft, CRC module and CRC Watchdog stops immediately on DEBUG SUSPEND (of CRC-controller). 1 : Free, CRC calcuation and CRC watchdog is not affected by DEBUG HALT (of CRC-controller) Reset type: CPUx.SYSRSn
3-0	RESERVED	R	0h	Reserved

9.5.2.3 BGCRC_CTRL2 Register (Offset = 4h) [Reset = 0h]

BGCRC_CTRL2 is shown in Figure 9-10 and described in Table 9-10.

Return to the Summary Table.

BGCRC Control register 2

Figure 9-10. BGCRC_CTRL2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				SCRUB_MODE			
R-0h				R/W-0h			
15	14	13	12	11	10	9	8
TEST_HALT				RESERVED		BLOCK_SIZE	
R/W-0h				R-0h		R/W-0h	
7	6	5	4	3	2	1	0
BLOCK_SIZE							
R/W-0h							

Table 9-10. BGCRC_CTRL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19-16	SCRUB_MODE	R/W	0h	Scrub mode configuration 1010 : Scrub mode, CRC of data is not compared with the golden CRC. Error check is done using the ECC/Parity logic. Any other value: CRC value is compared with golden CRC at the end in addition to the data correctness check by ECC/Parity logic. Notes: 1010 configuration is used for scrub mode (for data memories) where the memory value is read and ECC/Parity logic is used for the error detection. BGCRC_RESULT.CRC_VALUE is not updated in this configuration. Reset type: CPUx.SYSRSn
15-12	TEST_HALT	R/W	0h	Halt Bit : 1010 : Module operation is stopped Any other value : CRC calaculation will continue/resume from where it was halted Notes: BGCRC_EN.START = 1010 configuration with TEST_HALT = 1010 will halt the CRC calculation. The new check will resume when TEST_HALT is configured to a value other than 1010 Reset type: CPUx.SYSRSn
11-10	RESERVED	R	0h	Reserved
9-0	BLOCK_SIZE	R/W	0h	Configures the block size for the check 0x0 : 256 Byte (default) 0x1 : 512 Byte 0x2 : 768 Byte 0x3 : 1KB ... 0x3FF : 256KB (0xn : (n+1)*256Byte) Reset type: CPUx.SYSRSn

9.5.2.4 BGCRC_START_ADDR Register (Offset = 6h) [Reset = 0h]

BGCRC_START_ADDR is shown in Figure 9-11 and described in Table 9-11.

Return to the Summary Table.

Start address for the BGCRC check

Figure 9-11. BGCRC_START_ADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
START_ADDRESS																															
R/W-0h																															

Table 9-11. BGCRC_START_ADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	START_ADDRESS	R/W	0h	START_ADDRESS indicates the start point of the test. (For CPU_CRC, this will be the CPU address. For CLA_CRC, this will be the CLA address where the memory is mapped) Reset type: CPUx.SYSRSn

9.5.2.5 BGCRC_SEED Register (Offset = 8h) [Reset = 0h]

BGCRC_SEED is shown in Figure 9-12 and described in Table 9-12.

Return to the Summary Table.

Seed for CRC calculation

Figure 9-12. BGCRC_SEED Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEED																															
R/W-0h																															

Table 9-12. BGCRC_SEED Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SEED	R/W	0h	Initial value of CRC, this value is coiped to the CRC register on triggering CRC calculation by writing to START bit. Reset type: CPUx.SYSRSn

9.5.2.6 BGCRC_GOLDEN Register (Offset = Eh) [Reset = 0h]

BGCRC_GOLDEN is shown in Figure 9-13 and described in Table 9-13.

Return to the Summary Table.

Golden CRC to be compared against

Figure 9-13. BGCRC_GOLDEN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC_VALUE																															
R/W-0h																															

Table 9-13. BGCRC_GOLDEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CRC_VALUE	R/W	0h	Golden CRC register: If CRC check is enabled, the calcuated CRC value is compared with golden CRC and status is updated. Reset type: CPUx.SYSRSn

9.5.2.7 BGCRC_RESULT Register (Offset = 10h) [Reset = 0h]

BGCRC_RESULT is shown in Figure 9-14 and described in Table 9-14.

Return to the Summary Table.

CRC calculated

Figure 9-14. BGCRC_RESULT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC_VALUE																															
R-0h																															

Table 9-14. BGCRC_RESULT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CRC_VALUE	R	0h	CRC result register This reister value will be updated only on the completion of CRC check on a block of data as programmed by BGCRC_CTRL2.BLOCK_SIZE. Reset type: CPUx.SYSRSn

9.5.2.8 BGCRC_CURR_ADDR Register (Offset = 12h) [Reset = 0h]

BGCRC_CURR_ADDR is shown in Figure 9-15 and described in Table 9-15.

Return to the Summary Table.

Current address regsiter

Figure 9-15. BGCRC_CURR_ADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CURRENT_ADDR																															
R-0h																															

Table 9-15. BGCRC_CURR_ADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CURRENT_ADDR	R	0h	Current address from where the data is fetched. During a failure, the CURRENT_ADDR field indicates the value from where the last fetch happened. Reset type: CPUx.SYSRSn

9.5.2.9 BGCRC_WD_CFG Register (Offset = 1Ch) [Reset = 0h]

BGCRC_WD_CFG is shown in Figure 9-16 and described in Table 9-16.

Return to the Summary Table.

BGCRC windowed watchdog configuration

Figure 9-16. BGCRC_WD_CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												WDDIS			
R-0h												R/W-0h			

Table 9-16. BGCRC_WD_CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3-0	WDDIS	R/W	0h	1010: CRC Watchdog counter is disabled. Any other value: CRC watchdog is enabled Watchdog is an upcounter and starts counting when BGCRC_EN.START is asserted. Watchdog continues to count during TEST_HALT state also(BGCRC_CTRL2.TEST_HALT = "1010"). CRC watchdog can be disabled during TEST_HALT by explicit configuration. (BGCRC_WD_CFG.WDDIS = 1010). Once the watchdog is disabled and re-enabled, watchdog count resumes from the previous disabled point. Reset type: CPUx.SYSRSn

9.5.2.10 BGCRC_WD_MIN Register (Offset = 1Eh) [Reset = 0h]

BGCRC_WD_MIN is shown in Figure 9-17 and described in Table 9-17.

Return to the Summary Table.

BGCRC windowed watchdog min value

Figure 9-17. BGCRC_WD_MIN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MINVAL																															
R/W-0h																															

Table 9-17. BGCRC_WD_MIN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MINVAL	R/W	0h	If the CRC computation completes before BGCRC_WD_MIN.MINVAL FAIL_STATUS.WD_UNDERFLOW flag gets set. Reset type: CPUx.SYSRSn

9.5.2.11 BGCRC_WD_MAX Register (Offset = 20h) [Reset = FFFFFFFFh]

BGCRC_WD_MAX is shown in Figure 9-18 and described in Table 9-18.

Return to the Summary Table.

BGCRC windowed watchdog max value

Figure 9-18. BGCRC_WD_MAX Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAXVAL																															
R/W-FFFFFFFFh																															

Table 9-18. BGCRC_WD_MAX Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MAXVAL	R/W	FFFFFFFFh	If the CRC computation doesn't complete before BGCRC_WD_MIN.MAXVAL FAIL_STATUS.WD_OVERFLOW flag gets set. Reset type: CPUx.SYSRSn

9.5.2.12 BGCRC_WD_CNT Register (Offset = 22h) [Reset = 0h]

BGCRC_WD_CNT is shown in Figure 9-19 and described in Table 9-19.

Return to the Summary Table.

BGCRC windowed watchdog count

Figure 9-19. BGCRC_WD_CNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WD_CNT																															
R-0h																															

Table 9-19. BGCRC_WD_CNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	WD_CNT	R	0h	CRC windowed watchdog counter value Counter value freezes at the end of CRC computation and will be reloaded only by BGCRC_EN.START = "1010" configuration. BGCRC_WD_CNT register freezes when a failure occurs. Reset type: CPUx.SYSRSn

9.5.2.13 BGCRC_NMIFLG Register (Offset = 2Ah) [Reset = 0h]

BGCRC_NMIFLG is shown in Figure 9-20 and described in Table 9-20.

Return to the Summary Table.

BGCRC NMI flag register

Figure 9-20. BGCRC_NMIFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	WD_OVERFLOW	WD_UNDERFLOW	CORRECTABLE_ERR	UNCORRECTABLE_ERR	CRC_FAIL	RESERVED	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 9-20. BGCRC_NMIFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reserved
6	WD_OVERFLOW	R	0h	Windowed watchdog Overflow. 1 : Test did not complete before BGCRC_WD_MAX.MAXVAL 0 : No such errors Reset type: CPUx.SYSRSn
5	WD_UNDERFLOW	R	0h	Windowed watchdog underflow. 1 : Test completed before BGCRC_WD_MIN.MINVAL 0 : No such errors Reset type: CPUx.SYSRSn
4	CORRECTABLE_ERR	R	0h	Correctable error indication: 0 : No ECC correctable error during memory read 1 : Correctable ECC error during memory read Note: ECC computation is done during every memory read. (Correctable errors are not ignored since the module doesn't write-back corrected value. The error remains in the memory and required corrective action need to be taken by CPU/CLA as part of ISR) Reset type: CPUx.SYSRSn
3	UNCORRECTABLE_ERR	R	0h	Uncorrectable error indication: 0 : No ECC-uncorrectable/Parity error during memory read 1 : ECC-uncorrectable/Parity error during memory read Note: ECC/Parity check is done during every memory read. Reset type: CPUx.SYSRSn
2	CRC_FAIL	R	0h	CRC FAIL interrupt 0 : No failure in CRC check. 1 : CRC check failure Note: Comparion is enabled only after CRC calc is completed Reset type: CPUx.SYSRSn
1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

9.5.2.14 BGCRC_NMICLR Register (Offset = 2Ch) [Reset = 0h]

BGCRC_NMICLR is shown in Figure 9-21 and described in Table 9-21.

Return to the Summary Table.

BGCRC NMI flag clear register

Figure 9-21. BGCRC_NMICLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	WD_OVERFLOW	WD_UNDERFLOW	CORRECTABLE_ERR	UNCORRECTABLE_ERR	CRC_FAIL	RESERVED	RESERVED
R-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0h	R-0h

Table 9-21. BGCRC_NMICLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reserved
6	WD_OVERFLOW	R-0/W1S	0h	Clear WD_OVERFLOW NMI flag 0 No effect 1 Clears NMI flag Reset type: CPUx.SYSRSn
5	WD_UNDERFLOW	R-0/W1S	0h	Clear WD_UNDERFLOW NMI flag 0 No effect 1 Clears NMI flag Reset type: CPUx.SYSRSn
4	CORRECTABLE_ERR	R-0/W1S	0h	Clear CORRECTABLE_ERR NMI flag 0 No effect 1 Clears NMI flag Reset type: CPUx.SYSRSn
3	UNCORRECTABLE_ERR	R-0/W1S	0h	Clear UNCORRECTABLE_ERROR NMI flag 0 No effect 1 Clears NMI flag Reset type: CPUx.SYSRSn
2	CRC_FAIL	R-0/W1S	0h	Clear CRC_FAIL NMI flag 0 No effect 1 Clears NMI flag Reset type: CPUx.SYSRSn
1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

9.5.2.15 BGCRC_NMIFRC Register (Offset = 2Eh) [Reset = 0h]

BGCRC_NMIFRC is shown in Figure 9-22 and described in Table 9-22.

Return to the Summary Table.

BGCRC NMI flag force register

Figure 9-22. BGCRC_NMIFRC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	WD_OVERFLOW	WD_UNDERFLOW	CORRECTABLE_ERR	UNCORRECTABLE_ERR	CRC_FAIL	RESERVED	RESERVED
R-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0h	R-0h

Table 9-22. BGCRC_NMIFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reserved
6	WD_OVERFLOW	R-0/W1S	0h	Force WD_OVERFLOW NMI flag 0 No effect 1 force NMI flag Reset type: CPUx.SYSRSn
5	WD_UNDERFLOW	R-0/W1S	0h	Force WD_UNDERFLOW NMI flag 0 No effect 1 force NMI flag Reset type: CPUx.SYSRSn
4	CORRECTABLE_ERR	R-0/W1S	0h	Force CORRECTABLE_ERR NMI flag 0 No effect 1 force NMI flag Reset type: CPUx.SYSRSn
3	UNCORRECTABLE_ERR	R-0/W1S	0h	Force UNCORRECTABLE_ERR NMI flag 0 No effect 1 force NMI flag Reset type: CPUx.SYSRSn
2	CRC_FAIL	R-0/W1S	0h	Force CRC_FAIL NMI flag 0 No effect 1 force NMI flag Reset type: CPUx.SYSRSn
1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

9.5.2.16 BGCRC_INTEN Register (Offset = 34h) [Reset = 0h]

BGCRC_INTEN is shown in Figure 9-23 and described in Table 9-23.

Return to the Summary Table.

Interrupt enable

Figure 9-23. BGCRC_INTEN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	WD_OVERFLOW	WD_UNDERFLOW	CORRECTABLE_ERR	UNCORRECTABLE_ERR	CRC_FAIL	TEST_DONE	RESERVED
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h

Table 9-23. BGCRC_INTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reserved
6	WD_OVERFLOW	R/W	0h	0 WD_OVERFLOW Interrupt disabled 1 WD_OVERFLOW Interrupt enabled Reset type: CPUx.SYSRSn
5	WD_UNDERFLOW	R/W	0h	0 WD_UNDERFLOW Interrupt disabled 1 WD_UNDERFLOW Interrupt enabled Reset type: CPUx.SYSRSn
4	CORRECTABLE_ERR	R/W	0h	0 CORRECTABLE_ERR Interrupt disabled 1 CORRECTABLE_ERR Interrupt enabled Reset type: CPUx.SYSRSn
3	UNCORRECTABLE_ERR	R/W	0h	0 UNCORRECTABLE_ERR Interrupt disabled 1 UNCORRECTABLE_ERR Interrupt enabled Reset type: CPUx.SYSRSn
2	CRC_FAIL	R/W	0h	0 CRC_FAIL Interrupt disabled 1 CRC_FAIL Interrupt enabled Reset type: CPUx.SYSRSn
1	TEST_DONE	R/W	0h	0 TEST_DONE Interrupt disabled 1 TEST_DONE Interrupt enabled Reset type: CPUx.SYSRSn
0	RESERVED	R	0h	Reserved

9.5.2.17 BGCRC_INTFLG Register (Offset = 36h) [Reset = 0h]

BGCRC_INTFLG is shown in Figure 9-24 and described in Table 9-24.

Return to the Summary Table.

Interrupt flag

Figure 9-24. BGCRC_INTFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	WD_OVERFLOW	WD_UNDERFLOW	CORRECTABLE_ERR	UNCORRECTABLE_ERR	CRC_FAIL	TEST_DONE	INT
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 9-24. BGCRC_INTFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reserved
6	WD_OVERFLOW	R	0h	Windowed watchdog Overflow. 1 : Test did not completed before BGCRC_WD_MAX.MAXVAL 0 : No such errors Reset type: CPUx.SYSRSn
5	WD_UNDERFLOW	R	0h	Windowed watchdog underflow. 1 : Test completed before BGCRC_WD_MIN.MINVAL 0 : No such errors Reset type: CPUx.SYSRSn
4	CORRECTABLE_ERR	R	0h	Correctable error indication: 0 : No ECC correctable error during memory read 1 : Correctable ECC error during memory read Note: ECC computation is done during every memory read. (Correctable errors are not ignored since the module doesn't write-back corrected value. The error remains in the memory and required corrective action need to be taken by CPU/CLA as part of ISR) Reset type: CPUx.SYSRSn
3	UNCORRECTABLE_ERR	R	0h	uncorrectable error indication: 0 : No ECC-uncorrectable/Parity error during memory read 1 : ECC-uncorrectable/Parity error during memory read Note: ECC/Parity check is done during every memory read. Reset type: CPUx.SYSRSn
2	CRC_FAIL	R	0h	CRC fail interrupt 0 : No failure of CRC check 1 : CRC check failure Note: Comparion is enabled only after CRC calc is completed Reset type: CPUx.SYSRSn
1	TEST_DONE	R	0h	Done Interrupt Status flag 0 CRC calculation is in progress or CRC module is idle. 1 CRC calculation is done. Note: TEST_DONE flag will get set on CRC calculation completion even in case of CRC mismatch Reset type: CPUx.SYSRSn

Table 9-24. BGCRC_INTFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INT	R	0h	Global Interrupt Status flag 0 No interrupt generated 1 Interrupt was generated Reset type: CPUx.SYSRSn

9.5.2.18 BGCRC_INTCLR Register (Offset = 38h) [Reset = 0h]

BGCRC_INTCLR is shown in Figure 9-25 and described in Table 9-25.

Return to the Summary Table.

Interrupt flag clear

Figure 9-25. BGCRC_INTCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	WD_OVERFLOW	WD_UNDERFLOW	CORRECTABLE_ERR	UNCORRECTABLE_ERR	CRC_FAIL	TEST_DONE	INT
R-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 9-25. BGCRC_INTCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reserved
6	WD_OVERFLOW	R-0/W1S	0h	Clear interrupt flag 0 No effect 1 Clears the interrupt flag Reset type: CPUx.SYSRSn
5	WD_UNDERFLOW	R-0/W1S	0h	Clear interrupt flag 0 No effect 1 Clears the interrupt flag Reset type: CPUx.SYSRSn
4	CORRECTABLE_ERR	R-0/W1S	0h	Clear interrupt flag 0 No effect 1 Clears the interrupt flag Reset type: CPUx.SYSRSn
3	UNCORRECTABLE_ERR	R-0/W1S	0h	Clear interrupt flag 0 No effect 1 Clears the interrupt flag Reset type: CPUx.SYSRSn
2	CRC_FAIL	R-0/W1S	0h	Clear interrupt flag 0 No effect 1 Clears the interrupt flag Reset type: CPUx.SYSRSn
1	TEST_DONE	R-0/W1S	0h	Clear interrupt flag 0 No effect 1 Clears the interrupt flag Reset type: CPUx.SYSRSn
0	INT	R-0/W1S	0h	Global Interrupt Clear 0 No effect 1 Clears the interrupt flag and enables further interrupts to be generated if an event flags is set to 1. Reset type: CPUx.SYSRSn

9.5.2.19 BGCRC_INTFRC Register (Offset = 3Ah) [Reset = 0h]

BGCRC_INTFRC is shown in Figure 9-26 and described in Table 9-26.

Return to the Summary Table.

Interrupt flag force

Figure 9-26. BGCRC_INTFRC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	WD_OVERFLOW	WD_UNDERFLOW	CORRECTABLE_ERR	UNCORRECTABLE_ERR	CRC_FAIL	TEST_DONE	RESERVED
R-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0h

Table 9-26. BGCRC_INTFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reserved
6	WD_OVERFLOW	R-0/W1S	0h	Force interrupt flag 0 No effect 1 force the interrupt flag Reset type: CPUx.SYSRSn
5	WD_UNDERFLOW	R-0/W1S	0h	Force interrupt flag 0 No effect 1 force the interrupt flag Reset type: CPUx.SYSRSn
4	CORRECTABLE_ERR	R-0/W1S	0h	Force interrupt flag 0 No effect 1 force the interrupt flag Reset type: CPUx.SYSRSn
3	UNCORRECTABLE_ERR	R-0/W1S	0h	Force interrupt flag 0 No effect 1 force the interrupt flag Reset type: CPUx.SYSRSn
2	CRC_FAIL	R-0/W1S	0h	Force interrupt flag 0 No effect 1 force the interrupt flag Reset type: CPUx.SYSRSn
1	TEST_DONE	R-0/W1S	0h	Force interrupt flag 0 No effect 1 force the interrupt flag Reset type: CPUx.SYSRSn
0	RESERVED	R	0h	Reserved

9.5.2.20 BGCRC_LOCK Register (Offset = 3Ch) [Reset = 0h]

BGCRC_LOCK is shown in Figure 9-27 and described in Table 9-27.

Return to the Summary Table.

BGCRC register map lockconfiguration

Figure 9-27. BGCRC_LOCK Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	BGCRC_INTFRC	RESERVED	RESERVED	BGCRC_INTEN	RESERVED	RESERVED
R-0h	R-0h	R/W-0h	R-0h	R-0h	R/W-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
BGCRC_NMIFRC	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	BGCRC_WD_MAX
R/W-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R/W-0h
15	14	13	12	11	10	9	8
BGCRC_WD_MIN	BGCRC_WD_CFG	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
BGCRC_GOLDEN	RESERVED	RESERVED	BGCRC_SEED	BGCRC_START_ADDR	BGCRC_CTRL2	BGCRC_CTRL1	BGCRC_EN
R/W-0h	R-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 9-27. BGCRC_LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved
30	RESERVED	R	0h	Reserved
29	BGCRC_INTFRC	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: CPUx.SYSRSn
28	RESERVED	R	0h	Reserved
27	RESERVED	R	0h	Reserved
26	BGCRC_INTEN	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: CPUx.SYSRSn
25	RESERVED	R	0h	Reserved
24	RESERVED	R	0h	Reserved
23	BGCRC_NMIFRC	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: CPUx.SYSRSn
22	RESERVED	R	0h	Reserved
21	RESERVED	R	0h	Reserved
20	RESERVED	R	0h	Reserved
19	RESERVED	R	0h	Reserved
18	RESERVED	R	0h	Reserved
17	RESERVED	R	0h	Reserved
16	BGCRC_WD_MAX	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: CPUx.SYSRSn

Table 9-27. BGCRC_LOCK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15	BGCRC_WD_MIN	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: CPUx.SYSRSn
14	BGCRC_WD_CFG	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: CPUx.SYSRSn
13	RESERVED	R	0h	Reserved
12	RESERVED	R	0h	Reserved
11	RESERVED	R	0h	Reserved
10	RESERVED	R	0h	Reserved
9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	BGCRC_GOLDEN	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: CPUx.SYSRSn
6	RESERVED	R	0h	Reserved
5	RESERVED	R	0h	Reserved
4	BGCRC_SEED	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: CPUx.SYSRSn
3	BGCRC_START_ADDR	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: CPUx.SYSRSn
2	BGCRC_CTRL2	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: CPUx.SYSRSn
1	BGCRC_CTRL1	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: CPUx.SYSRSn
0	BGCRC_EN	R/W	0h	0: Register configuration is not locked. 1: Register configuration is locked. Reset type: CPUx.SYSRSn

9.5.2.21 BGCRC_COMMIT Register (Offset = 3Eh) [Reset = 0h]

BGCRC_COMMIT is shown in Figure 9-28 and described in Table 9-28.

Return to the Summary Table.

BGCRC register map commit configuration

Figure 9-28. BGCRC_COMMIT Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	BGCRC_INTFRC	RESERVED	RESERVED	BGCRC_INTEN	RESERVED	RESERVED
R-0h	R-0h	R/WSonce-0h	R-0h	R-0h	R/WSonce-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
BGCRC_NMIFRC	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	BGCRC_WD_MAX
R/WSonce-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R/WSonce-0h
15	14	13	12	11	10	9	8
BGCRC_WD_MIN	BGCRC_WD_CFG	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/WSonce-0h	R/WSonce-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
BGCRC_GOLDEN	RESERVED	RESERVED	BGCRC_SEED	BGCRC_START_ADDR	BGCRC_CTRL2	BGCRC_CTRL1	BGCRC_EN
R/WSonce-0h	R-0h	R-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 9-28. BGCRC_COMMIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved
30	RESERVED	R	0h	Reserved
29	BGCRC_INTFRC	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: CPUx.SYSRSn
28	RESERVED	R	0h	Reserved
27	RESERVED	R	0h	Reserved
26	BGCRC_INTEN	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: CPUx.SYSRSn
25	RESERVED	R	0h	Reserved
24	RESERVED	R	0h	Reserved
23	BGCRC_NMIFRC	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: CPUx.SYSRSn
22	RESERVED	R	0h	Reserved
21	RESERVED	R	0h	Reserved
20	RESERVED	R	0h	Reserved
19	RESERVED	R	0h	Reserved
18	RESERVED	R	0h	Reserved

Table 9-28. BGCRC_COMMIT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	RESERVED	R	0h	Reserved
16	BGCRC_WD_MAX	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: CPUx.SYSRSn
15	BGCRC_WD_MIN	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: CPUx.SYSRSn
14	BGCRC_WD_CFG	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: CPUx.SYSRSn
13	RESERVED	R	0h	Reserved
12	RESERVED	R	0h	Reserved
11	RESERVED	R	0h	Reserved
10	RESERVED	R	0h	Reserved
9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	BGCRC_GOLDEN	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: CPUx.SYSRSn
6	RESERVED	R	0h	Reserved
5	RESERVED	R	0h	Reserved
4	BGCRC_SEED	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: CPUx.SYSRSn
3	BGCRC_START_ADDR	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: CPUx.SYSRSn
2	BGCRC_CTRL2	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: CPUx.SYSRSn
1	BGCRC_CTRL1	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: CPUx.SYSRSn
0	BGCRC_EN	R/WSonce	0h	0: Register lock configuration is not committed. 1: Register configuration is committed. Once configuration is committed, only reset can change the configuration. Reset type: CPUx.SYSRSn

9.5.3 BGCRC Registers to Driverlib Functions

Table 9-29. BGCRC Registers to Driverlib Functions

File	Driverlib Function
EN	
bgcrc.h	BGCRC_start
bgcrc.h	BGCRC_getRunStatus
CTRL1	
bgcrc.h	BGCRC_setConfig
CTRL2	
bgcrc.h	BGCRC_setRegion
bgcrc.h	BGCRC_halt
bgcrc.h	BGCRC_resume
START_ADDR	
bgcrc.h	BGCRC_setRegion
SEED	
bgcrc.h	BGCRC_setSeedValue
GOLDEN	
bgcrc.h	BGCRC_setGoldenCRCValue
RESULT	
bgcrc.h	BGCRC_getResult
CURR_ADDR	
bgcrc.h	BGCRC_getCurrentAddress
WD_CFG	
bgcrc.h	BGCRC_enableWatchdog
bgcrc.h	BGCRC_disableWatchdog
WD_MIN	
bgcrc.h	BGCRC_setWatchdogWindow
WD_MAX	
bgcrc.h	BGCRC_setWatchdogWindow
WD_CNT	
bgcrc.h	BGCRC_getWatchdogCounterValue
NMIFLG	
bgcrc.h	BGCRC_getNMIStatus
NMICLR	
bgcrc.h	BGCRC_clearNMIStatus
NMIFRC	
bgcrc.h	BGCRC_forceNMI
INTEN	
bgcrc.h	BGCRC_enableInterrupt
bgcrc.h	BGCRC_disableInterrupt
INTFLG	
bgcrc.h	BGCRC_getInterruptStatus
INTCLR	
bgcrc.h	BGCRC_clearInterruptStatus
INTFRC	
bgcrc.h	BGCRC_forceInterrupt
LOCK	

Table 9-29. BGCRC Registers to Driverlib Functions (continued)

File	Driverlib Function
bgcrc.h	BGCRC_lockRegister
bgcrc.h	BGCRC_unlockRegister
COMMIT	
bgcrc.h	BGCRC_commitRegisterLock

Chapter 10

General-Purpose Input/Output (GPIO)

The GPIO module controls the device's digital and analog I/O multiplexing, which uses shared pins to maximize application flexibility. The pins are named by their general-purpose I/O name (for example, GPIO0, GPIO25, GPIO58). These pins can be individually selected to operate as digital I/O (also called GPIO mode), or connected to one of several peripheral I/O signals. The input signals can be qualified to remove unwanted noise.

10.1 Introduction

Up to twelve independent peripheral signals are multiplexed on a single GPIO-enabled pin in addition to the CPU-controlled I/O capability. Each pin output can be controlled by either a peripheral or one of the two CPU masters (CPU1, CPU1.CLA). There are two I/O ports:

Up to twelve independent peripheral signals are multiplexed on a single GPIO-enabled pin in addition to the CPU-controlled I/O capability. Each pin output can be controlled by either a peripheral or the CPU. There are two I/O ports:

- Port A consists of GPIO0-GPIO31
- Port B consists of GPIO32-GPIO59

The analog signals on this device are multiplexed with digital inputs. Some of these analog IO (AIO) pins do not have digital output capability. Others of these are analog pins capable of full digital input and output capability (AGPIO). Analog pins multiplexed with digital inputs (and outputs) are assigned to a single port:

- Port H consists of GPIO224-GPIO253

Note

By default, all analog pins with digital input support shall come up in analog mode. To turn ON the digital input functionality, GPHAMSEL register needs to be configured.

Figure 10-1 shows the GPIO logic for a single pin.

There are two key features to note in Figure 10-1. The first is that the input and output paths are entirely separate, connecting only at the pin. The second is that peripheral muxing takes place far from the pin. As a result, it is always possible for both CPUs and CLAs to read the physical state of the pin independent of CPU mastering and peripheral muxing. Likewise, external interrupts can be generated from peripheral activity. All pin options such as input qualification and open-drain output are valid for all peripherals. Table 10-1 provides details of GPIO registers accessible by different masters

Table 10-1. GPIO access by different masters

Register Type	CPU	CLA	DMA	HIC	Comments
GPIO_CTRL_REGS	Yes	No	No	No	
GPIO_DATA_REGS	Yes	Yes	No	No	Based on GPxCSEL configuration
GPIO_DATA_READ_REGS	Yes	Yes	No	Yes	

Note

JTAG uses a different signal path that does not support inversion or qualification.

GPIO18/X2 and GPIO19/X1 have different timings due to the load placed on them by the oscillator circuit. For information on using GPIO18/X2 and GPIO19/X1 as GPIOs, see the device data sheet and the Clocking section of this document.

If digital signals with sharp edges (high dv/dt) are connected to the AIOs or AGPIOs, cross-talk can occur with adjacent analog signals. Therefore, you should limit the edge rate of signals connected to AIOs or AGPIOs if adjacent channels are being used for analog functions.

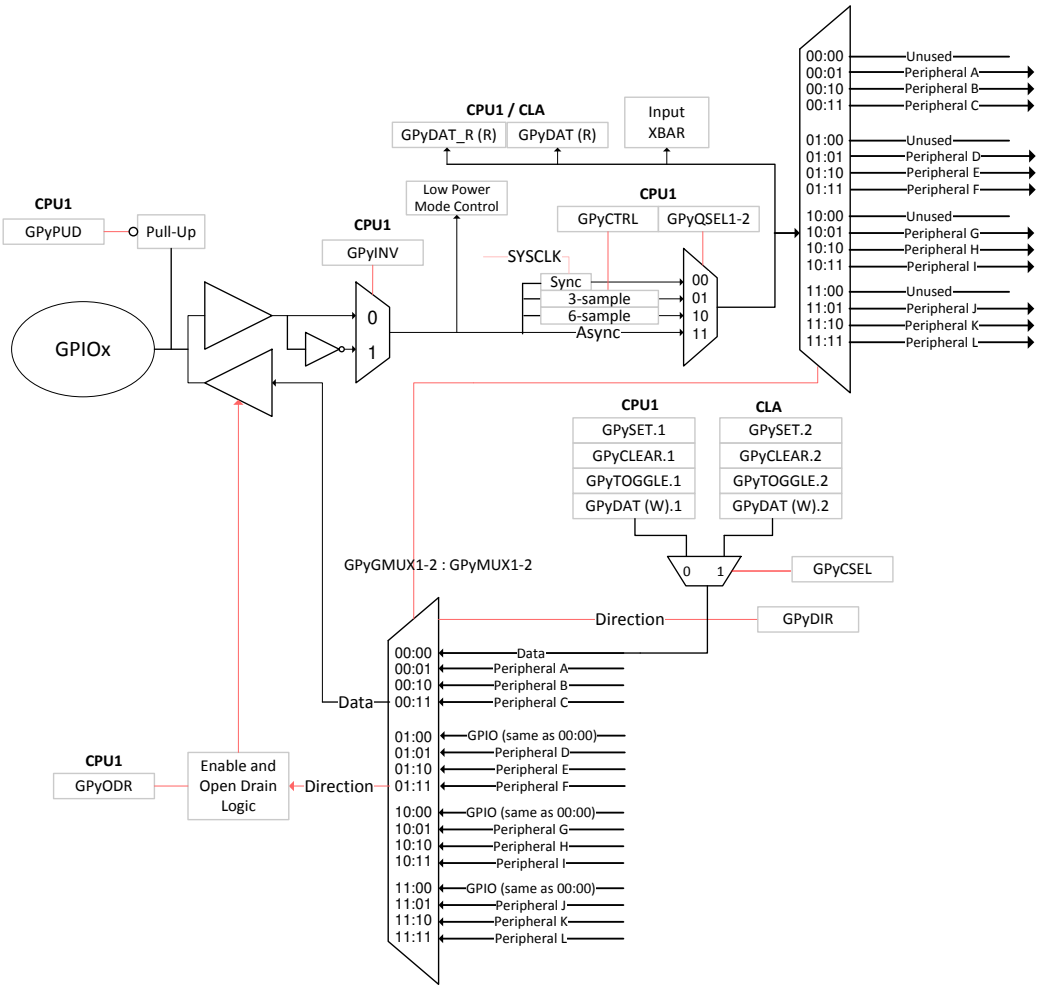


Figure 10-1. GPIO Logic for a Single Pin

10.2 GPIO Related Collateral

Foundational Materials

- C2000 Academy - System Initialization

Getting Started Materials

- *How to Maximize GPIO Usage in C2000 Devices Application Report*
- [FAQ] C2000 GPIO FAQ

10.3 Configuration Overview

I/O pin configuration consists of several steps:

1. **Plan the device pin-out**
   Make a list of all required peripherals for the application. Using the peripheral mux information in the device data manual, choose which GPIOs to use for the peripheral signals. Decide which of the remaining GPIOs to use as inputs and outputs for each CPU and CLA.

   Once the peripheral muxing has been chosen, it should be implemented by writing the appropriate values to the GPyMUX1/2 and GPyGMUX1/2 registers. When changing the GPyGMUX value for a pin, always set the corresponding GPyMUX bits to zero first to avoid glitching in the muxes. By default, all pins are general-purpose I/Os, not peripheral signals.

2. **(Optional) Enable internal pullup resistors**
   To enable or disable the pullup resistors, write to the appropriate bits in the GPIO pullup disable registers (GPyPUD). All pullups are disabled by default. Pullups can be used to keep input pins in a known state when there is no external signal driving them.
3. **Select input qualification**
   If the pin will be used as an input, specify the required input qualification, if any. The input qualification sampling period is selected in the GPyCTRL registers, while the type of qualification is selected in the GPyQSEL1 and GPyQSEL2 registers. By default, all qualification is synchronous with a sampling period equal to PLLSYSCLK. For an explanation of input qualification, see Section 10.5.
4. **Select the direction of any general-purpose I/O pins**
   For each pin configured as a GPIO, specify the direction of the pin as either input or output using the GPyDIR registers. By default, all GPIO pins are inputs. Before changing a pin to an output, load the output latch with the value to be driven by writing that value to the GPySET, GPyCLEAR, or GPyDAT registers. Once the latch is loaded, write to GPyDIR to change the pin direction. By default, all output latches are zero.

   The GPyDAT_R register can be used to read what value was written to the GPyDAT register.
5. **Select low-power mode wake-up sources**
   GPIOs 0-63 can be used to wake the system up from low power modes. To select one or more GPIOs for wake-up, write to the appropriate bits in the GPIOLPMSEL0 and GPIOLPMSEL1 registers. These registers are part of the CPU system register space. For more information on low-power modes and GPIO wake-up, see the Low Power Modes section in the *System Control and Interrupts* chapter.
6. **Select external interrupt sources**
   Configuring external interrupts is a two-step process. First, the interrupts themselves must be enabled and their polarity must be configured via the XINTnCR registers. Second, the XINT1-5 GPIO pins must be set by selecting the sources for Input X-BAR signals 4, 5, 6, 13, and 14, respectively. For more information on the Input X-BAR architecture, see the *Crossbar (X-BAR)* chapter.

10.4 Digital General-Purpose I/O Control

The values on the pins that are configured as GPIO can be changed by using the following registers.

- **GPyDAT Registers**

  Each I/O port has one data register. Each bit in the data register corresponds to one GPIO pin. No matter how the pin is configured (GPIO or peripheral function), the corresponding bit in the data register reflects the current state of the pin after qualification. Writing to the GPyDAT register clears or sets the corresponding output latch and if the pin is enabled as a general purpose output (GPIO output) the pin will also be driven either low or high. If the pin is not configured as a GPIO output then the value will be latched, but the pin will not be driven. Only if the pin is later configured as a GPIO output, will the latched value be driven onto the pin.

  When using the GPyDAT register to change the level of an output pin, you should be cautious not to accidentally change the level of another pin. For example, if you mean to change the output latch level of GPIOA1 by writing to the GPADAT register bit 0 using a read-modify-write instruction, a problem can occur if another I/O port A signal changes level between the read and the write stage of the instruction. Following is an analysis of why this happens:

  The GPyDAT registers reflect the state of the pin, not the latch. This means the register reflects the actual pin value. However, there is a lag between when the register is written to when the new pin value is reflected back in the register. This may pose a problem when this register is used in subsequent program statements to alter the state of GPIO pins. An example is shown below where two program statements attempt to drive two different GPIO pins that are currently low to a high state.

If Read-Modify-Write operations are used on the GPyDAT registers, because of the delay between the output and the input of the first instruction (I1), the second instruction (I2) will read the old value and write it back.

```
GpioDataRegs.GPADAT.bit.GPIO1 = 1;  //I1 performs read-modify-write of GPADAT
GpioDataRegs.GPADAT.bit.GPIO2 = 1;  //I2 also a read-modify-write of GPADAT
                                    //GPADAT gets the old value of GPIO1 due to the delay
```

The second instruction will wait for the first to finish its write due to the write-followed-by-read protection on this peripheral frame. There will be some lag, however, between the write of (I1) and the GPyDAT bit reflecting the new value (1) on the pin. During this lag, the second instruction will read the old value of GPIO1 (0) and write it back along with the new value of GPIO2 (1). Therefore, GPIO1 pin stays low.

One solution is to put some NOPs between instructions. A better solution is to use the GPySET/GPyCLEAR/GPyTOGGLE registers instead of the GPyDAT registers. These registers always read back a 0 and writes of 0 have no effect. Only bits that need to be changed can be specified without disturbing any other bit(s) that are currently in the process of changing.

- **GPyDAT_R Registers**

  The GPyDAT_R registers are read only registers which return the value written to the GPyDAT registers instead of pin status. Writes to these registers have no effect.
- **GPySET Registers**

  The set registers are used to drive specified GPIO pins high without disturbing other pins. Each I/O port has one set register and each bit corresponds to one GPIO pin. The set registers always read back 0. If the corresponding pin is configured as an output, then writing a 1 to that bit in the set register will set the output latch high and the corresponding pin will be driven high. If the pin is not configured as a GPIO output, then the value will be latched but the pin will not be driven. Only if the pin is later configured as a GPIO output will the latched value be driven onto the pin. Writing a 0 to any bit in the set registers has no effect.
- **GPyCLEAR Registers**

  The clear registers are used to drive specified GPIO pins low without disturbing other pins. Each I/O port has one clear register. The clear registers always read back 0. If the corresponding pin is configured as a general purpose output, then writing a 1 to the corresponding bit in the clear register will clear the output latch and the pin will be driven low. If the pin is not configured as a GPIO output, then the value will be latched but the pin will not be driven. Only if the pin is later configured as a GPIO output will the latched value be driven onto the pin. Writing a 0 to any bit in the clear registers has no effect.
- **GPyTOGGLE Registers**

  The toggle registers are used to drive specified GPIO pins to the opposite level without disturbing other pins. Each I/O port has one toggle register. The toggle registers always read back 0. If the corresponding pin is configured as an output, then writing a 1 to that bit in the toggle register flips the output latch and pulls the corresponding pin in the opposite direction. That is, if the output pin is driven low, then writing a 1 to the corresponding bit in the toggle register will pull the pin high. Likewise, if the output pin is high, then writing a 1 to the corresponding bit in the toggle register will pull the pin low. If the pin is not configured as a GPIO output, then the value will be latched but the pin will not be driven. Only if the pin is later configured as a GPIO output will the latched value be driven onto the pin. Writing a 0 to any bit in the toggle registers has no effect.

10.5 Input Qualification

The input qualification scheme has been designed to be very flexible. You can select the type of input qualification for each GPIO pin by configuring the GPyQSEL1 and GPyQSEL2 registers. In the case of a GPIO input pin, the qualification can be specified as only synchronize to SYSCLKOUT or qualification by a sampling window. For pins that are configured as peripheral inputs, the input can also be asynchronous in addition to synchronized to SYSCLKOUT or qualified by a sampling window. The remainder of this section describes the options available.

10.5.1 No Synchronization (Asynchronous Input)

This mode is used for peripherals where input synchronization is not required or the peripheral itself performs the synchronization. Examples include communication ports McBSP, SCI, SPI, and I^2C. In addition, it may be desirable to have the ePWM trip zone ($\overline{TZn}$) signals function independent of the presence of SYSCLKOUT.

The asynchronous option is not valid if the pin is used as a general purpose digital input pin (GPIO). If the pin is configured as a GPIO input and the asynchronous option is selected then the qualification defaults to synchronization to SYSCLKOUT as described in Section 10.5.2.

Note

Using input synchronization when the peripheral itself performs the synchronization may cause unexpected results. The user should ensure that the GPIO pin is configured for asynchronous in this case.

10.5.2 Synchronization to SYSCLKOUT Only

This is the default qualification mode of all the pins at reset. In this mode, the input signal is only synchronized to the system clock (SYSCLKOUT). Because the incoming signal is asynchronous, it can take up to a SYSCLKOUT period of delay in order for the input to the MCU to be changed. No further qualification is performed on the signal.

10.5.3 Qualification Using a Sampling Window

In this mode, the signal is first synchronized to the system clock (SYSCLKOUT) and then qualified by a specified number of cycles before the input is allowed to change. Figure 10-2 and Figure 10-3 show how the input qualification is performed to eliminate unwanted noise. Two parameters are specified by the user for this type of qualification: 1) the sampling period, or how often the signal is sampled, and 2) the number of samples to be taken.

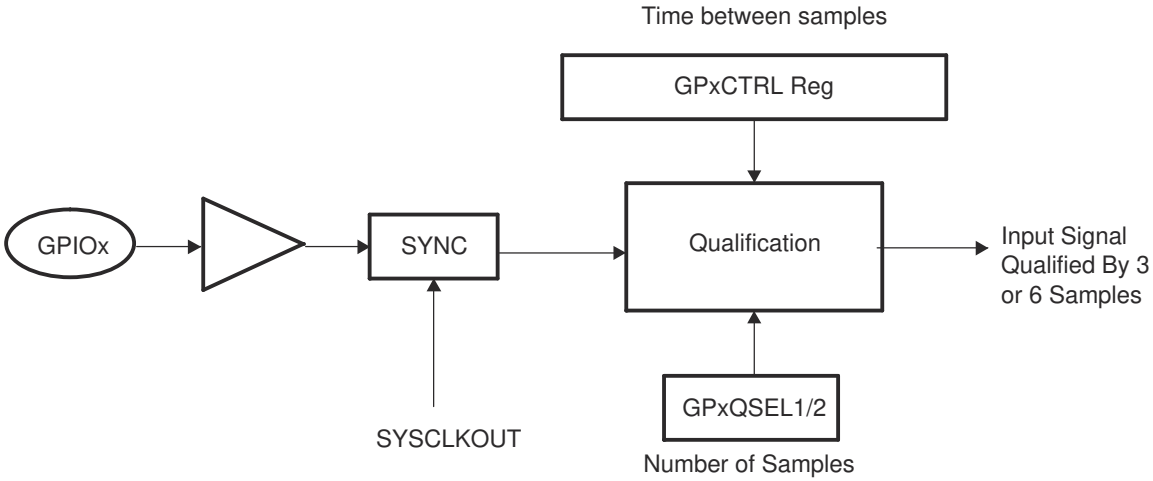


Figure 10-2. Input Qualification Using a Sampling Window

Time between samples (sampling period):

To qualify the signal, the input signal is sampled at a regular period. The sampling period is specified by the user and determines the time duration between samples, or how often the signal will be sampled, relative to the CPU clock (SYSCLKOUT).

The sampling period is specified by the qualification period (QUALPRDn) bits in the GPxCTRL register. The sampling period is configurable in groups of 8 input signals. For example, GPIO0 to GPIO7 use GPACTRL[QUALPRD0] setting and GPIO8 to GPIO15 use GPACTRL[QUALPRD1]. Table 10-2 and Table 10-3 show the relationship between the sampling period or sampling frequency and the GPxCTRL[QUALPRDn] setting.

Table 10-2. Sampling Period

	Sampling Period
If GPxCTRL[QUALPRDn] = 0	1 × $T_{SYSCLKOUT}$
If GPxCTRL[QUALPRDn] ≠ 0	2 × GPxCTRL[QUALPRDn] × $T_{SYSCLKOUT}$
	Where $T_{SYSCLKOUT}$ is the period in time of SYSCLKOUT

Table 10-3. Sampling Frequency

	Sampling Frequency
If GPxCTRL[QUALPRDn] = 0	$f_{SYSCLKOUT}$
If GPxCTRL[QUALPRDn] ≠ 0	$f_{SYSCLKOUT}$ × 1 ÷ (2 × GPxCTRL[QUALPRDn])
	Where $f_{SYSCLKOUT}$ is the frequency of SYSCLKOUT

From these equations, the minimum and maximum time between samples can be calculated for a given SYSCLKOUT frequency:

Example: Maximum Sampling Frequency:

If GPxCTRL[QUALPRDn] = 0

then the sampling frequency is $f_{SYSCLKOUT}$

If, for example, $f_{SYSCLKOUT}$ = 60 MHz

then the signal will be sampled at 60 MHz or one sample every 16.67 ns.

Example: Minimum Sampling Frequency:

If GPxCTRL[QUALPRDn] = 0xFF (255)

then the sampling frequency is $f_{SYSCLKOUT}$ × 1 ÷ (2 × GPxCTRL[QUALPRDn])

If, for example, $f_{SYSCLKOUT}$ = 60 MHz

then the signal will be sampled at 6 0 MHz × 1 ÷ (2 × 255) or one sample every 8.5 µs.

Number of samples:

The number of times the signal is sampled is either three samples or six samples as specified in the qualification selection (GPAQSEL1, GPAQSEL2, GPBQSEL1, and GPBQSEL2) registers. When three or six consecutive cycles are the same, then the input change will be passed through to the MCU.

Total Sampling Window Width:

The sampling window is the time during which the input signal will be sampled as shown in Figure 10-3. By using the equation for the sampling period along with the number of samples to be taken, the total width of the window can be determined.

For the input qualifier to detect a change in the input, the level of the signal must be stable for the duration of the sampling window width or longer.

The number of sampling periods within the window is always one less than the number of samples taken. For a three-sample window, the sampling window width is two sampling periods wide where the sampling period is

defined in Table 10-2. Likewise, for a six-sample window, the sampling window width is five sampling periods wide. Table 10-4 and Table 10-5 show the calculations that can be used to determine the total sampling window width based on GPxCTRL[QUALPRDn] and the number of samples taken.

Table 10-4. Case 1: Three-Sample Sampling Window Width

	Total Sampling Window Width
If GPxCTRL[QUALPRDn] = 0	2 × $T_{SYSCLKOUT}$
If GPxCTRL[QUALPRDn] ≠ 0	2 × 2 × GPxCTRL[QUALPRDn] × $T_{SYSCLKOUT}$
	Where $T_{SYSCLKOUT}$ is the period in time of SYSCLKOUT

Table 10-5. Case 2: Six-Sample Sampling Window Width

	Total Sampling Window Width
If GPxCTRL[QUALPRDn] = 0	5 × $T_{SYSCLKOUT}$
If GPxCTRL[QUALPRDn] ≠ 0	5 × 2 × GPxCTRL[QUALPRDn] × $T_{SYSCLKOUT}$
	Where $T_{SYSCLKOUT}$ is the period in time of SYSCLKOUT

Note

The external signal change is asynchronous with respect to both the sampling period and SYSCLKOUT. Due to the asynchronous nature of the external signal, the input should be held stable for a time greater than the sampling window width to make sure the logic detects a change in the signal. The extra time required can be up to an additional sampling period + $T_{SYSCLKOUT}$.

The required duration for an input signal to be stable for the qualification logic to detect a change is described in the data sheet.

Example Qualification Window:

For the example shown in Figure 10-3, the input qualification has been configured as follows:

- GPxQSEL1/2 = 1,0. This indicates a six-sample qualification.
- GPxCTRL[QUALPRDn] = 1. The sampling period is $t_w(SP)$ = 2 × GPxCTRL[QUALPRDn] × $T_{SYSCLKOUT}$ = 2 x $T_{SYSCLKOUT}$.

This configuration results in the following:

- The width of the sampling window is:

  $t_w(IQSW) = 5 \times t_w(SP) = 5 \times 2 \times GPxCTRL[QUALPRDn] \times T_{SYSCLKOUT} = 5 \times 2 \times T_{SYSCLKOUT}$

- If, for example, $T_{SYSCLKOUT}$ = 16.67 ns, then the duration of the sampling window is:

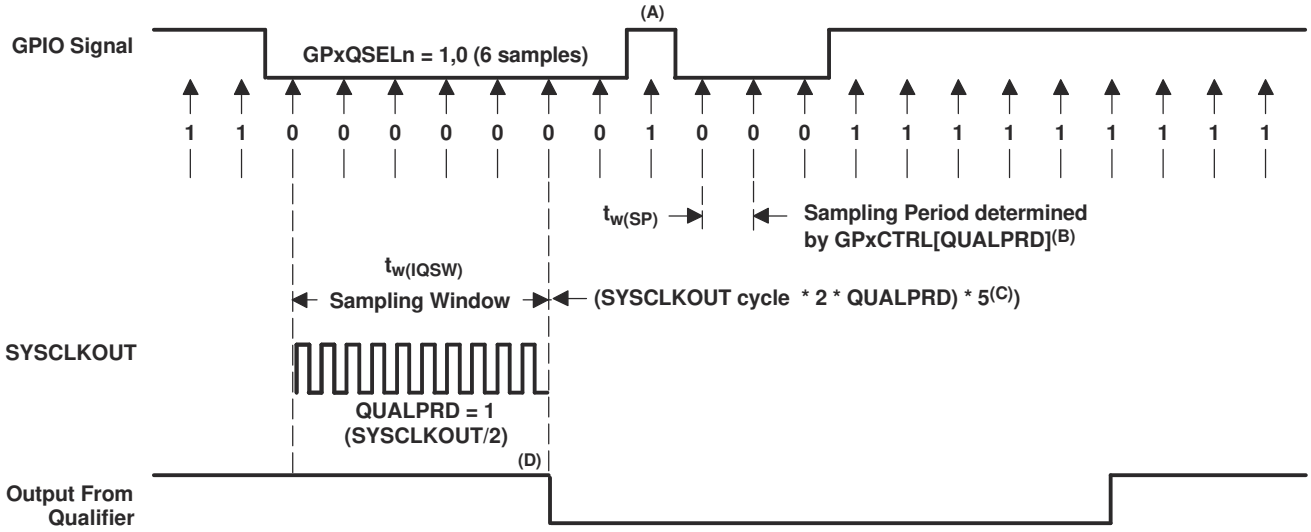
  Sampling period, $t_w(SP) = 2 \times T_{SYSCLKOUT} = 2 \times 16.67$ ns = 33.3 ns

  Sampling window, $t_w(IQSW) = 5 \times t_w(SP) = 5 \times 33.3$ ns =166.7 ns

- To account for the asynchronous nature of the input relative to the sampling period and SYSCLKOUT, up to an additional sampling period and SYSCLK period may be required to detect a change in the input signal. For this example:

  $t_w(IQSW) + t_w(SP) + T_{SYSCLKOUT}$ = 166.7 ns + 33.3 ns + 16.67 ns = 216.7 ns

- In Figure 10-3, the glitch (A) is shorter then the qualification window and will be ignored by the input qualifier.



A. This glitch will be ignored by the input qualifier. The QUALPRD bit field specifies the qualification sampling period. It can vary from 00 to 0xFF. If QUALPRD = 00, then the sampling period is 1 SYSCLKOUT cycle. For any other value “n”, the qualification sampling period in 2n SYSCLKOUT cycles (i.e., at every 2n SYSCLKOUT cycles, the GPIO pin will be sampled).
B. The qualification period selected via the GPxCTRL register applies to groups of 8 GPIO pins.
C. The qualification block can take either three or six samples. The GPxQSELn Register selects which sample mode is used.
D. In the example shown, for the qualifier to detect the change, the input should be stable for 10 SYSCLKOUT cycles or greater. In other words, the inputs should be stable for (5 x QUALPRD x 2) SYSCLKOUT cycles. That would ensure 5 sampling periods for detection to occur. Since external signals are driven asynchronously, an 13-SYSCLKOUT-wide pulse ensures reliable recognition.

Figure 10-3. Input Qualifier Clock Cycles

10.6 GPIO and Peripheral Muxing

10.6.1 GPIO Muxing

Up to twelve different peripheral functions are multiplexed to each pin along with a general-purpose input/output (GPIO) function. This allows you to choose the peripheral mix and pinout that works best for your particular application. Refer to Table 10-6 for muxing combinations and definitions.

Table 10-6. GPIO Muxed Pins

0, 4, 8, 12	1	2	3	5	6	7	9	10	11	13	14	15	ALT
GPIO0	EPWM1_A				I2CA_SDA	SPIA_STE	FSIRXA_CLK	MCAN_RX	CLB_OUTPUTX BAR8	EQEP1_INDEX	HIC_D7	HIC_BASESEL1	
GPIO1	EPWM1_B				I2CA_SCL	SPIA_SOMI		MCAN_TX	CLB_OUTPUTX BAR7	HIC_A2	FSITXA_TDM_ D1	HIC_D10	
GPIO2	EPWM2_A			OUTPUTXBAR1	PMBUSA_SDA	SPIA_SIMO	SCIA_TX	FSIRXA_D1	I2CB_SDA	HIC_A1	CANA_TX	HIC_D9	
GPIO3	EPWM2_B	OUTPUTXBAR2		OUTPUTXBAR2	PMBUSA_SCL	SPIA_CLK	SCIA_RX	FSIRXA_D0	I2CB_SCL	HIC_NOE	CANA_RX	HIC_D4	
GPIO4	EPWM3_A		MCAN_TX	OUTPUTXBAR3	CANA_TX	SPIB_CLK	EQEP2_ STROBE	FSIRXA_CLK	CLB_OUTPUTX BAR6	HIC_BASESEL2		HIC_NWE	
GPIO5	EPWM3_B		OUTPUTXBAR3	MCAN_RX	CANA_RX	SPIA_STE	FSITXA_D1	CLB_OUTPUTX BAR5		HIC_A7	HIC_D4	HIC_D15	
GPIO6	EPWM4_A	OUTPUTXBAR4	SYNCOUT	EQEP1_A		SPIB_SOMI	FSITXA_D0		FSITXA_D1	HIC_NBE1	CLB_OUTPUTX BAR8	HIC_D14	
GPIO7	EPWM4_B		OUTPUTXBAR5	EQEP1_B		SPIB_SIMO	FSITXA_CLK	CLB_OUTPUTX BAR2		HIC_A6		HIC_D14	
GPIO8	EPWM5_A		ADCSOCAO	EQEP1_ STROBE	SCIA_TX	SPIA_SIMO	I2CA_SCL	FSITXA_D1	CLB_OUTPUTX BAR5	HIC_A0	FSITXA_TDM_ CLK	HIC_D8	
GPIO9	EPWM5_B	SCIB_TX	OUTPUTXBAR6	EQEP1_INDEX	SCIA_RX	SPIA_CLK		FSITXA_D0	LINB_RX	HIC_BASESEL0	I2CB_SCL	HIC_NRDY	
GPIO10	EPWM6_A		ADCSOCBO	EQEP1_A	SCIB_TX	SPIA_SOMI	I2CA_SDA	FSITXA_CLK	LINB_TX	HIC_NWE	FSITXA_TDM_ D0	CLB_OUTPUTX BAR4	
GPIO11	EPWM6_B		OUTPUTXBAR7	EQEP1_B	SCIB_RX	SPIA_STE	FSIRXA_D1	LINB_RX	EQEP2_A	SPIA_SIMO	HIC_D6	HIC_NBE0	
GPIO12	EPWM7_A		MCAN_RX	EQEP1_ STROBE	SCIB_TX	PMBUSA_CTL	FSIRXA_D0	LINB_TX	SPIA_CLK	CANA_RX	HIC_D13	HIC_INT	
GPIO13	EPWM7_B		MCAN_TX	EQEP1_INDEX	SCIB_RX	PMBUSA_ ALERT	FSIRXA_CLK	LINB_RX	SPIA_SOMI	CANA_TX	HIC_D11	HIC_D5	
GPIO14	EPWM8_A	SCIB_TX		I2CB_SDA	OUTPUTXBAR3	PMBUSA_SDA	SPIB_CLK	EQEP2_A	LINB_TX	EPWM3_A	CLB_OUTPUTX BAR7	HIC_D15	
GPIO15	EPWM8_B	SCIB_RX		I2CB_SCL	OUTPUTXBAR4	PMBUSA_SCL	SPIB_STE	EQEP2_B	LINB_RX	EPWM3_B	CLB_OUTPUTX BAR6	HIC_D12	
GPIO16	SPIA_SIMO		OUTPUTXBAR7	EPWM5_A	SCIA_TX	SD1_D1	EQEP1_ STROBE	PMBUSA_SCL	XCLKOUT	EQEP2_B	SPIB_SOMI	HIC_D1	
GPIO17	SPIA_SOMI		OUTPUTXBAR8	EPWM5_B	SCIA_RX	SD1_C1	EQEP1_INDEX	PMBUSA_SDA	CANA_TX			HIC_D2	
GPIO18	SPIA_CLK	SCIB_TX	CANA_RX	EPWM6_A	I2CA_SCL	SD1_D2	EQEP2_A	PMBUSA_CTL	XCLKOUT	LINB_TX	FSITXA_TDM_ CLK	HIC_INT	X2
GPIO19	SPIA_STE	SCIB_RX	CANA_TX	EPWM6_B	I2CA_SDA	SD1_C2	EQEP2_B	PMBUSA_ ALERT	CLB_OUTPUTX BAR1	LINB_RX	FSITXA_TDM_ D0	HIC_NBE0	X1
GPIO20	EQEP1_A				SPIB_SIMO	SD1_D3	MCAN_TX						
GPIO21	EQEP1_B				SPIB_SOMI	SD1_C3	MCAN_RX						
GPIO22	EQEP1_ STROBE		SCIB_TX		SPIB_CLK	SD1_D4	LINA_TX	CLB_OUTPUTX BAR1	LINB_TX	HIC_A5	EPWM4_A	HIC_D13	
GPIO23	EQEP1_INDEX		SCIB_RX		SPIB_STE	SD1_C4	LINA_RX	CLB_OUTPUTX BAR3	LINB_RX	HIC_A3	EPWM4_B	HIC_D11	

Table 10-6. GPIO Muxed Pins (continued)

0, 4, 8, 12	1	2	3	5	6	7	9	10	11	13	14	15	ALT
GPIO24	OUTPUTXBAR1	EQEP2_A		EPWM8_A	SPIB_SIMO	SD2_D1	LINB_TX	PMBUSA_SCL	SCIA_TX	ERRORSTS		HIC_D3	
GPIO25	OUTPUTXBAR2	EQEP2_B		EQEP1_A	SPIB_SOMI	SD2_C1	FSITXA_D1	PMBUSA_SDA	SCIA_RX		HIC_BASESEL0		
GPIO26	OUTPUTXBAR3	EQEP2_INDEX		OUTPUTXBAR3	SPIB_CLK	SD2_D2	FSITXA_D0	PMBUSA_CTL	I2CA_SDA		HIC_D0	HIC_A1	
GPIO27	OUTPUTXBAR4	EQEP2_ STROBE		OUTPUTXBAR4	SPIB_STE	SD2_C2	FSITXA_CLK	PMBUSA_ ALERT	I2CA_SCL		HIC_D1	HIC_A4	
GPIO28	SCIA_RX		EPWM7_A	OUTPUTXBAR5	EQEP1_A	SD2_D3	EQEP2_ STROBE	LINA_TX	SPIB_CLK	ERRORSTS	I2CB_SDA	HIC_NOE	
GPIO29	SCIA_TX		EPWM7_B	OUTPUTXBAR6	EQEP1_B	SD2_C3	EQEP2_INDEX	LINA_RX	SPIB_STE	ERRORSTS	I2CB_SCL	HIC_NCS	AUX CLKIN
GPIO30	CANA_RX		SPIB_SIMO	OUTPUTXBAR7	EQEP1_ STROBE	SD2_D4	FSIRXA_CLK	MCAN_RX	EPWM1_A		HIC_D8		
GPIO31	CANA_TX		SPIB_SOMI	OUTPUTXBAR8	EQEP1_INDEX	SD2_C4	FSIRXA_D1	MCAN_TX	EPWM1_B		HIC_D10		
GPIO32	I2CA_SDA		SPIB_CLK	EPWM8_B	LINA_TX	SD1_D2	FSIRXA_D0	CANA_TX	PMBUSA_SDA	ADCSOCBO		HIC_INT	
GPIO33	I2CA_SCL		SPIB_STE	OUTPUTXBAR4	LINA_RX	SD1_C2	FSIRXA_CLK	CANA_RX	EQEP2_B	ADCSOCAO	SD1_C1	HIC_D0	
GPIO34	OUTPUTXBAR1				PMBUSA_SDA					HIC_NBE1	I2CB_SDA	HIC_D9	
GPIO35	SCIA_RX		I2CA_SDA	CANA_RX	PMBUSA_SCL	LINA_RX	EQEP1_A	PMBUSA_CTL	EPWM5_B	SD2_C1	HIC_NWE	TDI	
GPIO37	OUTPUTXBAR2		I2CA_SCL	SCIA_TX	CANA_TX	LINA_TX	EQEP1_B	PMBUSA_ ALERT			HIC_NRDY	TDO	
GPIO39					MCAN_RX	FSIRXA_CLK	EQEP2_INDEX		CLB_OUTPUTX BAR2	SYNCOUT	EQEP1_INDEX	HIC_D7	
GPIO40	SPIB_SIMO			EPWM2_B	PMBUSA_SDA	FSIRXA_D0	SCIB_TX	EQEP1_A	LINB_TX		HIC_NBE1	HIC_D5	
GPIO41				EPWM2_A	PMBUSA_SCL	FSIRXA_D1	SCIB_RX	EQEP1_B	LINB_RX	HIC_A4	SPIB_SOMI	HIC_D12	
GPIO42		LINA_RX	OUTPUTXBAR5	PMBUSA_CTL	I2CA_SDA			EQEP1_ STROBE	CLB_OUTPUTX BAR3		HIC_D2	HIC_A6	
GPIO43			OUTPUTXBAR6	PMBUSA_ ALERT	I2CA_SCL		PMBUSA_ ALERT	EQEP1_INDEX	CLB_OUTPUTX BAR4	SD2_D3	HIC_D3	HIC_A7	
GPIO44			OUTPUTXBAR7	EQEP1_A	PMBUSA_SDA	FSITXA_CLK	PMBUSA_CTL	CLB_OUTPUTX BAR3	FSIRXA_D0	HIC_D7	LINB_TX	HIC_D5	
GPIO45			OUTPUTXBAR8			FSITXA_D0	PMBUSA_ ALERT	CLB_OUTPUTX BAR4		SD2_C3		HIC_D6	
GPIO46			LINA_TX	MCAN_TX		FSITXA_D1	PMBUSA_SDA			SD2_C4		HIC_NWE	
GPIO47			LINA_RX	MCAN_RX		CLB_OUTPUTX BAR2	PMBUSA_SCL			SD2_D4	FSITXA_TDM_ CLK	HIC_A6	
GPIO48	OUTPUTXBAR3		CANA_TX		SCIA_TX	SD1_D1	PMBUSA_SDA					HIC_A7	
GPIO49	OUTPUTXBAR4		CANA_RX		SCIA_RX	SD1_C1	LINA_RX			SD2_D1	FSITXA_D0	HIC_D2	
GPIO50	EQEP1_A			MCAN_TX	SPIB_SIMO	SD1_D2	I2CB_SDA			SD2_D2	FSITXA_D1	HIC_D3	
GPIO51	EQEP1_B			MCAN_RX	SPIB_SOMI	SD1_C2	I2CB_SCL			SD2_D3	FSITXA_CLK	HIC_D6	
GPIO52	EQEP1_ STROBE			CLB_OUTPUTX BAR5	SPIB_CLK	SD1_D3	SYNCOUT			SD2_D4	FSIRXA_D0	HIC_NWE	
GPIO53	EQEP1_INDEX			CLB_OUTPUTX BAR6	SPIB_STE	SD1_C3	ADCSOCAO	CANA_RX		SD1_C1	FSIRXA_D1		
GPIO54	SPIA_SIMO			EQEP2_A	OUTPUTXBAR2	SD1_D4	ADCSOCBO	LINB_TX		SD1_C2	FSIRXA_CLK	FSITXA_TDM_ D1	
GPIO55	SPIA_SOMI			EQEP2_B	OUTPUTXBAR3	SD1_C4	ERRORSTS	LINB_RX		SD1_C3		HIC_A0	

Table 10-6. GPIO Muxed Pins (continued)

0, 4, 8, 12	1	2	3	5	6	7	9	10	11	13	14	15	ALT
GPIO56	SPIA_CLK	CLB_OUTPUTXBAR7	MCAN_TX	EQEP2_STROBE	SCIB_TX	SD2_D1	SPIB_SIMO	I2CA_SDA	EQEP1_A	SD1_C4	FSIRXA_D1	HIC_D6	
GPIO57	SPIA_STE	CLB_OUTPUTXBAR8	MCAN_RX	EQEP2_INDEX	SCIB_RX	SD2_C1	SPIB_SOMI	I2CA_SCL	EQEP1_B		FSIRXA_CLK	HIC_D4	
GPIO58				OUTPUTXBAR1	SPIB_CLK	SD2_D2	LINA_TX	CANA_TX	EQEP1_STROBE	SD2_C2	FSIRXA_D0	HIC_NRDY	
GPIO59				OUTPUTXBAR2	SPIB_STE	SD2_C2	LINA_RX	CANA_RX	EQEP1_INDEX	SD2_C3	FSITXA_TDM_D1		
GPIO60			MCAN_TX	OUTPUTXBAR3	SPIB_SIMO	SD2_D3				SD2_C4		HIC_A0	
GPIO61			MCAN_RX	OUTPUTXBAR4	SPIB_SOMI	SD2_C3					CANA_RX		
AIO228		SD2_C1										HIC_A0	
AIO226		SD2_D4										HIC_A1	
AIO242		SD2_D2										HIC_A2	
AIO224		SD2_D3										HIC_A3	
AIO233		SD2_D1										HIC_A4	
AIO229													
AIO239		SD1_D1										HIC_A5	
AIO237		SD1_D2										HIC_A6	
AIO244		SD1_D3										HIC_A7	
AIO232		SD1_D4										HIC_BASESEL0	
AIO231		SD1_C1										HIC_BASESEL1	
AIO238		SD2_C3										HIC_NCS	
AIO248													
AIO251													
AIO245		SD1_C2										HIC_NOE	
AIO252		SD2_C4											
AIO241		SD2_C1										HIC_NBE1	
AIO249													
AIO225		SD2_C2										HIC_NWE	
AIO240		SD2_C1										HIC_NBE1	
AIO227		SD1_C3										HIC_NBE0	
AIO236													
AIO230		SD1_C4										HIC_BASESEL2	
AIO253													
AIO247													

10.6.2 Peripheral Muxing

For example, multiplexing for the GPIO6 pin is controlled by writing to GPAGMUX[13:12] and GPAMUX[13:12]. By writing to these bits, GPIO6 is configured as either a general-purpose digital I/O or one of four different peripheral functions. The options are shown in Table 10-7.

Table 10-7. GPIO and Peripheral Muxing

GPAGMUX1[13:12]	GPAMUX1[13:12]	Pin Functionality
00	00	GPIO6
00	01	EPWM4_A
00	10	OUTPUTXBAR4
00	11	SYNCOUT
01	00	GPIO6
01	01	EQEP1_A
01	10	
01	11	SPIBSOMI
10	00	GPIO6
10	01	FSITXAD0
10	10	
10	11	FSITXAD1
11	00	GPIO6
11	01	HIC_NBE1
11	10	CLB_OUTPUTXBAR8
11	11	HIC_D14

The devices have different multiplexing schemes. If a peripheral is not available on a particular device, that mux selection is reserved on that device and should not be used.

CAUTION

If you select a reserved GPIO mux configuration that is not mapped to either a peripheral or GPIO mode, the state of the pin will be undefined and the pin may be driven. Unimplemented configurations are for future expansion and must not be selected. In the device mux table (see datasheet), these options are indicated as Reserved or left blank.

Some peripherals can be assigned to more than one pin by way of the mux registers. For example, OUTPUTXBAR1 can be assigned to GPIOs 2, 24, 34, or 58, depending on individual system requirements. An example of this is shown in Table 10-8.

If none, or more then one, of the GPIO pins are configured as peripheral input pins, then that GPIO will be set to a hard-wired default value.

Table 10-8. Peripheral Muxing (Multiple Pins Assigned)

GMUX Configuration	MUX Configuration	
Choice 1: GPIO2	GPAGMUX1[5:4]=01	GPAMUX1[5:4]=01
or Choice 2: GPIO24	GPAGMUX2[17:16]=00	GPAMUX2[17:16]=01
or Choice 3: GPIO34	GPBGMUX1[5:4]=00	GPBMUX1[5:4]=01
or Choice 4: GPIO58	GPBGMUX2[21:20]=01	GPBMUX2[21:20]=01

10.7 Internal Pullup Configuration Requirements

On reset, GPIOs are in input mode and have the internal pullups disabled. An un-driven input can float to a mid-rail voltage and cause wasted shoot-through current on the input buffer. The user should always put each GPIO in one of these configurations:

- Input mode and driven on the board by another component to a level above V_{ih} or below V_{il}
- Input mode with GPIO internal pullup enabled
- Output mode

On devices with lesser pin count packages, pull-ups on unbonded GPIOs are by default enabled to prevent floating inputs. The user should take care to avoid disabling these pullups in their application code.

On devices in the 176 PTP packages, the pullups for any internally unbonded GPIO must be enabled to prevent floating inputs. TI has provided functions in controlSUITE/C2000Ware which users can call to enable the pullup on any unbonded GPIO for the package they are using. This function, GPIO_EnabledUnbondedIOPullups(), resides in the (Device)_Sysctrl.c file and is called by default from InitSysCtrl(). The user should take care to avoid disabling these pullups in their application code.

10.8 Software

10.8.1 GPIO Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/gpio

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

10.8.1.1 Device GPIO Setup

FILE: gpio_ex1_setup.c

Configures the device GPIO into two different configurations This code is verbose to illustrate how the GPIO could be setup. In a real application, lines of code can be combined for improved code size and efficiency.

This example only sets-up the GPIO. Nothing is actually done with the pins after setup.

In general:

- All pullup resistors are enabled. For ePWMs this may not be desired.
- Input qual for communication ports (CAN, SPI, SCI, I2C) is asynchronous
- Input qual for Trip pins (TZ) is asynchronous
- Input qual for eCAP and eQEP signals is synch to SYSCLKOUT
- Input qual for some I/O's and __interrupts may have a sampling window

10.8.1.2 Device GPIO Toggle

FILE: gpio_ex2_toggle.c

Configures the device GPIO through the sysconfig file. The GPIO pin is toggled in the infinit loop.

10.8.1.3 Device GPIO Interrupt

FILE: gpio_ex3_interrupt.c

Configures the device GPIOs through the sysconfig file. One GPIO output pin, and one GPIO input pin is configured. The example then configures the GPIO input pin to be the source of an external interrupt which toggles the GPIO output pin.

10.8.1.4 External Interrupt (XINT)

FILE: gpio_ex4_aio_external_interrupt.c

In this example AIO pins are configured as digital inputs. Two other GPIO signals (connected externally to AIO pins) are toggled in software to trigger external interrupt through AIO224 and AIO225 (AIO224 assigned to XINT1 and AIO225 assigned to XINT2). The user is required to externally connect these signals for the program to work properly. Each interrupt is fired in sequence: XINT1 first and then XINT2.

GPIO34 will go high outside of the interrupts and low within the interrupts. This signal can be monitored on a scope.

External Connections

- Connect GPIO30 to AIO224. AIO224 will be assigned to XINT1
- Connect GPIO31 to AIO225. AIO225 will be assigned to XINT2
- GPIO34 can be monitored on an oscilloscope

Watch Variables

- xint1Count for the number of times through XINT1 interrupt
- xint2Count for the number of times through XINT2 interrupt
- loopCount for the number of times through the idle loop

10.8.2 LED Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/led

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

10.8.2.1 LED Blinky Example

FILE: led_ex1_blinky.c

This example demonstrates how to blink a LED.

External Connections

- None.

Watch Variables

- None.

10.9 GPIO Registers

This section describes the General-Purpose Input/Output Registers.

10.9.1 GPIO Base Address Table

Table 10-9. GPIO Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
GpioCtrlRegs	GPIO_CTRL_REGS	GPIOCTRL_BASE	0x0000_7C00	YES	-	-	-	YES
GpioDataRegs	GPIO_DATA_REGS	GPIODATA_BASE	0x0000_7F00	YES	-	-	YES	YES
GpioDataReadRegs	GPIO_DATA_READ_REGS	GPIODATAREAD_BASE	0x0000_7F80	YES	-	YES	YES	YES

10.9.2 GPIO_CTRL_REGS Registers

Table 10-10 lists the memory-mapped registers for the GPIO_CTRL_REGS registers. All register offset addresses not listed in Table 10-10 should be considered as reserved locations and the register contents should not be modified.

Table 10-10. GPIO_CTRL_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	GPACTRL	GPIO A Qualification Sampling Period Control (GPIO0 to 31)	EALLOW	Go
2h	GPAQSEL1	GPIO A Qualifier Select 1 Register (GPIO0 to 15)	EALLOW	Go
4h	GPAQSEL2	GPIO A Qualifier Select 2 Register (GPIO16 to 31)	EALLOW	Go
6h	GPAMUX1	GPIO A Mux 1 Register (GPIO0 to 15)	EALLOW	Go
8h	GPAMUX2	GPIO A Mux 2 Register (GPIO16 to 31)	EALLOW	Go
Ah	GPADIR	GPIO A Direction Register (GPIO0 to 31)	EALLOW	Go
Ch	GPAPUD	GPIO A Pull Up Disable Register (GPIO0 to 31)	EALLOW	Go
10h	GPAINV	GPIO A Input Polarity Invert Registers (GPIO0 to 31)	EALLOW	Go
12h	GPAODR	GPIO A Open Drain Output Register (GPIO0 to GPIO31)	EALLOW	Go
14h	GPAAMSEL	GPIO A Analog Mode Select register (GPIO0 to GPIO31)	EALLOW	Go
20h	GPAGMUX1	GPIO A Peripheral Group Mux (GPIO0 to 15)	EALLOW	Go
22h	GPAGMUX2	GPIO A Peripheral Group Mux (GPIO16 to 31)	EALLOW	Go
28h	GPACSEL1	GPIO A Core Select Register (GPIO0 to 7)	EALLOW	Go
2Ah	GPACSEL2	GPIO A Core Select Register (GPIO8 to 15)	EALLOW	Go
2Ch	GPACSEL3	GPIO A Core Select Register (GPIO16 to 23)	EALLOW	Go
2Eh	GPACSEL4	GPIO A Core Select Register (GPIO24 to 31)	EALLOW	Go
3Ch	GPALOCK	GPIO A Lock Configuration Register (GPIO0 to 31)	EALLOW	Go
3Eh	GPACR	GPIO A Lock Commit Register (GPIO0 to 31)	EALLOW	Go
40h	GPBCTRL	GPIO B Qualification Sampling Period Control (GPIO32 to 63)	EALLOW	Go
42h	GPBQSEL1	GPIO B Qualifier Select 1 Register (GPIO32 to 47)	EALLOW	Go
44h	GPBQSEL2	GPIO B Qualifier Select 2 Register (GPIO48 to 63)	EALLOW	Go
46h	GPBMUX1	GPIO B Mux 1 Register (GPIO32 to 47)	EALLOW	Go
48h	GPBMUX2	GPIO B Mux 2 Register (GPIO48 to 63)	EALLOW	Go
4Ah	GPBDIR	GPIO B Direction Register (GPIO32 to 63)	EALLOW	Go
4Ch	GPBPUD	GPIO B Pull Up Disable Register (GPIO32 to 63)	EALLOW	Go
50h	GPBINV	GPIO B Input Polarity Invert Registers (GPIO32 to 63)	EALLOW	Go
52h	GPBODR	GPIO B Open Drain Output Register (GPIO32 to GPIO63)	EALLOW	Go
60h	GPBGMUX1	GPIO B Peripheral Group Mux (GPIO32 to 47)	EALLOW	Go
62h	GPBGMUX2	GPIO B Peripheral Group Mux (GPIO48 to 63)	EALLOW	Go
68h	GPBCSEL1	GPIO B Core Select Register (GPIO32 to 39)	EALLOW	Go
6Ah	GPBCSEL2	GPIO B Core Select Register (GPIO40 to 47)	EALLOW	Go
6Ch	GPBCSEL3	GPIO B Core Select Register (GPIO48 to 55)	EALLOW	Go
6Eh	GPBCSEL4	GPIO B Core Select Register (GPIO56 to 63)	EALLOW	Go

Table 10-10. GPIO_CTRL_REGS Registers (continued)

Offset	Acronym	Register Name	Write Protection	Section
7Ch	GPBLOCK	GPIO B Lock Configuration Register (GPIO32 to 63)	EALLOW	Go
7Eh	GPBCR	GPIO B Lock Commit Register (GPIO32 to 63)	EALLOW	Go
1C0h	GPHCTRL	GPIO H Qualification Sampling Period Control (GPIO224 to 255)	EALLOW	Go
1C2h	GPHQSEL1	GPIO H Qualifier Select 1 Register (GPIO224 to 239)	EALLOW	Go
1C4h	GPHQSEL2	GPIO H Qualifier Select 2 Register (GPIO240 to 255)	EALLOW	Go
1C6h	GPHMUX1	GPIO H Mux 1 Register (GPIO224 to 239)	EALLOW	Go
1C8h	GPHMUX2	GPIO H Mux 2 Register (GPIO240 to 255)	EALLOW	Go
1D0h	GPHINV	GPIO H Input Polarity Invert Registers (GPIO224 to 255)	EALLOW	Go
1D4h	GPHAMSEL	GPIO H Analog Mode Select register (GPIO224 to GPIO255)	EALLOW	Go
1E0h	GPHGMUX1	GPIO H Peripheral Group Mux (GPIO224 to 239)	EALLOW	Go
1E2h	GPHGMUX2	GPIO H Peripheral Group Mux (GPIO240 to 255)	EALLOW	Go
1E8h	GPHCSEL1	GPIO H Core Select Register (GPIO224 to 231)	EALLOW	Go
1EAh	GPHCSEL2	GPIO H Core Select Register (GPIO232 to 239)	EALLOW	Go
1ECh	GPHCSEL3	GPIO H Core Select Register (GPIO240 to 247)	EALLOW	Go
1EEh	GPHCSEL4	GPIO H Core Select Register (GPIO248 to 255)	EALLOW	Go
1FCh	GPHLOCK	GPIO H Lock Configuration Register (GPIO224 to 255)	EALLOW	Go
1FEh	GPHCR	GPIO H Lock Commit Register (GPIO224 to 255)	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 10-11 shows the codes that are used for access types in this section.

Table 10-11. GPIO_CTRL_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.

Table 10-11. GPIO_CTRL_REGS Access Type Codes (continued)

Access Type	Code	Description
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

10.9.2.1 GPACTRL Register (Offset = 0h) [Reset = 0h]

GPACTRL is shown in Figure 10-4 and described in Table 10-12.

Return to the Summary Table.

GPIO A Qualification Sampling Period Control (GPIO0 to 31)

Figure 10-4. GPACTRL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QUALPRD3								QUALPRD2								QUALPRD1								QUALPRD0							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 10-12. GPACTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	QUALPRD3	R/W	0h	Qualification sampling period for GPIO24 to GPIO31: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: SYSRSn
23-16	QUALPRD2	R/W	0h	Qualification sampling period for GPIO16 to GPIO23: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: SYSRSn
15-8	QUALPRD1	R/W	0h	Qualification sampling period for GPIO8 to GPIO15: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: SYSRSn
7-0	QUALPRD0	R/W	0h	Qualification sampling period for GPIO0 to GPIO7: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: SYSRSn

10.9.2.2 GPAQSEL1 Register (Offset = 2h) [Reset = 0h]

GPAQSEL1 is shown in Figure 10-5 and described in Table 10-13.

Return to the Summary Table.

GPIO A Qualifier Select 1 Register (GPIO0 to 15)
Input qualification type:
0,0 Sync
0,1 Qualification (3 samples)
1,0 Qualification (6 samples)
1,1 Async (no Sync or Qualification)

Figure 10-5. GPAQSEL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO15		GPIO14		GPIO13		GPIO12		GPIO11		GPIO10		GPIO9		GPIO8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO7		GPIO6		GPIO5		GPIO4		GPIO3		GPIO2		GPIO1		GPIO0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 10-13. GPAQSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO15	R/W	0h	Select input qualification type for GPIO15: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
29-28	GPIO14	R/W	0h	Select input qualification type for GPIO14: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
27-26	GPIO13	R/W	0h	Select input qualification type for GPIO13: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
25-24	GPIO12	R/W	0h	Select input qualification type for GPIO12: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
23-22	GPIO11	R/W	0h	Select input qualification type for GPIO11: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
21-20	GPIO10	R/W	0h	Select input qualification type for GPIO10: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn

Table 10-13. GPAQSEL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	GPIO9	R/W	0h	Select input qualification type for GPIO9: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
17-16	GPIO8	R/W	0h	Select input qualification type for GPIO8: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
15-14	GPIO7	R/W	0h	Select input qualification type for GPIO7: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
13-12	GPIO6	R/W	0h	Select input qualification type for GPIO6: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
11-10	GPIO5	R/W	0h	Select input qualification type for GPIO5: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
9-8	GPIO4	R/W	0h	Select input qualification type for GPIO4: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
7-6	GPIO3	R/W	0h	Select input qualification type for GPIO3: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
5-4	GPIO2	R/W	0h	Select input qualification type for GPIO2: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
3-2	GPIO1	R/W	0h	Select input qualification type for GPIO1: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
1-0	GPIO0	R/W	0h	Select input qualification type for GPIO0: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn

10.9.2.3 GPAQSEL2 Register (Offset = 4h) [Reset = 0h]

GPAQSEL2 is shown in Figure 10-6 and described in Table 10-14.

Return to the Summary Table.

GPIO A Qualifier Select 2 Register (GPIO16 to 31)
Input qualification type:
0,0 Sync
0,1 Qualification (3 samples)
1,0 Qualification (6 samples)
1,1 Async (no Sync or Qualification)

Figure 10-6. GPAQSEL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO31		GPIO30		GPIO29		GPIO28		GPIO27		GPIO26		GPIO25		GPIO24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO23		GPIO22		GPIO21		GPIO20		GPIO19		GPIO18		GPIO17		GPIO16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 10-14. GPAQSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO31	R/W	0h	Select input qualification type for GPIO31: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
29-28	GPIO30	R/W	0h	Select input qualification type for GPIO30: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
27-26	GPIO29	R/W	0h	Select input qualification type for GPIO29: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
25-24	GPIO28	R/W	0h	Select input qualification type for GPIO28: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
23-22	GPIO27	R/W	0h	Select input qualification type for GPIO27: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
21-20	GPIO26	R/W	0h	Select input qualification type for GPIO26: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn

Table 10-14. GPAQSEL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	GPIO25	R/W	0h	Select input qualification type for GPIO25: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
17-16	GPIO24	R/W	0h	Select input qualification type for GPIO24: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
15-14	GPIO23	R/W	0h	Select input qualification type for GPIO23: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
13-12	GPIO22	R/W	0h	Select input qualification type for GPIO22: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
11-10	GPIO21	R/W	0h	Select input qualification type for GPIO21: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
9-8	GPIO20	R/W	0h	Select input qualification type for GPIO20: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
7-6	GPIO19	R/W	0h	Select input qualification type for GPIO19: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
5-4	GPIO18	R/W	0h	Select input qualification type for GPIO18: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
3-2	GPIO17	R/W	0h	Select input qualification type for GPIO17: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
1-0	GPIO16	R/W	0h	Select input qualification type for GPIO16: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn

10.9.2.4 GPAMUX1 Register (Offset = 6h) [Reset = 0h]

GPAMUX1 is shown in Figure 10-7 and described in Table 10-15.

Return to the Summary Table.

GPIO A Mux 1 Register (GPIO0 to 15)
Defines pin-muxing selection for GPIO.
Notes:
The respective GPyGMUXn.GPIOz must be configured prior to this register to avoid intermediate peripheral selects being mapped to the GPIO. Refer to GPIO chapter for more details.

Figure 10-7. GPAMUX1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO15		GPIO14		GPIO13		GPIO12		GPIO11		GPIO10		GPIO9		GPIO8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO7		GPIO6		GPIO5		GPIO4		GPIO3		GPIO2		GPIO1		GPIO0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 10-15. GPAMUX1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO15	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
29-28	GPIO14	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
27-26	GPIO13	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
25-24	GPIO12	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
23-22	GPIO11	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
21-20	GPIO10	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
19-18	GPIO9	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
17-16	GPIO8	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
15-14	GPIO7	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
13-12	GPIO6	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
11-10	GPIO5	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
9-8	GPIO4	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
7-6	GPIO3	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
5-4	GPIO2	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
3-2	GPIO1	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
1-0	GPIO0	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn

10.9.2.5 GPAMUX2 Register (Offset = 8h) [Reset = 0h]

GPAMUX2 is shown in Figure 10-8 and described in Table 10-16.

Return to the Summary Table.

GPIO A Mux 2 Register (GPIO16 to 31)
Defines pin-muxing selection for GPIO.
Notes:
The respective GPyGMUXn.GPIOz must be configured prior to this register to avoid intermediate peripheral selects being mapped to the GPIO. Refer to GPIO chapter for more details.

Figure 10-8. GPAMUX2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO31		GPIO30		GPIO29		GPIO28		GPIO27		GPIO26		GPIO25		GPIO24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO23		GPIO22		GPIO21		GPIO20		GPIO19		GPIO18		GPIO17		GPIO16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 10-16. GPAMUX2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO31	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
29-28	GPIO30	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
27-26	GPIO29	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
25-24	GPIO28	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
23-22	GPIO27	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
21-20	GPIO26	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
19-18	GPIO25	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
17-16	GPIO24	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
15-14	GPIO23	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
13-12	GPIO22	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
11-10	GPIO21	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
9-8	GPIO20	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
7-6	GPIO19	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
5-4	GPIO18	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
3-2	GPIO17	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
1-0	GPIO16	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn

10.9.2.6 GPADIR Register (Offset = Ah) [Reset = 0h]

GPADIR is shown in Figure 10-9 and described in Table 10-17.

Return to the Summary Table.

GPIO A Direction Register (GPIO0 to 31)
Controls direction of GPIO pins when the specified pin is configured in GPIO mode.
0: Configures pin as input.
1: Configures pin as output.
Reading the register returns the current value of the register setting.

Figure 10-9. GPADIR Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 10-17. GPADIR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
30	GPIO30	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
29	GPIO29	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
28	GPIO28	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
27	GPIO27	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
26	GPIO26	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
25	GPIO25	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
24	GPIO24	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
23	GPIO23	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
22	GPIO22	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
21	GPIO21	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
20	GPIO20	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
19	GPIO19	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn

Table 10-17. GPADIR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO18	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
17	GPIO17	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
16	GPIO16	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
15	GPIO15	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
14	GPIO14	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
13	GPIO13	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
12	GPIO12	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
11	GPIO11	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
10	GPIO10	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
9	GPIO9	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
8	GPIO8	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
7	GPIO7	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
6	GPIO6	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
5	GPIO5	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
4	GPIO4	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
3	GPIO3	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
2	GPIO2	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
1	GPIO1	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
0	GPIO0	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn

10.9.2.7 GPAPUD Register (Offset = Ch) [Reset = FFFFFFFFh]

GPAPUD is shown in Figure 10-10 and described in Table 10-18.

Return to the Summary Table.

GPIO A Pull Up Disable Register (GPIO0 to 31)
Disables the Pull-Up on GPIO.
0: Enables the Pull-Up.
1: Disables the Pull-Up.
Reading the register returns the current value of the register setting.
Note:
[1] The Pull-Ups on the GPIO pins are disabled asynchronously when IORSn signal is low.

Figure 10-10. GPAPUD Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h

Table 10-18. GPAPUD Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
30	GPIO30	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
29	GPIO29	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
28	GPIO28	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
27	GPIO27	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
26	GPIO26	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
25	GPIO25	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
24	GPIO24	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
23	GPIO23	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
22	GPIO22	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
21	GPIO21	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
20	GPIO20	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn

Table 10-18. GPAPUD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	GPIO19	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
18	GPIO18	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
17	GPIO17	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
16	GPIO16	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
15	GPIO15	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
14	GPIO14	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
13	GPIO13	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
12	GPIO12	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
11	GPIO11	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
10	GPIO10	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
9	GPIO9	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
8	GPIO8	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
7	GPIO7	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
6	GPIO6	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
5	GPIO5	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
4	GPIO4	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
3	GPIO3	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
2	GPIO2	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
1	GPIO1	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
0	GPIO0	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn

10.9.2.8 GPAINV Register (Offset = 10h) [Reset = 0h]

GPAINV is shown in Figure 10-11 and described in Table 10-19.

Return to the Summary Table.

GPIO A Input Polarity Invert Registers (GPIO0 to 31)
Selects between non-inverted and inverted GPIO input to the device.
0: selects non-inverted GPIO input
1: selects inverted GPIO input
Reading the register returns the current value of the register setting.

Figure 10-11. GPAINV Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 10-19. GPAINV Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
30	GPIO30	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
29	GPIO29	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
28	GPIO28	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
27	GPIO27	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
26	GPIO26	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
25	GPIO25	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
24	GPIO24	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
23	GPIO23	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
22	GPIO22	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
21	GPIO21	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
20	GPIO20	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
19	GPIO19	R/W	0h	Input inversion control for this pin Reset type: SYSRSn

Table 10-19. GPAINV Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO18	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
17	GPIO17	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
16	GPIO16	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
15	GPIO15	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
14	GPIO14	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
13	GPIO13	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
12	GPIO12	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
11	GPIO11	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
10	GPIO10	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
9	GPIO9	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
8	GPIO8	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
7	GPIO7	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
6	GPIO6	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
5	GPIO5	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
4	GPIO4	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
3	GPIO3	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
2	GPIO2	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
1	GPIO1	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
0	GPIO0	R/W	0h	Input inversion control for this pin Reset type: SYSRSn

10.9.2.9 GPAODR Register (Offset = 12h) [Reset = 0h]

GPAODR is shown in Figure 10-12 and described in Table 10-20.

Return to the Summary Table.

GPIO A Open Drain Output Register (GPIO0 to GPIO31)
Selects between normal and open-drain output for the GPIO pin.
0: Normal Output
1: Open Drain Output
Reading the register returns the current value of the register setting.
Note:
[1] In the Open Drain output mode, if the buffer is configured for output mode, a 0 value to be driven out comes out on the on the PAD while a 1 value to be driven out tri-states the buffer.

Figure 10-12. GPAODR Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 10-20. GPAODR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
30	GPIO30	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
29	GPIO29	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
28	GPIO28	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
27	GPIO27	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
26	GPIO26	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
25	GPIO25	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
24	GPIO24	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
23	GPIO23	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
22	GPIO22	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
21	GPIO21	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn

Table 10-20. GPAODR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	GPIO20	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
19	GPIO19	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
18	GPIO18	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
17	GPIO17	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
16	GPIO16	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
15	GPIO15	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
14	GPIO14	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
13	GPIO13	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
12	GPIO12	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
11	GPIO11	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
10	GPIO10	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
9	GPIO9	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
8	GPIO8	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
7	GPIO7	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
6	GPIO6	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
5	GPIO5	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
4	GPIO4	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
3	GPIO3	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
2	GPIO2	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
1	GPIO1	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
0	GPIO0	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn

10.9.2.10 GPAAMSEL Register (Offset = 14h) [Reset = 00300000h]

GPAAMSEL is shown in Figure 10-13 and described in Table 10-21.

Return to the Summary Table.

GPIO A Analog Mode Select register (GPIO0 to GPIO31)
Selects between digital and analog functionality for GPIO pins.
0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers
1: The analog function of the pin is enabled and the pin is capable of analog functions
Reading the register returns the current value of the register setting.
Note:
[1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect.

Figure 10-13. GPAAMSEL Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	RESERVED	GPIO21	GPIO20	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-1h	R/W-1h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 10-21. GPAAMSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30	RESERVED	R/W	0h	Reserved
29	RESERVED	R/W	0h	Reserved
28	RESERVED	R/W	0h	Reserved
27	RESERVED	R/W	0h	Reserved
26	RESERVED	R/W	0h	Reserved
25	RESERVED	R/W	0h	Reserved
24	RESERVED	R/W	0h	Reserved
23	RESERVED	R/W	0h	Reserved
22	RESERVED	R/W	0h	Reserved
21	GPIO21	R/W	1h	Analog Mode select for this pin Reset type: SYSRSn
20	GPIO20	R/W	1h	Analog Mode select for this pin Reset type: SYSRSn
19	RESERVED	R/W	0h	Reserved
18	RESERVED	R/W	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15	RESERVED	R/W	0h	Reserved

Table 10-21. GPAAMSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
14	RESERVED	R/W	0h	Reserved
13	RESERVED	R/W	0h	Reserved
12	RESERVED	R/W	0h	Reserved
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	RESERVED	R/W	0h	Reserved
8	RESERVED	R/W	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	RESERVED	R/W	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	RESERVED	R/W	0h	Reserved
0	RESERVED	R/W	0h	Reserved

10.9.2.11 GPAGMUX1 Register (Offset = 20h) [Reset = 0h]

GPAGMUX1 is shown in Figure 10-14 and described in Table 10-22.

Return to the Summary Table.

GPIO A Peripheral Group Mux (GPIO0 to 15)
Defines pin-muxing selection for GPIO.
Notes:
[1]For complete pin-mux selection on GPIOx, GPAMUXy.GPIOx configuration is also required.

Figure 10-14. GPAGMUX1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO15		GPIO14		GPIO13		GPIO12		GPIO11		GPIO10		GPIO9		GPIO8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO7		GPIO6		GPIO5		GPIO4		GPIO3		GPIO2		GPIO1		GPIO0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 10-22. GPAGMUX1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO15	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
29-28	GPIO14	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
27-26	GPIO13	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
25-24	GPIO12	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
23-22	GPIO11	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
21-20	GPIO10	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
19-18	GPIO9	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
17-16	GPIO8	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
15-14	GPIO7	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
13-12	GPIO6	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
11-10	GPIO5	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
9-8	GPIO4	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
7-6	GPIO3	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
5-4	GPIO2	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
3-2	GPIO1	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
1-0	GPIO0	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn

10.9.2.12 GPAGMUX2 Register (Offset = 22h) [Reset = 0h]

GPAGMUX2 is shown in Figure 10-15 and described in Table 10-23.

Return to the Summary Table.

GPIO A Peripheral Group Mux (GPIO16 to 31)
Defines pin-muxing selection for GPIO.
Notes:
[1]For complete pin-mux selection on GPIOx, GPAMUXy.GPIOx configuration is also required.

Figure 10-15. GPAGMUX2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO31		GPIO30		GPIO29		GPIO28		GPIO27		GPIO26		GPIO25		GPIO24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO23		GPIO22		GPIO21		GPIO20		GPIO19		GPIO18		GPIO17		GPIO16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 10-23. GPAGMUX2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO31	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
29-28	GPIO30	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
27-26	GPIO29	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
25-24	GPIO28	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
23-22	GPIO27	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
21-20	GPIO26	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
19-18	GPIO25	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
17-16	GPIO24	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
15-14	GPIO23	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
13-12	GPIO22	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
11-10	GPIO21	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
9-8	GPIO20	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
7-6	GPIO19	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
5-4	GPIO18	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
3-2	GPIO17	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
1-0	GPIO16	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn

10.9.2.13 GPACSEL1 Register (Offset = 28h) [Reset = 0h]

GPACSEL1 is shown in Figure 10-16 and described in Table 10-24.

Return to the Summary Table.

Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected (Reserved)
0011: CPU2.CLA1 selected (Reserved)
0100: CM selected (Reserved)
0101: HIC selected (Reserved)

Figure 10-16. GPACSEL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO7				GPIO6				GPIO5				GPIO4			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO3				GPIO2				GPIO1				GPIO0			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 10-24. GPACSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO7	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
27-24	GPIO6	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
23-20	GPIO5	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
19-16	GPIO4	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
15-12	GPIO3	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
11-8	GPIO2	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
7-4	GPIO1	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
3-0	GPIO0	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn

10.9.2.14 GPACSEL2 Register (Offset = 2Ah) [Reset = 0h]

GPACSEL2 is shown in Figure 10-17 and described in Table 10-25.

Return to the Summary Table.

Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected (Reserved)
0011: CPU2.CLA1 selected (Reserved)
0100: CM selected (Reserved)
0101: HIC selected (Reserved)

Figure 10-17. GPACSEL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO15				GPIO14				GPIO13				GPIO12			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO11				GPIO10				GPIO9				GPIO8			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 10-25. GPACSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO15	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
27-24	GPIO14	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
23-20	GPIO13	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
19-16	GPIO12	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
15-12	GPIO11	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
11-8	GPIO10	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
7-4	GPIO9	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
3-0	GPIO8	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn

10.9.2.15 GPACSEL3 Register (Offset = 2Ch) [Reset = 0h]

GPACSEL3 is shown in Figure 10-18 and described in Table 10-26.

Return to the Summary Table.

Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected (Reserved)
0011: CPU2.CLA1 selected (Reserved)
0100: CM selected (Reserved)
0101: HIC selected (Reserved)

Figure 10-18. GPACSEL3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO23				GPIO22				GPIO21				GPIO20			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO19				GPIO18				GPIO17				GPIO16			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 10-26. GPACSEL3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO23	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
27-24	GPIO22	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
23-20	GPIO21	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
19-16	GPIO20	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
15-12	GPIO19	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
11-8	GPIO18	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
7-4	GPIO17	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
3-0	GPIO16	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn

10.9.2.16 GPACSEL4 Register (Offset = 2Eh) [Reset = 0h]

GPACSEL4 is shown in Figure 10-19 and described in Table 10-27.

Return to the Summary Table.

Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected (Reserved)
0011: CPU2.CLA1 selected (Reserved)
0100: CM selected (Reserved)
0101: HIC selected (Reserved)

Figure 10-19. GPACSEL4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO31				GPIO30				GPIO29				GPIO28			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO27				GPIO26				GPIO25				GPIO24			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 10-27. GPACSEL4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO31	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
27-24	GPIO30	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
23-20	GPIO29	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
19-16	GPIO28	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
15-12	GPIO27	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
11-8	GPIO26	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
7-4	GPIO25	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
3-0	GPIO24	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn

10.9.2.17 GPALOCK Register (Offset = 3Ch) [Reset = 0h]

GPALOCK is shown in Figure 10-20 and described in Table 10-28.

Return to the Summary Table.

GPIO A Lock Configuration Register (GPIO0 to 31)
GPIO Configuration Lock for GPIO.
0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed
1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin

Figure 10-20. GPALOCK Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 10-28. GPALOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
30	GPIO30	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
29	GPIO29	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
28	GPIO28	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
27	GPIO27	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
26	GPIO26	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
25	GPIO25	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
24	GPIO24	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
23	GPIO23	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
22	GPIO22	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
21	GPIO21	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
20	GPIO20	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn

Table 10-28. GPALOCK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	GPIO19	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
18	GPIO18	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
17	GPIO17	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
16	GPIO16	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
15	GPIO15	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
14	GPIO14	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
13	GPIO13	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
12	GPIO12	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
11	GPIO11	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
10	GPIO10	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
9	GPIO9	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
8	GPIO8	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
7	GPIO7	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
6	GPIO6	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
5	GPIO5	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
4	GPIO4	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
3	GPIO3	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
2	GPIO2	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
1	GPIO1	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
0	GPIO0	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn

10.9.2.18 GPACR Register (Offset = 3Eh) [Reset = 0h]

GPACR is shown in Figure 10-21 and described in Table 10-29.

Return to the Summary Table.

GPIO A Lock Commit Register (GPIO0 to 31)
GPIO Configuration Lock Commit for GPIO:
1: Locks changes to the bit in GPyLOCK register which controls the same pin
0: Bit in the GPyLOCK register which controls the same pin can be changed

Figure 10-21. GPACR Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 10-29. GPACR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
30	GPIO30	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
29	GPIO29	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
28	GPIO28	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
27	GPIO27	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
26	GPIO26	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
25	GPIO25	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
24	GPIO24	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
23	GPIO23	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
22	GPIO22	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
21	GPIO21	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
20	GPIO20	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
19	GPIO19	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn

Table 10-29. GPACR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO18	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
17	GPIO17	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
16	GPIO16	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
15	GPIO15	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
14	GPIO14	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
13	GPIO13	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
12	GPIO12	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
11	GPIO11	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
10	GPIO10	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
9	GPIO9	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
8	GPIO8	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
7	GPIO7	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
6	GPIO6	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
5	GPIO5	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
4	GPIO4	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
3	GPIO3	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
2	GPIO2	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
1	GPIO1	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
0	GPIO0	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn

10.9.2.19 GPBCTRL Register (Offset = 40h) [Reset = 0h]

GPBCTRL is shown in Figure 10-22 and described in Table 10-30.

Return to the Summary Table.

GPIO B Qualification Sampling Period Control (GPIO32 to 63)

Figure 10-22. GPBCTRL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QUALPRD3								QUALPRD2								QUALPRD1								QUALPRD0							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 10-30. GPBCTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	QUALPRD3	R/W	0h	Qualification sampling period for GPIO56 to GPIO63: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: SYSRSn
23-16	QUALPRD2	R/W	0h	Qualification sampling period for GPIO48 to GPIO55: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: SYSRSn
15-8	QUALPRD1	R/W	0h	Qualification sampling period for GPIO40 to GPIO47: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: SYSRSn
7-0	QUALPRD0	R/W	0h	Qualification sampling period for GPIO32 to GPIO39: 0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: SYSRSn

10.9.2.20 GPBQSEL1 Register (Offset = 42h) [Reset = CC0h]

GPBQSEL1 is shown in Figure 10-23 and described in Table 10-31.

Return to the Summary Table.

GPIO B Qualifier Select 1 Register (GPIO32 to 47)
Input qualification type:
0,0 Sync
0,1 Qualification (3 samples)
1,0 Qualification (6 samples)
1,1 Async (no Sync or Qualification)

Figure 10-23. GPBQSEL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO47		GPIO46		GPIO45		GPIO44		GPIO43		GPIO42		GPIO41		GPIO40	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO39		RESERVED		GPIO37		RESERVED		GPIO35		GPIO34		GPIO33		GPIO32	
R/W-0h		R/W-0h		R/W-3h		R/W-0h		R/W-3h		R/W-0h		R/W-0h		R/W-0h	

Table 10-31. GPBQSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO47	R/W	0h	Select input qualification type for GPIO47: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
29-28	GPIO46	R/W	0h	Select input qualification type for GPIO46: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
27-26	GPIO45	R/W	0h	Select input qualification type for GPIO45: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
25-24	GPIO44	R/W	0h	Select input qualification type for GPIO44: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
23-22	GPIO43	R/W	0h	Select input qualification type for GPIO43: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
21-20	GPIO42	R/W	0h	Select input qualification type for GPIO42: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn

Table 10-31. GPBQSEL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	GPIO41	R/W	0h	Select input qualification type for GPIO41: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
17-16	GPIO40	R/W	0h	Select input qualification type for GPIO40: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
15-14	GPIO39	R/W	0h	Select input qualification type for GPIO39: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
13-12	RESERVED	R/W	0h	Reserved
11-10	GPIO37	R/W	3h	Select input qualification type for GPIO37: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
9-8	RESERVED	R/W	0h	Reserved
7-6	GPIO35	R/W	3h	Select input qualification type for GPIO35: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
5-4	GPIO34	R/W	0h	Select input qualification type for GPIO34: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
3-2	GPIO33	R/W	0h	Select input qualification type for GPIO33: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
1-0	GPIO32	R/W	0h	Select input qualification type for GPIO32: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn

10.9.2.21 GPBQSEL2 Register (Offset = 44h) [Reset = 0h]

GPBQSEL2 is shown in Figure 10-24 and described in Table 10-32.

Return to the Summary Table.

GPIO B Qualifier Select 2 Register (GPIO48 to 63)
Input qualification type:
0,0 Sync
0,1 Qualification (3 samples)
1,0 Qualification (6 samples)
1,1 Async (no Sync or Qualification)

Figure 10-24. GPBQSEL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED		RESERVED		GPIO61		GPIO60		GPIO59		GPIO58		GPIO57		GPIO56	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO55		GPIO54		GPIO53		GPIO52		GPIO51		GPIO50		GPIO49		GPIO48	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 10-32. GPBQSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R/W	0h	Reserved
29-28	RESERVED	R/W	0h	Reserved
27-26	GPIO61	R/W	0h	Select input qualification type for GPIO61: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
25-24	GPIO60	R/W	0h	Select input qualification type for GPIO60: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
23-22	GPIO59	R/W	0h	Select input qualification type for GPIO59: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
21-20	GPIO58	R/W	0h	Select input qualification type for GPIO58: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
19-18	GPIO57	R/W	0h	Select input qualification type for GPIO57: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn

Table 10-32. GPBQSEL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17-16	GPIO56	R/W	0h	Select input qualification type for GPIO56: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
15-14	GPIO55	R/W	0h	Select input qualification type for GPIO55: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
13-12	GPIO54	R/W	0h	Select input qualification type for GPIO54: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
11-10	GPIO53	R/W	0h	Select input qualification type for GPIO53: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
9-8	GPIO52	R/W	0h	Select input qualification type for GPIO52: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
7-6	GPIO51	R/W	0h	Select input qualification type for GPIO51: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
5-4	GPIO50	R/W	0h	Select input qualification type for GPIO50: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
3-2	GPIO49	R/W	0h	Select input qualification type for GPIO49: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
1-0	GPIO48	R/W	0h	Select input qualification type for GPIO48: 0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn

10.9.2.22 GPBMUX1 Register (Offset = 46h) [Reset = CC0h]

GPBMUX1 is shown in Figure 10-25 and described in Table 10-33.

Return to the Summary Table.

GPIO B Mux 1 Register (GPIO32 to 47)
Defines pin-muxing selection for GPIO.
Notes:
The respective GPyGMUXn.GPIOz must be configured prior to this register to avoid intermediate peripheral selects being mapped to the GPIO. Refer to GPIO chapter for more details.

Figure 10-25. GPBMUX1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO47		GPIO46		GPIO45		GPIO44		GPIO43		GPIO42		GPIO41		GPIO40	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO39		RESERVED		GPIO37		RESERVED		GPIO35		GPIO34		GPIO33		GPIO32	
R/W-0h		R/W-0h		R/W-3h		R/W-0h		R/W-3h		R/W-0h		R/W-0h		R/W-0h	

Table 10-33. GPBMUX1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO47	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
29-28	GPIO46	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
27-26	GPIO45	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
25-24	GPIO44	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
23-22	GPIO43	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
21-20	GPIO42	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
19-18	GPIO41	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
17-16	GPIO40	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
15-14	GPIO39	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
13-12	RESERVED	R/W	0h	Reserved
11-10	GPIO37	R/W	3h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
9-8	RESERVED	R/W	0h	Reserved
7-6	GPIO35	R/W	3h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
5-4	GPIO34	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
3-2	GPIO33	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
1-0	GPIO32	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn

10.9.2.23 GPBMUX2 Register (Offset = 48h) [Reset = 0h]

GPBMUX2 is shown in Figure 10-26 and described in Table 10-34.

Return to the Summary Table.

GPIO B Mux 2 Register (GPIO48 to 63)
Defines pin-muxing selection for GPIO.
Notes:
The respective GPyGMUXn.GPIOz must be configured prior to this register to avoid intermediate peripheral selects being mapped to the GPIO. Refer to GPIO chapter for more details.

Figure 10-26. GPBMUX2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED		RESERVED		GPIO61		GPIO60		GPIO59		GPIO58		GPIO57		GPIO56	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO55		GPIO54		GPIO53		GPIO52		GPIO51		GPIO50		GPIO49		GPIO48	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 10-34. GPBMUX2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R/W	0h	Reserved
29-28	RESERVED	R/W	0h	Reserved
27-26	GPIO61	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
25-24	GPIO60	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
23-22	GPIO59	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
21-20	GPIO58	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
19-18	GPIO57	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
17-16	GPIO56	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
15-14	GPIO55	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
13-12	GPIO54	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
11-10	GPIO53	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
9-8	GPIO52	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
7-6	GPIO51	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
5-4	GPIO50	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
3-2	GPIO49	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
1-0	GPIO48	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn

10.9.2.24 GPBDIR Register (Offset = 4Ah) [Reset = 0h]

GPBDIR is shown in Figure 10-27 and described in Table 10-35.

Return to the Summary Table.

GPIO B Direction Register (GPIO32 to 63)
Controls direction of GPIO pins when the specified pin is configured in GPIO mode.
0: Configures pin as input.
1: Configures pin as output.
Reading the register returns the current value of the register setting.

Figure 10-27. GPBDIR Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	GPIO61	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO39	RESERVED	GPIO37	RESERVED	GPIO35	GPIO34	GPIO33	GPIO32
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 10-35. GPBDIR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30	RESERVED	R/W	0h	Reserved
29	GPIO61	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
28	GPIO60	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
27	GPIO59	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
26	GPIO58	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
25	GPIO57	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
24	GPIO56	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
23	GPIO55	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
22	GPIO54	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
21	GPIO53	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
20	GPIO52	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
19	GPIO51	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
18	GPIO50	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn

Table 10-35. GPBDIR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	GPIO49	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
16	GPIO48	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
15	GPIO47	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
14	GPIO46	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
13	GPIO45	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
12	GPIO44	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
11	GPIO43	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
10	GPIO42	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
9	GPIO41	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
8	GPIO40	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
7	GPIO39	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
6	RESERVED	R/W	0h	Reserved
5	GPIO37	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
4	RESERVED	R/W	0h	Reserved
3	GPIO35	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
2	GPIO34	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
1	GPIO33	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn
0	GPIO32	R/W	0h	Defines direction for this pin in GPIO mode Reset type: SYSRSn

10.9.2.25 GPBPUD Register (Offset = 4Ch) [Reset = FFFFFFFFh]

GPBPUD is shown in Figure 10-28 and described in Table 10-36.

Return to the Summary Table.

GPIO B Pull Up Disable Register (GPIO32 to 63)
Disables the Pull-Up on GPIO.
0: Enables the Pull-Up.
1: Disables the Pull-Up.
Reading the register returns the current value of the register setting.
Note:
[1] The Pull-Ups on the GPIO pins are disabled asynchronously when IORSn signal is low.

Figure 10-28. GPBPUD Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	GPIO61	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
7	6	5	4	3	2	1	0
GPIO39	RESERVED	GPIO37	RESERVED	GPIO35	GPIO34	GPIO33	GPIO32
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h

Table 10-36. GPBPUD Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	1h	Reserved
30	RESERVED	R/W	1h	Reserved
29	GPIO61	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
28	GPIO60	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
27	GPIO59	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
26	GPIO58	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
25	GPIO57	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
24	GPIO56	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
23	GPIO55	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
22	GPIO54	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
21	GPIO53	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
20	GPIO52	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
19	GPIO51	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn

Table 10-36. GPBPUD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO50	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
17	GPIO49	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
16	GPIO48	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
15	GPIO47	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
14	GPIO46	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
13	GPIO45	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
12	GPIO44	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
11	GPIO43	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
10	GPIO42	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
9	GPIO41	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
8	GPIO40	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
7	GPIO39	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
6	RESERVED	R/W	1h	Reserved
5	GPIO37	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
4	RESERVED	R/W	1h	Reserved
3	GPIO35	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
2	GPIO34	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
1	GPIO33	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn
0	GPIO32	R/W	1h	Pull-Up Disable control for this pin Reset type: SYSRSn

10.9.2.26 GPBINV Register (Offset = 50h) [Reset = 0h]

GPBINV is shown in Figure 10-29 and described in Table 10-37.

Return to the Summary Table.

GPIO B Input Polarity Invert Registers (GPIO32 to 63)
Selects between non-inverted and inverted GPIO input to the device.
0: selects non-inverted GPIO input
1: selects inverted GPIO input
Reading the register returns the current value of the register setting.

Figure 10-29. GPBINV Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	GPIO61	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO39	RESERVED	GPIO37	RESERVED	GPIO35	GPIO34	GPIO33	GPIO32
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 10-37. GPBINV Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30	RESERVED	R/W	0h	Reserved
29	GPIO61	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
28	GPIO60	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
27	GPIO59	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
26	GPIO58	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
25	GPIO57	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
24	GPIO56	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
23	GPIO55	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
22	GPIO54	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
21	GPIO53	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
20	GPIO52	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
19	GPIO51	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
18	GPIO50	R/W	0h	Input inversion control for this pin Reset type: SYSRSn

Table 10-37. GPBINV Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	GPIO49	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
16	GPIO48	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
15	GPIO47	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
14	GPIO46	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
13	GPIO45	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
12	GPIO44	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
11	GPIO43	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
10	GPIO42	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
9	GPIO41	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
8	GPIO40	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
7	GPIO39	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
6	RESERVED	R/W	0h	Reserved
5	GPIO37	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
4	RESERVED	R/W	0h	Reserved
3	GPIO35	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
2	GPIO34	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
1	GPIO33	R/W	0h	Input inversion control for this pin Reset type: SYSRSn
0	GPIO32	R/W	0h	Input inversion control for this pin Reset type: SYSRSn

10.9.2.27 GPBODR Register (Offset = 52h) [Reset = 0h]

GPBODR is shown in Figure 10-30 and described in Table 10-38.

Return to the Summary Table.

GPIO B Open Drain Output Register (GPIO32 to GPIO63)
Selects between normal and open-drain output for the GPIO pin.
0: Normal Output
1: Open Drain Output
Reading the register returns the current value of the register setting.
Note:
[1] In the Open Drain output mode, if the buffer is configured for output mode, a 0 value to be driven out comes out on the on the PAD while a 1 value to be driven out tri-states the buffer.

Figure 10-30. GPBODR Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	GPIO61	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO39	RESERVED	GPIO37	RESERVED	GPIO35	GPIO34	GPIO33	GPIO32
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 10-38. GPBODR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30	RESERVED	R/W	0h	Reserved
29	GPIO61	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
28	GPIO60	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
27	GPIO59	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
26	GPIO58	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
25	GPIO57	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
24	GPIO56	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
23	GPIO55	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
22	GPIO54	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
21	GPIO53	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
20	GPIO52	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn

Table 10-38. GPBODR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	GPIO51	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
18	GPIO50	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
17	GPIO49	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
16	GPIO48	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
15	GPIO47	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
14	GPIO46	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
13	GPIO45	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
12	GPIO44	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
11	GPIO43	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
10	GPIO42	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
9	GPIO41	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
8	GPIO40	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
7	GPIO39	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
6	RESERVED	R/W	0h	Reserved
5	GPIO37	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
4	RESERVED	R/W	0h	Reserved
3	GPIO35	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
2	GPIO34	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
1	GPIO33	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn
0	GPIO32	R/W	0h	Outpout Open-Drain control for this pin Reset type: SYSRSn

10.9.2.28 GPBGMUX1 Register (Offset = 60h) [Reset = CC0h]

GPBGMUX1 is shown in Figure 10-31 and described in Table 10-39.

Return to the Summary Table.

GPIO B Peripheral Group Mux (GPIO32 to 47)
Defines pin-muxing selection for GPIO.
Notes:
[1]For complete pin-mux selection on GPIOx, GPAMUXy.GPIOx configuration is also required.

Figure 10-31. GPBGMUX1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO47		GPIO46		GPIO45		GPIO44		GPIO43		GPIO42		GPIO41		GPIO40	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO39		RESERVED		GPIO37		RESERVED		GPIO35		GPIO34		GPIO33		GPIO32	
R/W-0h		R/W-0h		R/W-3h		R/W-0h		R/W-3h		R/W-0h		R/W-0h		R/W-0h	

Table 10-39. GPBGMUX1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO47	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
29-28	GPIO46	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
27-26	GPIO45	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
25-24	GPIO44	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
23-22	GPIO43	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
21-20	GPIO42	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
19-18	GPIO41	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
17-16	GPIO40	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
15-14	GPIO39	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
13-12	RESERVED	R/W	0h	Reserved
11-10	GPIO37	R/W	3h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
9-8	RESERVED	R/W	0h	Reserved
7-6	GPIO35	R/W	3h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
5-4	GPIO34	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
3-2	GPIO33	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
1-0	GPIO32	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn

10.9.2.29 GPBGMUX2 Register (Offset = 62h) [Reset = 0h]

GPBGMUX2 is shown in Figure 10-32 and described in Table 10-40.

Return to the Summary Table.

GPIO B Peripheral Group Mux (GPIO48 to 63)
Defines pin-muxing selection for GPIO.
Notes:
[1]For complete pin-mux selection on GPIOx, GPAMUXy.GPIOx configuration is also required.

Figure 10-32. GPBGMUX2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED		RESERVED		GPIO61		GPIO60		GPIO59		GPIO58		GPIO57		GPIO56	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO55		GPIO54		GPIO53		GPIO52		GPIO51		GPIO50		GPIO49		GPIO48	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 10-40. GPBGMUX2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R/W	0h	Reserved
29-28	RESERVED	R/W	0h	Reserved
27-26	GPIO61	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
25-24	GPIO60	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
23-22	GPIO59	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
21-20	GPIO58	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
19-18	GPIO57	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
17-16	GPIO56	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
15-14	GPIO55	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
13-12	GPIO54	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
11-10	GPIO53	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
9-8	GPIO52	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
7-6	GPIO51	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
5-4	GPIO50	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
3-2	GPIO49	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
1-0	GPIO48	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn

10.9.2.30 GPBCSEL1 Register (Offset = 68h) [Reset = 0h]

GPBCSEL1 is shown in Figure 10-33 and described in Table 10-41.

Return to the Summary Table.

Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected (Reserved)
0011: CPU2.CLA1 selected (Reserved)
0100: CM selected (Reserved)
0101: HIC selected (Reserved)

Figure 10-33. GPBCSEL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO39				RESERVED				GPIO37				RESERVED			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO35				GPIO34				GPIO33				GPIO32			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 10-41. GPBCSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO39	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
27-24	RESERVED	R/W	0h	Reserved
23-20	GPIO37	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
19-16	RESERVED	R/W	0h	Reserved
15-12	GPIO35	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
11-8	GPIO34	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
7-4	GPIO33	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
3-0	GPIO32	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn

10.9.2.31 GPBCSEL2 Register (Offset = 6Ah) [Reset = 0h]

GPBCSEL2 is shown in Figure 10-34 and described in Table 10-42.

Return to the Summary Table.

Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected (Reserved)
0011: CPU2.CLA1 selected (Reserved)
0100: CM selected (Reserved)
0101: HIC selected (Reserved)

Figure 10-34. GPBCSEL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO47				GPIO46				GPIO45				GPIO44			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO43				GPIO42				GPIO41				GPIO40			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 10-42. GPBCSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO47	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
27-24	GPIO46	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
23-20	GPIO45	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
19-16	GPIO44	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
15-12	GPIO43	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
11-8	GPIO42	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
7-4	GPIO41	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
3-0	GPIO40	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn

10.9.2.32 GPBCSEL3 Register (Offset = 6Ch) [Reset = 0h]

GPBCSEL3 is shown in Figure 10-35 and described in Table 10-43.

Return to the Summary Table.

Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected (Reserved)
0011: CPU2.CLA1 selected (Reserved)
0100: CM selected (Reserved)
0101: HIC selected (Reserved)

Figure 10-35. GPBCSEL3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO55				GPIO54				GPIO53				GPIO52			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO51				GPIO50				GPIO49				GPIO48			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 10-43. GPBCSEL3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO55	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
27-24	GPIO54	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
23-20	GPIO53	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
19-16	GPIO52	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
15-12	GPIO51	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
11-8	GPIO50	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
7-4	GPIO49	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
3-0	GPIO48	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn

10.9.2.33 GPBCSEL4 Register (Offset = 6Eh) [Reset = 0h]

GPBCSEL4 is shown in Figure 10-36 and described in Table 10-44.

Return to the Summary Table.

Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected (Reserved)
0011: CPU2.CLA1 selected (Reserved)
0100: CM selected (Reserved)
0101: HIC selected (Reserved)

Figure 10-36. GPBCSEL4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				RESERVED				GPIO61				GPIO60			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO59				GPIO58				GPIO57				GPIO56			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 10-44. GPBCSEL4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R/W	0h	Reserved
27-24	RESERVED	R/W	0h	Reserved
23-20	GPIO61	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
19-16	GPIO60	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
15-12	GPIO59	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
11-8	GPIO58	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
7-4	GPIO57	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
3-0	GPIO56	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn

10.9.2.34 GPBLOCK Register (Offset = 7Ch) [Reset = 0h]

GPBLOCK is shown in Figure 10-37 and described in Table 10-45.

Return to the Summary Table.

GPIO B Lock Configuration Register (GPIO32 to 63)
GPIO Configuration Lock for GPIO.
0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed
1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin

Figure 10-37. GPBLOCK Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	GPIO61	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO39	RESERVED	GPIO37	RESERVED	GPIO35	GPIO34	GPIO33	GPIO32
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 10-45. GPBLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30	RESERVED	R/W	0h	Reserved
29	GPIO61	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
28	GPIO60	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
27	GPIO59	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
26	GPIO58	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
25	GPIO57	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
24	GPIO56	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
23	GPIO55	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
22	GPIO54	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
21	GPIO53	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
20	GPIO52	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
19	GPIO51	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn

Table 10-45. GPBLOCK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO50	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
17	GPIO49	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
16	GPIO48	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
15	GPIO47	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
14	GPIO46	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
13	GPIO45	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
12	GPIO44	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
11	GPIO43	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
10	GPIO42	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
9	GPIO41	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
8	GPIO40	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
7	GPIO39	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
6	RESERVED	R/W	0h	Reserved
5	GPIO37	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
4	RESERVED	R/W	0h	Reserved
3	GPIO35	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
2	GPIO34	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
1	GPIO33	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn
0	GPIO32	R/W	0h	Configuration Lock bit for this pin Reset type: SYSRSn

10.9.2.35 GPBCR Register (Offset = 7Eh) [Reset = 0h]

GPBCR is shown in Figure 10-38 and described in Table 10-46.

Return to the Summary Table.

GPIO B Lock Commit Register (GPIO32 to 63)
GPIO Configuration Lock Commit for GPIO:
1: Locks changes to the bit in GPyLOCK register which controls the same pin
0: Bit in the GPyLOCK register which controls the same pin can be changed

Figure 10-38. GPBCR Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	GPIO61	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h
7	6	5	4	3	2	1	0
GPIO39	RESERVED	GPIO37	RESERVED	GPIO35	GPIO34	GPIO33	GPIO32
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 10-46. GPBCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/WSonce	0h	Reserved
30	RESERVED	R/WSonce	0h	Reserved
29	GPIO61	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
28	GPIO60	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
27	GPIO59	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
26	GPIO58	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
25	GPIO57	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
24	GPIO56	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
23	GPIO55	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
22	GPIO54	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
21	GPIO53	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
20	GPIO52	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
19	GPIO51	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
18	GPIO50	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn

Table 10-46. GPBCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	GPIO49	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
16	GPIO48	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
15	GPIO47	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
14	GPIO46	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
13	GPIO45	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
12	GPIO44	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
11	GPIO43	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
10	GPIO42	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
9	GPIO41	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
8	GPIO40	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
7	GPIO39	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
6	RESERVED	R/WSonce	0h	Reserved
5	GPIO37	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
4	RESERVED	R/WSonce	0h	Reserved
3	GPIO35	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
2	GPIO34	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
1	GPIO33	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn
0	GPIO32	R/WSonce	0h	Configuration lock commit bit for this pin Reset type: SYSRSn

10.9.2.36 GPHCTRL Register (Offset = 1C0h) [Reset = 0h]

GPHCTRL is shown in Figure 10-39 and described in Table 10-47.

Return to the Summary Table.

GPIO H Qualification Sampling Period Control (GPIO224 to 255)

Figure 10-39. GPHCTRL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QUALPRD3								QUALPRD2								QUALPRD1								QUALPRD0							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 10-47. GPHCTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	QUALPRD3	R/W	0h	0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/513 Reset type: SYSRSn
23-16	QUALPRD2	R/W	0h	0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/512 Reset type: SYSRSn
15-8	QUALPRD1	R/W	0h	0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/511 Reset type: SYSRSn
7-0	QUALPRD0	R/W	0h	0x00,QUALPRDx = PLLSYSCLK 0x01,QUALPRDx = PLLSYSCLK/2 0x02,QUALPRDx = PLLSYSCLK/4 .... 0xFF,QUALPRDx = PLLSYSCLK/510 Reset type: SYSRSn

10.9.2.37 GPHQSEL1 Register (Offset = 1C2h) [Reset = 0h]

GPHQSEL1 is shown in Figure 10-40 and described in Table 10-48.

Return to the Summary Table.

GPIO H Qualifier Select 1 Register (GPIO224 to 239)
Input qualification type:
0,0 Sync
0,1 Qualification (3 samples)
1,0 Qualification (6 samples)
1,1 Async (no Sync or Qualification)

Figure 10-40. GPHQSEL1 Register

31	30	29	28	27	26	25	24
GPIO239		GPIO238		GPIO237		GPIO236	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
23	22	21	20	19	18	17	16
RESERVED		RESERVED		GPIO233		GPIO232	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
GPIO231		GPIO230		GPIO229		GPIO228	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
GPIO227		GPIO226		GPIO225		GPIO224	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 10-48. GPHQSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO239	R/W	0h	0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
29-28	GPIO238	R/W	0h	0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
27-26	GPIO237	R/W	0h	0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
25-24	GPIO236	R/W	0h	0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
23-22	RESERVED	R/W	0h	Reserved
21-20	RESERVED	R/W	0h	Reserved
19-18	GPIO233	R/W	0h	0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn

Table 10-48. GPHQSEL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17-16	GPIO232	R/W	0h	0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
15-14	GPIO231	R/W	0h	0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
13-12	GPIO230	R/W	0h	0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
11-10	GPIO229	R/W	0h	0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
9-8	GPIO228	R/W	0h	0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
7-6	GPIO227	R/W	0h	0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
5-4	GPIO226	R/W	0h	0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
3-2	GPIO225	R/W	0h	0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
1-0	GPIO224	R/W	0h	0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn

10.9.2.38 GPHQSEL2 Register (Offset = 1C4h) [Reset = 0h]

GPHQSEL2 is shown in Figure 10-41 and described in Table 10-49.

Return to the Summary Table.

GPIO H Qualifier Select 2 Register (GPIO240 to 255)
Input qualification type:
0,0 Sync
0,1 Qualification (3 samples)
1,0 Qualification (6 samples)
1,1 Async (no Sync or Qualification)

Figure 10-41. GPHQSEL2 Register

31	30	29	28	27	26	25	24
RESERVED		RESERVED		GPIO253		GPIO252	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
23	22	21	20	19	18	17	16
GPIO251		RESERVED		GPIO249		GPIO248	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
GPIO247		RESERVED		GPIO245		GPIO244	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		GPIO242		GPIO241		GPIO240	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 10-49. GPHQSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R/W	0h	Reserved
29-28	RESERVED	R/W	0h	Reserved
27-26	GPIO253	R/W	0h	0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
25-24	GPIO252	R/W	0h	0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
23-22	GPIO251	R/W	0h	0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
21-20	RESERVED	R/W	0h	Reserved
19-18	GPIO249	R/W	0h	0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn

Table 10-49. GPHQSEL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17-16	GPIO248	R/W	0h	0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
15-14	GPIO247	R/W	0h	0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
13-12	RESERVED	R/W	0h	Reserved
11-10	GPIO245	R/W	0h	0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
9-8	GPIO244	R/W	0h	0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
7-6	RESERVED	R/W	0h	Reserved
5-4	GPIO242	R/W	0h	0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
3-2	GPIO241	R/W	0h	0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn
1-0	GPIO240	R/W	0h	0,0,Sync 0,1,Qualification (3 samples) 1,0,Qualification (6 samples) 1,1,Async (no Sync or Qualification) Reset type: SYSRSn

10.9.2.39 GPHMUX1 Register (Offset = 1C6h) [Reset = 0h]

GPHMUX1 is shown in Figure 10-42 and described in Table 10-50.

Return to the Summary Table.

GPIO H Mux 1 Register (GPIO224 to 239)
Defines pin-muxing selection for GPIO.
Notes:
The respective GPyGMUXn.GPIOz must be configured prior to this register to avoid intermediate peripheral selects being mapped to the GPIO. Refer to GPIO chapter for more details.

Figure 10-42. GPHMUX1 Register

31	30	29	28	27	26	25	24
GPIO239		GPIO238		GPIO237		RESERVED	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
23	22	21	20	19	18	17	16
RESERVED		RESERVED		GPIO233		GPIO232	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
GPIO231		GPIO230		RESERVED		GPIO228	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
GPIO227		GPIO226		GPIO225		GPIO224	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 10-50. GPHMUX1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO239	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
29-28	GPIO238	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
27-26	GPIO237	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
25-24	RESERVED	R/W	0h	Reserved
23-22	RESERVED	R/W	0h	Reserved
21-20	RESERVED	R/W	0h	Reserved
19-18	GPIO233	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
17-16	GPIO232	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
15-14	GPIO231	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
13-12	GPIO230	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
11-10	RESERVED	R/W	0h	Reserved
9-8	GPIO228	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
7-6	GPIO227	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
5-4	GPIO226	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn

Table 10-50. GPHMUX1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	GPIO225	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
1-0	GPIO224	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn

10.9.2.40 GPHMUX2 Register (Offset = 1C8h) [Reset = 0h]

GPHMUX2 is shown in Figure 10-43 and described in Table 10-51.

Return to the Summary Table.

GPIO H Mux 2 Register (GPIO240 to 255)
Defines pin-muxing selection for GPIO.
Notes:
The respective GPyGMUXn.GPIOz must be configured prior to this register to avoid intermediate peripheral selects being mapped to the GPIO. Refer to GPIO chapter for more details.

Figure 10-43. GPHMUX2 Register

31	30	29	28	27	26	25	24
RESERVED		RESERVED		RESERVED		GPIO252	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
23	22	21	20	19	18	17	16
RESERVED		RESERVED		RESERVED		RESERVED	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
RESERVED		RESERVED		GPIO245		GPIO244	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		GPIO242		GPIO241		GPIO240	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 10-51. GPHMUX2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R/W	0h	Reserved
29-28	RESERVED	R/W	0h	Reserved
27-26	RESERVED	R/W	0h	Reserved
25-24	GPIO252	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
23-22	RESERVED	R/W	0h	Reserved
21-20	RESERVED	R/W	0h	Reserved
19-18	RESERVED	R/W	0h	Reserved
17-16	RESERVED	R/W	0h	Reserved
15-14	RESERVED	R/W	0h	Reserved
13-12	RESERVED	R/W	0h	Reserved
11-10	GPIO245	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
9-8	GPIO244	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
7-6	RESERVED	R/W	0h	Reserved
5-4	GPIO242	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
3-2	GPIO241	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
1-0	GPIO240	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn

10.9.2.41 GPHINV Register (Offset = 1D0h) [Reset = 0h]

GPHINV is shown in Figure 10-44 and described in Table 10-52.

Return to the Summary Table.

GPIO H Input Polarity Invert Registers (GPIO224 to 255)
Selects between non-inverted and inverted GPIO input to the device.
0: selects non-inverted GPIO input
1: selects inverted GPIO input
Reading the register returns the current value of the register setting.

Figure 10-44. GPHINV Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	GPIO253	GPIO252	GPIO251	RESERVED	GPIO249	GPIO248
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO247	RESERVED	GPIO245	GPIO244	RESERVED	GPIO242	GPIO241	GPIO240
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO239	GPIO238	GPIO237	GPIO236	RESERVED	RESERVED	GPIO233	GPIO232
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO231	GPIO230	GPIO229	GPIO228	GPIO227	GPIO226	GPIO225	GPIO224
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 10-52. GPHINV Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30	RESERVED	R/W	0h	Reserved
29	GPIO253	R/W	0h	0: selects non-inverted GPIO input 1: selects inverted GPIO input Notes: [1] Reading the register returns the current value of the register setting. Reset type: SYSRSn
28	GPIO252	R/W	0h	0: selects non-inverted GPIO input 1: selects inverted GPIO input Notes: [1] Reading the register returns the current value of the register setting. Reset type: SYSRSn
27	GPIO251	R/W	0h	0: selects non-inverted GPIO input 1: selects inverted GPIO input Notes: [1] Reading the register returns the current value of the register setting. Reset type: SYSRSn
26	RESERVED	R/W	0h	Reserved
25	GPIO249	R/W	0h	0: selects non-inverted GPIO input 1: selects inverted GPIO input Notes: [1] Reading the register returns the current value of the register setting. Reset type: SYSRSn

Table 10-52. GPHINV Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
24	GPIO248	R/W	0h	0: selects non-inverted GPIO input 1: selects inverted GPIO input Notes: [1] Reading the register returns the current value of the register setting. Reset type: SYSRSn
23	GPIO247	R/W	0h	0: selects non-inverted GPIO input 1: selects inverted GPIO input Notes: [1] Reading the register returns the current value of the register setting. Reset type: SYSRSn
22	RESERVED	R/W	0h	Reserved
21	GPIO245	R/W	0h	0: selects non-inverted GPIO input 1: selects inverted GPIO input Notes: [1] Reading the register returns the current value of the register setting. Reset type: SYSRSn
20	GPIO244	R/W	0h	0: selects non-inverted GPIO input 1: selects inverted GPIO input Notes: [1] Reading the register returns the current value of the register setting. Reset type: SYSRSn
19	RESERVED	R/W	0h	Reserved
18	GPIO242	R/W	0h	0: selects non-inverted GPIO input 1: selects inverted GPIO input Notes: [1] Reading the register returns the current value of the register setting. Reset type: SYSRSn
17	GPIO241	R/W	0h	0: selects non-inverted GPIO input 1: selects inverted GPIO input Notes: [1] Reading the register returns the current value of the register setting. Reset type: SYSRSn
16	GPIO240	R/W	0h	0: selects non-inverted GPIO input 1: selects inverted GPIO input Notes: [1] Reading the register returns the current value of the register setting. Reset type: SYSRSn
15	GPIO239	R/W	0h	0: selects non-inverted GPIO input 1: selects inverted GPIO input Notes: [1] Reading the register returns the current value of the register setting. Reset type: SYSRSn
14	GPIO238	R/W	0h	0: selects non-inverted GPIO input 1: selects inverted GPIO input Notes: [1] Reading the register returns the current value of the register setting. Reset type: SYSRSn

Table 10-52. GPHINV Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	GPIO237	R/W	0h	0: selects non-inverted GPIO input 1: selects inverted GPIO input Notes: [1] Reading the register returns the current value of the register setting. Reset type: SYSRSn
12	GPIO236	R/W	0h	0: selects non-inverted GPIO input 1: selects inverted GPIO input Notes: [1] Reading the register returns the current value of the register setting. Reset type: SYSRSn
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	GPIO233	R/W	0h	0: selects non-inverted GPIO input 1: selects inverted GPIO input Notes: [1] Reading the register returns the current value of the register setting. Reset type: SYSRSn
8	GPIO232	R/W	0h	0: selects non-inverted GPIO input 1: selects inverted GPIO input Notes: [1] Reading the register returns the current value of the register setting. Reset type: SYSRSn
7	GPIO231	R/W	0h	0: selects non-inverted GPIO input 1: selects inverted GPIO input Notes: [1] Reading the register returns the current value of the register setting. Reset type: SYSRSn
6	GPIO230	R/W	0h	0: selects non-inverted GPIO input 1: selects inverted GPIO input Notes: [1] Reading the register returns the current value of the register setting. Reset type: SYSRSn
5	GPIO229	R/W	0h	0: selects non-inverted GPIO input 1: selects inverted GPIO input Notes: [1] Reading the register returns the current value of the register setting. Reset type: SYSRSn
4	GPIO228	R/W	0h	0: selects non-inverted GPIO input 1: selects inverted GPIO input Notes: [1] Reading the register returns the current value of the register setting. Reset type: SYSRSn
3	GPIO227	R/W	0h	0: selects non-inverted GPIO input 1: selects inverted GPIO input Notes: [1] Reading the register returns the current value of the register setting. Reset type: SYSRSn

Table 10-52. GPHINV Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	GPIO226	R/W	0h	0: selects non-inverted GPIO input 1: selects inverted GPIO input Notes: [1] Reading the register returns the current value of the register setting. Reset type: SYSRSn
1	GPIO225	R/W	0h	0: selects non-inverted GPIO input 1: selects inverted GPIO input Notes: [1] Reading the register returns the current value of the register setting. Reset type: SYSRSn
0	GPIO224	R/W	0h	0: selects non-inverted GPIO input 1: selects inverted GPIO input Notes: [1] Reading the register returns the current value of the register setting. Reset type: SYSRSn

10.9.2.42 GPHAMSEL Register (Offset = 1D4h) [Reset = FFFFFFFFh]

GPHAMSEL is shown in Figure 10-45 and described in Table 10-53.

Return to the Summary Table.

GPIO H Analog Mode Select register (GPIO224 to GPIO255)
Selects between digital and analog functionality for GPIO pins.
0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers
1: The analog function of the pin is enabled and the pin is capable of analog functions
Reading the register returns the current value of the register setting.
Note:
[1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, t

Figure 10-45. GPHAMSEL Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	GPIO253	GPIO252	GPIO251	RESERVED	GPIO249	GPIO248
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
23	22	21	20	19	18	17	16
GPIO247	RESERVED	GPIO245	GPIO244	RESERVED	GPIO242	GPIO241	GPIO240
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
15	14	13	12	11	10	9	8
GPIO239	GPIO238	GPIO237	GPIO236	RESERVED	RESERVED	GPIO233	GPIO232
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
7	6	5	4	3	2	1	0
GPIO231	GPIO230	GPIO229	GPIO228	GPIO227	GPIO226	GPIO225	GPIO224
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h

Table 10-53. GPHAMSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	1h	Reserved
30	RESERVED	R/W	1h	Reserved
29	GPIO253	R/W	1h	0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers 1: The analog function of the pin is enabled and the pin is capable of analog functions Reading the register returns the current value of the register setting. Note: [1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect. Reset type: SYSRSn
28	GPIO252	R/W	1h	0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers 1: The analog function of the pin is enabled and the pin is capable of analog functions Reading the register returns the current value of the register setting. Note: [1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect. Reset type: SYSRSn

Table 10-53. GPHAMSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	GPIO251	R/W	1h	0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers 1: The analog function of the pin is enabled and the pin is capable of analog functions Reading the register returns the current value of the register setting. Note: [1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect. Reset type: SYSRSn
26	RESERVED	R/W	1h	Reserved
25	GPIO249	R/W	1h	0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers 1: The analog function of the pin is enabled and the pin is capable of analog functions Reading the register returns the current value of the register setting. Note: [1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect. Reset type: SYSRSn
24	GPIO248	R/W	1h	0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers 1: The analog function of the pin is enabled and the pin is capable of analog functions Reading the register returns the current value of the register setting. Note: [1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect. Reset type: SYSRSn
23	GPIO247	R/W	1h	0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers 1: The analog function of the pin is enabled and the pin is capable of analog functions Reading the register returns the current value of the register setting. Note: [1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect. Reset type: SYSRSn
22	RESERVED	R/W	1h	Reserved
21	GPIO245	R/W	1h	0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers 1: The analog function of the pin is enabled and the pin is capable of analog functions Reading the register returns the current value of the register setting. Note: [1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect. Reset type: SYSRSn

Table 10-53. GPHAMSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	GPIO244	R/W	1h	0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers 1: The analog function of the pin is enabled and the pin is capable of analog functions Reading the register returns the current value of the register setting. Note: [1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect. Reset type: SYSRSn
19	RESERVED	R/W	1h	Reserved
18	GPIO242	R/W	1h	0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers 1: The analog function of the pin is enabled and the pin is capable of analog functions Reading the register returns the current value of the register setting. Note: [1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect. Reset type: SYSRSn
17	GPIO241	R/W	1h	0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers 1: The analog function of the pin is enabled and the pin is capable of analog functions Reading the register returns the current value of the register setting. Note: [1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect. Reset type: SYSRSn
16	GPIO240	R/W	1h	0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers 1: The analog function of the pin is enabled and the pin is capable of analog functions Reading the register returns the current value of the register setting. Note: [1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect. Reset type: SYSRSn
15	GPIO239	R/W	1h	0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers 1: The analog function of the pin is enabled and the pin is capable of analog functions Reading the register returns the current value of the register setting. Note: [1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect. Reset type: SYSRSn

Table 10-53. GPHAMSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
14	GPIO238	R/W	1h	0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers 1: The analog function of the pin is enabled and the pin is capable of analog functions Reading the register returns the current value of the register setting. Note: [1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect. Reset type: SYSRSn
13	GPIO237	R/W	1h	0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers 1: The analog function of the pin is enabled and the pin is capable of analog functions Reading the register returns the current value of the register setting. Note: [1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect. Reset type: SYSRSn
12	GPIO236	R/W	1h	0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers 1: The analog function of the pin is enabled and the pin is capable of analog functions Reading the register returns the current value of the register setting. Note: [1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect. Reset type: SYSRSn
11	RESERVED	R/W	1h	Reserved
10	RESERVED	R/W	1h	Reserved
9	GPIO233	R/W	1h	0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers 1: The analog function of the pin is enabled and the pin is capable of analog functions Reading the register returns the current value of the register setting. Note: [1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect. Reset type: SYSRSn
8	GPIO232	R/W	1h	0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers 1: The analog function of the pin is enabled and the pin is capable of analog functions Reading the register returns the current value of the register setting. Note: [1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect. Reset type: SYSRSn

Table 10-53. GPHAMSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	GPIO231	R/W	1h	0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers 1: The analog function of the pin is enabled and the pin is capable of analog functions Reading the register returns the current value of the register setting. Note: [1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect. Reset type: SYSRSn
6	GPIO230	R/W	1h	0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers 1: The analog function of the pin is enabled and the pin is capable of analog functions Reading the register returns the current value of the register setting. Note: [1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect. Reset type: SYSRSn
5	GPIO229	R/W	1h	0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers 1: The analog function of the pin is enabled and the pin is capable of analog functions Reading the register returns the current value of the register setting. Note: [1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect. Reset type: SYSRSn
4	GPIO228	R/W	1h	0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers 1: The analog function of the pin is enabled and the pin is capable of analog functions Reading the register returns the current value of the register setting. Note: [1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect. Reset type: SYSRSn
3	GPIO227	R/W	1h	0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers 1: The analog function of the pin is enabled and the pin is capable of analog functions Reading the register returns the current value of the register setting. Note: [1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect. Reset type: SYSRSn

Table 10-53. GPHAMSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	GPIO226	R/W	1h	0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers 1: The analog function of the pin is enabled and the pin is capable of analog functions Reading the register returns the current value of the register setting. Note: [1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect. Reset type: SYSRSn
1	GPIO225	R/W	1h	0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers 1: The analog function of the pin is enabled and the pin is capable of analog functions Reading the register returns the current value of the register setting. Note: [1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect. Reset type: SYSRSn
0	GPIO224	R/W	1h	0: The analog function of the pin is disabled and the pin is capable of digital functions as specified by the other GPIO configuration registers 1: The analog function of the pin is enabled and the pin is capable of analog functions Reading the register returns the current value of the register setting. Note: [1] This register and bits are only valid for GPIO signals that share analog function through a unified I/O pad. For all the IOs, the corresponding bits in these registers dont have any affect. Reset type: SYSRSn

10.9.2.43 GPHGMUX1 Register (Offset = 1E0h) [Reset = 0h]

GPHGMUX1 is shown in Figure 10-46 and described in Table 10-54.

Return to the Summary Table.

GPIO H Peripheral Group Mux (GPIO224 to 239)
Defines pin-muxing selection for GPIO.
Notes:
[1]For complete pin-mux selection on GPIOx, GPAMUXy.GPIOx configuration is also required.

Figure 10-46. GPHGMUX1 Register

31	30	29	28	27	26	25	24
GPIO239		GPIO238		GPIO237		RESERVED	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
23	22	21	20	19	18	17	16
RESERVED		RESERVED		GPIO233		GPIO232	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
GPIO231		GPIO230		RESERVED		GPIO228	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
GPIO227		GPIO226		GPIO225		GPIO224	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 10-54. GPHGMUX1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	GPIO239	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
29-28	GPIO238	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
27-26	GPIO237	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
25-24	RESERVED	R/W	0h	Reserved
23-22	RESERVED	R/W	0h	Reserved
21-20	RESERVED	R/W	0h	Reserved
19-18	GPIO233	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
17-16	GPIO232	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
15-14	GPIO231	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
13-12	GPIO230	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
11-10	RESERVED	R/W	0h	Reserved
9-8	GPIO228	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
7-6	GPIO227	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
5-4	GPIO226	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
3-2	GPIO225	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn

Table 10-54. GPHGMUX1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	GPIO224	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn

10.9.2.44 GPHGMUX2 Register (Offset = 1E2h) [Reset = 0h]

GPHGMUX2 is shown in Figure 10-47 and described in Table 10-55.

Return to the Summary Table.

GPIO H Peripheral Group Mux (GPIO240 to 255)
Defines pin-muxing selection for GPIO.
Notes:
[1]For complete pin-mux selection on GPIOx, GPAMUXy.GPIOx configuration is also required.

Figure 10-47. GPHGMUX2 Register

31	30	29	28	27	26	25	24
RESERVED		RESERVED		RESERVED		GPIO252	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
23	22	21	20	19	18	17	16
RESERVED		RESERVED		RESERVED		RESERVED	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8
RESERVED		RESERVED		GPIO245		GPIO244	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		GPIO242		GPIO241		GPIO240	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 10-55. GPHGMUX2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R/W	0h	Reserved
29-28	RESERVED	R/W	0h	Reserved
27-26	RESERVED	R/W	0h	Reserved
25-24	GPIO252	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
23-22	RESERVED	R/W	0h	Reserved
21-20	RESERVED	R/W	0h	Reserved
19-18	RESERVED	R/W	0h	Reserved
17-16	RESERVED	R/W	0h	Reserved
15-14	RESERVED	R/W	0h	Reserved
13-12	RESERVED	R/W	0h	Reserved
11-10	GPIO245	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
9-8	GPIO244	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
7-6	RESERVED	R/W	0h	Reserved
5-4	GPIO242	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
3-2	GPIO241	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn
1-0	GPIO240	R/W	0h	Defines pin-muxing selection for GPIO Reset type: SYSRSn

10.9.2.45 GPHCSEL1 Register (Offset = 1E8h) [Reset = 0h]

GPHCSEL1 is shown in Figure 10-48 and described in Table 10-56.

Return to the Summary Table.

Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected (Reserved)
0011: CPU2.CLA1 selected (Reserved)
0100: CM selected (Reserved)
0101: HIC selected (Reserved)

Figure 10-48. GPHCSEL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO231				GPIO230				GPIO229				GPIO228			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO227				GPIO226				GPIO225				GPIO224			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 10-56. GPHCSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO231	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
27-24	GPIO230	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
23-20	GPIO229	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
19-16	GPIO228	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
15-12	GPIO227	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
11-8	GPIO226	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
7-4	GPIO225	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
3-0	GPIO224	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn

10.9.2.46 GPHCSEL2 Register (Offset = 1EAh) [Reset = 0h]

GPHCSEL2 is shown in Figure 10-49 and described in Table 10-57.

Return to the Summary Table.

Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected (Reserved)
0011: CPU2.CLA1 selected (Reserved)
0100: CM selected (Reserved)
0101: HIC selected (Reserved)

Figure 10-49. GPHCSEL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO239				GPIO238				GPIO237				GPIO236			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				RESERVED				GPIO233				GPIO232			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 10-57. GPHCSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO239	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
27-24	GPIO238	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
23-20	GPIO237	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
19-16	GPIO236	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
15-12	RESERVED	R/W	0h	Reserved
11-8	RESERVED	R/W	0h	Reserved
7-4	GPIO233	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
3-0	GPIO232	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn

10.9.2.47 GPHCSEL3 Register (Offset = 1ECh) [Reset = 0h]

GPHCSEL3 is shown in Figure 10-50 and described in Table 10-58.

Return to the Summary Table.

Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected (Reserved)
0011: CPU2.CLA1 selected (Reserved)
0100: CM selected (Reserved)
0101: HIC selected (Reserved)

Figure 10-50. GPHCSEL3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GPIO247				RESERVED				GPIO245				GPIO244			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED				GPIO242				GPIO241				GPIO240			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 10-58. GPHCSEL3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GPIO247	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
27-24	RESERVED	R/W	0h	Reserved
23-20	GPIO245	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
19-16	GPIO244	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
15-12	RESERVED	R/W	0h	Reserved
11-8	GPIO242	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
7-4	GPIO241	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
3-0	GPIO240	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn

10.9.2.48 GPHCSEL4 Register (Offset = 1EEh) [Reset = 0h]

GPHCSEL4 is shown in Figure 10-51 and described in Table 10-59.

Return to the Summary Table.

Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin
0000: CPU1 selected
0001: CPU1.CLA1 selected
0010: CPU2 selected (Reserved)
0011: CPU2.CLA1 selected (Reserved)
0100: CM selected (Reserved)
0101: HIC selected (Reserved)

Figure 10-51. GPHCSEL4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				RESERVED				GPIO253				GPIO252			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GPIO251				RESERVED				GPIO249				GPIO248			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 10-59. GPHCSEL4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R/W	0h	Reserved
27-24	RESERVED	R/W	0h	Reserved
23-20	GPIO253	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
19-16	GPIO252	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
15-12	GPIO251	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
11-8	RESERVED	R/W	0h	Reserved
7-4	GPIO249	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn
3-0	GPIO248	R/W	0h	Selects which master's GPIODAT/SET/CLEAR/TOGGLE registers control this GPIO pin Reset type: SYSRSn

10.9.2.49 GPHLOCK Register (Offset = 1FCh) [Reset = 0h]

GPHLOCK is shown in Figure 10-52 and described in Table 10-60.

Return to the Summary Table.

GPIO H Lock Configuration Register (GPIO224 to 255)
GPIO Configuration Lock for GPIO.
0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed
1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin

Figure 10-52. GPHLOCK Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	GPIO253	GPIO252	GPIO251	RESERVED	GPIO249	GPIO248
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO247	RESERVED	GPIO245	GPIO244	RESERVED	GPIO242	GPIO241	GPIO240
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO239	GPIO238	GPIO237	GPIO236	RESERVED	RESERVED	GPIO233	GPIO232
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO231	GPIO230	GPIO229	GPIO228	GPIO227	GPIO226	GPIO225	GPIO224
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 10-60. GPHLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30	RESERVED	R/W	0h	Reserved
29	GPIO253	R/W	0h	1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin 0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed Reset type: SYSRSn
28	GPIO252	R/W	0h	1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin 0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed Reset type: SYSRSn
27	GPIO251	R/W	0h	1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin 0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed Reset type: SYSRSn
26	RESERVED	R/W	0h	Reserved

Table 10-60. GPHLOCK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	GPIO249	R/W	0h	1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin 0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed Reset type: SYSRSn
24	GPIO248	R/W	0h	1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin 0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed Reset type: SYSRSn
23	GPIO247	R/W	0h	1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin 0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed Reset type: SYSRSn
22	RESERVED	R/W	0h	Reserved
21	GPIO245	R/W	0h	1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin 0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed Reset type: SYSRSn
20	GPIO244	R/W	0h	1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin 0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed Reset type: SYSRSn
19	RESERVED	R/W	0h	Reserved
18	GPIO242	R/W	0h	1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin 0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed Reset type: SYSRSn
17	GPIO241	R/W	0h	1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin 0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed Reset type: SYSRSn
16	GPIO240	R/W	0h	1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin 0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed Reset type: SYSRSn

Table 10-60. GPHLOCK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15	GPIO239	R/W	0h	1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin 0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed Reset type: SYSRSn
14	GPIO238	R/W	0h	1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin 0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed Reset type: SYSRSn
13	GPIO237	R/W	0h	1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin 0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed Reset type: SYSRSn
12	GPIO236	R/W	0h	1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin 0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed Reset type: SYSRSn
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	GPIO233	R/W	0h	1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin 0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed Reset type: SYSRSn
8	GPIO232	R/W	0h	1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin 0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed Reset type: SYSRSn
7	GPIO231	R/W	0h	1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin 0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed Reset type: SYSRSn
6	GPIO230	R/W	0h	1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin 0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed Reset type: SYSRSn

Table 10-60. GPHLOCK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	GPIO229	R/W	0h	1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin 0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed Reset type: SYSRSn
4	GPIO228	R/W	0h	1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin 0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed Reset type: SYSRSn
3	GPIO227	R/W	0h	1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin 0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed Reset type: SYSRSn
2	GPIO226	R/W	0h	1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin 0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed Reset type: SYSRSn
1	GPIO225	R/W	0h	1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin 0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed Reset type: SYSRSn
0	GPIO224	R/W	0h	1: Locks changes to the bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx registers which control the same pin 0: Bits in GPyMUX1, GPyMUX2, GPyDIR, GPyINV, GPyODR, GPyAMSEL, GPyGMUX1, GPyGMUX2 and GPyCSELx register which control the same pin can be changed Reset type: SYSRSn

10.9.2.50 GPHCR Register (Offset = 1FEh) [Reset = 0h]

GPHCR is shown in Figure 10-53 and described in Table 10-61.

Return to the Summary Table.

GPIO H Lock Commit Register (GPIO224 to 255)
GPIO Configuration Lock Commit for GPIO:
1: Locks changes to the bit in GPyLOCK register which controls the same pin
0: Bit in the GPyLOCK register which controls the same pin can be changed

Figure 10-53. GPHCR Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	GPIO253	GPIO252	GPIO251	RESERVED	GPIO249	GPIO248
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO247	RESERVED	GPIO245	GPIO244	RESERVED	GPIO242	GPIO241	GPIO240
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO239	GPIO238	GPIO237	GPIO236	RESERVED	RESERVED	GPIO233	GPIO232
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO231	GPIO230	GPIO229	GPIO228	GPIO227	GPIO226	GPIO225	GPIO224
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 10-61. GPHCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30	RESERVED	R/W	0h	Reserved
29	GPIO253	R/W	0h	1: Locks changes to the bit in GPyLOCK register which controls the same pin 0: Bit in the GPyLOCK register which controls the same pin can be changed Reset type: SYSRSn
28	GPIO252	R/W	0h	1: Locks changes to the bit in GPyLOCK register which controls the same pin 0: Bit in the GPyLOCK register which controls the same pin can be changed Reset type: SYSRSn
27	GPIO251	R/W	0h	1: Locks changes to the bit in GPyLOCK register which controls the same pin 0: Bit in the GPyLOCK register which controls the same pin can be changed Reset type: SYSRSn
26	RESERVED	R/W	0h	Reserved
25	GPIO249	R/W	0h	1: Locks changes to the bit in GPyLOCK register which controls the same pin 0: Bit in the GPyLOCK register which controls the same pin can be changed Reset type: SYSRSn
24	GPIO248	R/W	0h	1: Locks changes to the bit in GPyLOCK register which controls the same pin 0: Bit in the GPyLOCK register which controls the same pin can be changed Reset type: SYSRSn

Table 10-61. GPHCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23	GPIO247	R/W	0h	1: Locks changes to the bit in GPyLOCK register which controls the same pin 0: Bit in the GPyLOCK register which controls the same pin can be changed Reset type: SYSRSn
22	RESERVED	R/W	0h	Reserved
21	GPIO245	R/W	0h	1: Locks changes to the bit in GPyLOCK register which controls the same pin 0: Bit in the GPyLOCK register which controls the same pin can be changed Reset type: SYSRSn
20	GPIO244	R/W	0h	1: Locks changes to the bit in GPyLOCK register which controls the same pin 0: Bit in the GPyLOCK register which controls the same pin can be changed Reset type: SYSRSn
19	RESERVED	R/W	0h	Reserved
18	GPIO242	R/W	0h	1: Locks changes to the bit in GPyLOCK register which controls the same pin 0: Bit in the GPyLOCK register which controls the same pin can be changed Reset type: SYSRSn
17	GPIO241	R/W	0h	1: Locks changes to the bit in GPyLOCK register which controls the same pin 0: Bit in the GPyLOCK register which controls the same pin can be changed Reset type: SYSRSn
16	GPIO240	R/W	0h	1: Locks changes to the bit in GPyLOCK register which controls the same pin 0: Bit in the GPyLOCK register which controls the same pin can be changed Reset type: SYSRSn
15	GPIO239	R/W	0h	1: Locks changes to the bit in GPyLOCK register which controls the same pin 0: Bit in the GPyLOCK register which controls the same pin can be changed Reset type: SYSRSn
14	GPIO238	R/W	0h	1: Locks changes to the bit in GPyLOCK register which controls the same pin 0: Bit in the GPyLOCK register which controls the same pin can be changed Reset type: SYSRSn
13	GPIO237	R/W	0h	1: Locks changes to the bit in GPyLOCK register which controls the same pin 0: Bit in the GPyLOCK register which controls the same pin can be changed Reset type: SYSRSn
12	GPIO236	R/W	0h	1: Locks changes to the bit in GPyLOCK register which controls the same pin 0: Bit in the GPyLOCK register which controls the same pin can be changed Reset type: SYSRSn
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	GPIO233	R/W	0h	1: Locks changes to the bit in GPyLOCK register which controls the same pin 0: Bit in the GPyLOCK register which controls the same pin can be changed Reset type: SYSRSn

Table 10-61. GPHCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	GPIO232	R/W	0h	1: Locks changes to the bit in GPyLOCK register which controls the same pin 0: Bit in the GPyLOCK register which controls the same pin can be changed Reset type: SYSRSn
7	GPIO231	R/W	0h	1: Locks changes to the bit in GPyLOCK register which controls the same pin 0: Bit in the GPyLOCK register which controls the same pin can be changed Reset type: SYSRSn
6	GPIO230	R/W	0h	1: Locks changes to the bit in GPyLOCK register which controls the same pin 0: Bit in the GPyLOCK register which controls the same pin can be changed Reset type: SYSRSn
5	GPIO229	R/W	0h	1: Locks changes to the bit in GPyLOCK register which controls the same pin 0: Bit in the GPyLOCK register which controls the same pin can be changed Reset type: SYSRSn
4	GPIO228	R/W	0h	1: Locks changes to the bit in GPyLOCK register which controls the same pin 0: Bit in the GPyLOCK register which controls the same pin can be changed Reset type: SYSRSn
3	GPIO227	R/W	0h	1: Locks changes to the bit in GPyLOCK register which controls the same pin 0: Bit in the GPyLOCK register which controls the same pin can be changed Reset type: SYSRSn
2	GPIO226	R/W	0h	1: Locks changes to the bit in GPyLOCK register which controls the same pin 0: Bit in the GPyLOCK register which controls the same pin can be changed Reset type: SYSRSn
1	GPIO225	R/W	0h	1: Locks changes to the bit in GPyLOCK register which controls the same pin 0: Bit in the GPyLOCK register which controls the same pin can be changed Reset type: SYSRSn
0	GPIO224	R/W	0h	1: Locks changes to the bit in GPyLOCK register which controls the same pin 0: Bit in the GPyLOCK register which controls the same pin can be changed Reset type: SYSRSn

10.9.3 GPIO_DATA_REGS Registers

Table 10-62 lists the memory-mapped registers for the GPIO_DATA_REGS registers. All register offset addresses not listed in Table 10-62 should be considered as reserved locations and the register contents should not be modified.

Table 10-62. GPIO_DATA_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	GPADAT	GPIO A Data Register (GPIO0 to 31)		Go
2h	GPASET	GPIO A Data Set Register (GPIO0 to 31)		Go
4h	GPACLEAR	GPIO A Data Clear Register (GPIO0 to 31)		Go
6h	GPATOGGLE	GPIO A Data Toggle Register (GPIO0 to 31)		Go
8h	GPBDAT	GPIO B Data Register (GPIO32 to 63)		Go
Ah	GPBSET	GPIO B Data Set Register (GPIO32 to 63)		Go
Ch	GPBCLEAR	GPIO B Data Clear Register (GPIO32 to 63)		Go
Eh	GPBTOGGLE	GPIO B Data Toggle Register (GPIO32 to 63)		Go
38h	GPHDAT	GPIO H Data Register (GPIO224 to 255)		Go

Complex bit access types are encoded to fit into small table cells. Table 10-63 shows the codes that are used for access types in this section.

Table 10-63. GPIO_DATA_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

10.9.3.1 GPADAT Register (Offset = 0h) [Reset = 0h]

GPADAT is shown in Figure 10-54 and described in Table 10-64.

Return to the Summary Table.

GPIO A Data Register (GPIO0 to 31)
Reading this register indicates the current status of the GPIO pin, irrespective of which mode the pin is in. Writing to this register will set the GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero.
DESIGNER NOTE:
[1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register.

Figure 10-54. GPADAT Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 10-64. GPADAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R/W	0h	Data Register for this pin Reset type: SYSRSn
30	GPIO30	R/W	0h	Data Register for this pin Reset type: SYSRSn
29	GPIO29	R/W	0h	Data Register for this pin Reset type: SYSRSn
28	GPIO28	R/W	0h	Data Register for this pin Reset type: SYSRSn
27	GPIO27	R/W	0h	Data Register for this pin Reset type: SYSRSn
26	GPIO26	R/W	0h	Data Register for this pin Reset type: SYSRSn
25	GPIO25	R/W	0h	Data Register for this pin Reset type: SYSRSn
24	GPIO24	R/W	0h	Data Register for this pin Reset type: SYSRSn
23	GPIO23	R/W	0h	Data Register for this pin Reset type: SYSRSn
22	GPIO22	R/W	0h	Data Register for this pin Reset type: SYSRSn
21	GPIO21	R/W	0h	Data Register for this pin Reset type: SYSRSn

Table 10-64. GPADAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	GPIO20	R/W	0h	Data Register for this pin Reset type: SYSRSn
19	GPIO19	R/W	0h	Data Register for this pin Reset type: SYSRSn
18	GPIO18	R/W	0h	Data Register for this pin Reset type: SYSRSn
17	GPIO17	R/W	0h	Data Register for this pin Reset type: SYSRSn
16	GPIO16	R/W	0h	Data Register for this pin Reset type: SYSRSn
15	GPIO15	R/W	0h	Data Register for this pin Reset type: SYSRSn
14	GPIO14	R/W	0h	Data Register for this pin Reset type: SYSRSn
13	GPIO13	R/W	0h	Data Register for this pin Reset type: SYSRSn
12	GPIO12	R/W	0h	Data Register for this pin Reset type: SYSRSn
11	GPIO11	R/W	0h	Data Register for this pin Reset type: SYSRSn
10	GPIO10	R/W	0h	Data Register for this pin Reset type: SYSRSn
9	GPIO9	R/W	0h	Data Register for this pin Reset type: SYSRSn
8	GPIO8	R/W	0h	Data Register for this pin Reset type: SYSRSn
7	GPIO7	R/W	0h	Data Register for this pin Reset type: SYSRSn
6	GPIO6	R/W	0h	Data Register for this pin Reset type: SYSRSn
5	GPIO5	R/W	0h	Data Register for this pin Reset type: SYSRSn
4	GPIO4	R/W	0h	Data Register for this pin Reset type: SYSRSn
3	GPIO3	R/W	0h	Data Register for this pin Reset type: SYSRSn
2	GPIO2	R/W	0h	Data Register for this pin Reset type: SYSRSn
1	GPIO1	R/W	0h	Data Register for this pin Reset type: SYSRSn
0	GPIO0	R/W	0h	Data Register for this pin Reset type: SYSRSn

10.9.3.2 GPASET Register (Offset = 2h) [Reset = 0h]

GPASET is shown in Figure 10-55 and described in Table 10-65.

Return to the Summary Table.

GPIO A Data Set Register (GPIO0 to 31)
Writing a 1 will force GPIO output data latch to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 10-55. GPASET Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h

Table 10-65. GPASET Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
30	GPIO30	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
29	GPIO29	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
28	GPIO28	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
27	GPIO27	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
26	GPIO26	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
25	GPIO25	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
24	GPIO24	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
23	GPIO23	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
22	GPIO22	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
21	GPIO21	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
20	GPIO20	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
19	GPIO19	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn

Table 10-65. GPASET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO18	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
17	GPIO17	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
16	GPIO16	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
15	GPIO15	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
14	GPIO14	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
13	GPIO13	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
12	GPIO12	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
11	GPIO11	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
10	GPIO10	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
9	GPIO9	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
8	GPIO8	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
7	GPIO7	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
6	GPIO6	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
5	GPIO5	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
4	GPIO4	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
3	GPIO3	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
2	GPIO2	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
1	GPIO1	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
0	GPIO0	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn

10.9.3.3 GPACLEAR Register (Offset = 4h) [Reset = 0h]

GPACLEAR is shown in Figure 10-56 and described in Table 10-66.

Return to the Summary Table.

GPIO A Data Clear Register (GPIO0 to 31)
Writing a 1 will force GPIO0 output data latch to 0.
Writes of 0 are ignored.
Always reads back a 0.

Figure 10-56. GPACLEAR Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h

Table 10-66. GPACLEAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
30	GPIO30	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
29	GPIO29	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
28	GPIO28	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
27	GPIO27	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
26	GPIO26	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
25	GPIO25	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
24	GPIO24	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
23	GPIO23	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
22	GPIO22	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
21	GPIO21	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
20	GPIO20	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
19	GPIO19	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn

Table 10-66. GPACLEAR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO18	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
17	GPIO17	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
16	GPIO16	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
15	GPIO15	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
14	GPIO14	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
13	GPIO13	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
12	GPIO12	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
11	GPIO11	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
10	GPIO10	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
9	GPIO9	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
8	GPIO8	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
7	GPIO7	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
6	GPIO6	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
5	GPIO5	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
4	GPIO4	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
3	GPIO3	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
2	GPIO2	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
1	GPIO1	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
0	GPIO0	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn

10.9.3.4 GPATOGGLE Register (Offset = 6h) [Reset = 0h]

GPATOGGLE is shown in Figure 10-57 and described in Table 10-67.

Return to the Summary Table.

GPIO A Data Toggle Register (GPIO0 to 31)
Writing a 1 will toggle GPIO0 output data latch 1 to 0 or 0 to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 10-57. GPATOGGLE Register

31	30	29	28	27	26	25	24
GPIO31	GPIO30	GPIO29	GPIO28	GPIO27	GPIO26	GPIO25	GPIO24
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h
23	22	21	20	19	18	17	16
GPIO23	GPIO22	GPIO21	GPIO20	GPIO19	GPIO18	GPIO17	GPIO16
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h
15	14	13	12	11	10	9	8
GPIO15	GPIO14	GPIO13	GPIO12	GPIO11	GPIO10	GPIO9	GPIO8
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h
7	6	5	4	3	2	1	0
GPIO7	GPIO6	GPIO5	GPIO4	GPIO3	GPIO2	GPIO1	GPIO0
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h

Table 10-67. GPATOGGLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	GPIO31	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
30	GPIO30	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
29	GPIO29	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
28	GPIO28	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
27	GPIO27	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
26	GPIO26	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
25	GPIO25	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
24	GPIO24	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
23	GPIO23	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
22	GPIO22	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
21	GPIO21	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
20	GPIO20	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
19	GPIO19	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn

Table 10-67. GPATOGGLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	GPIO18	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
17	GPIO17	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
16	GPIO16	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
15	GPIO15	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
14	GPIO14	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
13	GPIO13	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
12	GPIO12	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
11	GPIO11	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
10	GPIO10	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
9	GPIO9	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
8	GPIO8	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
7	GPIO7	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
6	GPIO6	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
5	GPIO5	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
4	GPIO4	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
3	GPIO3	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
2	GPIO2	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
1	GPIO1	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
0	GPIO0	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn

10.9.3.5 GPBDAT Register (Offset = 8h) [Reset = 0h]

GPBDAT is shown in Figure 10-58 and described in Table 10-68.

Return to the Summary Table.

GPIO B Data Register (GPIO32 to 63)
Reading this register indicates the current status of the GPIO pin, irrespective of which mode the pin is in. Writing to this register will set the GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero.
DESIGNER NOTE:
[1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register.

Figure 10-58. GPBDAT Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	GPIO61	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO39	RESERVED	GPIO37	RESERVED	GPIO35	GPIO34	GPIO33	GPIO32
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 10-68. GPBDAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30	RESERVED	R/W	0h	Reserved
29	GPIO61	R/W	0h	Data Register for this pin Reset type: SYSRSn
28	GPIO60	R/W	0h	Data Register for this pin Reset type: SYSRSn
27	GPIO59	R/W	0h	Data Register for this pin Reset type: SYSRSn
26	GPIO58	R/W	0h	Data Register for this pin Reset type: SYSRSn
25	GPIO57	R/W	0h	Data Register for this pin Reset type: SYSRSn
24	GPIO56	R/W	0h	Data Register for this pin Reset type: SYSRSn
23	GPIO55	R/W	0h	Data Register for this pin Reset type: SYSRSn
22	GPIO54	R/W	0h	Data Register for this pin Reset type: SYSRSn
21	GPIO53	R/W	0h	Data Register for this pin Reset type: SYSRSn
20	GPIO52	R/W	0h	Data Register for this pin Reset type: SYSRSn

Table 10-68. GPBDAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19	GPIO51	R/W	0h	Data Register for this pin Reset type: SYSRSn
18	GPIO50	R/W	0h	Data Register for this pin Reset type: SYSRSn
17	GPIO49	R/W	0h	Data Register for this pin Reset type: SYSRSn
16	GPIO48	R/W	0h	Data Register for this pin Reset type: SYSRSn
15	GPIO47	R/W	0h	Data Register for this pin Reset type: SYSRSn
14	GPIO46	R/W	0h	Data Register for this pin Reset type: SYSRSn
13	GPIO45	R/W	0h	Data Register for this pin Reset type: SYSRSn
12	GPIO44	R/W	0h	Data Register for this pin Reset type: SYSRSn
11	GPIO43	R/W	0h	Data Register for this pin Reset type: SYSRSn
10	GPIO42	R/W	0h	Data Register for this pin Reset type: SYSRSn
9	GPIO41	R/W	0h	Data Register for this pin Reset type: SYSRSn
8	GPIO40	R/W	0h	Data Register for this pin Reset type: SYSRSn
7	GPIO39	R/W	0h	Data Register for this pin Reset type: SYSRSn
6	RESERVED	R/W	0h	Reserved
5	GPIO37	R/W	0h	Data Register for this pin Reset type: SYSRSn
4	RESERVED	R/W	0h	Reserved
3	GPIO35	R/W	0h	Data Register for this pin Reset type: SYSRSn
2	GPIO34	R/W	0h	Data Register for this pin Reset type: SYSRSn
1	GPIO33	R/W	0h	Data Register for this pin Reset type: SYSRSn
0	GPIO32	R/W	0h	Data Register for this pin Reset type: SYSRSn

10.9.3.6 GPBSET Register (Offset = Ah) [Reset = 0h]

GPBSET is shown in Figure 10-59 and described in Table 10-69.

Return to the Summary Table.

GPIO B Data Set Register (GPIO32 to 63)
Writing a 1 will force GPIO output data latch to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 10-59. GPBSET Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	GPIO61	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h
7	6	5	4	3	2	1	0
GPIO39	RESERVED	GPIO37	RESERVED	GPIO35	GPIO34	GPIO33	GPIO32
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h

Table 10-69. GPBSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R-0/W	0h	Reserved
30	RESERVED	R-0/W	0h	Reserved
29	GPIO61	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
28	GPIO60	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
27	GPIO59	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
26	GPIO58	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
25	GPIO57	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
24	GPIO56	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
23	GPIO55	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
22	GPIO54	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
21	GPIO53	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
20	GPIO52	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
19	GPIO51	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
18	GPIO50	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn

Table 10-69. GPBSET Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	GPIO49	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
16	GPIO48	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
15	GPIO47	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
14	GPIO46	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
13	GPIO45	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
12	GPIO44	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
11	GPIO43	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
10	GPIO42	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
9	GPIO41	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
8	GPIO40	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
7	GPIO39	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
6	RESERVED	R-0/W	0h	Reserved
5	GPIO37	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
4	RESERVED	R-0/W	0h	Reserved
3	GPIO35	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
2	GPIO34	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
1	GPIO33	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn
0	GPIO32	R-0/W	0h	Output Set bit for this pin Reset type: SYSRSn

10.9.3.7 GPBCLEAR Register (Offset = Ch) [Reset = 0h]

GPBCLEAR is shown in Figure 10-60 and described in Table 10-70.

Return to the Summary Table.

GPIO B Data Clear Register (GPIO32 to 63)
Writing a 1 will force GPIO0 output data latch to 0.
Writes of 0 are ignored.
Always reads back a 0.

Figure 10-60. GPBCLEAR Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	GPIO61	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h
7	6	5	4	3	2	1	0
GPIO39	RESERVED	GPIO37	RESERVED	GPIO35	GPIO34	GPIO33	GPIO32
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h

Table 10-70. GPBCLEAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R-0/W	0h	Reserved
30	RESERVED	R-0/W	0h	Reserved
29	GPIO61	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
28	GPIO60	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
27	GPIO59	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
26	GPIO58	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
25	GPIO57	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
24	GPIO56	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
23	GPIO55	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
22	GPIO54	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
21	GPIO53	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
20	GPIO52	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
19	GPIO51	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
18	GPIO50	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn

Table 10-70. GPBCLEAR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	GPIO49	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
16	GPIO48	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
15	GPIO47	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
14	GPIO46	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
13	GPIO45	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
12	GPIO44	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
11	GPIO43	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
10	GPIO42	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
9	GPIO41	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
8	GPIO40	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
7	GPIO39	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
6	RESERVED	R-0/W	0h	Reserved
5	GPIO37	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
4	RESERVED	R-0/W	0h	Reserved
3	GPIO35	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
2	GPIO34	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
1	GPIO33	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn
0	GPIO32	R-0/W	0h	Output Clear bit for this pin Reset type: SYSRSn

10.9.3.8 GPBTOGGLE Register (Offset = Eh) [Reset = 0h]

GPBTOGGLE is shown in Figure 10-61 and described in Table 10-71.

Return to the Summary Table.

GPIO B Data Toggle Register (GPIO32 to 63)
Writing a 1 will toggle GPIO0 output data latch 1 to 0 or 0 to 1.
Writes of 0 are ignored.
Always reads back a 0.

Figure 10-61. GPBTOGGLE Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	GPIO61	GPIO60	GPIO59	GPIO58	GPIO57	GPIO56
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h
23	22	21	20	19	18	17	16
GPIO55	GPIO54	GPIO53	GPIO52	GPIO51	GPIO50	GPIO49	GPIO48
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h
15	14	13	12	11	10	9	8
GPIO47	GPIO46	GPIO45	GPIO44	GPIO43	GPIO42	GPIO41	GPIO40
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h
7	6	5	4	3	2	1	0
GPIO39	RESERVED	GPIO37	RESERVED	GPIO35	GPIO34	GPIO33	GPIO32
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h

Table 10-71. GPBTOGGLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R-0/W	0h	Reserved
30	RESERVED	R-0/W	0h	Reserved
29	GPIO61	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
28	GPIO60	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
27	GPIO59	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
26	GPIO58	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
25	GPIO57	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
24	GPIO56	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
23	GPIO55	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
22	GPIO54	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
21	GPIO53	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
20	GPIO52	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
19	GPIO51	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
18	GPIO50	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn

Table 10-71. GPBTOGGLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	GPIO49	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
16	GPIO48	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
15	GPIO47	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
14	GPIO46	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
13	GPIO45	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
12	GPIO44	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
11	GPIO43	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
10	GPIO42	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
9	GPIO41	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
8	GPIO40	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
7	GPIO39	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
6	RESERVED	R-0/W	0h	Reserved
5	GPIO37	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
4	RESERVED	R-0/W	0h	Reserved
3	GPIO35	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
2	GPIO34	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
1	GPIO33	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn
0	GPIO32	R-0/W	0h	Output Toggle Register GPIO pin Reset type: SYSRSn

10.9.3.9 GPHDAT Register (Offset = 38h) [Reset = 0h]

GPHDAT is shown in Figure 10-62 and described in Table 10-72.

Return to the Summary Table.

GPIO H Data Register (GPIO224 to 255)
Reading this register indicates the current status of the GPIO pin, irrespective of which mode the pin is in. Writing to this register will set the GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero.
DESIGNER NOTE:
[1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the

Figure 10-62. GPHDAT Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	GPIO253	GPIO252	GPIO251	RESERVED	GPIO249	GPIO248
R/W-0h	R/W-0h	R-0h	R/W-0h	R-0h	R/W-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
GPIO247	RESERVED	GPIO245	GPIO244	RESERVED	GPIO242	GPIO241	GPIO240
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
GPIO239	GPIO238	GPIO237	GPIO236	RESERVED	RESERVED	GPIO233	GPIO232
R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
GPIO231	GPIO230	GPIO229	GPIO228	GPIO227	GPIO226	GPIO225	GPIO224
R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 10-72. GPHDAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30	RESERVED	R/W	0h	Reserved
29	GPIO253	R	0h	Reading this register indicates the current status of this GPIO pin, irrespective of which mode the pin is in. Writing to this register will set this GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero. DESIGNER NOTE: [1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register. Reset type: SYSRSn
28	GPIO252	R/W	0h	Reading this register indicates the current status of this GPIO pin, irrespective of which mode the pin is in. Writing to this register will set this GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero. DESIGNER NOTE: [1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register. Reset type: SYSRSn

Table 10-72. GPHDAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	GPIO251	R	0h	Reading this register indicates the current status of this GPIO pin, irrespective of which mode the pin is in. Writing to this register will set this GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero. DESIGNER NOTE: [1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register. Reset type: SYSRSn
26	RESERVED	R/W	0h	Reserved
25	GPIO249	R	0h	Reading this register indicates the current status of this GPIO pin, irrespective of which mode the pin is in. Writing to this register will set this GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero. DESIGNER NOTE: [1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register. Reset type: SYSRSn
24	GPIO248	R	0h	Reading this register indicates the current status of this GPIO pin, irrespective of which mode the pin is in. Writing to this register will set this GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero. DESIGNER NOTE: [1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register. Reset type: SYSRSn
23	GPIO247	R	0h	Reading this register indicates the current status of this GPIO pin, irrespective of which mode the pin is in. Writing to this register will set this GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero. DESIGNER NOTE: [1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register. Reset type: SYSRSn
22	RESERVED	R/W	0h	Reserved
21	GPIO245	R/W	0h	Reading this register indicates the current status of this GPIO pin, irrespective of which mode the pin is in. Writing to this register will set this GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero. DESIGNER NOTE: [1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register. Reset type: SYSRSn

Table 10-72. GPHDAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	GPIO244	R/W	0h	Reading this register indicates the current status of this GPIO pin, irrespective of which mode the pin is in. Writing to this register will set this GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero. DESIGNER NOTE: [1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register. Reset type: SYSRSn
19	RESERVED	R/W	0h	Reserved
18	GPIO242	R/W	0h	Reading this register indicates the current status of this GPIO pin, irrespective of which mode the pin is in. Writing to this register will set this GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero. DESIGNER NOTE: [1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register. Reset type: SYSRSn
17	GPIO241	R/W	0h	Reading this register indicates the current status of this GPIO pin, irrespective of which mode the pin is in. Writing to this register will set this GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero. DESIGNER NOTE: [1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register. Reset type: SYSRSn
16	GPIO240	R/W	0h	Reading this register indicates the current status of this GPIO pin, irrespective of which mode the pin is in. Writing to this register will set this GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero. DESIGNER NOTE: [1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register. Reset type: SYSRSn
15	GPIO239	R/W	0h	Reading this register indicates the current status of this GPIO pin, irrespective of which mode the pin is in. Writing to this register will set this GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero. DESIGNER NOTE: [1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register. Reset type: SYSRSn

Table 10-72. GPHDAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
14	GPIO238	R/W	0h	Reading this register indicates the current status of this GPIO pin, irrespective of which mode the pin is in. Writing to this register will set this GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero. DESIGNER NOTE: [1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register. Reset type: SYSRSn
13	GPIO237	R/W	0h	Reading this register indicates the current status of this GPIO pin, irrespective of which mode the pin is in. Writing to this register will set this GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero. DESIGNER NOTE: [1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register. Reset type: SYSRSn
12	GPIO236	R	0h	Reading this register indicates the current status of this GPIO pin, irrespective of which mode the pin is in. Writing to this register will set this GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero. DESIGNER NOTE: [1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register. Reset type: SYSRSn
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	GPIO233	R/W	0h	Reading this register indicates the current status of this GPIO pin, irrespective of which mode the pin is in. Writing to this register will set this GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero. DESIGNER NOTE: [1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register. Reset type: SYSRSn
8	GPIO232	R/W	0h	Reading this register indicates the current status of this GPIO pin, irrespective of which mode the pin is in. Writing to this register will set this GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero. DESIGNER NOTE: [1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register. Reset type: SYSRSn

Table 10-72. GPHDAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	GPIO231	R/W	0h	Reading this register indicates the current status of this GPIO pin, irrespective of which mode the pin is in. Writing to this register will set this GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero. DESIGNER NOTE: [1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register. Reset type: SYSRSn
6	GPIO230	R/W	0h	Reading this register indicates the current status of this GPIO pin, irrespective of which mode the pin is in. Writing to this register will set this GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero. DESIGNER NOTE: [1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register. Reset type: SYSRSn
5	GPIO229	R	0h	Reading this register indicates the current status of this GPIO pin, irrespective of which mode the pin is in. Writing to this register will set this GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero. DESIGNER NOTE: [1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register. Reset type: SYSRSn
4	GPIO228	R/W	0h	Reading this register indicates the current status of this GPIO pin, irrespective of which mode the pin is in. Writing to this register will set this GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero. DESIGNER NOTE: [1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register. Reset type: SYSRSn
3	GPIO227	R/W	0h	Reading this register indicates the current status of this GPIO pin, irrespective of which mode the pin is in. Writing to this register will set this GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero. DESIGNER NOTE: [1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register. Reset type: SYSRSn

Table 10-72. GPHDAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	GPIO226	R/W	0h	Reading this register indicates the current status of this GPIO pin, irrespective of which mode the pin is in. Writing to this register will set this GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero. DESIGNER NOTE: [1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register. Reset type: SYSRSn
1	GPIO225	R/W	0h	Reading this register indicates the current status of this GPIO pin, irrespective of which mode the pin is in. Writing to this register will set this GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero. DESIGNER NOTE: [1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register. Reset type: SYSRSn
0	GPIO224	R/W	0h	Reading this register indicates the current status of this GPIO pin, irrespective of which mode the pin is in. Writing to this register will set this GPIO pin high or low if the pin is enabled for GPIO output mode, otherwise the value written is latched but ignored. The state of the output register latch will remain in its current state until the next write operation. A system reset will clear all bits and latched values to zero. DESIGNER NOTE: [1] Reading the GPIODAT register should reflect the state of the PIN (after qualification), not the state of the output latch of the GPIODAT register. Reset type: SYSRSn

10.9.4 GPIO_DATA_READ_REGS Registers

Table 10-73 lists the memory-mapped registers for the GPIO_DATA_READ_REGS registers. All register offset addresses not listed in Table 10-73 should be considered as reserved locations and the register contents should not be modified.

Table 10-73. GPIO_DATA_READ_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	GPADAT_R	GPIO A Data Read Register		Go
2h	GPBDAT_R	GPIO B Data Read Register		Go
Eh	GPHDAT_R	GPIO H Data Read Register		Go

Complex bit access types are encoded to fit into small table cells. Table 10-74 shows the codes that are used for access types in this section.

Table 10-74. GPIO_DATA_READ_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

10.9.4.1 GPADAT_R Register (Offset = 0h) [Reset = 0h]

GPADAT_R is shown in Figure 10-63 and described in Table 10-75.

Return to the Summary Table.

GPIO A Data Read Register.
Returns the contents of GPADAT register on a read, write to this register has no effect

Figure 10-63. GPADAT_R Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R-0h																															

Table 10-75. GPADAT_R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R	0h	A read from this register returns the contents of GPADAT register, writes have no impact Reset type: CPU1.SYSRSn

10.9.4.2 GPBDAT_R Register (Offset = 2h) [Reset = 0h]

GPBDAT_R is shown in Figure 10-64 and described in Table 10-76.

Return to the Summary Table.

GPIO B Data Read Register.
Returns the contents of GPBDAT register on a read, write to this register has no effect

Figure 10-64. GPBDAT_R Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R-0h																															

Table 10-76. GPBDAT_R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R	0h	A read from this register returns the contents of GPBDAT register, writes have no impact Reset type: CPU1.SYSRSn

10.9.4.3 GPHDAT_R Register (Offset = Eh) [Reset = 0h]

GPHDAT_R is shown in Figure 10-65 and described in Table 10-77.

Return to the Summary Table.

GPIO H Data Read Register.
Returns the contents of GPHDAT register on a read, write to this register has no effect

Figure 10-65. GPHDAT_R Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R-0h																															

Table 10-77. GPHDAT_R Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R	0h	A read from this register returns the contents of GPHDAT register, writes have no impact Reset type: CPU1.SYSRSn

10.9.5 GPIO Registers to Driverlib Functions

Table 10-78. GPIO Registers to Driverlib Functions

File	Driverlib Function
GPACTRL	
gpio.c	GPIO_setQualificationPeriod
GPAQSEL1	
gpio.c	GPIO_setQualificationMode
gpio.c	GPIO_getQualificationMode
GPAQSEL2	
-	See GPAQSEL1
GPAMUX1	
gpio.c	GPIO_setPinConfig
GPAMUX2	
-	See GPAMUX1
GPADIR	
gpio.c	GPIO_setDirectionMode
gpio.c	GPIO_getDirectionMode
GPAPUD	
gpio.c	GPIO_setPadConfig
gpio.c	GPIO_getPadConfig
GPAINV	
gpio.c	GPIO_setPadConfig
gpio.c	GPIO_getPadConfig
GPAODR	
gpio.c	GPIO_setPadConfig
gpio.c	GPIO_getPadConfig
GPAAMSEL	
-	
GPAGMUX1	
gpio.c	GPIO_setPinConfig
GPAGMUX2	

Table 10-78. GPIO Registers to Driverlib Functions (continued)

File	Driverlib Function
-	See GPAGMUX1
GPACSEL1	
gpio.c	GPIO_setMasterCore
GPACSEL2	
-	See GPACSEL1
GPACSEL3	
-	See GPACSEL1
GPACSEL4	
-	See GPACSEL1
GPALOCK	
gpio.h	GPIO_lockPortConfig
gpio.h	GPIO_unlockPortConfig
GPACR	
gpio.h	GPIO_commitPortConfig
GPBCTRL	
-	See GPACTRL
GPBQSEL1	
-	See GPAQSEL1
GPBQSEL2	
-	See GPAQSEL1
GPBMUX1	
-	See GPAMUX1
GPBMUX2	
-	See GPAMUX1
GPBDIR	
-	See GPADIR
GPBPUD	
-	See GPAPUD
GPBINV	
-	See GPAINV
GPBODR	
-	See GPAODR
GPBGMUX1	
-	See GPAGMUX1
GPBGMUX2	
-	See GPAGMUX1
GPBCSEL1	
-	See GPACSEL1
GPBCSEL2	
-	See GPACSEL1
GPBCSEL3	
-	See GPACSEL1
GPBCSEL4	
-	See GPACSEL1
GPBLOCK	

Table 10-78. GPIO Registers to Driverlib Functions (continued)

File	Driverlib Function
-	See GPALOCK
GPBCR	
-	See GPACR
GPHCTRL	
-	See GPACTRL
GPHQSEL1	
-	See GPAQSEL1
GPHQSEL2	
-	See GPAQSEL1
GPHMUX1	
-	See GPAMUX1
GPHMUX2	
-	See GPAMUX1
GPHINV	
-	See GPAINV
GPHAMSEL	
-	
GPHGMUX1	
-	See GPAGMUX1
GPHGMUX2	
-	See GPAGMUX1
GPHCSEL1	
-	See GPACSEL1
GPHCSEL2	
-	See GPACSEL1
GPHCSEL3	
-	See GPACSEL1
GPHCSEL4	
-	See GPACSEL1
GPHLOCK	
-	See GPALOCK
GPHCR	
-	See GPACR
GPADAT	
gpio.h	GPIO_readPin
gpio.h	GPIO_readPortData
gpio.h	GPIO_writePortData
GPASET	
gpio.h	GPIO_writePin
gpio.h	GPIO_setPortPins
GPACLEAR	
gpio.h	GPIO_writePin
gpio.h	GPIO_clearPortPins
GPATOGGLE	
gpio.h	GPIO_togglePin

Table 10-78. GPIO Registers to Driverlib Functions (continued)

File	Driverlib Function
gpio.h	GPIO_togglePortPins
GPBDAT	
-	See GPADAT
GPBSET	
-	See GPASET
GPBCLEAR	
-	See GPACLEAR
GPBTOGGLE	
-	See GPATOGGLE
GPHDAT	
-	See GPADAT
GPADAT_R	
-	
GPBDAT_R	
-	
GPHDAT_R	
-	

This page intentionally left blank.

Chapter 11

Crossbar (X-BAR)

The crossbars (referred to as X-BAR throughout this chapter) provide flexibility to connect device inputs, outputs, and internal resources in a variety of configurations.

The device contains a total of six X-BARs:

- Input X-BAR
- CLB Input X-BAR
- Output X-BAR
- CLB Output X-BAR
- CLB X-BAR
- ePWM X-BAR

Each of the X-BARs is named according to where they take signals. For example, the Input X-BAR and CLB Input X-BAR bring external signals “in” to the device. The Output X-BAR and CLB Output X-BAR take internal signals “out” of the device to a GPIO. The CLB X-BAR and ePWM X-BAR take signals to the CLB and ePWM modules, respectively.

You can read more about each of these X-BARs in the following sections.

11.1 Input X-BAR and CLB Input X-BAR

On this device, the Input X-BAR is used to route signals from a GPIO to many different IP blocks such as the ADC(s), eCAP(s), ePWM(s), and external interrupts. The Input X-BAR has access to every GPIO and can route each signal to any (or multiple) of the IP blocks previously mentioned. The digital input of AIOs are also available on the Input X-BAR. This flexibility relieves some of the constraints on peripheral muxing by just requiring any GPIO pin to be available. It is important to note that the function selected on the GPIO mux does not affect the Input X-BAR. The Input X-BAR simply connects the signal on the input buffer to the selected destination. Therefore, you can do things such as route the output of one peripheral to another (that is, measure the output of an ePWM with an eCAP for a frequency test).

The Input X-BAR is configured by way of the INPUTxSELECT registers. The available IP destinations for each INPUTx is shown in Figure 11-1 and Table 11-1 . For more information on configuration, see the INPUT_XBAR_REGS register definitions in the *XBAR Registers* section.

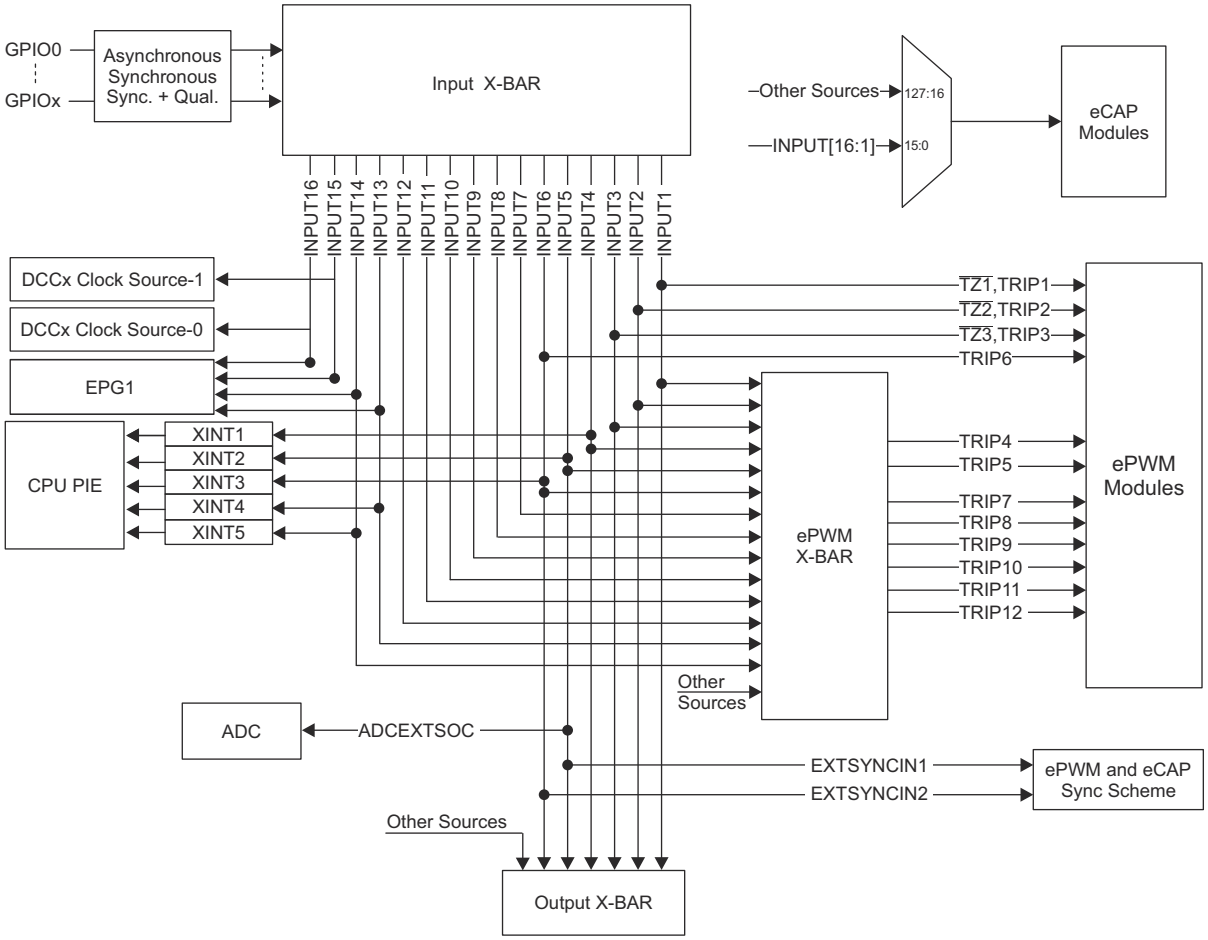


Figure 11-1. Input X-BAR

Table 11-1. Input X-BAR Destinations

Input	eCAP / HRCAP	ePWM X-BAR	CLB X-BAR	Output X-BAR	CPU XINT	ePWM Trip	ADC Start of Conversion	ePWM / eCAP Sync	DCC	ERAD	EPG
INPUT1	Yes	Yes	Yes	Yes		TZ1, TRIP1				Yes	
INPUT2	Yes	Yes	Yes	Yes		TZ2, TRIP2				Yes	
INPUT3	Yes	Yes	Yes	Yes		TZ3, TRIP3				Yes	
INPUT4	Yes	Yes	Yes	Yes	XINT1					Yes	
INPUT5	Yes	Yes	Yes	Yes	XINT2		ADCEXTSOC	EXTSYNCIN1		Yes	
INPUT6	Yes	Yes	Yes	Yes	XINT3	TRIP6		EXTSYNCIN2		Yes	
INPUT7	Yes	Yes	Yes							Yes	
INPUT8	Yes	Yes	Yes							Yes	
INPUT9	Yes	Yes	Yes							Yes	
INPUT10	Yes	Yes	Yes							Yes	
INPUT11	Yes	Yes	Yes						Yes	Yes	
INPUT12	Yes	Yes	Yes						Yes	Yes	
INPUT13	Yes	Yes	Yes		XINT4					Yes	EPG1 EPGIN1
INPUT14	Yes	Yes	Yes		XINT5					Yes	EPG1 EPGIN2
INPUT15	Yes								Yes	Yes	EPG1 EPGIN3
INPUT16	Yes								Yes	Yes	EPG1 EPGIN4

11.1.1 CLB Input X-BAR

The CLB Input X-BAR is architecturally identical to the Input X-BAR. The only difference is the destination for each INPUTx. The destination for each INPUTx is only the CLB Tiles as shown in Table 11-2. This allows for GPIOs to be accessed by the CLB tiles without using the combination of Input X-BAR and CLB X-BAR.

Table 11-2. CLB Input X-BAR Destinations

Input	Destinations
INPUT1	CLB Tiles
INPUT2	CLB Tiles
INPUT3	CLB Tiles
INPUT4	CLB Tiles
INPUT5	CLB Tiles
INPUT6	CLB Tiles
INPUT7	CLB Tiles, EPWM XBAR
INPUT8	CLB Tiles, EPWM XBAR
INPUT9	CLB Tiles, EPWM XBAR
INPUT10	CLB Tiles, EPWM XBAR
INPUT11	CLB Tiles, EPWM XBAR, DCC
INPUT12	CLB Tiles, EPWM XBAR, DCC
INPUT13	CLB Tiles, EPWM XBAR
INPUT14	CLB Tiles, EPWM XBAR
INPUT15	CLB Tiles
INPUT16	CLB Tiles

11.2 ePWM, CLB, and GPIO Output X-BAR

This section describes the ePWM, CLB, and GPIO Output X-BAR.

11.2.1 ePWM X-BAR

The ePWM X-BAR brings signals to the ePWM modules. Specifically, the ePWM X-BAR is connected to the Digital Compare (DC) submodule of each ePWM module for actions such as tripzones and syncing. Refer to the *Enhanced Pulse Width Modulator (ePWM)* chapter for more information on additional ways the DC submodule can be used. Figure 11-2 shows the architecture of the ePWM X-BAR. Note that the architecture of the ePWM X-BAR is identical to the architecture of the GPIO Output X-BAR (with the exception of the output latch).

11.2.1.1 ePWM X-BAR Architecture

The ePWM X-BAR has eight outputs that are routed to each ePWM module. Figure 11-2 represents the architecture of a single output but it is identical to the architecture of all of the other outputs.

First, determine the signal(s) that should be passed to the ePWM by referencing Table 11-3. You may select up to one signal per mux (32 total muxes) for each TRIPx output. Select the inputs to each mux via the TRIPxMUX0TO15CFG and TRIPxMUX16TO31CFG registers. In order to pass any signal through to the ePWM, you must also enable the mux in the TRIPxMUXENABLE register. All muxes that are enabled will be logically ORed before being passed on to the respective TRIPx signal on the ePWM. You may also optionally invert the signal via the TRIPOUTINV register.

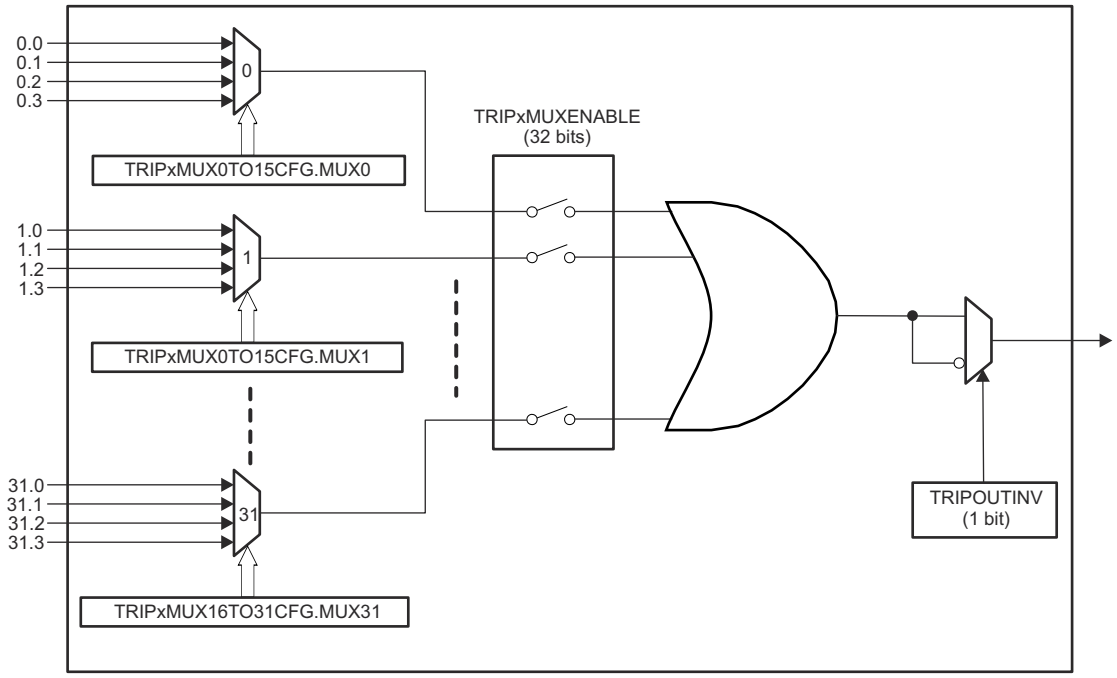


Figure 11-2. ePWM X-BAR Architecture - Single Output

Table 11-3. ePWM X-BAR Mux Configuration Table

Mux	0	1	2	3
0	CMPSS1.CTRIPH	CMPSS1.CTRIPH_OR_CTRIPL	ADCAEVT1	ECAP1OUT
1	CMPSS1.CTRIPL	INPUTXBAR1	CLB1_OUT4	ADCCEVT1
2	CMPSS2.CTRIPH	CMPSS2.CTRIPH_OR_CTRIPL	ADCAEVT2	ECAP2OUT
3	CMPSS2.CTRIPL	INPUTXBAR2	CLB1_OUT5	ADCCEVT2
4	CMPSS3.CTRIPH	CMPSS3.CTRIPH_OR_CTRIPL	ADCAEVT3	ECAP3OUT
5	CMPSS3.CTRIPL	INPUTXBAR3	CLB2_OUT4	ADCCEVT3
6	CMPSS4.CTRIPH	CMPSS4.CTRIPH_OR_CTRIPL	ADCAEVT4	Reserved
7	CMPSS4.CTRIPL	INPUTXBAR4	CLB2_OUT5	ADCCEVT4
8	Reserved	Reserved	ADCBEVT1	Reserved
9	Reserved	INPUTXBAR5	CLB3_OUT4	Reserved
10	Reserved	Reserved	ADCBEVT2	Reserved
11	Reserved	INPUTXBAR6	CLB3_OUT5	Reserved
12	Reserved	Reserved	ADCBEVT3	Reserved
13	Reserved	ADCSOCAO	CLB4_OUT4	Reserved
14	Reserved	Reserved	ADCBEVT4	EXTSYNCOUT
15	Reserved	ADCSOCBO	CLB4_OUT5	Reserved
16	SD1FLT1.CEVT1	SD1FLT1.CEVT1_OR_CEVT2	Reserved	Reserved
17	SD1FLT1.CEVT2	INPUTXBAR7	CLB INPUTXBAR7	CLA HALT
18	SD1FLT2.CEVT1	SD1FLT2.CEVT1_OR_CEVT2	Reserved	Reserved
19	SD1FLT2.CEVT2	INPUTXBAR8	CLB INPUTXBAR8	ERRORSTS ERROR
20	SD1FLT3.CEVT1	SD1FLT3.CEVT1_OR_CEVT2	Reserved	FSIA RX_TRIG1
21	SD1FLT3.CEVT2	INPUTXBAR9	CLB INPUTXBAR9	Reserved
22	SD1FLT4.CEVT1	SD1FLT4.CEVT1_OR_CEVT2	Reserved	Reserved
23	SD1FLT4.CEVT2	INPUTXBAR10	CLB INPUTXBAR10	Reserved
24	SD2FLT1.CEVT1	SD2FLT1.CEVT1_OR_CEVT2	Reserved	Reserved
25	SD2FLT1.CEVT2	INPUTXBAR11	MCANA.FEVT0	CLB INPUTXBAR11
26	SD2FLT2.CEVT1	SD2FLT2.CEVT1_OR_CEVT2	Reserved	Reserved
27	SD2FLT2.CEVT2	INPUTXBAR12	MCANA.FEVT1	CLB INPUTXBAR12
28	SD2FLT3.CEVT1	SD2FLT3.CEVT1_OR_CEVT2	Reserved	Reserved
29	SD2FLT3.CEVT2	INPUTXBAR13	MCANA.FEVT2	CLB INPUTXBAR13
30	SD2FLT4.CEVT1	SD2FLT4.CEVT1_OR_CEVT2	Reserved	Reserved
31	SD2FLT4.CEVT2	INPUTXBAR14	ERRORSTS ERROR	CLB INPUTXBAR14

Note

Do not use "Reserved" signals in your application.

11.2.2 CLB X-BAR

The CLB X-BAR brings signals to the CLB modules. Figure 11-3 shows the architecture of the CLB X-BAR. Note that the architecture of the CLB X-BAR is identical to the architecture of the GPIO Output X-BAR (with the exception of the output latch).

11.2.2.1 CLB X-BAR Architecture

The CLB X-BAR has eight outputs that are routed to each CLB module. Figure 11-3 represents the architecture of a single output but it is identical to the architecture of all of the other outputs.

First, determine the signal(s) that should be passed to the CLB by referencing Table 11-4. You may select up to one signal per mux (31 total muxes) for each AUXSIGx output. Select the inputs to each mux via the AUXSIGxMUX0TO15CFG and AUXSIGxMUX16TO31CFG registers. In order to pass any signal through to the CLB, you must also enable the mux in the AUXSIGxMUXENABLE register. All muxes which are enabled will be logically ORed before being passed on to the respective AUXSIGx signal on the CLB. You may also optionally invert the signal via the AUXSIGOUTINV register.

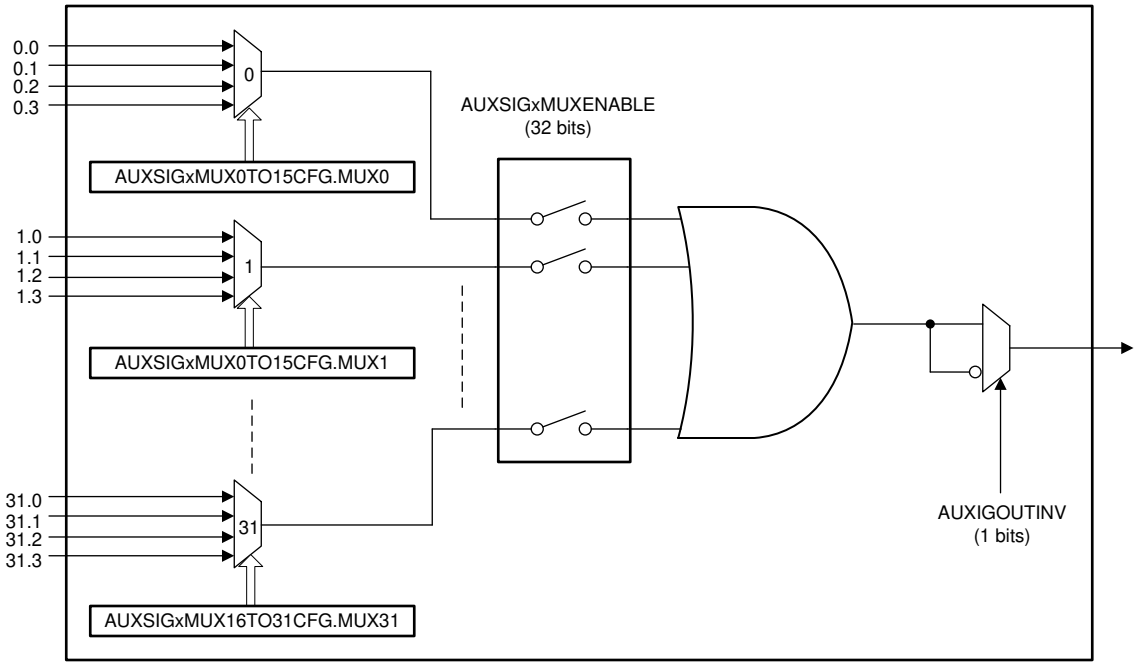


Figure 11-3. CLB X-BAR Architecture - Single Output

Table 11-4. CLB X-BAR Mux Configuration Table

Mux	0	1	2	3
0	CMPSS1.CTRIPH	CMPSS1.CTRIPH_OR_CTRIPL	ADCAEVT1	ECAP1OUT
1	CMPSS1.CTRIPL	INPUTXBAR1	CLB1_OUT4	ADCCEVT1
2	CMPSS2.CTRIPH	CMPSS2.CTRIPH_OR_CTRIPL	ADCAEVT2	ECAP2OUT
3	CMPSS2.CTRIPL	INPUTXBAR2	CLB1_OUT5	ADCCEVT2
4	CMPSS3.CTRIPH	CMPSS3.CTRIPH_OR_CTRIPL	ADCAEVT3	ECAP3OUT
5	CMPSS3.CTRIPL	INPUTXBAR3	CLB2_OUT4	ADCCEVT3
6	CMPSS4.CTRIPH	CMPSS4.CTRIPH_OR_CTRIPL	ADCAEVT4	Reserved
7	CMPSS4.CTRIPL	INPUTXBAR4	CLB2_OUT5	ADCCEVT4
8	Reserved	Reserved	ADCBEVT1	Reserved
9	Reserved	INPUTXBAR5	CLB3_4	Reserved
10	Reserved	Reserved	ADCBEVT2	Reserved
11	Reserved	INPUTXBAR6	CLB3_5	Reserved
12	Reserved	Reserved	ADCBEVT3	Reserved
13	Reserved	ADCSOCAO	CLB4_4	Reserved
14	Reserved	Reserved	ADCBEVT4	EXTSYNCOUT
15	Reserved	ADCSOCBO	CLB4_5	Reserved
16	SD1FLT1.COMPH	SD1FLT1.COMPH_OR_COMPL	SD1FLT1.COMPZ	SD1FLT1.DRINT
17	SD1FLT1.COMPL	INPUTXBAR7	Reserved	CLA HALT
18	SD1FLT2.COMPH	SD1FLT2.COMPH_OR_COMPL	SD1FLT2.COMPZ	SD1FLT2.DRINT
19	SD1FLT2.COMPL	INPUTXBAR8	Reserved	ERRORSTS ERROR
20	SD1FLT3.COMPH	SD1FLT3.COMPH_OR_COMPL	SD1FLT3.COMPZ	SD1FLT3.DRINT
21	SD1FLT3.COMPL	INPUTXBAR9	Reserved	Reserved
22	SD1FLT4.COMPH	SD1FLT4.COMPH_OR_COMPL	SD1FLT4.COMPZ	SD1FLT4.DRINT
23	SD1FLT4.COMPL	INPUTXBAR10	Reserved	Reserved
24	SD2FLT1.COMPH	SD2FLT1.COMPH_OR_COMPL	SD2FLT1.COMPZ	SD2FLT1.DRINT
25	SD2FLT1.COMPL	INPUTXBAR11	MCANA.FEVT0	Reserved
26	SD2FLT2.COMPH	SD2FLT2.COMPH_OR_COMPL	SD2FLT2.COMPZ	SD2FLT2.DRINT
27	SD2FLT2.COMPL	INPUTXBAR12	MCANA.FEVT1	Reserved
28	SD2FLT3.COMPH	SD2FLT3.COMPH_OR_COMPL	SD2FLT3.COMPZ	SD2FLT3.DRINT
29	SD2FLT3.COMPL	INPUTXBAR13	MCANA.FEVT2	Reserved
30	SD2FLT4.COMPH	SD2FLT4.COMPH_OR_COMPL	SD2FLT4.COMPZ	SD2FLT4.DRINT
31	SD2FLT4.COMPL	INPUTXBAR14	ERRORSTS ERROR	Reserved

11.2.3 GPIO Output X-BAR

The GPIO Output X-BAR takes signals from inside the device and brings them out to a GPIO. Figure 11-4 shows the architecture of the GPIO Output X-BAR. The X-BAR contains eight outputs and each will contain at least one position on the GPIO mux, denoted as OUTPUTXBARx. The X-BAR allows the selection of a single signal or a logical OR of up to 32 signals.

11.2.3.1 GPIO Output X-BAR Architecture

The GPIO Output X-BAR has eight outputs that are routed to the GPIO module. Figure 11-4 represents the architecture of a single output, but it is identical to the architecture of all of the other outputs. It is worth noting that the architecture of the Output X-BAR (with the exception of the output latch) is identical to the architecture of the ePWM X-BAR.

First, determine the signal(s) that should be passed to the GPIO by referencing Table 11-5. You may select up to one signal per mux (32 total muxes) for each OUTPUTXBARx output. Select the inputs to each mux via the OUTPUTxMUX0TO15CFG and OUTPUTxMUX16TO31CFG registers.

In order to pass any signal through to the GPIO, you must also enable the mux in the OUTPUTxMUXENABLE register. All muxes which are enabled will be logically ORed before being passed on to the respective OUTPUTx signal on the GPIO module. You may also optionally invert the signal via the OUTPUTINV register. The signal will only be seen on the GPIO if the proper OUTPUTx muxing options are selected via the GpioCtrlRegs.GPxMUX and GpioCtrlRegs.GPxGMUX registers.

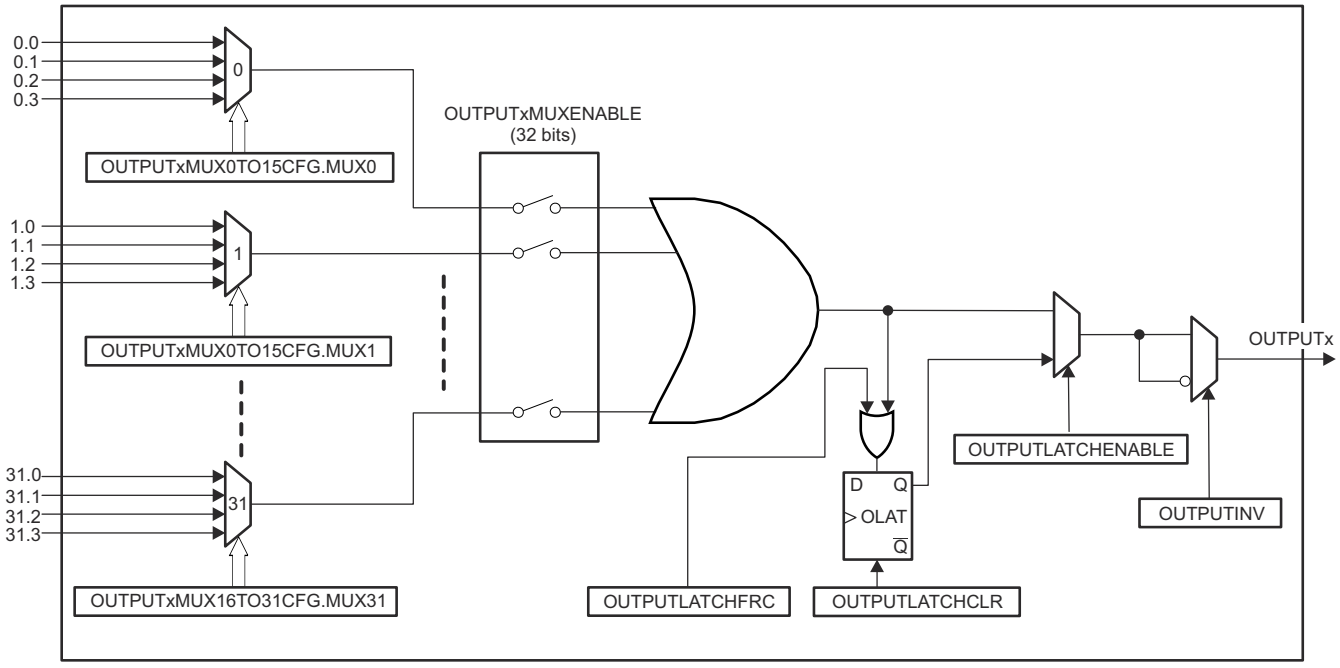


Figure 11-4. GPIO Output X-BAR Architecture

Table 11-5. OUTPUT X-BAR Mux Configuration Table

Mux	0	1	2	3
0	CMPSS1.CTRIPOUTH	CMPSS1.CTRIPOUTH_OR_CTRIPOUTL	ADCAEVT1	ECAP1OUT
1	CMPSS1.CTRIPOUTL	INPUTXBAR1	CLB1_OUT4_1	ADCCEVT1
2	CMPSS2.CTRIPOUTH	CMPSS2.CTRIPOUTH_OR_CTRIPOUTL	ADCAEVT2	ECAP2OUT
3	CMPSS2.CTRIPOUTL	INPUTXBAR2	CLB1_OUT5_1	ADCCEVT2
4	CMPSS3.CTRIPOUTH	CMPSS3.CTRIPOUTH_OR_CTRIPOUTL	ADCAEVT3	ECAP3OUT
5	CMPSS3.CTRIPOUTL	INPUTXBAR3	CLB2_OUT4_1	ADCCEVT3
6	CMPSS4.CTRIPOUTH	CMPSS4.CTRIPOUTH_OR_CTRIPOUTL	ADCAEVT4	Reserved
7	CMPSS4.CTRIPOUTL	INPUTXBAR4	CLB2_OUT5_1	ADCCEVT4
8	Reserved	Reserved	Reserved	Reserved
9	Reserved	INPUTXBAR5	Reserved	Reserved
10	Reserved	Reserved	Reserved	Reserved
11	Reserved	INPUTXBAR6	Reserved	Reserved
12	Reserved	Reserved	Reserved	Reserved
13	Reserved	ADCSOCAO	Reserved	Reserved
14	Reserved	Reserved	Reserved	EXTSYNCOUT
15	Reserved	ADCSOCBO	Reserved	Reserved
16	SD1FLT1.CEVT1	SD1FLT1.CEVT1_OR_CEVT2	Reserved	Reserved
17	SD1FLT1.CEVT2	Reserved	Reserved	CLA HALT
18	SD1FLT2.CEVT1	SD1FLT2.CEVT1_OR_CEVT2	Reserved	Reserved
19	SD1FLT2.CEVT2	Reserved	Reserved	ERRORSTS ERROR
20	SD1FLT3.CEVT1	SD1FLT3.CEVT1_OR_CEVT2	Reserved	Reserved
21	SD1FLT3.CEVT2	Reserved	Reserved	FSIA RX_TRIG2
22	SD1FLT4.CEVT1	SD1FLT4.CEVT1_OR_CEVT2	Reserved	Reserved
23	SD1FLT4.CEVT2	Reserved	Reserved	Reserved
24	SD2FLT1.CEVT1	SD2FLT1.CEVT1_OR_CEVT2	Reserved	Reserved
25	SD2FLT1.CEVT2	Reserved	Reserved	Reserved
26	SD2FLT2.CEVT1	SD2FLT2.CEVT1_OR_CEVT2	Reserved	Reserved
27	SD2FLT2.CEVT2	Reserved	Reserved	Reserved
28	SD2FLT3.CEVT1	SD2FLT3.CEVT1_OR_CEVT2	Reserved	Reserved
29	SD2FLT3.CEVT2	Reserved	Reserved	Reserved
30	SD2FLT4.CEVT1	SD2FLT4.CEVT1_OR_CEVT2	Reserved	EPG1 EPGOUT0
31	SD2FLT4.CEVT2	Reserved	ERRORSTS ERROR	EPG1 EPGOUT1

Note

Do not use "Reserved" signals in your application.

11.2.4 CLB Output X-BAR

The CLB Output X-BAR takes signals from inside the CLB Tiles and brings them out to a GPIO.

11.2.4.1 CLB Output X-BAR Architecture

The CLB Output X-BAR has eight outputs that are routed to the GPIO module. CLB Output X-BAR architecture is identical to the architecture of the GPIO Output X-BAR. First, determine the signal(s) that should be passed to the GPIO by referencing Table 11-6.

Table 11-6. CLB Output X-BAR Mux Configuration Table

Mux	0	1	2	3
0	CLB1_OUT0	Reserved	Reserved	OUTPUT XBAR1
1	CLB1_OUT1	Reserved	Reserved	OUTPUT XBAR2
2	CLB1_OUT2	Reserved	Reserved	Reserved
3	CLB1_OUT3	Reserved	Reserved	Reserved
4	CLB1_OUT4	Reserved	Reserved	Reserved
5	CLB1_OUT5	Reserved	Reserved	Reserved
6	CLB1_OUT6	Reserved	Reserved	Reserved
7	CLB1_OUT7	Reserved	Reserved	Reserved
8	CLB2_OUT0	Reserved	Reserved	OUTPUT XBAR3
9	CLB2_OUT1	Reserved	Reserved	OUTPUT XBAR4
10	CLB2_OUT2	Reserved	Reserved	Reserved
11	CLB2_OUT3	Reserved	Reserved	Reserved
12	CLB2_OUT4	Reserved	Reserved	Reserved
13	CLB2_OUT5	Reserved	Reserved	Reserved
14	CLB2_OUT6	Reserved	Reserved	Reserved
15	CLB2_OUT7	Reserved	Reserved	Reserved
16	CLB3_OUT0	Reserved	Reserved	OUTPUT XBAR5
17	CLB3_OUT1	Reserved	Reserved	OUTPUT XBAR6
18	CLB3_OUT2	Reserved	Reserved	Reserved
19	CLB3_OUT3	Reserved	Reserved	Reserved
20	CLB3_OUT4	Reserved	Reserved	Reserved
21	CLB3_OUT5	Reserved	Reserved	Reserved
22	CLB3_OUT6	Reserved	Reserved	Reserved
23	CLB3_OUT7	Reserved	Reserved	Reserved
24	CLB4_OUT0	Reserved	Reserved	OUTPUT XBAR7
25	CLB4_OUT1	Reserved	Reserved	OUTPUT XBAR8
26	CLB4_OUT2	Reserved	Reserved	Reserved
27	CLB4_OUT3	Reserved	Reserved	Reserved
28	CLB4_OUT4	Reserved	Reserved	Reserved
29	CLB4_OUT5	Reserved	Reserved	Reserved
30	CLB4_OUT6	Reserved	Reserved	EPG1 EPGOUT2
31	CLB4_OUT7	Reserved	Reserved	EPG1 EPGOUT3

Note

Do not use "Reserved" signals in your application.

11.2.5 X-BAR Flags

With the exception of the CMPSS signals, the ePWM X-BAR and the Output X-BAR have all of the same input signals. Due to the inputs being similar, the ePWM X-BAR and Output X-BAR leverage a single set of input flags to indicate which input signals have been triggered. This allows software to check the input flags when an event occurs. See Figure 11-5 for more information. There is a bit allocated for each input signal in one of the XBARFLGx registers. The flag will remain set until cleared through the appropriate XBARCLRx register.

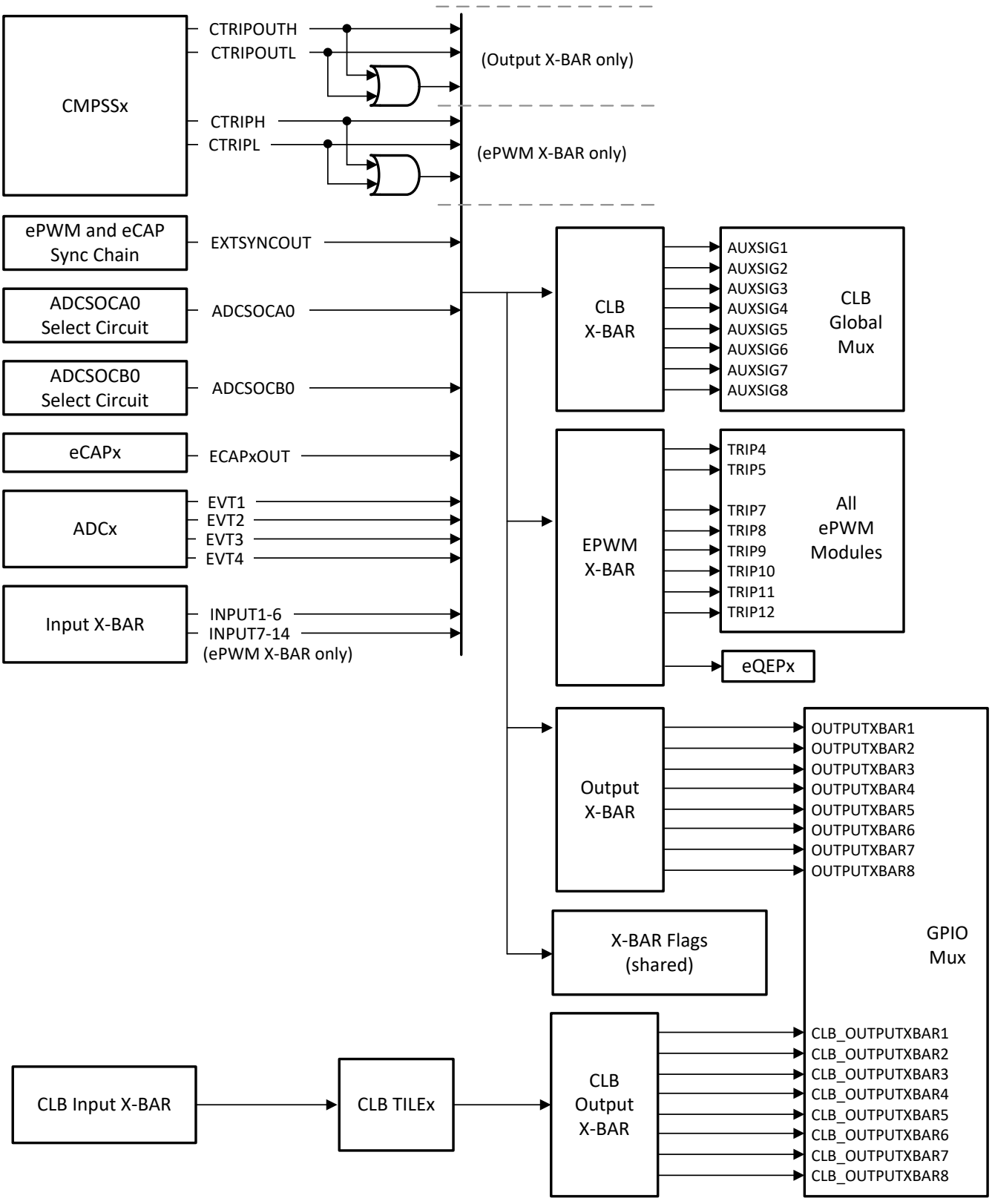


Figure 11-5. ePWM and Output X-BARs Sources

11.3 XBAR Registers

This section describes the Crossbar registers.

11.3.1 XBAR Base Address Table

Table 11-7. XBAR Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
InputXbarRegs	INPUT_XBAR_REGS	INPUTXBAR_BASE	0x0000_7900	YES	-	-	-	YES
XbarRegs	XBAR_REGS	XBAR_BASE	0x0000_7920	YES	-	-	-	YES
ClbInputXbarRegs	INPUT_XBAR_REGS	CLBINPUTXBAR_BASE	0x0000_7960	YES	-	-	-	YES
EPwmXbarRegs	EPWM_XBAR_REGS	EPWMXBAR_BASE	0x0000_7A00	YES	-	-	-	YES
ClbXbarRegs	CLB_XBAR_REGS	CLBXBAR_BASE	0x0000_7A40	YES	-	-	-	YES
OutputXbarRegs	OUTPUT_XBAR_REGS	OUTPUTXBAR_BASE	0x0000_7A80	YES	-	-	-	YES
ClbOutputXbarRegs	OUTPUT_XBAR_REGS	CLBOUTPUTXBAR_BASE	0x0000_7BC0	YES	-	-	-	YES

11.3.2 INPUT_XBAR_REGS Registers

Table 11-8 lists the memory-mapped registers for the INPUT_XBAR_REGS registers. All register offset addresses not listed in Table 11-8 should be considered as reserved locations and the register contents should not be modified.

Table 11-8. INPUT_XBAR_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	INPUT1SELECT	INPUT1 Input Select Register (GPIO0 to x)	EALLOW	Go
1h	INPUT2SELECT	INPUT2 Input Select Register (GPIO0 to x)	EALLOW	Go
2h	INPUT3SELECT	INPUT3 Input Select Register (GPIO0 to x)	EALLOW	Go
3h	INPUT4SELECT	INPUT4 Input Select Register (GPIO0 to x)	EALLOW	Go
4h	INPUT5SELECT	INPUT5 Input Select Register (GPIO0 to x)	EALLOW	Go
5h	INPUT6SELECT	INPUT6 Input Select Register (GPIO0 to x)	EALLOW	Go
6h	INPUT7SELECT	INPUT7 Input Select Register (GPIO0 to x)	EALLOW	Go
7h	INPUT8SELECT	INPUT8 Input Select Register (GPIO0 to x)	EALLOW	Go
8h	INPUT9SELECT	INPUT9 Input Select Register (GPIO0 to x)	EALLOW	Go
9h	INPUT10SELECT	INPUT10 Input Select Register (GPIO0 to x)	EALLOW	Go
Ah	INPUT11SELECT	INPUT11 Input Select Register (GPIO0 to x)	EALLOW	Go
Bh	INPUT12SELECT	INPUT12 Input Select Register (GPIO0 to x)	EALLOW	Go
Ch	INPUT13SELECT	INPUT13 Input Select Register (GPIO0 to x)	EALLOW	Go
Dh	INPUT14SELECT	INPUT14 Input Select Register (GPIO0 to x)	EALLOW	Go
Eh	INPUT15SELECT	INPUT15 Input Select Register (GPIO0 to x)	EALLOW	Go
Fh	INPUT16SELECT	INPUT16 Input Select Register (GPIO0 to x)	EALLOW	Go
1Eh	INPUTSELECTLOCK	Input Select Lock Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 11-9 shows the codes that are used for access types in this section.

Table 11-9. INPUT_XBAR_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.

Table 11-9. INPUT_XBAR_REGS Access Type Codes (continued)

Access Type	Code	Description
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

11.3.2.1 INPUT1SELECT Register (Offset = 0h) [Reset = FFFEh]

INPUT1SELECT is shown in Figure 11-6 and described in Table 11-10.

Return to the Summary Table.

INPUT1 Input Select Register (GPIO0 to x)

Figure 11-6. INPUT1SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 11-10. INPUT1SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT1 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

11.3.2.2 INPUT2SELECT Register (Offset = 1h) [Reset = FFFEh]

INPUT2SELECT is shown in Figure 11-7 and described in Table 11-11.

Return to the Summary Table.

INPUT2 Input Select Register (GPIO0 to x)

Figure 11-7. INPUT2SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 11-11. INPUT2SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT2 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

11.3.2.3 INPUT3SELECT Register (Offset = 2h) [Reset = FFFEh]

INPUT3SELECT is shown in Figure 11-8 and described in Table 11-12.

Return to the Summary Table.

INPUT3 Input Select Register (GPIO0 to x)

Figure 11-8. INPUT3SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 11-12. INPUT3SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT3 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

11.3.2.4 INPUT4SELECT Register (Offset = 3h) [Reset = FFFEh]

INPUT4SELECT is shown in Figure 11-9 and described in Table 11-13.

Return to the Summary Table.

INPUT4 Input Select Register (GPIO0 to x)

Figure 11-9. INPUT4SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 11-13. INPUT4SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT4 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

11.3.2.5 INPUT5SELECT Register (Offset = 4h) [Reset = FFFEh]

INPUT5SELECT is shown in Figure 11-10 and described in Table 11-14.

Return to the Summary Table.

INPUT5 Input Select Register (GPIO0 to x)

Figure 11-10. INPUT5SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 11-14. INPUT5SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT5 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

11.3.2.6 INPUT6SELECT Register (Offset = 5h) [Reset = FFFEh]

INPUT6SELECT is shown in Figure 11-11 and described in Table 11-15.

Return to the Summary Table.

INPUT6 Input Select Register (GPIO0 to x)

Figure 11-11. INPUT6SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 11-15. INPUT6SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT6 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

11.3.2.7 INPUT7SELECT Register (Offset = 6h) [Reset = FFFEh]

INPUT7SELECT is shown in Figure 11-12 and described in Table 11-16.

Return to the Summary Table.

INPUT7 Input Select Register (GPIO0 to x)

Figure 11-12. INPUT7SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 11-16. INPUT7SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT7 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

11.3.2.8 INPUT8SELECT Register (Offset = 7h) [Reset = FFFEh]

INPUT8SELECT is shown in Figure 11-13 and described in Table 11-17.

Return to the Summary Table.

INPUT8 Input Select Register (GPIO0 to x)

Figure 11-13. INPUT8SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 11-17. INPUT8SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT8 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

11.3.2.9 INPUT9SELECT Register (Offset = 8h) [Reset = FFFEh]

INPUT9SELECT is shown in Figure 11-14 and described in Table 11-18.

Return to the Summary Table.

INPUT9 Input Select Register (GPIO0 to x)

Figure 11-14. INPUT9SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 11-18. INPUT9SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT9 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

11.3.2.10 INPUT10SELECT Register (Offset = 9h) [Reset = FFFEh]

INPUT10SELECT is shown in Figure 11-15 and described in Table 11-19.

Return to the Summary Table.

INPUT10 Input Select Register (GPIO0 to x)

Figure 11-15. INPUT10SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 11-19. INPUT10SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT10 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

11.3.2.11 INPUT11SELECT Register (Offset = Ah) [Reset = FFFEh]

INPUT11SELECT is shown in Figure 11-16 and described in Table 11-20.

Return to the Summary Table.

INPUT11 Input Select Register (GPIO0 to x)

Figure 11-16. INPUT11SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 11-20. INPUT11SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT11 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

11.3.2.12 INPUT12SELECT Register (Offset = Bh) [Reset = FFFEh]

INPUT12SELECT is shown in Figure 11-17 and described in Table 11-21.

Return to the Summary Table.

INPUT12 Input Select Register (GPIO0 to x)

Figure 11-17. INPUT12SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 11-21. INPUT12SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT12 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

11.3.2.13 INPUT13SELECT Register (Offset = Ch) [Reset = FFFEh]

INPUT13SELECT is shown in Figure 11-18 and described in Table 11-22.

Return to the Summary Table.

INPUT13 Input Select Register (GPIO0 to x)

Figure 11-18. INPUT13SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 11-22. INPUT13SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT13 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

11.3.2.14 INPUT14SELECT Register (Offset = Dh) [Reset = FFFEh]

INPUT14SELECT is shown in Figure 11-19 and described in Table 11-23.

Return to the Summary Table.

INPUT14 Input Select Register (GPIO0 to x)

Figure 11-19. INPUT14SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 11-23. INPUT14SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT14 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

11.3.2.15 INPUT15SELECT Register (Offset = Eh) [Reset = FFFEh]

INPUT15SELECT is shown in Figure 11-20 and described in Table 11-24.

Return to the Summary Table.

INPUT15 Input Select Register (GPIO0 to x)

Figure 11-20. INPUT15SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 11-24. INPUT15SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT15 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

11.3.2.16 INPUT16SELECT Register (Offset = Fh) [Reset = FFFEh]

INPUT16SELECT is shown in Figure 11-21 and described in Table 11-25.

Return to the Summary Table.

INPUT16 Input Select Register (GPIO0 to x)

Figure 11-21. INPUT16SELECT Register

15	14	13	12	11	10	9	8
SELECT							
R/W-FFFEh							
7	6	5	4	3	2	1	0
SELECT							
R/W-FFFEh							

Table 11-25. INPUT16SELECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SELECT	R/W	FFFEh	Select GPIO for INPUT16 signal: 0x0 : Select GPIO0 0x1 : Select GPIO1 0x2 : Select GPIO2 ... 0xFFFD: '1' will be driven to the destination 0xFFFE: '1' will be driven to the destination 0xFFFF: '0' will be driven to the destination NOTE: SELECT value greater than the available number of GPIO pins on a device (except 0xFFFF) will cause the destination to be driven '1'. Reset type: CPU1.SYSRSn

11.3.2.17 INPUTSELECTLOCK Register (Offset = 1Eh) [Reset = 0h]

INPUTSELECTLOCK is shown in Figure 11-22 and described in Table 11-26.

Return to the Summary Table.

Input Select Lock Register.
Any bit in this register, once set can only be cleared through SYSRSn. Write of 0 to any bit of this register has no effect. Reads to the registers which have LOCK protection are always allowed.

Figure 11-22. INPUTSELECTLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
INPUT16SELECT	INPUT15SELECT	INPUT14SELECT	INPUT13SELECT	INPUT12SELECT	INPUT11SELECT	INPUT10SELECT	INPUT9SELECT
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h
7	6	5	4	3	2	1	0
INPUT8SELECT	INPUT7SELECT	INPUT6SELECT	INPUT5SELECT	INPUT4SELECT	INPUT3SELECT	INPUT2SELECT	INPUT1SELECT
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 11-26. INPUTSELECTLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	INPUT16SELECT	R/WSonce	0h	Lock bit for INPUT16SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
14	INPUT15SELECT	R/WSonce	0h	Lock bit for INPUT15SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
13	INPUT14SELECT	R/WSonce	0h	Lock bit for INPUT14SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
12	INPUT13SELECT	R/WSonce	0h	Lock bit for INPUT13SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
11	INPUT12SELECT	R/WSonce	0h	Lock bit for INPUT12SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
10	INPUT11SELECT	R/WSonce	0h	Lock bit for INPUT11SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
9	INPUT10SELECT	R/WSonce	0h	Lock bit for INPUT10SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn

Table 11-26. INPUTSELECTLOCK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	INPUT9SELECT	R/WSonce	0h	Lock bit for INPUT9SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
7	INPUT8SELECT	R/WSonce	0h	Lock bit for INPUT8SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
6	INPUT7SELECT	R/WSonce	0h	Lock bit for INPUT7SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
5	INPUT6SELECT	R/WSonce	0h	Lock bit for INPUT6SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
4	INPUT5SELECT	R/WSonce	0h	Lock bit for INPUT5SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
3	INPUT4SELECT	R/WSonce	0h	Lock bit for INPUT4SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
2	INPUT3SELECT	R/WSonce	0h	Lock bit for INPUT3SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
1	INPUT2SELECT	R/WSonce	0h	Lock bit for INPUT2SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn
0	INPUT1SELECT	R/WSonce	0h	Lock bit for INPUT1SELECT Register 0: Register is not locked 1: Register is locked Reset type: CPU1.SYSRSn

11.3.3 XBAR_REGS Registers

Table 11-27 lists the memory-mapped registers for the XBAR_REGS registers. All register offset addresses not listed in Table 11-27 should be considered as reserved locations and the register contents should not be modified.

Table 11-27. XBAR_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	XBARFLG1	X-Bar Input Flag Register 1		Go
2h	XBARFLG2	X-Bar Input Flag Register 2		Go
4h	XBARFLG3	X-Bar Input Flag Register 3		Go
6h	XBARFLG4	X-Bar Input Flag Register 4		Go
8h	XBARCLR1	X-Bar Input Flag Clear Register 1		Go
Ah	XBARCLR2	X-Bar Input Flag Clear Register 2		Go
Ch	XBARCLR3	X-Bar Input Flag Clear Register 3		Go
Eh	XBARCLR4	X-Bar Input Flag Clear Register 4		Go

Complex bit access types are encoded to fit into small table cells. Table 11-28 shows the codes that are used for access types in this section.

Table 11-28. XBAR_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

11.3.3.1 XBARFLG1 Register (Offset = 0h) [Reset = 0h]

XBARFLG1 is shown in Figure 11-23 and described in Table 11-29.

Return to the Summary Table.

X-Bar Input Flag Register 1

Figure 11-23. XBARFLG1 Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
CMPSS4_CTRIPOUTH	CMPSS4_CTRIPOUTL	CMPSS3_CTRIPOUTH	CMPSS3_CTRIPOUTL	CMPSS2_CTRIPOUTH	CMPSS2_CTRIPOUTL	CMPSS1_CTRIPOUTH	CMPSS1_CTRIPOUTL
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
CMPSS4_CTRIPH	CMPSS4_CTRIPL	CMPSS3_CTRIPH	CMPSS3_CTRIPL	CMPSS2_CTRIPH	CMPSS2_CTRIPL	CMPSS1_CTRIPH	CMPSS1_CTRIPL
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 11-29. XBARFLG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved
30	RESERVED	R	0h	Reserved
29	RESERVED	R	0h	Reserved
28	RESERVED	R	0h	Reserved
27	RESERVED	R	0h	Reserved
26	RESERVED	R	0h	Reserved
25	RESERVED	R	0h	Reserved
24	RESERVED	R	0h	Reserved
23	CMPSS4_CTRIPOUTH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
22	CMPSS4_CTRIPOUTL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
21	CMPSS3_CTRIPOUTH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 11-29. XBARFLG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	CMPSS3_CTRIPOUTL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
19	CMPSS2_CTRIPOUTH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
18	CMPSS2_CTRIPOUTL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
17	CMPSS1_CTRIPOUTH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
16	CMPSS1_CTRIPOUTL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
15	RESERVED	R	0h	Reserved
14	RESERVED	R	0h	Reserved
13	RESERVED	R	0h	Reserved
12	RESERVED	R	0h	Reserved
11	RESERVED	R	0h	Reserved
10	RESERVED	R	0h	Reserved
9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	CMPSS4_CTRIPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
6	CMPSS4_CTRIPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 11-29. XBARFLG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	CMPSS3_CTRIPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
4	CMPSS3_CTRIPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
3	CMPSS2_CTRIPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
2	CMPSS2_CTRIPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
1	CMPSS1_CTRIPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
0	CMPSS1_CTRIPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

11.3.3.2 XBARFLG2 Register (Offset = 2h) [Reset = 0h]

XBARFLG2 is shown in Figure 11-24 and described in Table 11-30.

Return to the Summary Table.

X-Bar Input Flag Register 2

Figure 11-24. XBARFLG2 Register

31	30	29	28	27	26	25	24
ADCCEVT1	ADCBEVT4	ADCBEVT3	ADCBEVT2	ADCBEVT1	ADCAEVT4	ADCAEVT3	ADCAEVT2
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
ADCAEVT1	EXTSYNCOUT	RESERVED	RESERVED	RESERVED	ECAP3_OUT	ECAP2_OUT	ECAP1_OUT
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
INPUT14	INPUT13	INPUT12	INPUT11	INPUT10	INPUT9	INPUT8	INPUT7
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
ADCSOCB	ADCSOCA	INPUT6	INPUT5	INPUT4	INPUT3	INPUT2	INPUT1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 11-30. XBARFLG2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ADCCEVT1	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
30	ADCBEVT4	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
29	ADCBEVT3	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
28	ADCBEVT2	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
27	ADCBEVT1	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 11-30. XBARFLG2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	ADCAEVT4	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
25	ADCAEVT3	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
24	ADCAEVT2	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
23	ADCAEVT1	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
22	EXTSYNCOUT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
21	RESERVED	R	0h	Reserved
20	RESERVED	R	0h	Reserved
19	RESERVED	R	0h	Reserved
18	ECAP3_OUT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
17	ECAP2_OUT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
16	ECAP1_OUT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 11-30. XBARFLG2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15	INPUT14	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
14	INPUT13	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
13	INPUT12	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
12	INPUT11	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
11	INPUT10	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
10	INPUT9	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
9	INPUT8	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
8	INPUT7	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 11-30. XBARFLG2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	ADCSOCB	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
6	ADCSOCA	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
5	INPUT6	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
4	INPUT5	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
3	INPUT4	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
2	INPUT3	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
1	INPUT2	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
0	INPUT1	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

11.3.3.3 XBARFLG3 Register (Offset = 4h) [Reset = 0h]

XBARFLG3 is shown in Figure 11-25 and described in Table 11-31.

Return to the Summary Table.

X-Bar Input Flag Register 3

Figure 11-25. XBARFLG3 Register

31	30	29	28	27	26	25	24
SD1FLT4_DRINT	SD1FLT4_COMPZ	SD1FLT3_DRINT	SD1FLT3_COMPZ	SD1FLT2_DRINT	SD1FLT2_COMPZ	SD1FLT1_DRINT	SD1FLT1_COMPZ
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
RESERVED	SD2FLT4_COMPH	SD2FLT4_COMPL	SD2FLT3_COMPH	SD2FLT3_COMPL	SD2FLT2_COMPH	SD2FLT2_COMPL	SD2FLT1_COMPH
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
SD2FLT1_COMPL	SD1FLT4_COMPH	SD1FLT4_COMPL	SD1FLT3_COMPH	SD1FLT3_COMPL	SD1FLT2_COMPH	SD1FLT2_COMPL	SD1FLT1_COMPH
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
SD1FLT1_COMPL	RESERVED	RESERVED	RESERVED	RESERVED	ADCCEVT4	ADCCEVT3	ADCCEVT2
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 11-31. XBARFLG3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SD1FLT4_DRINT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
30	SD1FLT4_COMPZ	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
29	SD1FLT3_DRINT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
28	SD1FLT3_COMPZ	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 11-31. XBARFLG3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	SD1FLT2_DRINT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
26	SD1FLT2_COMPZ	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
25	SD1FLT1_DRINT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
24	SD1FLT1_COMPZ	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
23	RESERVED	R	0h	Reserved
22	SD2FLT4_COMPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
21	SD2FLT4_COMPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
20	SD2FLT3_COMPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
19	SD2FLT3_COMPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 11-31. XBARFLG3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	SD2FLT2_COMPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
17	SD2FLT2_COMPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
16	SD2FLT1_COMPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
15	SD2FLT1_COMPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
14	SD1FLT4_COMPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
13	SD1FLT4_COMPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
12	SD1FLT3_COMPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
11	SD1FLT3_COMPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 11-31. XBARFLG3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	SD1FLT2_COMPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
9	SD1FLT2_COMPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
8	SD1FLT1_COMPH	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
7	SD1FLT1_COMPL	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
6	RESERVED	R	0h	Reserved
5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3	RESERVED	R	0h	Reserved
2	ADCCEVT4	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
1	ADCCEVT3	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
0	ADCCEVT2	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

11.3.3.4 XBARFLG4 Register (Offset = 6h) [Reset = 0h]

XBARFLG4 is shown in Figure 11-26 and described in Table 11-32.

Return to the Summary Table.

X-Bar Input Flag Register 4

Figure 11-26. XBARFLG4 Register

31	30	29	28	27	26	25	24
CLAHALT	RESERVED	RESERVED	ERRORSTS_ERROR	RESERVED	RESERVED	RESERVED	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
CLB4_5_1	CLB4_4_1	CLB3_5_1	CLB3_4_1	CLB2_5_1	CLB2_4_1	CLB1_5_1	CLB1_4_1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	MCANA_FEVT2	MCANA_FEVT1	MCANA_FEVT0	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
SD2FLT4_DRINT	SD2FLT4_COMPZ	SD2FLT3_DRINT	SD2FLT3_COMPZ	SD2FLT2_DRINT	SD2FLT2_COMPZ	SD2FLT1_DRINT	SD2FLT1_COMPZ
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 11-32. XBARFLG4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CLAHALT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
30	RESERVED	R	0h	Reserved
29	RESERVED	R	0h	Reserved
28	ERRORSTS_ERROR	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: ERRORSTS_ERROR input was triggered 0: ERRORSTS_ERROR Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
27	RESERVED	R	0h	Reserved
26	RESERVED	R	0h	Reserved
25	RESERVED	R	0h	Reserved
24	RESERVED	R	0h	Reserved
23	CLB4_5_1	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 11-32. XBARFLG4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
22	CLB4_4_1	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
21	CLB3_5_1	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
20	CLB3_4_1	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
19	CLB2_5_1	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
18	CLB2_4_1	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
17	CLB1_5_1	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
16	CLB1_4_1	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
15	RESERVED	R	0h	Reserved
14	RESERVED	R	0h	Reserved
13	RESERVED	R	0h	Reserved
12	RESERVED	R	0h	Reserved
11	MCANA_FEVT2	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: MCANA_FEVT2 input was triggered 0: MCANA_FEVT2 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 11-32. XBARFLG4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MCANA_FEVT1	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: MCANA_FEVT1 input was triggered 0: MCANA_FEVT1 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
9	MCANA_FEVT0	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: MCANA_FEVT0 input was triggered 0: MCANA_FEVT0 Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
8	RESERVED	R	0h	Reserved
7	SD2FLT4_DRINT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
6	SD2FLT4_COMPZ	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
5	SD2FLT3_DRINT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
4	SD2FLT3_COMPZ	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
3	SD2FLT2_DRINT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
2	SD2FLT2_COMPZ	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 11-32. XBARFLG4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	SD2FLT1_DRINT	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
0	SD2FLT1_COMPZ	R	0h	This register is used to Flag the inputs of the X-Bars to provide software knowledge of the input sources which got triggered. 1: Corresponding Input was triggered 0: Corresponding Input was not triggered Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

11.3.3.5 XBARCLR1 Register (Offset = 8h) [Reset = 0h]

XBARCLR1 is shown in Figure 11-27 and described in Table 11-33.

Return to the Summary Table.

X-Bar Input Flag Clear Register 1

Figure 11-27. XBARCLR1 Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
CMPSS4_CTRIPOUTH	CMPSS4_CTRIPOUTL	CMPSS3_CTRIPOUTH	CMPSS3_CTRIPOUTL	CMPSS2_CTRIPOUTH	CMPSS2_CTRIPOUTL	CMPSS1_CTRIPOUTH	CMPSS1_CTRIPOUTL
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
CMPSS4_CTRIPH	CMPSS4_CTRIPL	CMPSS3_CTRIPH	CMPSS3_CTRIPL	CMPSS2_CTRIPH	CMPSS2_CTRIPL	CMPSS1_CTRIPH	CMPSS1_CTRIPL
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 11-33. XBARCLR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R-0/W1S	0h	Reserved
30	RESERVED	R-0/W1S	0h	Reserved
29	RESERVED	R-0/W1S	0h	Reserved
28	RESERVED	R-0/W1S	0h	Reserved
27	RESERVED	R-0/W1S	0h	Reserved
26	RESERVED	R-0/W1S	0h	Reserved
25	RESERVED	R-0/W1S	0h	Reserved
24	RESERVED	R-0/W1S	0h	Reserved
23	CMPSS4_CTRIPOUTH	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
22	CMPSS4_CTRIPOUTL	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
21	CMPSS3_CTRIPOUTH	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
20	CMPSS3_CTRIPOUTL	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
19	CMPSS2_CTRIPOUTH	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn

Table 11-33. XBARCLR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	CMPSS2_CTRIPOUTL	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
17	CMPSS1_CTRIPOUTH	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
16	CMPSS1_CTRIPOUTL	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
15	RESERVED	R-0/W1S	0h	Reserved
14	RESERVED	R-0/W1S	0h	Reserved
13	RESERVED	R-0/W1S	0h	Reserved
12	RESERVED	R-0/W1S	0h	Reserved
11	RESERVED	R-0/W1S	0h	Reserved
10	RESERVED	R-0/W1S	0h	Reserved
9	RESERVED	R-0/W1S	0h	Reserved
8	RESERVED	R-0/W1S	0h	Reserved
7	CMPSS4_CTRIPH	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
6	CMPSS4_CTRIPL	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
5	CMPSS3_CTRIPH	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
4	CMPSS3_CTRIPL	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
3	CMPSS2_CTRIPH	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
2	CMPSS2_CTRIPL	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
1	CMPSS1_CTRIPH	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
0	CMPSS1_CTRIPL	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARFLG1 register. Writing 0 has no effect Reset type: CPU1.SYSRSn

11.3.3.6 XBARCLR2 Register (Offset = Ah) [Reset = 0h]

XBARCLR2 is shown in Figure 11-28 and described in Table 11-34.

Return to the Summary Table.

X-Bar Input Flag Clear Register 2

Figure 11-28. XBARCLR2 Register

31	30	29	28	27	26	25	24
ADCCEVT1	ADCBEVT4	ADCBEVT3	ADCBEVT2	ADCBEVT1	ADCAEVT4	ADCAEVT3	ADCAEVT2
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
ADCAEVT1	EXTSYNCOUT	RESERVED	RESERVED	RESERVED	ECAP3_OUT	ECAP2_OUT	ECAP1_OUT
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
INPUT14	INPUT13	INPUT12	INPUT11	INPUT10	INPUT9	INPUT8	INPUT7
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
ADCSOCB	ADCSOCA	INPUT6	INPUT5	INPUT4	INPUT3	INPUT2	INPUT1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 11-34. XBARCLR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ADCCEVT1	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
30	ADCBEVT4	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
29	ADCBEVT3	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
28	ADCBEVT2	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
27	ADCBEVT1	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
26	ADCAEVT4	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
25	ADCAEVT3	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
24	ADCAEVT2	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn

Table 11-34. XBARCLR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23	ADCAEVT1	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
22	EXTSYNCOUT	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
21	RESERVED	R-0/W1S	0h	Reserved
20	RESERVED	R-0/W1S	0h	Reserved
19	RESERVED	R-0/W1S	0h	Reserved
18	ECAP3_OUT	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
17	ECAP2_OUT	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
16	ECAP1_OUT	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
15	INPUT14	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
14	INPUT13	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
13	INPUT12	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
12	INPUT11	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
11	INPUT10	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
10	INPUT9	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
9	INPUT8	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
8	INPUT7	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn

Table 11-34. XBARCLR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	ADCSOCB	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
6	ADCSOCA	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
5	INPUT6	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
4	INPUT5	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
3	INPUT4	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
2	INPUT3	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
1	INPUT2	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
0	INPUT1	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT2 register. Writing 0 has no effect Reset type: CPU1.SYSRSn

11.3.3.7 XBARCLR3 Register (Offset = Ch) [Reset = 0h]

XBARCLR3 is shown in Figure 11-29 and described in Table 11-35.

Return to the Summary Table.

X-Bar Input Flag Clear Register 3

Figure 11-29. XBARCLR3 Register

31	30	29	28	27	26	25	24
SD1FLT4_DRINT	SD1FLT4_COMPZ	SD1FLT3_DRINT	SD1FLT3_COMPZ	SD1FLT2_DRINT	SD1FLT2_COMPZ	SD1FLT1_DRINT	SD1FLT1_COMPZ
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
RESERVED	SD2FLT4_COMPH	SD2FLT4_COMPL	SD2FLT3_COMPH	SD2FLT3_COMPL	SD2FLT2_COMPH	SD2FLT2_COMPL	SD2FLT1_COMPH
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
SD2FLT1_COMPL	SD1FLT4_COMPH	SD1FLT4_COMPL	SD1FLT3_COMPH	SD1FLT3_COMPL	SD1FLT2_COMPH	SD1FLT2_COMPL	SD1FLT1_COMPH
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
SD1FLT1_COMPL	RESERVED	RESERVED	RESERVED	RESERVED	ADCCEVT4	ADCCEVT3	ADCCEVT2
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 11-35. XBARCLR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SD1FLT4_DRINT	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
30	SD1FLT4_COMPZ	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
29	SD1FLT3_DRINT	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
28	SD1FLT3_COMPZ	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
27	SD1FLT2_DRINT	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
26	SD1FLT2_COMPZ	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
25	SD1FLT1_DRINT	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn

Table 11-35. XBARCLR3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
24	SD1FLT1_COMPZ	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
23	RESERVED	R-0/W1S	0h	Reserved
22	SD2FLT4_COMPH	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
21	SD2FLT4_COMPL	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
20	SD2FLT3_COMPH	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
19	SD2FLT3_COMPL	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
18	SD2FLT2_COMPH	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
17	SD2FLT2_COMPL	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
16	SD2FLT1_COMPH	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
15	SD2FLT1_COMPL	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
14	SD1FLT4_COMPH	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
13	SD1FLT4_COMPL	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
12	SD1FLT3_COMPH	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
11	SD1FLT3_COMPL	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
10	SD1FLT2_COMPH	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn

Table 11-35. XBARCLR3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	SD1FLT2_COMPL	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
8	SD1FLT1_COMPH	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
7	SD1FLT1_COMPL	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
6	RESERVED	R-0/W1S	0h	Reserved
5	RESERVED	R-0/W1S	0h	Reserved
4	RESERVED	R-0/W1S	0h	Reserved
3	RESERVED	R-0/W1S	0h	Reserved
2	ADCCEVT4	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
1	ADCCEVT3	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
0	ADCCEVT2	R-0/W1S	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn

11.3.3.8 XBARCLR4 Register (Offset = Eh) [Reset = 0h]

XBARCLR4 is shown in Figure 11-30 and described in Table 11-36.

Return to the Summary Table.

X-Bar Input Flag Clear Register 4

Figure 11-30. XBARCLR4 Register

31	30	29	28	27	26	25	24
CLAHALT	RESERVED	RESERVED	ERRORSTS_ERROR	RESERVED	RESERVED	RESERVED	RESERVED
R-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
23	22	21	20	19	18	17	16
CLB4_5_1	CLB4_4_1	CLB3_5_1	CLB3_4_1	CLB2_5_1	CLB2_4_1	CLB1_5_1	CLB1_4_1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	MCANA_FEVT2	MCANA_FEVT1	MCANA_FEVT0	RESERVED
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
SD2FLT4_DRINT	SD2FLT4_COMPZ	SD2FLT3_DRINT	SD2FLT3_COMPZ	SD2FLT2_DRINT	SD2FLT2_COMPZ	SD2FLT1_DRINT	SD2FLT1_COMPZ
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 11-36. XBARCLR4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CLAHALT	R	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
30	RESERVED	R-0/W1S	0h	Reserved
29	RESERVED	R-0/W1S	0h	Reserved
28	ERRORSTS_ERROR	R-0/W1S	0h	Writing 1 to this bit clears the ERRORSTS_ERROR bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
27	RESERVED	R-0/W1S	0h	Reserved
26	RESERVED	R-0/W1S	0h	Reserved
25	RESERVED	R-0/W1S	0h	Reserved
24	RESERVED	R-0/W1S	0h	Reserved
23	CLB4_5_1	R	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
22	CLB4_4_1	R	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
21	CLB3_5_1	R	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn

Table 11-36. XBARCLR4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
20	CLB3_4_1	R	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
19	CLB2_5_1	R	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
18	CLB2_4_1	R	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
17	CLB1_5_1	R	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
16	CLB1_4_1	R	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
15	RESERVED	R-0/W1S	0h	Reserved
14	RESERVED	R-0/W1S	0h	Reserved
13	RESERVED	R-0/W1S	0h	Reserved
12	RESERVED	R-0/W1S	0h	Reserved
11	MCANA_FEVT2	R-0/W1S	0h	Writing 1 to this bit clears the MCANA_FEVT2 bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
10	MCANA_FEVT1	R-0/W1S	0h	Writing 1 to this bit clears the MCANA_FEVT1 bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
9	MCANA_FEVT0	R-0/W1S	0h	Writing 1 to this bit clears the MCANA_FEVT0 bit in the XBARFLG4 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
8	RESERVED	R-0/W1S	0h	Reserved
7	SD2FLT4_DRINT	R	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
6	SD2FLT4_COMPZ	R	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
5	SD2FLT3_DRINT	R	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
4	SD2FLT3_COMPZ	R	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
3	SD2FLT2_DRINT	R	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn

Table 11-36. XBARCLR4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	SD2FLT2_COMPZ	R	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
1	SD2FLT1_DRINT	R	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn
0	SD2FLT1_COMPZ	R	0h	Writing 1 to a bit in this register clears the corresponding bit in the XBARILAT3 register. Writing 0 has no effect Reset type: CPU1.SYSRSn

11.3.4 EPWM_XBAR_REGS Registers

Table 11-37 lists the memory-mapped registers for the EPWM_XBAR_REGS registers. All register offset addresses not listed in Table 11-37 should be considered as reserved locations and the register contents should not be modified.

Table 11-37. EPWM_XBAR_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	TRIP4MUX0TO15CFG	ePWM XBAR Mux Configuration for TRIP4	EALLOW	Go
2h	TRIP4MUX16TO31CFG	ePWM XBAR Mux Configuration for TRIP4	EALLOW	Go
4h	TRIP5MUX0TO15CFG	ePWM XBAR Mux Configuration for TRIP5	EALLOW	Go
6h	TRIP5MUX16TO31CFG	ePWM XBAR Mux Configuration for TRIP5	EALLOW	Go
8h	TRIP7MUX0TO15CFG	ePWM XBAR Mux Configuration for TRIP7	EALLOW	Go
Ah	TRIP7MUX16TO31CFG	ePWM XBAR Mux Configuration for TRIP7	EALLOW	Go
Ch	TRIP8MUX0TO15CFG	ePWM XBAR Mux Configuration for TRIP8	EALLOW	Go
Eh	TRIP8MUX16TO31CFG	ePWM XBAR Mux Configuration for TRIP8	EALLOW	Go
10h	TRIP9MUX0TO15CFG	ePWM XBAR Mux Configuration for TRIP9	EALLOW	Go
12h	TRIP9MUX16TO31CFG	ePWM XBAR Mux Configuration for TRIP9	EALLOW	Go
14h	TRIP10MUX0TO15CFG	ePWM XBAR Mux Configuration for TRIP10	EALLOW	Go
16h	TRIP10MUX16TO31CFG	ePWM XBAR Mux Configuration for TRIP10	EALLOW	Go
18h	TRIP11MUX0TO15CFG	ePWM XBAR Mux Configuration for TRIP11	EALLOW	Go
1Ah	TRIP11MUX16TO31CFG	ePWM XBAR Mux Configuration for TRIP11	EALLOW	Go
1Ch	TRIP12MUX0TO15CFG	ePWM XBAR Mux Configuration for TRIP12	EALLOW	Go
1Eh	TRIP12MUX16TO31CFG	ePWM XBAR Mux Configuration for TRIP12	EALLOW	Go
20h	TRIP4MUXENABLE	ePWM XBAR Mux Enable for TRIP4	EALLOW	Go
22h	TRIP5MUXENABLE	ePWM XBAR Mux Enable for TRIP5	EALLOW	Go
24h	TRIP7MUXENABLE	ePWM XBAR Mux Enable for TRIP7	EALLOW	Go
26h	TRIP8MUXENABLE	ePWM XBAR Mux Enable for TRIP8	EALLOW	Go
28h	TRIP9MUXENABLE	ePWM XBAR Mux Enable for TRIP9	EALLOW	Go
2Ah	TRIP10MUXENABLE	ePWM XBAR Mux Enable for TRIP10	EALLOW	Go
2Ch	TRIP11MUXENABLE	ePWM XBAR Mux Enable for TRIP11	EALLOW	Go
2Eh	TRIP12MUXENABLE	ePWM XBAR Mux Enable for TRIP12	EALLOW	Go
38h	TRIPOUTINV	ePWM XBAR Output Inversion Register	EALLOW	Go
3Eh	TRIPLOCK	ePWM XBAR Configuration Lock register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 11-38 shows the codes that are used for access types in this section.

Table 11-38. EPWM_XBAR_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
WSonce	W Sonce	Write Set once
Reset or Default Value		

Table 11-38. EPWM_XBAR_REGS Access Type Codes (continued)

Access Type	Code	Description
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

11.3.4.1 TRIP4MUX0TO15CFG Register (Offset = 0h) [Reset = 0h]

TRIP4MUX0TO15CFG is shown in Figure 11-31 and described in Table 11-39.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP4

Figure 11-31. TRIP4MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-39. TRIP4MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-39. TRIP4MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-39. TRIP4MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.4.2 TRIP4MUX16TO31CFG Register (Offset = 2h) [Reset = 0h]

TRIP4MUX16TO31CFG is shown in Figure 11-32 and described in Table 11-40.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP4

Figure 11-32. TRIP4MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-40. TRIP4MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-40. TRIP4MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-40. TRIP4MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for EPWM-XBAR TRIP4 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.4.3 TRIP5MUX0TO15CFG Register (Offset = 4h) [Reset = 0h]

TRIP5MUX0TO15CFG is shown in Figure 11-33 and described in Table 11-41.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP5

Figure 11-33. TRIP5MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-41. TRIP5MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-41. TRIP5MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-41. TRIP5MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.4.4 TRIP5MUX16TO31CFG Register (Offset = 6h) [Reset = 0h]

TRIP5MUX16TO31CFG is shown in Figure 11-34 and described in Table 11-42.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP5

Figure 11-34. TRIP5MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-42. TRIP5MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-42. TRIP5MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-42. TRIP5MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for EPWM-XBAR TRIP5 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.4.5 TRIP7MUX0TO15CFG Register (Offset = 8h) [Reset = 0h]

TRIP7MUX0TO15CFG is shown in Figure 11-35 and described in Table 11-43.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP7

Figure 11-35. TRIP7MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-43. TRIP7MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-43. TRIP7MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-43. TRIP7MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.4.6 TRIP7MUX16TO31CFG Register (Offset = Ah) [Reset = 0h]

TRIP7MUX16TO31CFG is shown in Figure 11-36 and described in Table 11-44.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP7

Figure 11-36. TRIP7MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-44. TRIP7MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-44. TRIP7MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-44. TRIP7MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for EPWM-XBAR TRIP7 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.4.7 TRIP8MUX0TO15CFG Register (Offset = Ch) [Reset = 0h]

TRIP8MUX0TO15CFG is shown in Figure 11-37 and described in Table 11-45.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP8

Figure 11-37. TRIP8MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-45. TRIP8MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-45. TRIP8MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-45. TRIP8MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.4.8 TRIP8MUX16TO31CFG Register (Offset = Eh) [Reset = 0h]

TRIP8MUX16TO31CFG is shown in Figure 11-38 and described in Table 11-46.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP8

Figure 11-38. TRIP8MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-46. TRIP8MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-46. TRIP8MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-46. TRIP8MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for EPWM-XBAR TRIP8 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.4.9 TRIP9MUX0TO15CFG Register (Offset = 10h) [Reset = 0h]

TRIP9MUX0TO15CFG is shown in Figure 11-39 and described in Table 11-47.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP9

Figure 11-39. TRIP9MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-47. TRIP9MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-47. TRIP9MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-47. TRIP9MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.4.10 TRIP9MUX16TO31CFG Register (Offset = 12h) [Reset = 0h]

TRIP9MUX16TO31CFG is shown in Figure 11-40 and described in Table 11-48.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP9

Figure 11-40. TRIP9MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-48. TRIP9MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-48. TRIP9MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-48. TRIP9MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for EPWM-XBAR TRIP9 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.4.11 TRIP10MUX0TO15CFG Register (Offset = 14h) [Reset = 0h]

TRIP10MUX0TO15CFG is shown in Figure 11-41 and described in Table 11-49.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP10

Figure 11-41. TRIP10MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-49. TRIP10MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-49. TRIP10MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-49. TRIP10MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.4.12 TRIP10MUX16TO31CFG Register (Offset = 16h) [Reset = 0h]

TRIP10MUX16TO31CFG is shown in Figure 11-42 and described in Table 11-50.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP10

Figure 11-42. TRIP10MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-50. TRIP10MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-50. TRIP10MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-50. TRIP10MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for EPWM-XBAR TRIP10 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.4.13 TRIP11MUX0TO15CFG Register (Offset = 18h) [Reset = 0h]

TRIP11MUX0TO15CFG is shown in Figure 11-43 and described in Table 11-51.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP11

Figure 11-43. TRIP11MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-51. TRIP11MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-51. TRIP11MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-51. TRIP11MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.4.14 TRIP11MUX16TO31CFG Register (Offset = 1Ah) [Reset = 0h]

TRIP11MUX16TO31CFG is shown in Figure 11-44 and described in Table 11-52.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP11

Figure 11-44. TRIP11MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-52. TRIP11MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-52. TRIP11MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-52. TRIP11MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for EPWM-XBAR TRIP11 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.4.15 TRIP12MUX0TO15CFG Register (Offset = 1Ch) [Reset = 0h]

TRIP12MUX0TO15CFG is shown in Figure 11-45 and described in Table 11-53.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP12

Figure 11-45. TRIP12MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-53. TRIP12MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-53. TRIP12MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-53. TRIP12MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.4.16 TRIP12MUX16TO31CFG Register (Offset = 1Eh) [Reset = 0h]

TRIP12MUX16TO31CFG is shown in Figure 11-46 and described in Table 11-54.

Return to the Summary Table.

ePWM XBAR Mux Configuration for TRIP12

Figure 11-46. TRIP12MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-54. TRIP12MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-54. TRIP12MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-54. TRIP12MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for EPWM-XBAR TRIP12 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.4.17 TRIP4MUXENABLE Register (Offset = 20h) [Reset = 0h]

TRIP4MUXENABLE is shown in Figure 11-47 and described in Table 11-55.

Return to the Summary Table.

ePWM XBAR Mux Enable for TRIP4

Figure 11-47. TRIP4MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-55. TRIP4MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux31 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux31 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux30 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux30 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux29 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux29 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux28 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux28 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux27 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux27 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-55. TRIP4MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux26 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux26 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux25 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux25 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux24 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux24 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux23 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux23 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux22 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux22 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux21 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux21 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux20 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux20 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux19 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux19 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-55. TRIP4MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux18 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux18 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux17 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux17 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux16 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux16 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux15 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux15 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux14 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux14 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux13 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux13 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux12 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux12 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux11 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux11 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-55. TRIP4MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux10 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux10 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux9 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux9 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux8 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux8 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux7 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux7 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux6 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux6 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux5 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux5 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux4 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux4 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux3 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux3 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-55. TRIP4MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux2 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux2 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux1 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux1 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of Mux0 to drive TRIP4 of EPWM-XBAR 0: Respective output of Mux0 is disabled to drive the TRIP4 of EPWM-XBAR 1: Respective output of Mux0 is enabled to drive the TRIP4 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.4.18 TRIP5MUXENABLE Register (Offset = 22h) [Reset = 0h]

TRIP5MUXENABLE is shown in Figure 11-48 and described in Table 11-56.

Return to the Summary Table.

ePWM XBAR Mux Enable for TRIP5

Figure 11-48. TRIP5MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-56. TRIP5MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux31 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux31 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux30 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux30 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux29 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux29 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux28 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux28 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux27 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux27 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-56. TRIP5MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux26 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux26 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux25 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux25 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux24 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux24 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux23 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux23 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux22 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux22 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux21 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux21 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux20 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux20 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux19 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux19 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-56. TRIP5MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux18 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux18 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux17 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux17 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux16 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux16 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux15 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux15 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux14 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux14 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux13 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux13 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux12 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux12 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux11 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux11 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-56. TRIP5MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux10 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux10 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux9 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux9 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux8 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux8 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux7 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux7 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux6 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux6 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux5 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux5 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux4 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux4 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux3 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux3 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-56. TRIP5MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux2 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux2 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux1 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux1 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive TRIP5 of EPWM-XBAR 0: Respective output of Mux0 is disabled to drive the TRIP5 of EPWM-XBAR 1: Respective output of Mux0 is enabled to drive the TRIP5 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.4.19 TRIP7MUXENABLE Register (Offset = 24h) [Reset = 0h]

TRIP7MUXENABLE is shown in Figure 11-49 and described in Table 11-57.

Return to the Summary Table.

ePWM XBAR Mux Enable for TRIP7

Figure 11-49. TRIP7MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-57. TRIP7MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux31 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux31 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux30 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux30 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux29 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux29 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux28 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux28 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux27 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux27 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-57. TRIP7MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux26 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux26 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux25 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux25 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux24 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux24 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux23 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux23 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux22 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux22 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux21 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux21 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux20 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux20 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux19 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux19 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-57. TRIP7MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux18 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux18 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux17 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux17 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux16 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux16 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux15 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux15 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux14 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux14 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux13 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux13 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux12 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux12 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux11 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux11 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-57. TRIP7MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux10 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux10 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux9 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux9 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux8 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux8 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux7 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux7 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux6 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux6 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux5 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux5 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux4 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux4 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux3 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux3 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-57. TRIP7MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux2 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux2 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux1 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux1 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive TRIP7 of EPWM-XBAR 0: Respective output of Mux0 is disabled to drive the TRIP7 of EPWM-XBAR 1: Respective output of Mux0 is enabled to drive the TRIP7 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.4.20 TRIP8MUXENABLE Register (Offset = 26h) [Reset = 0h]

TRIP8MUXENABLE is shown in Figure 11-50 and described in Table 11-58.

Return to the Summary Table.

ePWM XBAR Mux Enable for TRIP8

Figure 11-50. TRIP8MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-58. TRIP8MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux31 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux31 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux30 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux30 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux29 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux29 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux28 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux28 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux27 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux27 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-58. TRIP8MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux26 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux26 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux25 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux25 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux24 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux24 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux23 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux23 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux22 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux22 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux21 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux21 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux20 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux20 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux19 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux19 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-58. TRIP8MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux18 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux18 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux17 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux17 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux16 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux16 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux15 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux15 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux14 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux14 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux13 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux13 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux12 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux12 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux11 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux11 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-58. TRIP8MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux10 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux10 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux9 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux9 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux8 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux8 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux7 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux7 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux6 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux6 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux5 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux5 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux4 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux4 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux3 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux3 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-58. TRIP8MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux2 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux2 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux1 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux1 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive TRIP8 of EPWM-XBAR 0: Respective output of Mux0 is disabled to drive the TRIP8 of EPWM-XBAR 1: Respective output of Mux0 is enabled to drive the TRIP8 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.4.21 TRIP9MUXENABLE Register (Offset = 28h) [Reset = 0h]

TRIP9MUXENABLE is shown in Figure 11-51 and described in Table 11-59.

Return to the Summary Table.

ePWM XBAR Mux Enable for TRIP9

Figure 11-51. TRIP9MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-59. TRIP9MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux31 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux31 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux30 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux30 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux29 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux29 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux28 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux28 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux27 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux27 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-59. TRIP9MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux26 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux26 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux25 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux25 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux24 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux24 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux23 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux23 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux22 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux22 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux21 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux21 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux20 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux20 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux19 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux19 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-59. TRIP9MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux18 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux18 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux17 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux17 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux16 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux16 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux15 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux15 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux14 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux14 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux13 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux13 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux12 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux12 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux11 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux11 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-59. TRIP9MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux10 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux10 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux9 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux9 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux8 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux8 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux7 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux7 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux6 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux6 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux5 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux5 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux4 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux4 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux3 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux3 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-59. TRIP9MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux2 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux2 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux1 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux1 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive TRIP9 of EPWM-XBAR 0: Respective output of Mux0 is disabled to drive the TRIP9 of EPWM-XBAR 1: Respective output of Mux0 is enabled to drive the TRIP9 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.4.22 TRIP10MUXENABLE Register (Offset = 2Ah) [Reset = 0h]

TRIP10MUXENABLE is shown in Figure 11-52 and described in Table 11-60.

Return to the Summary Table.

ePWM XBAR Mux Enable for TRIP10

Figure 11-52. TRIP10MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-60. TRIP10MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux31 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux31 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux30 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux30 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux29 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux29 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux28 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux28 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux27 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux27 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-60. TRIP10MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux26 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux26 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux25 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux25 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux24 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux24 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux23 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux23 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux22 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux22 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux21 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux21 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux20 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux20 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux19 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux19 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-60. TRIP10MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux18 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux18 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux17 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux17 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux16 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux16 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux15 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux15 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux14 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux14 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux13 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux13 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux12 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux12 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux11 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux11 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-60. TRIP10MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux10 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux10 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux9 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux9 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux8 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux8 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux7 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux7 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux6 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux6 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux5 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux5 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux4 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux4 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux3 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux3 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-60. TRIP10MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux2 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux2 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux1 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux1 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive TRIP10 of EPWM-XBAR 0: Respective output of Mux0 is disabled to drive the TRIP10 of EPWM-XBAR 1: Respective output of Mux0 is enabled to drive the TRIP10 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.4.23 TRIP11MUXENABLE Register (Offset = 2Ch) [Reset = 0h]

TRIP11MUXENABLE is shown in Figure 11-53 and described in Table 11-61.

Return to the Summary Table.

ePWM XBAR Mux Enable for TRIP11

Figure 11-53. TRIP11MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-61. TRIP11MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux31 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux31 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux30 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux30 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux29 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux29 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux28 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux28 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux27 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux27 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-61. TRIP11MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux26 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux26 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux25 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux25 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux24 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux24 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux23 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux23 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux22 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux22 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux21 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux21 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux20 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux20 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux19 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux19 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-61. TRIP11MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux18 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux18 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux17 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux17 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux16 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux16 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux15 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux15 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux14 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux14 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux13 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux13 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux12 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux12 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux11 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux11 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-61. TRIP11MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux10 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux10 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux9 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux9 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux8 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux8 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux7 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux7 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux6 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux6 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux5 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux5 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux4 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux4 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux3 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux3 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-61. TRIP11MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux2 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux2 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux1 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux1 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive TRIP11 of EPWM-XBAR 0: Respective output of Mux0 is disabled to drive the TRIP11 of EPWM-XBAR 1: Respective output of Mux0 is enabled to drive the TRIP11 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.4.24 TRIP12MUXENABLE Register (Offset = 2Eh) [Reset = 0h]

TRIP12MUXENABLE is shown in Figure 11-54 and described in Table 11-62.

Return to the Summary Table.

ePWM XBAR Mux Enable for TRIP12

Figure 11-54. TRIP12MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-62. TRIP12MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux31 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux31 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux30 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux30 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux29 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux29 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux28 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux28 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux27 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux27 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-62. TRIP12MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux26 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux26 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux25 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux25 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux24 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux24 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux23 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux23 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux22 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux22 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux21 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux21 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux20 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux20 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux19 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux19 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-62. TRIP12MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux18 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux18 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux17 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux17 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux16 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux16 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux15 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux15 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux14 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux14 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux13 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux13 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux12 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux12 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux11 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux11 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-62. TRIP12MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux10 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux10 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux9 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux9 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux8 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux8 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux7 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux7 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux6 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux6 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux5 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux5 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux4 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux4 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux3 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux3 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-62. TRIP12MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux2 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux2 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux1 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux1 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive TRIP12 of EPWM-XBAR 0: Respective output of Mux0 is disabled to drive the TRIP12 of EPWM-XBAR 1: Respective output of Mux0 is enabled to drive the TRIP12 of EPWM-XBAR Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.4.25 TRIPOUTINV Register (Offset = 38h) [Reset = 0h]

TRIPOUTINV is shown in Figure 11-55 and described in Table 11-63.

Return to the Summary Table.

ePWM XBAR Output Inversion Register

Figure 11-55. TRIPOUTINV Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
TRIP12	TRIP11	TRIP10	TRIP9	TRIP8	TRIP7	TRIP5	TRIP4
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-63. TRIPOUTINV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7	TRIP12	R/W	0h	Selects polarity for TRIP12 of EPWM-XBAR 0: drives active high output 1: drives active-low output Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	TRIP11	R/W	0h	Selects polarity for TRIP11 of EPWM-XBAR 0: drives active high output 1: drives active-low output Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	TRIP10	R/W	0h	Selects polarity for TRIP10 of EPWM-XBAR 0: drives active high output 1: drives active-low output Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	TRIP9	R/W	0h	Selects polarity for TRIP9 of EPWM-XBAR 0: drives active high output 1: drives active-low output Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	TRIP8	R/W	0h	Selects polarity for TRIP8 of EPWM-XBAR 0: drives active high output 1: drives active-low output Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
2	TRIP7	R/W	0h	Selects polarity for TRIP7 of EPWM-XBAR 0: drives active high output 1: drives active-low output Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-63. TRIPOUTINV Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	TRIP5	R/W	0h	Selects polarity for TRIP5 of EPWM-XBAR 0: drives active high output 1: drives active-low output Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	TRIP4	R/W	0h	Selects polarity for TRIP4 of EPWM-XBAR 0: drives active high output 1: drives active-low output Refer to the EPWM X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.4.26 TRIPLOCK Register (Offset = 3Eh) [Reset = 0h]

TRIPLOCK is shown in Figure 11-56 and described in Table 11-64.

Return to the Summary Table.

ePWM XBAR Configuration Lock register

Figure 11-56. TRIPLOCK Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							LOCK
R-0h							R/WSonce-0h

Table 11-64. TRIPLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Bit-0 of this register can be set only if KEY= 0x5a5a Reset type: CPU1.SYSRSn
15-1	RESERVED	R	0h	Reserved
0	LOCK	R/WSonce	0h	Locks the configuration for EPWM-XBAR. Once the configuration is locked, writes to the below registers for EPWM-XBAR is blocked. Registers Affected by the LOCK mechanism: EPWM-XBAROUTyMUX0TO15CFG EPWM-XBAROUTyMUX16TO31CFG EPWM-XBAROUTyMUXENABLE EPWM-XBAROUTLATEN EPWM-XBAROUTINV 0: Writes to the above registers are allowed 1: Writes to the above registers are blocked Note: [1] LOCK mechanism only apples to writes. Reads are never blocked. Reset type: CPU1.SYSRSn

11.3.5 CLB_XBAR_REGS Registers

Table 11-65 lists the memory-mapped registers for the CLB_XBAR_REGS registers. All register offset addresses not listed in Table 11-65 should be considered as reserved locations and the register contents should not be modified.

Table 11-65. CLB_XBAR_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	AUXSIG0MUX0TO15CFG	CLB XBAR Mux Configuration for Output-0	EALLOW	Go
2h	AUXSIG0MUX16TO31CFG	CLB XBAR Mux Configuration for Output-0	EALLOW	Go
4h	AUXSIG1MUX0TO15CFG	CLB XBAR Mux Configuration for Output-1	EALLOW	Go
6h	AUXSIG1MUX16TO31CFG	CLB XBAR Mux Configuration for Output-1	EALLOW	Go
8h	AUXSIG2MUX0TO15CFG	CLB XBAR Mux Configuration for Output-2	EALLOW	Go
Ah	AUXSIG2MUX16TO31CFG	CLB XBAR Mux Configuration for Output-2	EALLOW	Go
Ch	AUXSIG3MUX0TO15CFG	CLB XBAR Mux Configuration for Output-3	EALLOW	Go
Eh	AUXSIG3MUX16TO31CFG	CLB XBAR Mux Configuration for Output-3	EALLOW	Go
10h	AUXSIG4MUX0TO15CFG	CLB XBAR Mux Configuration for Output-4	EALLOW	Go
12h	AUXSIG4MUX16TO31CFG	CLB XBAR Mux Configuration for Output-4	EALLOW	Go
14h	AUXSIG5MUX0TO15CFG	CLB XBAR Mux Configuration for Output-5	EALLOW	Go
16h	AUXSIG5MUX16TO31CFG	CLB XBAR Mux Configuration for Output-5	EALLOW	Go
18h	AUXSIG6MUX0TO15CFG	CLB XBAR Mux Configuration for Output-6	EALLOW	Go
1Ah	AUXSIG6MUX16TO31CFG	CLB XBAR Mux Configuration for Output-6	EALLOW	Go
1Ch	AUXSIG7MUX0TO15CFG	CLB XBAR Mux Configuration for Output-7	EALLOW	Go
1Eh	AUXSIG7MUX16TO31CFG	CLB XBAR Mux Configuration for Output-7	EALLOW	Go
20h	AUXSIG0MUXENABLE	CLB XBAR Mux Enable Register for Output-0	EALLOW	Go
22h	AUXSIG1MUXENABLE	CLB XBAR Mux Enable Register for Output-1	EALLOW	Go
24h	AUXSIG2MUXENABLE	CLB XBAR Mux Enable Register for Output-2	EALLOW	Go
26h	AUXSIG3MUXENABLE	CLB XBAR Mux Enable Register for Output-3	EALLOW	Go
28h	AUXSIG4MUXENABLE	CLB XBAR Mux Enable Register for Output-4	EALLOW	Go
2Ah	AUXSIG5MUXENABLE	CLB XBAR Mux Enable Register for Output-5	EALLOW	Go
2Ch	AUXSIG6MUXENABLE	CLB XBAR Mux Enable Register for Output-6	EALLOW	Go
2Eh	AUXSIG7MUXENABLE	CLB XBAR Mux Enable Register for Output-7	EALLOW	Go
38h	AUXSIGOUTINV	CLB XBAR Output Inversion Register	EALLOW	Go
3Eh	AUXSIGLOCK	ClbXbar Configuration Lock register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 11-66 shows the codes that are used for access types in this section.

Table 11-66. CLB_XBAR_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
WSonce	W Sonce	Write Set once
Reset or Default Value		

Table 11-66. CLB_XBAR_REGS Access Type Codes (continued)

Access Type	Code	Description
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

11.3.5.1 AUXSIG0MUX0TO15CFG Register (Offset = 0h) [Reset = 0h]

AUXSIG0MUX0TO15CFG is shown in Figure 11-57 and described in Table 11-67.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-0

Figure 11-57. AUXSIG0MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-67. AUXSIG0MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Group Select Bits for MUX15: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX15 01 : Select .1 input for MUX15 10 : Select .2 input for MUX15 11 : Select .3 input for MUX15 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Group Select Bits for MUX14: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX14 01 : Select .1 input for MUX14 10 : Select .2 input for MUX14 11 : Select .3 input for MUX14 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Group Select Bits for MUX13: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX13 01 : Select .1 input for MUX13 10 : Select .2 input for MUX13 11 : Select .3 input for MUX13 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Group Select Bits for MUX12: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX12 01 : Select .1 input for MUX12 10 : Select .2 input for MUX12 11 : Select .3 input for MUX12 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Group Select Bits for MUX11: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX11 01 : Select .1 input for MUX11 10 : Select .2 input for MUX11 11 : Select .3 input for MUX11 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-67. AUXSIG0MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX10	R/W	0h	Group Select Bits for MUX10: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX10 01 : Select .1 input for MUX10 10 : Select .2 input for MUX10 11 : Select .3 input for MUX10 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX9	R/W	0h	Group Select Bits for MUX9: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX9 01 : Select .1 input for MUX9 10 : Select .2 input for MUX9 11 : Select .3 input for MUX9 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Group Select Bits for MUX8: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX8 01 : Select .1 input for MUX8 10 : Select .2 input for MUX8 11 : Select .3 input for MUX8 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Group Select Bits for MUX7: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX7 01 : Select .1 input for MUX7 10 : Select .2 input for MUX7 11 : Select .3 input for MUX7 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Group Select Bits for MUX6: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX6 01 : Select .1 input for MUX6 10 : Select .2 input for MUX6 11 : Select .3 input for MUX6 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Group Select Bits for MUX5: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX5 01 : Select .1 input for MUX5 10 : Select .2 input for MUX5 11 : Select .3 input for MUX5 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Group Select Bits for MUX4: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX4 01 : Select .1 input for MUX4 10 : Select .2 input for MUX4 11 : Select .3 input for MUX4 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-67. AUXSIG0MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX3	R/W	0h	Group Select Bits for MUX3: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX3 01 : Select .1 input for MUX3 10 : Select .2 input for MUX3 11 : Select .3 input for MUX3 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Group Select Bits for MUX2: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX2 01 : Select .1 input for MUX2 10 : Select .2 input for MUX2 11 : Select .3 input for MUX2 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX1	R/W	0h	Group Select Bits for MUX1: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX1 01 : Select .1 input for MUX1 10 : Select .2 input for MUX1 11 : Select .3 input for MUX1 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Group Select Bits for MUX0: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX0 01 : Select .1 input for MUX0 10 : Select .2 input for MUX0 11 : Select .3 input for MUX0 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.5.2 AUXSIG0MUX16TO31CFG Register (Offset = 2h) [Reset = 0h]

AUXSIG0MUX16TO31CFG is shown in Figure 11-58 and described in Table 11-68.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-0

Figure 11-58. AUXSIG0MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-68. AUXSIG0MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Group Select Bits for MUX31: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX31 01 : Select .1 input for MUX31 10 : Select .2 input for MUX31 11 : Select .3 input for MUX31 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Group Select Bits for MUX30: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX30 01 : Select .1 input for MUX30 10 : Select .2 input for MUX30 11 : Select .3 input for MUX30 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Group Select Bits for MUX29: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX29 01 : Select .1 input for MUX29 10 : Select .2 input for MUX29 11 : Select .3 input for MUX29 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Group Select Bits for MUX28: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX28 01 : Select .1 input for MUX28 10 : Select .2 input for MUX28 11 : Select .3 input for MUX28 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Group Select Bits for MUX27: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX27 01 : Select .1 input for MUX27 10 : Select .2 input for MUX27 11 : Select .3 input for MUX27 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-68. AUXSIG0MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX26	R/W	0h	Group Select Bits for MUX26: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX26 01 : Select .1 input for MUX26 10 : Select .2 input for MUX26 11 : Select .3 input for MUX26 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX25	R/W	0h	Group Select Bits for MUX25: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX25 01 : Select .1 input for MUX25 10 : Select .2 input for MUX25 11 : Select .3 input for MUX25 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Group Select Bits for MUX24: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX24 01 : Select .1 input for MUX24 10 : Select .2 input for MUX24 11 : Select .3 input for MUX24 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Group Select Bits for MUX23: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX23 01 : Select .1 input for MUX23 10 : Select .2 input for MUX23 11 : Select .3 input for MUX23 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Group Select Bits for MUX22: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX22 01 : Select .1 input for MUX22 10 : Select .2 input for MUX22 11 : Select .3 input for MUX22 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Group Select Bits for MUX21: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX21 01 : Select .1 input for MUX21 10 : Select .2 input for MUX21 11 : Select .3 input for MUX21 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Group Select Bits for MUX20: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX20 01 : Select .1 input for MUX20 10 : Select .2 input for MUX20 11 : Select .3 input for MUX20 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-68. AUXSIG0MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX19	R/W	0h	Group Select Bits for MUX19: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX19 01 : Select .1 input for MUX19 10 : Select .2 input for MUX19 11 : Select .3 input for MUX19 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Group Select Bits for MUX18: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX18 01 : Select .1 input for MUX18 10 : Select .2 input for MUX18 11 : Select .3 input for MUX18 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX17	R/W	0h	Group Select Bits for MUX17: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX17 01 : Select .1 input for MUX17 10 : Select .2 input for MUX17 11 : Select .3 input for MUX17 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Group Select Bits for MUX16: Selects 4X1 group output for AUXSIG0 of CLB-XBAR 00 : Select .0 input for MUX16 01 : Select .1 input for MUX16 10 : Select .2 input for MUX16 11 : Select .3 input for MUX16 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.5.3 AUXSIG1MUX0TO15CFG Register (Offset = 4h) [Reset = 0h]

AUXSIG1MUX0TO15CFG is shown in Figure 11-59 and described in Table 11-69.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-1

Figure 11-59. AUXSIG1MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-69. AUXSIG1MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Group Select Bits for MUX15: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX15 01 : Select .1 input for MUX15 10 : Select .2 input for MUX15 11 : Select .3 input for MUX15 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Group Select Bits for MUX14: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX14 01 : Select .1 input for MUX14 10 : Select .2 input for MUX14 11 : Select .3 input for MUX14 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Group Select Bits for MUX13: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX13 01 : Select .1 input for MUX13 10 : Select .2 input for MUX13 11 : Select .3 input for MUX13 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Group Select Bits for MUX12: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX12 01 : Select .1 input for MUX12 10 : Select .2 input for MUX12 11 : Select .3 input for MUX12 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Group Select Bits for MUX11: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX11 01 : Select .1 input for MUX11 10 : Select .2 input for MUX11 11 : Select .3 input for MUX11 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-69. AUXSIG1MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX10	R/W	0h	Group Select Bits for MUX10: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX10 01 : Select .1 input for MUX10 10 : Select .2 input for MUX10 11 : Select .3 input for MUX10 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX9	R/W	0h	Group Select Bits for MUX9: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX9 01 : Select .1 input for MUX9 10 : Select .2 input for MUX9 11 : Select .3 input for MUX9 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Group Select Bits for MUX8: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX8 01 : Select .1 input for MUX8 10 : Select .2 input for MUX8 11 : Select .3 input for MUX8 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Group Select Bits for MUX7: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX7 01 : Select .1 input for MUX7 10 : Select .2 input for MUX7 11 : Select .3 input for MUX7 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Group Select Bits for MUX6: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX6 01 : Select .1 input for MUX6 10 : Select .2 input for MUX6 11 : Select .3 input for MUX6 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Group Select Bits for MUX5: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX5 01 : Select .1 input for MUX5 10 : Select .2 input for MUX5 11 : Select .3 input for MUX5 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Group Select Bits for MUX4: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX4 01 : Select .1 input for MUX4 10 : Select .2 input for MUX4 11 : Select .3 input for MUX4 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-69. AUXSIG1MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX3	R/W	0h	Group Select Bits for MUX3: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX3 01 : Select .1 input for MUX3 10 : Select .2 input for MUX3 11 : Select .3 input for MUX3 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Group Select Bits for MUX2: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX2 01 : Select .1 input for MUX2 10 : Select .2 input for MUX2 11 : Select .3 input for MUX2 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX1	R/W	0h	Group Select Bits for MUX1: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX1 01 : Select .1 input for MUX1 10 : Select .2 input for MUX1 11 : Select .3 input for MUX1 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Group Select Bits for MUX0: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX0 01 : Select .1 input for MUX0 10 : Select .2 input for MUX0 11 : Select .3 input for MUX0 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.5.4 AUXSIG1MUX16TO31CFG Register (Offset = 6h) [Reset = 0h]

AUXSIG1MUX16TO31CFG is shown in Figure 11-60 and described in Table 11-70.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-1

Figure 11-60. AUXSIG1MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-70. AUXSIG1MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Group Select Bits for MUX31: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX31 01 : Select .1 input for MUX31 10 : Select .2 input for MUX31 11 : Select .3 input for MUX31 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Group Select Bits for MUX30: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX30 01 : Select .1 input for MUX30 10 : Select .2 input for MUX30 11 : Select .3 input for MUX30 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Group Select Bits for MUX29: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX29 01 : Select .1 input for MUX29 10 : Select .2 input for MUX29 11 : Select .3 input for MUX29 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Group Select Bits for MUX28: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX28 01 : Select .1 input for MUX28 10 : Select .2 input for MUX28 11 : Select .3 input for MUX28 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Group Select Bits for MUX27: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX27 01 : Select .1 input for MUX27 10 : Select .2 input for MUX27 11 : Select .3 input for MUX27 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-70. AUXSIG1MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX26	R/W	0h	Group Select Bits for MUX26: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX26 01 : Select .1 input for MUX26 10 : Select .2 input for MUX26 11 : Select .3 input for MUX26 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX25	R/W	0h	Group Select Bits for MUX25: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX25 01 : Select .1 input for MUX25 10 : Select .2 input for MUX25 11 : Select .3 input for MUX25 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Group Select Bits for MUX24: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX24 01 : Select .1 input for MUX24 10 : Select .2 input for MUX24 11 : Select .3 input for MUX24 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Group Select Bits for MUX23: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX23 01 : Select .1 input for MUX23 10 : Select .2 input for MUX23 11 : Select .3 input for MUX23 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Group Select Bits for MUX22: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX22 01 : Select .1 input for MUX22 10 : Select .2 input for MUX22 11 : Select .3 input for MUX22 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Group Select Bits for MUX21: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX21 01 : Select .1 input for MUX21 10 : Select .2 input for MUX21 11 : Select .3 input for MUX21 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Group Select Bits for MUX20: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX20 01 : Select .1 input for MUX20 10 : Select .2 input for MUX20 11 : Select .3 input for MUX20 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-70. AUXSIG1MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX19	R/W	0h	Group Select Bits for MUX19: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX19 01 : Select .1 input for MUX19 10 : Select .2 input for MUX19 11 : Select .3 input for MUX19 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Group Select Bits for MUX18: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX18 01 : Select .1 input for MUX18 10 : Select .2 input for MUX18 11 : Select .3 input for MUX18 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX17	R/W	0h	Group Select Bits for MUX17: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX17 01 : Select .1 input for MUX17 10 : Select .2 input for MUX17 11 : Select .3 input for MUX17 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Group Select Bits for MUX16: Selects 4X1 group output for AUXSIG1 of CLB-XBAR 00 : Select .0 input for MUX16 01 : Select .1 input for MUX16 10 : Select .2 input for MUX16 11 : Select .3 input for MUX16 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.5.5 AUXSIG2MUX0TO15CFG Register (Offset = 8h) [Reset = 0h]

AUXSIG2MUX0TO15CFG is shown in Figure 11-61 and described in Table 11-71.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-2

Figure 11-61. AUXSIG2MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-71. AUXSIG2MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Group Select Bits for MUX15: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX15 01 : Select .1 input for MUX15 10 : Select .2 input for MUX15 11 : Select .3 input for MUX15 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Group Select Bits for MUX14: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX14 01 : Select .1 input for MUX14 10 : Select .2 input for MUX14 11 : Select .3 input for MUX14 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Group Select Bits for MUX13: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX13 01 : Select .1 input for MUX13 10 : Select .2 input for MUX13 11 : Select .3 input for MUX13 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Group Select Bits for MUX12: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX12 01 : Select .1 input for MUX12 10 : Select .2 input for MUX12 11 : Select .3 input for MUX12 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Group Select Bits for MUX11: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX11 01 : Select .1 input for MUX11 10 : Select .2 input for MUX11 11 : Select .3 input for MUX11 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-71. AUXSIG2MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX10	R/W	0h	Group Select Bits for MUX10: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX10 01 : Select .1 input for MUX10 10 : Select .2 input for MUX10 11 : Select .3 input for MUX10 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX9	R/W	0h	Group Select Bits for MUX9: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX9 01 : Select .1 input for MUX9 10 : Select .2 input for MUX9 11 : Select .3 input for MUX9 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Group Select Bits for MUX8: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX8 01 : Select .1 input for MUX8 10 : Select .2 input for MUX8 11 : Select .3 input for MUX8 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Group Select Bits for MUX7: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX7 01 : Select .1 input for MUX7 10 : Select .2 input for MUX7 11 : Select .3 input for MUX7 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Group Select Bits for MUX6: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX6 01 : Select .1 input for MUX6 10 : Select .2 input for MUX6 11 : Select .3 input for MUX6 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Group Select Bits for MUX5: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX5 01 : Select .1 input for MUX5 10 : Select .2 input for MUX5 11 : Select .3 input for MUX5 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Group Select Bits for MUX4: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX4 01 : Select .1 input for MUX4 10 : Select .2 input for MUX4 11 : Select .3 input for MUX4 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-71. AUXSIG2MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX3	R/W	0h	Group Select Bits for MUX3: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX3 01 : Select .1 input for MUX3 10 : Select .2 input for MUX3 11 : Select .3 input for MUX3 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Group Select Bits for MUX2: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX2 01 : Select .1 input for MUX2 10 : Select .2 input for MUX2 11 : Select .3 input for MUX2 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX1	R/W	0h	Group Select Bits for MUX1: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX1 01 : Select .1 input for MUX1 10 : Select .2 input for MUX1 11 : Select .3 input for MUX1 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Group Select Bits for MUX0: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX0 01 : Select .1 input for MUX0 10 : Select .2 input for MUX0 11 : Select .3 input for MUX0 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.5.6 AUXSIG2MUX16TO31CFG Register (Offset = Ah) [Reset = 0h]

AUXSIG2MUX16TO31CFG is shown in Figure 11-62 and described in Table 11-72.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-2

Figure 11-62. AUXSIG2MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-72. AUXSIG2MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Group Select Bits for MUX31: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX31 01 : Select .1 input for MUX31 10 : Select .2 input for MUX31 11 : Select .3 input for MUX31 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Group Select Bits for MUX30: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX30 01 : Select .1 input for MUX30 10 : Select .2 input for MUX30 11 : Select .3 input for MUX30 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Group Select Bits for MUX29: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX29 01 : Select .1 input for MUX29 10 : Select .2 input for MUX29 11 : Select .3 input for MUX29 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Group Select Bits for MUX28: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX28 01 : Select .1 input for MUX28 10 : Select .2 input for MUX28 11 : Select .3 input for MUX28 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Group Select Bits for MUX27: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX27 01 : Select .1 input for MUX27 10 : Select .2 input for MUX27 11 : Select .3 input for MUX27 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-72. AUXSIG2MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX26	R/W	0h	Group Select Bits for MUX26: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX26 01 : Select .1 input for MUX26 10 : Select .2 input for MUX26 11 : Select .3 input for MUX26 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX25	R/W	0h	Group Select Bits for MUX25: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX25 01 : Select .1 input for MUX25 10 : Select .2 input for MUX25 11 : Select .3 input for MUX25 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Group Select Bits for MUX24: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX24 01 : Select .1 input for MUX24 10 : Select .2 input for MUX24 11 : Select .3 input for MUX24 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Group Select Bits for MUX23: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX23 01 : Select .1 input for MUX23 10 : Select .2 input for MUX23 11 : Select .3 input for MUX23 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Group Select Bits for MUX22: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX22 01 : Select .1 input for MUX22 10 : Select .2 input for MUX22 11 : Select .3 input for MUX22 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Group Select Bits for MUX21: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX21 01 : Select .1 input for MUX21 10 : Select .2 input for MUX21 11 : Select .3 input for MUX21 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Group Select Bits for MUX20: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX20 01 : Select .1 input for MUX20 10 : Select .2 input for MUX20 11 : Select .3 input for MUX20 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-72. AUXSIG2MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX19	R/W	0h	Group Select Bits for MUX19: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX19 01 : Select .1 input for MUX19 10 : Select .2 input for MUX19 11 : Select .3 input for MUX19 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Group Select Bits for MUX18: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX18 01 : Select .1 input for MUX18 10 : Select .2 input for MUX18 11 : Select .3 input for MUX18 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX17	R/W	0h	Group Select Bits for MUX17: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX17 01 : Select .1 input for MUX17 10 : Select .2 input for MUX17 11 : Select .3 input for MUX17 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Group Select Bits for MUX16: Selects 4X1 group output for AUXSIG2 of CLB-XBAR 00 : Select .0 input for MUX16 01 : Select .1 input for MUX16 10 : Select .2 input for MUX16 11 : Select .3 input for MUX16 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.5.7 AUXSIG3MUX0TO15CFG Register (Offset = Ch) [Reset = 0h]

AUXSIG3MUX0TO15CFG is shown in Figure 11-63 and described in Table 11-73.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-3

Figure 11-63. AUXSIG3MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-73. AUXSIG3MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Group Select Bits for MUX15: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX15 01 : Select .1 input for MUX15 10 : Select .2 input for MUX15 11 : Select .3 input for MUX15 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Group Select Bits for MUX14: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX14 01 : Select .1 input for MUX14 10 : Select .2 input for MUX14 11 : Select .3 input for MUX14 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Group Select Bits for MUX13: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX13 01 : Select .1 input for MUX13 10 : Select .2 input for MUX13 11 : Select .3 input for MUX13 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Group Select Bits for MUX12: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX12 01 : Select .1 input for MUX12 10 : Select .2 input for MUX12 11 : Select .3 input for MUX12 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Group Select Bits for MUX11: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX11 01 : Select .1 input for MUX11 10 : Select .2 input for MUX11 11 : Select .3 input for MUX11 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-73. AUXSIG3MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX10	R/W	0h	Group Select Bits for MUX10: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX10 01 : Select .1 input for MUX10 10 : Select .2 input for MUX10 11 : Select .3 input for MUX10 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX9	R/W	0h	Group Select Bits for MUX9: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX9 01 : Select .1 input for MUX9 10 : Select .2 input for MUX9 11 : Select .3 input for MUX9 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Group Select Bits for MUX8: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX8 01 : Select .1 input for MUX8 10 : Select .2 input for MUX8 11 : Select .3 input for MUX8 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Group Select Bits for MUX7: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX7 01 : Select .1 input for MUX7 10 : Select .2 input for MUX7 11 : Select .3 input for MUX7 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Group Select Bits for MUX6: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX6 01 : Select .1 input for MUX6 10 : Select .2 input for MUX6 11 : Select .3 input for MUX6 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Group Select Bits for MUX5: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX5 01 : Select .1 input for MUX5 10 : Select .2 input for MUX5 11 : Select .3 input for MUX5 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Group Select Bits for MUX4: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX4 01 : Select .1 input for MUX4 10 : Select .2 input for MUX4 11 : Select .3 input for MUX4 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-73. AUXSIG3MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX3	R/W	0h	Group Select Bits for MUX3: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX3 01 : Select .1 input for MUX3 10 : Select .2 input for MUX3 11 : Select .3 input for MUX3 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Group Select Bits for MUX2: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX2 01 : Select .1 input for MUX2 10 : Select .2 input for MUX2 11 : Select .3 input for MUX2 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX1	R/W	0h	Group Select Bits for MUX1: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX1 01 : Select .1 input for MUX1 10 : Select .2 input for MUX1 11 : Select .3 input for MUX1 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Group Select Bits for MUX0: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX0 01 : Select .1 input for MUX0 10 : Select .2 input for MUX0 11 : Select .3 input for MUX0 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.5.8 AUXSIG3MUX16TO31CFG Register (Offset = Eh) [Reset = 0h]

AUXSIG3MUX16TO31CFG is shown in Figure 11-64 and described in Table 11-74.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-3

Figure 11-64. AUXSIG3MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-74. AUXSIG3MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Group Select Bits for MUX31: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX31 01 : Select .1 input for MUX31 10 : Select .2 input for MUX31 11 : Select .3 input for MUX31 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Group Select Bits for MUX30: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX30 01 : Select .1 input for MUX30 10 : Select .2 input for MUX30 11 : Select .3 input for MUX30 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Group Select Bits for MUX29: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX29 01 : Select .1 input for MUX29 10 : Select .2 input for MUX29 11 : Select .3 input for MUX29 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Group Select Bits for MUX28: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX28 01 : Select .1 input for MUX28 10 : Select .2 input for MUX28 11 : Select .3 input for MUX28 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Group Select Bits for MUX27: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX27 01 : Select .1 input for MUX27 10 : Select .2 input for MUX27 11 : Select .3 input for MUX27 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-74. AUXSIG3MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX26	R/W	0h	Group Select Bits for MUX26: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX26 01 : Select .1 input for MUX26 10 : Select .2 input for MUX26 11 : Select .3 input for MUX26 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX25	R/W	0h	Group Select Bits for MUX25: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX25 01 : Select .1 input for MUX25 10 : Select .2 input for MUX25 11 : Select .3 input for MUX25 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Group Select Bits for MUX24: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX24 01 : Select .1 input for MUX24 10 : Select .2 input for MUX24 11 : Select .3 input for MUX24 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Group Select Bits for MUX23: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX23 01 : Select .1 input for MUX23 10 : Select .2 input for MUX23 11 : Select .3 input for MUX23 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Group Select Bits for MUX22: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX22 01 : Select .1 input for MUX22 10 : Select .2 input for MUX22 11 : Select .3 input for MUX22 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Group Select Bits for MUX21: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX21 01 : Select .1 input for MUX21 10 : Select .2 input for MUX21 11 : Select .3 input for MUX21 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Group Select Bits for MUX20: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX20 01 : Select .1 input for MUX20 10 : Select .2 input for MUX20 11 : Select .3 input for MUX20 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-74. AUXSIG3MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX19	R/W	0h	Group Select Bits for MUX19: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX19 01 : Select .1 input for MUX19 10 : Select .2 input for MUX19 11 : Select .3 input for MUX19 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Group Select Bits for MUX18: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX18 01 : Select .1 input for MUX18 10 : Select .2 input for MUX18 11 : Select .3 input for MUX18 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX17	R/W	0h	Group Select Bits for MUX17: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX17 01 : Select .1 input for MUX17 10 : Select .2 input for MUX17 11 : Select .3 input for MUX17 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Group Select Bits for MUX16: Selects 4X1 group output for AUXSIG3 of CLB-XBAR 00 : Select .0 input for MUX16 01 : Select .1 input for MUX16 10 : Select .2 input for MUX16 11 : Select .3 input for MUX16 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.5.9 AUXSIG4MUX0TO15CFG Register (Offset = 10h) [Reset = 0h]

AUXSIG4MUX0TO15CFG is shown in Figure 11-65 and described in Table 11-75.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-4

Figure 11-65. AUXSIG4MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-75. AUXSIG4MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Group Select Bits for MUX15: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX15 01 : Select .1 input for MUX15 10 : Select .2 input for MUX15 11 : Select .3 input for MUX15 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Group Select Bits for MUX14: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX14 01 : Select .1 input for MUX14 10 : Select .2 input for MUX14 11 : Select .3 input for MUX14 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Group Select Bits for MUX13: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX13 01 : Select .1 input for MUX13 10 : Select .2 input for MUX13 11 : Select .3 input for MUX13 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Group Select Bits for MUX12: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX12 01 : Select .1 input for MUX12 10 : Select .2 input for MUX12 11 : Select .3 input for MUX12 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Group Select Bits for MUX11: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX11 01 : Select .1 input for MUX11 10 : Select .2 input for MUX11 11 : Select .3 input for MUX11 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-75. AUXSIG4MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX10	R/W	0h	Group Select Bits for MUX10: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX10 01 : Select .1 input for MUX10 10 : Select .2 input for MUX10 11 : Select .3 input for MUX10 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX9	R/W	0h	Group Select Bits for MUX9: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX9 01 : Select .1 input for MUX9 10 : Select .2 input for MUX9 11 : Select .3 input for MUX9 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Group Select Bits for MUX8: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX8 01 : Select .1 input for MUX8 10 : Select .2 input for MUX8 11 : Select .3 input for MUX8 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Group Select Bits for MUX7: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX7 01 : Select .1 input for MUX7 10 : Select .2 input for MUX7 11 : Select .3 input for MUX7 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Group Select Bits for MUX6: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX6 01 : Select .1 input for MUX6 10 : Select .2 input for MUX6 11 : Select .3 input for MUX6 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Group Select Bits for MUX5: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX5 01 : Select .1 input for MUX5 10 : Select .2 input for MUX5 11 : Select .3 input for MUX5 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Group Select Bits for MUX4: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX4 01 : Select .1 input for MUX4 10 : Select .2 input for MUX4 11 : Select .3 input for MUX4 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-75. AUXSIG4MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX3	R/W	0h	Group Select Bits for MUX3: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX3 01 : Select .1 input for MUX3 10 : Select .2 input for MUX3 11 : Select .3 input for MUX3 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Group Select Bits for MUX2: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX2 01 : Select .1 input for MUX2 10 : Select .2 input for MUX2 11 : Select .3 input for MUX2 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX1	R/W	0h	Group Select Bits for MUX1: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX1 01 : Select .1 input for MUX1 10 : Select .2 input for MUX1 11 : Select .3 input for MUX1 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Group Select Bits for MUX0: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX0 01 : Select .1 input for MUX0 10 : Select .2 input for MUX0 11 : Select .3 input for MUX0 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.5.10 AUXSIG4MUX16TO31CFG Register (Offset = 12h) [Reset = 0h]

AUXSIG4MUX16TO31CFG is shown in Figure 11-66 and described in Table 11-76.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-4

Figure 11-66. AUXSIG4MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-76. AUXSIG4MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Group Select Bits for MUX31: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX31 01 : Select .1 input for MUX31 10 : Select .2 input for MUX31 11 : Select .3 input for MUX31 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Group Select Bits for MUX30: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX30 01 : Select .1 input for MUX30 10 : Select .2 input for MUX30 11 : Select .3 input for MUX30 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Group Select Bits for MUX29: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX29 01 : Select .1 input for MUX29 10 : Select .2 input for MUX29 11 : Select .3 input for MUX29 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Group Select Bits for MUX28: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX28 01 : Select .1 input for MUX28 10 : Select .2 input for MUX28 11 : Select .3 input for MUX28 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Group Select Bits for MUX27: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX27 01 : Select .1 input for MUX27 10 : Select .2 input for MUX27 11 : Select .3 input for MUX27 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-76. AUXSIG4MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX26	R/W	0h	Group Select Bits for MUX26: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX26 01 : Select .1 input for MUX26 10 : Select .2 input for MUX26 11 : Select .3 input for MUX26 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX25	R/W	0h	Group Select Bits for MUX25: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX25 01 : Select .1 input for MUX25 10 : Select .2 input for MUX25 11 : Select .3 input for MUX25 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Group Select Bits for MUX24: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX24 01 : Select .1 input for MUX24 10 : Select .2 input for MUX24 11 : Select .3 input for MUX24 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Group Select Bits for MUX23: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX23 01 : Select .1 input for MUX23 10 : Select .2 input for MUX23 11 : Select .3 input for MUX23 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Group Select Bits for MUX22: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX22 01 : Select .1 input for MUX22 10 : Select .2 input for MUX22 11 : Select .3 input for MUX22 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Group Select Bits for MUX21: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX21 01 : Select .1 input for MUX21 10 : Select .2 input for MUX21 11 : Select .3 input for MUX21 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Group Select Bits for MUX20: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX20 01 : Select .1 input for MUX20 10 : Select .2 input for MUX20 11 : Select .3 input for MUX20 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-76. AUXSIG4MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX19	R/W	0h	Group Select Bits for MUX19: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX19 01 : Select .1 input for MUX19 10 : Select .2 input for MUX19 11 : Select .3 input for MUX19 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Group Select Bits for MUX18: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX18 01 : Select .1 input for MUX18 10 : Select .2 input for MUX18 11 : Select .3 input for MUX18 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX17	R/W	0h	Group Select Bits for MUX17: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX17 01 : Select .1 input for MUX17 10 : Select .2 input for MUX17 11 : Select .3 input for MUX17 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Group Select Bits for MUX16: Selects 4X1 group output for AUXSIG4 of CLB-XBAR 00 : Select .0 input for MUX16 01 : Select .1 input for MUX16 10 : Select .2 input for MUX16 11 : Select .3 input for MUX16 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.5.11 AUXSIG5MUX0TO15CFG Register (Offset = 14h) [Reset = 0h]

AUXSIG5MUX0TO15CFG is shown in Figure 11-67 and described in Table 11-77.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-5

Figure 11-67. AUXSIG5MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-77. AUXSIG5MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Group Select Bits for MUX15: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX15 01 : Select .1 input for MUX15 10 : Select .2 input for MUX15 11 : Select .3 input for MUX15 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Group Select Bits for MUX14: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX14 01 : Select .1 input for MUX14 10 : Select .2 input for MUX14 11 : Select .3 input for MUX14 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Group Select Bits for MUX13: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX13 01 : Select .1 input for MUX13 10 : Select .2 input for MUX13 11 : Select .3 input for MUX13 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Group Select Bits for MUX12: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX12 01 : Select .1 input for MUX12 10 : Select .2 input for MUX12 11 : Select .3 input for MUX12 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Group Select Bits for MUX11: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX11 01 : Select .1 input for MUX11 10 : Select .2 input for MUX11 11 : Select .3 input for MUX11 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-77. AUXSIG5MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX10	R/W	0h	Group Select Bits for MUX10: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX10 01 : Select .1 input for MUX10 10 : Select .2 input for MUX10 11 : Select .3 input for MUX10 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX9	R/W	0h	Group Select Bits for MUX9: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX9 01 : Select .1 input for MUX9 10 : Select .2 input for MUX9 11 : Select .3 input for MUX9 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Group Select Bits for MUX8: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX8 01 : Select .1 input for MUX8 10 : Select .2 input for MUX8 11 : Select .3 input for MUX8 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Group Select Bits for MUX7: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX7 01 : Select .1 input for MUX7 10 : Select .2 input for MUX7 11 : Select .3 input for MUX7 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Group Select Bits for MUX6: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX6 01 : Select .1 input for MUX6 10 : Select .2 input for MUX6 11 : Select .3 input for MUX6 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Group Select Bits for MUX5: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX5 01 : Select .1 input for MUX5 10 : Select .2 input for MUX5 11 : Select .3 input for MUX5 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Group Select Bits for MUX4: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX4 01 : Select .1 input for MUX4 10 : Select .2 input for MUX4 11 : Select .3 input for MUX4 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-77. AUXSIG5MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX3	R/W	0h	Group Select Bits for MUX3: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX3 01 : Select .1 input for MUX3 10 : Select .2 input for MUX3 11 : Select .3 input for MUX3 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Group Select Bits for MUX2: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX2 01 : Select .1 input for MUX2 10 : Select .2 input for MUX2 11 : Select .3 input for MUX2 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX1	R/W	0h	Group Select Bits for MUX1: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX1 01 : Select .1 input for MUX1 10 : Select .2 input for MUX1 11 : Select .3 input for MUX1 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Group Select Bits for MUX0: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX0 01 : Select .1 input for MUX0 10 : Select .2 input for MUX0 11 : Select .3 input for MUX0 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.5.12 AUXSIG5MUX16TO31CFG Register (Offset = 16h) [Reset = 0h]

AUXSIG5MUX16TO31CFG is shown in Figure 11-68 and described in Table 11-78.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-5

Figure 11-68. AUXSIG5MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-78. AUXSIG5MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Group Select Bits for MUX31: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX31 01 : Select .1 input for MUX31 10 : Select .2 input for MUX31 11 : Select .3 input for MUX31 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Group Select Bits for MUX30: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX30 01 : Select .1 input for MUX30 10 : Select .2 input for MUX30 11 : Select .3 input for MUX30 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Group Select Bits for MUX29: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX29 01 : Select .1 input for MUX29 10 : Select .2 input for MUX29 11 : Select .3 input for MUX29 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Group Select Bits for MUX28: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX28 01 : Select .1 input for MUX28 10 : Select .2 input for MUX28 11 : Select .3 input for MUX28 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Group Select Bits for MUX27: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX27 01 : Select .1 input for MUX27 10 : Select .2 input for MUX27 11 : Select .3 input for MUX27 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-78. AUXSIG5MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX26	R/W	0h	Group Select Bits for MUX26: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX26 01 : Select .1 input for MUX26 10 : Select .2 input for MUX26 11 : Select .3 input for MUX26 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX25	R/W	0h	Group Select Bits for MUX25: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX25 01 : Select .1 input for MUX25 10 : Select .2 input for MUX25 11 : Select .3 input for MUX25 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Group Select Bits for MUX24: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX24 01 : Select .1 input for MUX24 10 : Select .2 input for MUX24 11 : Select .3 input for MUX24 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Group Select Bits for MUX23: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX23 01 : Select .1 input for MUX23 10 : Select .2 input for MUX23 11 : Select .3 input for MUX23 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Group Select Bits for MUX22: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX22 01 : Select .1 input for MUX22 10 : Select .2 input for MUX22 11 : Select .3 input for MUX22 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Group Select Bits for MUX21: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX21 01 : Select .1 input for MUX21 10 : Select .2 input for MUX21 11 : Select .3 input for MUX21 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Group Select Bits for MUX20: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX20 01 : Select .1 input for MUX20 10 : Select .2 input for MUX20 11 : Select .3 input for MUX20 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-78. AUXSIG5MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX19	R/W	0h	Group Select Bits for MUX19: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX19 01 : Select .1 input for MUX19 10 : Select .2 input for MUX19 11 : Select .3 input for MUX19 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Group Select Bits for MUX18: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX18 01 : Select .1 input for MUX18 10 : Select .2 input for MUX18 11 : Select .3 input for MUX18 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX17	R/W	0h	Group Select Bits for MUX17: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX17 01 : Select .1 input for MUX17 10 : Select .2 input for MUX17 11 : Select .3 input for MUX17 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Group Select Bits for MUX16: Selects 4X1 group output for AUXSIG5 of CLB-XBAR 00 : Select .0 input for MUX16 01 : Select .1 input for MUX16 10 : Select .2 input for MUX16 11 : Select .3 input for MUX16 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.5.13 AUXSIG6MUX0TO15CFG Register (Offset = 18h) [Reset = 0h]

AUXSIG6MUX0TO15CFG is shown in Figure 11-69 and described in Table 11-79.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-6

Figure 11-69. AUXSIG6MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-79. AUXSIG6MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Group Select Bits for MUX15: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX15 01 : Select .1 input for MUX15 10 : Select .2 input for MUX15 11 : Select .3 input for MUX15 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Group Select Bits for MUX14: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX14 01 : Select .1 input for MUX14 10 : Select .2 input for MUX14 11 : Select .3 input for MUX14 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Group Select Bits for MUX13: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX13 01 : Select .1 input for MUX13 10 : Select .2 input for MUX13 11 : Select .3 input for MUX13 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Group Select Bits for MUX12: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX12 01 : Select .1 input for MUX12 10 : Select .2 input for MUX12 11 : Select .3 input for MUX12 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Group Select Bits for MUX11: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX11 01 : Select .1 input for MUX11 10 : Select .2 input for MUX11 11 : Select .3 input for MUX11 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-79. AUXSIG6MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX10	R/W	0h	Group Select Bits for MUX10: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX10 01 : Select .1 input for MUX10 10 : Select .2 input for MUX10 11 : Select .3 input for MUX10 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX9	R/W	0h	Group Select Bits for MUX9: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX9 01 : Select .1 input for MUX9 10 : Select .2 input for MUX9 11 : Select .3 input for MUX9 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Group Select Bits for MUX8: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX8 01 : Select .1 input for MUX8 10 : Select .2 input for MUX8 11 : Select .3 input for MUX8 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Group Select Bits for MUX7: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX7 01 : Select .1 input for MUX7 10 : Select .2 input for MUX7 11 : Select .3 input for MUX7 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Group Select Bits for MUX6: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX6 01 : Select .1 input for MUX6 10 : Select .2 input for MUX6 11 : Select .3 input for MUX6 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Group Select Bits for MUX5: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX5 01 : Select .1 input for MUX5 10 : Select .2 input for MUX5 11 : Select .3 input for MUX5 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Group Select Bits for MUX4: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX4 01 : Select .1 input for MUX4 10 : Select .2 input for MUX4 11 : Select .3 input for MUX4 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-79. AUXSIG6MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX3	R/W	0h	Group Select Bits for MUX3: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX3 01 : Select .1 input for MUX3 10 : Select .2 input for MUX3 11 : Select .3 input for MUX3 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Group Select Bits for MUX2: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX2 01 : Select .1 input for MUX2 10 : Select .2 input for MUX2 11 : Select .3 input for MUX2 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX1	R/W	0h	Group Select Bits for MUX1: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX1 01 : Select .1 input for MUX1 10 : Select .2 input for MUX1 11 : Select .3 input for MUX1 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Group Select Bits for MUX0: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX0 01 : Select .1 input for MUX0 10 : Select .2 input for MUX0 11 : Select .3 input for MUX0 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.5.14 AUXSIG6MUX16TO31CFG Register (Offset = 1Ah) [Reset = 0h]

AUXSIG6MUX16TO31CFG is shown in Figure 11-70 and described in Table 11-80.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-6

Figure 11-70. AUXSIG6MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-80. AUXSIG6MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Group Select Bits for MUX31: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX31 01 : Select .1 input for MUX31 10 : Select .2 input for MUX31 11 : Select .3 input for MUX31 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Group Select Bits for MUX30: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX30 01 : Select .1 input for MUX30 10 : Select .2 input for MUX30 11 : Select .3 input for MUX30 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Group Select Bits for MUX29: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX29 01 : Select .1 input for MUX29 10 : Select .2 input for MUX29 11 : Select .3 input for MUX29 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Group Select Bits for MUX28: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX28 01 : Select .1 input for MUX28 10 : Select .2 input for MUX28 11 : Select .3 input for MUX28 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Group Select Bits for MUX27: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX27 01 : Select .1 input for MUX27 10 : Select .2 input for MUX27 11 : Select .3 input for MUX27 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-80. AUXSIG6MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX26	R/W	0h	Group Select Bits for MUX26: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX26 01 : Select .1 input for MUX26 10 : Select .2 input for MUX26 11 : Select .3 input for MUX26 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX25	R/W	0h	Group Select Bits for MUX25: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX25 01 : Select .1 input for MUX25 10 : Select .2 input for MUX25 11 : Select .3 input for MUX25 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Group Select Bits for MUX24: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX24 01 : Select .1 input for MUX24 10 : Select .2 input for MUX24 11 : Select .3 input for MUX24 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Group Select Bits for MUX23: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX23 01 : Select .1 input for MUX23 10 : Select .2 input for MUX23 11 : Select .3 input for MUX23 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Group Select Bits for MUX22: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX22 01 : Select .1 input for MUX22 10 : Select .2 input for MUX22 11 : Select .3 input for MUX22 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Group Select Bits for MUX21: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX21 01 : Select .1 input for MUX21 10 : Select .2 input for MUX21 11 : Select .3 input for MUX21 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Group Select Bits for MUX20: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX20 01 : Select .1 input for MUX20 10 : Select .2 input for MUX20 11 : Select .3 input for MUX20 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-80. AUXSIG6MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX19	R/W	0h	Group Select Bits for MUX19: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX19 01 : Select .1 input for MUX19 10 : Select .2 input for MUX19 11 : Select .3 input for MUX19 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Group Select Bits for MUX18: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX18 01 : Select .1 input for MUX18 10 : Select .2 input for MUX18 11 : Select .3 input for MUX18 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX17	R/W	0h	Group Select Bits for MUX17: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX17 01 : Select .1 input for MUX17 10 : Select .2 input for MUX17 11 : Select .3 input for MUX17 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Group Select Bits for MUX16: Selects 4X1 group output for AUXSIG6 of CLB-XBAR 00 : Select .0 input for MUX16 01 : Select .1 input for MUX16 10 : Select .2 input for MUX16 11 : Select .3 input for MUX16 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.5.15 AUXSIG7MUX0TO15CFG Register (Offset = 1Ch) [Reset = 0h]

AUXSIG7MUX0TO15CFG is shown in Figure 11-71 and described in Table 11-81.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-7

Figure 11-71. AUXSIG7MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-81. AUXSIG7MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Group Select Bits for MUX15: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX15 01 : Select .1 input for MUX15 10 : Select .2 input for MUX15 11 : Select .3 input for MUX15 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Group Select Bits for MUX14: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX14 01 : Select .1 input for MUX14 10 : Select .2 input for MUX14 11 : Select .3 input for MUX14 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Group Select Bits for MUX13: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX13 01 : Select .1 input for MUX13 10 : Select .2 input for MUX13 11 : Select .3 input for MUX13 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Group Select Bits for MUX12: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX12 01 : Select .1 input for MUX12 10 : Select .2 input for MUX12 11 : Select .3 input for MUX12 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Group Select Bits for MUX11: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX11 01 : Select .1 input for MUX11 10 : Select .2 input for MUX11 11 : Select .3 input for MUX11 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-81. AUXSIG7MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX10	R/W	0h	Group Select Bits for MUX10: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX10 01 : Select .1 input for MUX10 10 : Select .2 input for MUX10 11 : Select .3 input for MUX10 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX9	R/W	0h	Group Select Bits for MUX9: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX9 01 : Select .1 input for MUX9 10 : Select .2 input for MUX9 11 : Select .3 input for MUX9 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Group Select Bits for MUX8: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX8 01 : Select .1 input for MUX8 10 : Select .2 input for MUX8 11 : Select .3 input for MUX8 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Group Select Bits for MUX7: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX7 01 : Select .1 input for MUX7 10 : Select .2 input for MUX7 11 : Select .3 input for MUX7 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Group Select Bits for MUX6: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX6 01 : Select .1 input for MUX6 10 : Select .2 input for MUX6 11 : Select .3 input for MUX6 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Group Select Bits for MUX5: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX5 01 : Select .1 input for MUX5 10 : Select .2 input for MUX5 11 : Select .3 input for MUX5 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Group Select Bits for MUX4: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX4 01 : Select .1 input for MUX4 10 : Select .2 input for MUX4 11 : Select .3 input for MUX4 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-81. AUXSIG7MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX3	R/W	0h	Group Select Bits for MUX3: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX3 01 : Select .1 input for MUX3 10 : Select .2 input for MUX3 11 : Select .3 input for MUX3 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Group Select Bits for MUX2: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX2 01 : Select .1 input for MUX2 10 : Select .2 input for MUX2 11 : Select .3 input for MUX2 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX1	R/W	0h	Group Select Bits for MUX1: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX1 01 : Select .1 input for MUX1 10 : Select .2 input for MUX1 11 : Select .3 input for MUX1 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Group Select Bits for MUX0: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX0 01 : Select .1 input for MUX0 10 : Select .2 input for MUX0 11 : Select .3 input for MUX0 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.5.16 AUXSIG7MUX16TO31CFG Register (Offset = 1Eh) [Reset = 0h]

AUXSIG7MUX16TO31CFG is shown in Figure 11-72 and described in Table 11-82.

Return to the Summary Table.

CLB XBAR Mux Configuration for Output-7

Figure 11-72. AUXSIG7MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-82. AUXSIG7MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Group Select Bits for MUX31: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX31 01 : Select .1 input for MUX31 10 : Select .2 input for MUX31 11 : Select .3 input for MUX31 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Group Select Bits for MUX30: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX30 01 : Select .1 input for MUX30 10 : Select .2 input for MUX30 11 : Select .3 input for MUX30 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Group Select Bits for MUX29: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX29 01 : Select .1 input for MUX29 10 : Select .2 input for MUX29 11 : Select .3 input for MUX29 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Group Select Bits for MUX28: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX28 01 : Select .1 input for MUX28 10 : Select .2 input for MUX28 11 : Select .3 input for MUX28 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Group Select Bits for MUX27: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX27 01 : Select .1 input for MUX27 10 : Select .2 input for MUX27 11 : Select .3 input for MUX27 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-82. AUXSIG7MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21-20	MUX26	R/W	0h	Group Select Bits for MUX26: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX26 01 : Select .1 input for MUX26 10 : Select .2 input for MUX26 11 : Select .3 input for MUX26 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19-18	MUX25	R/W	0h	Group Select Bits for MUX25: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX25 01 : Select .1 input for MUX25 10 : Select .2 input for MUX25 11 : Select .3 input for MUX25 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Group Select Bits for MUX24: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX24 01 : Select .1 input for MUX24 10 : Select .2 input for MUX24 11 : Select .3 input for MUX24 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Group Select Bits for MUX23: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX23 01 : Select .1 input for MUX23 10 : Select .2 input for MUX23 11 : Select .3 input for MUX23 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Group Select Bits for MUX22: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX22 01 : Select .1 input for MUX22 10 : Select .2 input for MUX22 11 : Select .3 input for MUX22 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Group Select Bits for MUX21: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX21 01 : Select .1 input for MUX21 10 : Select .2 input for MUX21 11 : Select .3 input for MUX21 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Group Select Bits for MUX20: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX20 01 : Select .1 input for MUX20 10 : Select .2 input for MUX20 11 : Select .3 input for MUX20 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-82. AUXSIG7MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	MUX19	R/W	0h	Group Select Bits for MUX19: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX19 01 : Select .1 input for MUX19 10 : Select .2 input for MUX19 11 : Select .3 input for MUX19 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Group Select Bits for MUX18: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX18 01 : Select .1 input for MUX18 10 : Select .2 input for MUX18 11 : Select .3 input for MUX18 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3-2	MUX17	R/W	0h	Group Select Bits for MUX17: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX17 01 : Select .1 input for MUX17 10 : Select .2 input for MUX17 11 : Select .3 input for MUX17 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Group Select Bits for MUX16: Selects 4X1 group output for AUXSIG7 of CLB-XBAR 00 : Select .0 input for MUX16 01 : Select .1 input for MUX16 10 : Select .2 input for MUX16 11 : Select .3 input for MUX16 Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.5.17 AUXSIG0MUXENABLE Register (Offset = 20h) [Reset = 0h]

AUXSIG0MUXENABLE is shown in Figure 11-73 and described in Table 11-83.

Return to the Summary Table.

CLB XBAR Mux Enable Register for Output-0

Figure 11-73. AUXSIG0MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-83. AUXSIG0MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of MUX31 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX31 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX31 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of MUX30 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX30 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX30 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of MUX29 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX29 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX29 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of MUX28 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX28 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX28 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of MUX27 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX27 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX27 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-83. AUXSIG0MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of MUX26 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX26 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX26 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of MUX25 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX25 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX25 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of MUX24 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX24 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX24 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of MUX23 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX23 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX23 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of MUX22 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX22 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX22 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of MUX21 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX21 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX21 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of MUX20 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX20 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX20 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of MUX19 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX19 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX19 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-83. AUXSIG0MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of MUX18 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX18 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX18 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of MUX17 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX17 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX17 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of MUX16 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX16 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX16 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of MUX15 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX15 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX15 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of MUX14 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX14 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX14 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of MUX13 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX13 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX13 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of MUX12 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX12 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX12 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of MUX11 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX11 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX11 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-83. AUXSIG0MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of MUX10 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX10 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX10 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of MUX9 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX9 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX9 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of MUX8 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX8 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX8 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of MUX7 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX7 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX7 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of MUX6 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX6 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX6 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of MUX5 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX5 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX5 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of MUX4 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX4 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX4 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of MUX3 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX3 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX3 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-83. AUXSIG0MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of MUX2 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX2 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX2 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of MUX1 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX1 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX1 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive AUXSIG0 of CLB-XBAR 0: Respective output of MUX0 is disabled to drive the AUXSIG0 of CLB-XBAR 1: Respective output of MUX0 is enabled to drive the AUXSIG0 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.5.18 AUXSIG1MUXENABLE Register (Offset = 22h) [Reset = 0h]

AUXSIG1MUXENABLE is shown in Figure 11-74 and described in Table 11-84.

Return to the Summary Table.

CLB XBAR Mux Enable Register for Output-1

Figure 11-74. AUXSIG1MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-84. AUXSIG1MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of MUX31 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX31 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX31 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of MUX30 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX30 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX30 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of MUX29 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX29 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX29 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of MUX28 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX28 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX28 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of MUX27 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX27 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX27 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-84. AUXSIG1MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of MUX26 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX26 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX26 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of MUX25 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX25 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX25 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of MUX24 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX24 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX24 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of MUX23 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX23 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX23 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of MUX22 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX22 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX22 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of MUX21 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX21 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX21 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of MUX20 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX20 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX20 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of MUX19 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX19 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX19 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-84. AUXSIG1MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of MUX18 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX18 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX18 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of MUX17 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX17 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX17 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of MUX16 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX16 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX16 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of MUX15 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX15 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX15 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of MUX14 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX14 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX14 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of MUX13 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX13 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX13 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of MUX12 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX12 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX12 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of MUX11 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX11 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX11 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-84. AUXSIG1MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of MUX10 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX10 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX10 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of MUX9 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX9 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX9 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of MUX8 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX8 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX8 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of MUX7 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX7 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX7 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of MUX6 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX6 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX6 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of MUX5 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX5 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX5 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of MUX4 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX4 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX4 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of MUX3 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX3 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX3 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-84. AUXSIG1MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of MUX2 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX2 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX2 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of MUX1 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX1 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX1 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive AUXSIG1 of CLB-XBAR 0: Respective output of MUX0 is disabled to drive the AUXSIG1 of CLB-XBAR 1: Respective output of MUX0 is enabled to drive the AUXSIG1 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.5.19 AUXSIG2MUXENABLE Register (Offset = 24h) [Reset = 0h]

AUXSIG2MUXENABLE is shown in Figure 11-75 and described in Table 11-85.

Return to the Summary Table.

CLB XBAR Mux Enable Register for Output-2

Figure 11-75. AUXSIG2MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-85. AUXSIG2MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of MUX31 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX31 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX31 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of MUX30 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX30 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX30 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of MUX29 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX29 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX29 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of MUX28 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX28 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX28 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of MUX27 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX27 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX27 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-85. AUXSIG2MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of MUX26 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX26 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX26 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of MUX25 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX25 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX25 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of MUX24 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX24 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX24 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of MUX23 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX23 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX23 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of MUX22 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX22 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX22 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of MUX21 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX21 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX21 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of MUX20 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX20 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX20 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of MUX19 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX19 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX19 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-85. AUXSIG2MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of MUX18 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX18 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX18 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of MUX17 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX17 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX17 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of MUX16 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX16 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX16 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of MUX15 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX15 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX15 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of MUX14 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX14 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX14 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of MUX13 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX13 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX13 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of MUX12 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX12 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX12 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of MUX11 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX11 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX11 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-85. AUXSIG2MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of MUX10 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX10 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX10 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of MUX9 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX9 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX9 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of MUX8 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX8 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX8 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of MUX7 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX7 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX7 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of MUX6 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX6 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX6 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of MUX5 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX5 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX5 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of MUX4 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX4 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX4 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of MUX3 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX3 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX3 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-85. AUXSIG2MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of MUX2 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX2 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX2 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of MUX1 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX1 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX1 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive AUXSIG2 of CLB-XBAR 0: Respective output of MUX0 is disabled to drive the AUXSIG2 of CLB-XBAR 1: Respective output of MUX0 is enabled to drive the AUXSIG2 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.5.20 AUXSIG3MUXENABLE Register (Offset = 26h) [Reset = 0h]

AUXSIG3MUXENABLE is shown in Figure 11-76 and described in Table 11-86.

Return to the Summary Table.

CLB XBAR Mux Enable Register for Output-3

Figure 11-76. AUXSIG3MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-86. AUXSIG3MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of MUX31 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX31 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX31 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of MUX30 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX30 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX30 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of MUX29 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX29 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX29 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of MUX28 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX28 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX28 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of MUX27 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX27 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX27 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-86. AUXSIG3MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of MUX26 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX26 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX26 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of MUX25 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX25 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX25 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of MUX24 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX24 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX24 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of MUX23 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX23 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX23 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of MUX22 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX22 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX22 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of MUX21 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX21 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX21 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of MUX20 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX20 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX20 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of MUX19 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX19 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX19 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-86. AUXSIG3MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of MUX18 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX18 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX18 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of MUX17 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX17 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX17 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of MUX16 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX16 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX16 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of MUX15 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX15 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX15 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of MUX14 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX14 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX14 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of MUX13 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX13 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX13 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of MUX12 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX12 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX12 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of MUX11 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX11 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX11 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-86. AUXSIG3MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of MUX10 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX10 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX10 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of MUX9 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX9 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX9 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of MUX8 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX8 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX8 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of MUX7 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX7 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX7 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of MUX6 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX6 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX6 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of MUX5 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX5 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX5 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of MUX4 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX4 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX4 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of MUX3 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX3 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX3 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-86. AUXSIG3MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of MUX2 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX2 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX2 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of MUX1 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX1 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX1 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive AUXSIG3 of CLB-XBAR 0: Respective output of MUX0 is disabled to drive the AUXSIG3 of CLB-XBAR 1: Respective output of MUX0 is enabled to drive the AUXSIG3 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.5.21 AUXSIG4MUXENABLE Register (Offset = 28h) [Reset = 0h]

AUXSIG4MUXENABLE is shown in Figure 11-77 and described in Table 11-87.

Return to the Summary Table.

CLB XBAR Mux Enable Register for Output-4

Figure 11-77. AUXSIG4MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-87. AUXSIG4MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of MUX31 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX31 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX31 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of MUX30 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX30 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX30 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of MUX29 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX29 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX29 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of MUX28 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX28 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX28 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of MUX27 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX27 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX27 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-87. AUXSIG4MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of MUX26 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX26 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX26 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of MUX25 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX25 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX25 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of MUX24 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX24 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX24 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of MUX23 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX23 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX23 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of MUX22 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX22 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX22 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of MUX21 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX21 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX21 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of MUX20 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX20 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX20 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of MUX19 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX19 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX19 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-87. AUXSIG4MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of MUX18 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX18 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX18 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of MUX17 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX17 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX17 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of MUX16 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX16 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX16 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of MUX15 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX15 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX15 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of MUX14 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX14 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX14 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of MUX13 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX13 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX13 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of MUX12 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX12 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX12 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of MUX11 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX11 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX11 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-87. AUXSIG4MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of MUX10 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX10 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX10 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of MUX9 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX9 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX9 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of MUX8 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX8 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX8 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of MUX7 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX7 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX7 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of MUX6 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX6 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX6 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of MUX5 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX5 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX5 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of MUX4 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX4 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX4 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of MUX3 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX3 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX3 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-87. AUXSIG4MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of MUX2 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX2 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX2 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of MUX1 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX1 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX1 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive AUXSIG4 of CLB-XBAR 0: Respective output of MUX0 is disabled to drive the AUXSIG4 of CLB-XBAR 1: Respective output of MUX0 is enabled to drive the AUXSIG4 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.5.22 AUXSIG5MUXENABLE Register (Offset = 2Ah) [Reset = 0h]

AUXSIG5MUXENABLE is shown in Figure 11-78 and described in Table 11-88.

Return to the Summary Table.

CLB XBAR Mux Enable Register for Output-5

Figure 11-78. AUXSIG5MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-88. AUXSIG5MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of MUX31 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX31 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX31 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of MUX30 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX30 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX30 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of MUX29 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX29 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX29 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of MUX28 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX28 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX28 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of MUX27 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX27 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX27 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-88. AUXSIG5MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of MUX26 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX26 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX26 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of MUX25 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX25 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX25 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of MUX24 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX24 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX24 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of MUX23 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX23 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX23 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of MUX22 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX22 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX22 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of MUX21 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX21 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX21 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of MUX20 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX20 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX20 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of MUX19 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX19 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX19 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-88. AUXSIG5MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of MUX18 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX18 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX18 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of MUX17 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX17 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX17 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of MUX16 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX16 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX16 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of MUX15 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX15 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX15 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of MUX14 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX14 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX14 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of MUX13 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX13 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX13 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of MUX12 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX12 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX12 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of MUX11 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX11 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX11 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-88. AUXSIG5MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of MUX10 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX10 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX10 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of MUX9 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX9 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX9 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of MUX8 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX8 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX8 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of MUX7 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX7 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX7 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of MUX6 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX6 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX6 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of MUX5 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX5 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX5 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of MUX4 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX4 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX4 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of MUX3 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX3 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX3 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-88. AUXSIG5MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of MUX2 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX2 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX2 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of MUX1 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX1 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX1 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive AUXSIG5 of CLB-XBAR 0: Respective output of MUX0 is disabled to drive the AUXSIG5 of CLB-XBAR 1: Respective output of MUX0 is enabled to drive the AUXSIG5 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.5.23 AUXSIG6MUXENABLE Register (Offset = 2Ch) [Reset = 0h]

AUXSIG6MUXENABLE is shown in Figure 11-79 and described in Table 11-89.

Return to the Summary Table.

CLB XBAR Mux Enable Register for Output-6

Figure 11-79. AUXSIG6MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-89. AUXSIG6MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of MUX31 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX31 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX31 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of MUX30 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX30 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX30 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of MUX29 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX29 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX29 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of MUX28 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX28 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX28 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of MUX27 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX27 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX27 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-89. AUXSIG6MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of MUX26 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX26 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX26 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of MUX25 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX25 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX25 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of MUX24 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX24 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX24 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of MUX23 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX23 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX23 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of MUX22 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX22 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX22 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of MUX21 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX21 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX21 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of MUX20 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX20 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX20 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of MUX19 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX19 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX19 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-89. AUXSIG6MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of MUX18 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX18 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX18 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of MUX17 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX17 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX17 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of MUX16 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX16 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX16 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of MUX15 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX15 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX15 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of MUX14 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX14 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX14 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of MUX13 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX13 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX13 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of MUX12 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX12 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX12 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of MUX11 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX11 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX11 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-89. AUXSIG6MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of MUX10 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX10 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX10 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of MUX9 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX9 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX9 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of MUX8 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX8 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX8 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of MUX7 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX7 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX7 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of MUX6 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX6 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX6 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of MUX5 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX5 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX5 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of MUX4 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX4 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX4 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of MUX3 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX3 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX3 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-89. AUXSIG6MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of MUX2 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX2 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX2 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of MUX1 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX1 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX1 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive AUXSIG6 of CLB-XBAR 0: Respective output of MUX0 is disabled to drive the AUXSIG6 of CLB-XBAR 1: Respective output of MUX0 is enabled to drive the AUXSIG6 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.5.24 AUXSIG7MUXENABLE Register (Offset = 2Eh) [Reset = 0h]

AUXSIG7MUXENABLE is shown in Figure 11-80 and described in Table 11-90.

Return to the Summary Table.

CLB XBAR Mux Enable Register for Output-7

Figure 11-80. AUXSIG7MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-90. AUXSIG7MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of MUX31 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX31 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX31 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of MUX30 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX30 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX30 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of MUX29 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX29 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX29 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of MUX28 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX28 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX28 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of MUX27 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX27 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX27 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-90. AUXSIG7MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of MUX26 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX26 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX26 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of MUX25 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX25 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX25 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of MUX24 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX24 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX24 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of MUX23 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX23 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX23 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of MUX22 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX22 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX22 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of MUX21 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX21 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX21 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of MUX20 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX20 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX20 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of MUX19 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX19 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX19 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-90. AUXSIG7MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of MUX18 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX18 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX18 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of MUX17 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX17 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX17 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of MUX16 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX16 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX16 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of MUX15 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX15 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX15 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of MUX14 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX14 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX14 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of MUX13 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX13 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX13 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of MUX12 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX12 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX12 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of MUX11 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX11 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX11 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-90. AUXSIG7MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of MUX10 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX10 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX10 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of MUX9 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX9 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX9 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of MUX8 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX8 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX8 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of MUX7 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX7 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX7 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of MUX6 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX6 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX6 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of MUX5 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX5 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX5 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of MUX4 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX4 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX4 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of MUX3 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX3 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX3 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-90. AUXSIG7MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of MUX2 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX2 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX2 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of MUX1 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX1 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX1 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive AUXSIG7 of CLB-XBAR 0: Respective output of MUX0 is disabled to drive the AUXSIG7 of CLB-XBAR 1: Respective output of MUX0 is enabled to drive the AUXSIG7 of CLB-XBAR Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.5.25 AUXSIGOUTINV Register (Offset = 38h) [Reset = 0h]

AUXSIGOUTINV is shown in Figure 11-81 and described in Table 11-91.

Return to the Summary Table.

CLB XBAR Output Inversion Register

Figure 11-81. AUXSIGOUTINV Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
OUT7	OUT6	OUT5	OUT4	OUT3	OUT2	OUT1	OUT0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-91. AUXSIGOUTINV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7	OUT7	R/W	0h	Selects polarity for AUXSIG7 of CLB-XBAR 0: drives active high output 1: drives active-low output Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	OUT6	R/W	0h	Selects polarity for AUXSIG6 of CLB-XBAR 0: drives active high output 1: drives active-low output Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	OUT5	R/W	0h	Selects polarity for AUXSIG5 of CLB-XBAR 0: drives active high output 1: drives active-low output Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	OUT4	R/W	0h	Selects polarity for AUXSIG4 of CLB-XBAR 0: drives active high output 1: drives active-low output Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	OUT3	R/W	0h	Selects polarity for AUXSIG3 of CLB-XBAR 0: drives active high output 1: drives active-low output Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
2	OUT2	R/W	0h	Selects polarity for AUXSIG2 of CLB-XBAR 0: drives active high output 1: drives active-low output Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-91. AUXSIGOUTINV Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	OUT1	R/W	0h	Selects polarity for AUXSIG1 of CLB-XBAR 0: drives active high output 1: drives active-low output Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	OUT0	R/W	0h	Selects polarity for AUXSIG0 of CLB-XBAR 0: drives active high output 1: drives active-low output Refer to the CLB X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.5.26 AUXSIGLOCK Register (Offset = 3Eh) [Reset = 0h]

AUXSIGLOCK is shown in Figure 11-82 and described in Table 11-92.

Return to the Summary Table.

ClbXbar Configuration Lock register

Figure 11-82. AUXSIGLOCK Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							LOCK
R-0h							R/WSonce-0h

Table 11-92. AUXSIGLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Bit-0 of this register can be set only if KEY= 0x5a5a Reset type: CPU1.SYSRSn
15-1	RESERVED	R	0h	Reserved
0	LOCK	R/WSonce	0h	Locks the configuration for CLB-XBAR. Once the configuration is locked, writes to the below registers for CLB-XBAR is blocked. Registers Affected by the LOCK mechanism: CLB-XBAROUTyMUX0TO15CFG CLB-XBAROUTyMUX16TO31CFG CLB-XBAROUTyMUXENABLE CLB-XBAROUTLATEN CLB-XBAROUTINV 0: Writes to the above registers are allowed 1: Writes to the above registers are blocked Note: [1] LOCK mechanism only apples to writes. Reads are never blocked. Reset type: CPU1.SYSRSn

11.3.6 OUTPUT_XBAR_REGS Registers

Table 11-93 lists the memory-mapped registers for the OUTPUT_XBAR_REGS registers. All register offset addresses not listed in Table 11-93 should be considered as reserved locations and the register contents should not be modified.

Table 11-93. OUTPUT_XBAR_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	OUTPUT1MUX0TO15CFG	Output X-BAR Mux Configuration for Output 1	EALLOW	Go
2h	OUTPUT1MUX16TO31CFG	Output X-BAR Mux Configuration for Output 1	EALLOW	Go
4h	OUTPUT2MUX0TO15CFG	Output X-BAR Mux Configuration for Output 2	EALLOW	Go
6h	OUTPUT2MUX16TO31CFG	Output X-BAR Mux Configuration for Output 2	EALLOW	Go
8h	OUTPUT3MUX0TO15CFG	Output X-BAR Mux Configuration for Output 3	EALLOW	Go
Ah	OUTPUT3MUX16TO31CFG	Output X-BAR Mux Configuration for Output 3	EALLOW	Go
Ch	OUTPUT4MUX0TO15CFG	Output X-BAR Mux Configuration for Output 4	EALLOW	Go
Eh	OUTPUT4MUX16TO31CFG	Output X-BAR Mux Configuration for Output 4	EALLOW	Go
10h	OUTPUT5MUX0TO15CFG	Output X-BAR Mux Configuration for Output 5	EALLOW	Go
12h	OUTPUT5MUX16TO31CFG	Output X-BAR Mux Configuration for Output 5	EALLOW	Go
14h	OUTPUT6MUX0TO15CFG	Output X-BAR Mux Configuration for Output 6	EALLOW	Go
16h	OUTPUT6MUX16TO31CFG	Output X-BAR Mux Configuration for Output 6	EALLOW	Go
18h	OUTPUT7MUX0TO15CFG	Output X-BAR Mux Configuration for Output 7	EALLOW	Go
1Ah	OUTPUT7MUX16TO31CFG	Output X-BAR Mux Configuration for Output 7	EALLOW	Go
1Ch	OUTPUT8MUX0TO15CFG	Output X-BAR Mux Configuration for Output 8	EALLOW	Go
1Eh	OUTPUT8MUX16TO31CFG	Output X-BAR Mux Configuration for Output 8	EALLOW	Go
20h	OUTPUT1MUXENABLE	Output X-BAR Mux Enable for Output 1	EALLOW	Go
22h	OUTPUT2MUXENABLE	Output X-BAR Mux Enable for Output 2	EALLOW	Go
24h	OUTPUT3MUXENABLE	Output X-BAR Mux Enable for Output 3	EALLOW	Go
26h	OUTPUT4MUXENABLE	Output X-BAR Mux Enable for Output 4	EALLOW	Go
28h	OUTPUT5MUXENABLE	Output X-BAR Mux Enable for Output 5	EALLOW	Go
2Ah	OUTPUT6MUXENABLE	Output X-BAR Mux Enable for Output 6	EALLOW	Go
2Ch	OUTPUT7MUXENABLE	Output X-BAR Mux Enable for Output 7	EALLOW	Go
2Eh	OUTPUT8MUXENABLE	Output X-BAR Mux Enable for Output 8	EALLOW	Go
30h	OUTPUTLATCH	Output X-BAR Output Latch		Go
32h	OUTPUTLATCHCLR	Output X-BAR Output Latch Clear		Go
34h	OUTPUTLATCHFRC	Output X-BAR Output Latch Clear		Go
36h	OUTPUTLATCHENABLE	Output X-BAR Output Latch Enable	EALLOW	Go
38h	OUTPUTINV	Output X-BAR Output Inversion	EALLOW	Go
3Eh	OUTPUTLOCK	Output X-BAR Configuration Lock register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 11-94 shows the codes that are used for access types in this section.

Table 11-94. OUTPUT_XBAR_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		

Table 11-94. OUTPUT_XBAR_REGS Access Type Codes (continued)

Access Type	Code	Description
W	W	Write
W1S	W 1S	Write 1 to set
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

11.3.6.1 OUTPUT1MUX0TO15CFG Register (Offset = 0h) [Reset = 0h]

OUTPUT1MUX0TO15CFG is shown in Figure 11-83 and described in Table 11-95.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 1

Figure 11-83. OUTPUT1MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-95. OUTPUT1MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for OUTPUT1 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for OUTPUT1 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for OUTPUT1 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for OUTPUT1 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for OUTPUT1 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for OUTPUT1 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-95. OUTPUT1MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for OUTPUT1 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for OUTPUT1 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for OUTPUT1 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for OUTPUT1 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for OUTPUT1 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for OUTPUT1 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for OUTPUT1 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for OUTPUT1 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-95. OUTPUT1MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for OUTPUT1 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for OUTPUT1 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.2 OUTPUT1MUX16TO31CFG Register (Offset = 2h) [Reset = 0h]

OUTPUT1MUX16TO31CFG is shown in Figure 11-84 and described in Table 11-96.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 1

Figure 11-84. OUTPUT1MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-96. OUTPUT1MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for OUTPUT1 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for OUTPUT1 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for OUTPUT1 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for OUTPUT1 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for OUTPUT1 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for OUTPUT1 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-96. OUTPUT1MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for OUTPUT1 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for OUTPUT1 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for OUTPUT1 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for OUTPUT1 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for OUTPUT1 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for OUTPUT1 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for OUTPUT1 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for OUTPUT1 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-96. OUTPUT1MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for OUTPUT1 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for OUTPUT1 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.3 OUTPUT2MUX0TO15CFG Register (Offset = 4h) [Reset = 0h]

OUTPUT2MUX0TO15CFG is shown in Figure 11-85 and described in Table 11-97.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 2

Figure 11-85. OUTPUT2MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-97. OUTPUT2MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for OUTPUT2 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for OUTPUT2 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for OUTPUT2 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for OUTPUT2 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for OUTPUT2 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for OUTPUT2 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-97. OUTPUT2MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for OUTPUT2 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for OUTPUT2 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for OUTPUT2 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for OUTPUT2 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for OUTPUT2 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for OUTPUT2 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for OUTPUT2 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for OUTPUT2 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-97. OUTPUT2MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for OUTPUT2 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for OUTPUT2 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.4 OUTPUT2MUX16TO31CFG Register (Offset = 6h) [Reset = 0h]

OUTPUT2MUX16TO31CFG is shown in Figure 11-86 and described in Table 11-98.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 2

Figure 11-86. OUTPUT2MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-98. OUTPUT2MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for OUTPUT2 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for OUTPUT2 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for OUTPUT2 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for OUTPUT2 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for OUTPUT2 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for OUTPUT2 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-98. OUTPUT2MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for OUTPUT2 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for OUTPUT2 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for OUTPUT2 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for OUTPUT2 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for OUTPUT2 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for OUTPUT2 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for OUTPUT2 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for OUTPUT2 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-98. OUTPUT2MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for OUTPUT2 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for OUTPUT2 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.5 OUTPUT3MUX0TO15CFG Register (Offset = 8h) [Reset = 0h]

OUTPUT3MUX0TO15CFG is shown in Figure 11-87 and described in Table 11-99.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 3

Figure 11-87. OUTPUT3MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-99. OUTPUT3MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for OUTPUT3 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for OUTPUT3 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for OUTPUT3 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for OUTPUT3 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for OUTPUT3 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for OUTPUT3 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-99. OUTPUT3MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for OUTPUT3 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for OUTPUT3 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for OUTPUT3 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for OUTPUT3 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for OUTPUT3 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for OUTPUT3 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for OUTPUT3 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for OUTPUT3 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-99. OUTPUT3MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for OUTPUT3 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for OUTPUT3 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.6 OUTPUT3MUX16TO31CFG Register (Offset = Ah) [Reset = 0h]

OUTPUT3MUX16TO31CFG is shown in Figure 11-88 and described in Table 11-100.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 3

Figure 11-88. OUTPUT3MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-100. OUTPUT3MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for OUTPUT3 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for OUTPUT3 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for OUTPUT3 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for OUTPUT3 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for OUTPUT3 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for OUTPUT3 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-100. OUTPUT3MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for OUTPUT3 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for OUTPUT3 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for OUTPUT3 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for OUTPUT3 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for OUTPUT3 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for OUTPUT3 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for OUTPUT3 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for OUTPUT3 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-100. OUTPUT3MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for OUTPUT3 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for OUTPUT3 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.7 OUTPUT4MUX0TO15CFG Register (Offset = Ch) [Reset = 0h]

OUTPUT4MUX0TO15CFG is shown in Figure 11-89 and described in Table 11-101.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 4

Figure 11-89. OUTPUT4MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-101. OUTPUT4MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for OUTPUT4 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for OUTPUT4 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for OUTPUT4 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for OUTPUT4 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for OUTPUT4 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for OUTPUT4 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-101. OUTPUT4MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for OUTPUT4 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for OUTPUT4 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for OUTPUT4 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for OUTPUT4 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for OUTPUT4 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for OUTPUT4 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for OUTPUT4 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for OUTPUT4 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-101. OUTPUT4MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for OUTPUT4 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for OUTPUT4 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.8 OUTPUT4MUX16TO31CFG Register (Offset = Eh) [Reset = 0h]

OUTPUT4MUX16TO31CFG is shown in Figure 11-90 and described in Table 11-102.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 4

Figure 11-90. OUTPUT4MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-102. OUTPUT4MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for OUTPUT4 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for OUTPUT4 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for OUTPUT4 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for OUTPUT4 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for OUTPUT4 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for OUTPUT4 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-102. OUTPUT4MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for OUTPUT4 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for OUTPUT4 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for OUTPUT4 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for OUTPUT4 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for OUTPUT4 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for OUTPUT4 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for OUTPUT4 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for OUTPUT4 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-102. OUTPUT4MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for OUTPUT4 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for OUTPUT4 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.9 OUTPUT5MUX0TO15CFG Register (Offset = 10h) [Reset = 0h]

OUTPUT5MUX0TO15CFG is shown in Figure 11-91 and described in Table 11-103.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 5

Figure 11-91. OUTPUT5MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-103. OUTPUT5MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for OUTPUT5 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for OUTPUT5 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for OUTPUT5 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for OUTPUT5 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for OUTPUT5 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for OUTPUT5 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-103. OUTPUT5MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for OUTPUT5 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for OUTPUT5 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for OUTPUT5 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for OUTPUT5 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for OUTPUT5 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for OUTPUT5 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for OUTPUT5 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for OUTPUT5 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-103. OUTPUT5MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for OUTPUT5 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for OUTPUT5 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.10 OUTPUT5MUX16TO31CFG Register (Offset = 12h) [Reset = 0h]

OUTPUT5MUX16TO31CFG is shown in Figure 11-92 and described in Table 11-104.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 5

Figure 11-92. OUTPUT5MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-104. OUTPUT5MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for OUTPUT5 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for OUTPUT5 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for OUTPUT5 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for OUTPUT5 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for OUTPUT5 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for OUTPUT5 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-104. OUTPUT5MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for OUTPUT5 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for OUTPUT5 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for OUTPUT5 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for OUTPUT5 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for OUTPUT5 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for OUTPUT5 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for OUTPUT5 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for OUTPUT5 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-104. OUTPUT5MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for OUTPUT5 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for OUTPUT5 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.11 OUTPUT6MUX0TO15CFG Register (Offset = 14h) [Reset = 0h]

OUTPUT6MUX0TO15CFG is shown in Figure 11-93 and described in Table 11-105.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 6

Figure 11-93. OUTPUT6MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-105. OUTPUT6MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for OUTPUT6 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for OUTPUT6 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for OUTPUT6 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for OUTPUT6 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for OUTPUT6 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for OUTPUT6 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-105. OUTPUT6MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for OUTPUT6 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for OUTPUT6 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for OUTPUT6 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for OUTPUT6 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for OUTPUT6 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for OUTPUT6 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for OUTPUT6 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for OUTPUT6 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-105. OUTPUT6MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for OUTPUT6 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for OUTPUT6 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.12 OUTPUT6MUX16TO31CFG Register (Offset = 16h) [Reset = 0h]

OUTPUT6MUX16TO31CFG is shown in Figure 11-94 and described in Table 11-106.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 6

Figure 11-94. OUTPUT6MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-106. OUTPUT6MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for OUTPUT6 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for OUTPUT6 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for OUTPUT6 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for OUTPUT6 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for OUTPUT6 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for OUTPUT6 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-106. OUTPUT6MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for OUTPUT6 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for OUTPUT6 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for OUTPUT6 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for OUTPUT6 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for OUTPUT6 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for OUTPUT6 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for OUTPUT6 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for OUTPUT6 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-106. OUTPUT6MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for OUTPUT6 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for OUTPUT6 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.13 OUTPUT7MUX0TO15CFG Register (Offset = 18h) [Reset = 0h]

OUTPUT7MUX0TO15CFG is shown in Figure 11-95 and described in Table 11-107.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 7

Figure 11-95. OUTPUT7MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-107. OUTPUT7MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for OUTPUT7 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for OUTPUT7 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for OUTPUT7 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for OUTPUT7 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for OUTPUT7 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for OUTPUT7 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-107. OUTPUT7MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for OUTPUT7 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for OUTPUT7 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for OUTPUT7 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for OUTPUT7 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for OUTPUT7 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for OUTPUT7 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for OUTPUT7 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for OUTPUT7 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-107. OUTPUT7MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for OUTPUT7 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for OUTPUT7 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.14 OUTPUT7MUX16TO31CFG Register (Offset = 1Ah) [Reset = 0h]

OUTPUT7MUX16TO31CFG is shown in Figure 11-96 and described in Table 11-108.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 7

Figure 11-96. OUTPUT7MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-108. OUTPUT7MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for OUTPUT7 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for OUTPUT7 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for OUTPUT7 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for OUTPUT7 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for OUTPUT7 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for OUTPUT7 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-108. OUTPUT7MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for OUTPUT7 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for OUTPUT7 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for OUTPUT7 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for OUTPUT7 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for OUTPUT7 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for OUTPUT7 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for OUTPUT7 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for OUTPUT7 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-108. OUTPUT7MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for OUTPUT7 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for OUTPUT7 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.15 OUTPUT8MUX0TO15CFG Register (Offset = 1Ch) [Reset = 0h]

OUTPUT8MUX0TO15CFG is shown in Figure 11-97 and described in Table 11-109.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 8

Figure 11-97. OUTPUT8MUX0TO15CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX15		MUX14		MUX13		MUX12		MUX11		MUX10		MUX9		MUX8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX7		MUX6		MUX5		MUX4		MUX3		MUX2		MUX1		MUX0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-109. OUTPUT8MUX0TO15CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX15	R/W	0h	Select Bits for OUTPUT8 Mux15: 00 : Select .0 input for Mux15 01 : Select .1 input for Mux15 10 : Select .2 input for Mux15 11 : Select .3 input for Mux15 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX14	R/W	0h	Select Bits for OUTPUT8 Mux14: 00 : Select .0 input for Mux14 01 : Select .1 input for Mux14 10 : Select .2 input for Mux14 11 : Select .3 input for Mux14 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX13	R/W	0h	Select Bits for OUTPUT8 Mux13: 00 : Select .0 input for Mux13 01 : Select .1 input for Mux13 10 : Select .2 input for Mux13 11 : Select .3 input for Mux13 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX12	R/W	0h	Select Bits for OUTPUT8 Mux12: 00 : Select .0 input for Mux12 01 : Select .1 input for Mux12 10 : Select .2 input for Mux12 11 : Select .3 input for Mux12 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX11	R/W	0h	Select Bits for OUTPUT8 Mux11: 00 : Select .0 input for Mux11 01 : Select .1 input for Mux11 10 : Select .2 input for Mux11 11 : Select .3 input for Mux11 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX10	R/W	0h	Select Bits for OUTPUT8 Mux10: 00 : Select .0 input for Mux10 01 : Select .1 input for Mux10 10 : Select .2 input for Mux10 11 : Select .3 input for Mux10 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-109. OUTPUT8MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX9	R/W	0h	Select Bits for OUTPUT8 Mux9: 00 : Select .0 input for Mux9 01 : Select .1 input for Mux9 10 : Select .2 input for Mux9 11 : Select .3 input for Mux9 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX8	R/W	0h	Select Bits for OUTPUT8 Mux8: 00 : Select .0 input for Mux8 01 : Select .1 input for Mux8 10 : Select .2 input for Mux8 11 : Select .3 input for Mux8 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX7	R/W	0h	Select Bits for OUTPUT8 Mux7: 00 : Select .0 input for Mux7 01 : Select .1 input for Mux7 10 : Select .2 input for Mux7 11 : Select .3 input for Mux7 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX6	R/W	0h	Select Bits for OUTPUT8 Mux6: 00 : Select .0 input for Mux6 01 : Select .1 input for Mux6 10 : Select .2 input for Mux6 11 : Select .3 input for Mux6 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX5	R/W	0h	Select Bits for OUTPUT8 Mux5: 00 : Select .0 input for Mux5 01 : Select .1 input for Mux5 10 : Select .2 input for Mux5 11 : Select .3 input for Mux5 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX4	R/W	0h	Select Bits for OUTPUT8 Mux4: 00 : Select .0 input for Mux4 01 : Select .1 input for Mux4 10 : Select .2 input for Mux4 11 : Select .3 input for Mux4 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX3	R/W	0h	Select Bits for OUTPUT8 Mux3: 00 : Select .0 input for Mux3 01 : Select .1 input for Mux3 10 : Select .2 input for Mux3 11 : Select .3 input for Mux3 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX2	R/W	0h	Select Bits for OUTPUT8 Mux2: 00 : Select .0 input for Mux2 01 : Select .1 input for Mux2 10 : Select .2 input for Mux2 11 : Select .3 input for Mux2 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-109. OUTPUT8MUX0TO15CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX1	R/W	0h	Select Bits for OUTPUT8 Mux1: 00 : Select .0 input for Mux1 01 : Select .1 input for Mux1 10 : Select .2 input for Mux1 11 : Select .3 input for Mux1 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX0	R/W	0h	Select Bits for OUTPUT8 Mux0: 00 : Select .0 input for Mux0 01 : Select .1 input for Mux0 10 : Select .2 input for Mux0 11 : Select .3 input for Mux0 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.16 OUTPUT8MUX16TO31CFG Register (Offset = 1Eh) [Reset = 0h]

OUTPUT8MUX16TO31CFG is shown in Figure 11-98 and described in Table 11-110.

Return to the Summary Table.

Output X-BAR Mux Configuration for Output 8

Figure 11-98. OUTPUT8MUX16TO31CFG Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MUX31		MUX30		MUX29		MUX28		MUX27		MUX26		MUX25		MUX24	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MUX23		MUX22		MUX21		MUX20		MUX19		MUX18		MUX17		MUX16	
R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 11-110. OUTPUT8MUX16TO31CFG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	MUX31	R/W	0h	Select Bits for OUTPUT8 Mux31: 00 : Select .0 input for Mux31 01 : Select .1 input for Mux31 10 : Select .2 input for Mux31 11 : Select .3 input for Mux31 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29-28	MUX30	R/W	0h	Select Bits for OUTPUT8 Mux30: 00 : Select .0 input for Mux30 01 : Select .1 input for Mux30 10 : Select .2 input for Mux30 11 : Select .3 input for Mux30 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27-26	MUX29	R/W	0h	Select Bits for OUTPUT8 Mux29: 00 : Select .0 input for Mux29 01 : Select .1 input for Mux29 10 : Select .2 input for Mux29 11 : Select .3 input for Mux29 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25-24	MUX28	R/W	0h	Select Bits for OUTPUT8 Mux28: 00 : Select .0 input for Mux28 01 : Select .1 input for Mux28 10 : Select .2 input for Mux28 11 : Select .3 input for Mux28 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23-22	MUX27	R/W	0h	Select Bits for OUTPUT8 Mux27: 00 : Select .0 input for Mux27 01 : Select .1 input for Mux27 10 : Select .2 input for Mux27 11 : Select .3 input for Mux27 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21-20	MUX26	R/W	0h	Select Bits for OUTPUT8 Mux26: 00 : Select .0 input for Mux26 01 : Select .1 input for Mux26 10 : Select .2 input for Mux26 11 : Select .3 input for Mux26 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-110. OUTPUT8MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
19-18	MUX25	R/W	0h	Select Bits for OUTPUT8 Mux25: 00 : Select .0 input for Mux25 01 : Select .1 input for Mux25 10 : Select .2 input for Mux25 11 : Select .3 input for Mux25 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17-16	MUX24	R/W	0h	Select Bits for OUTPUT8 Mux24: 00 : Select .0 input for Mux24 01 : Select .1 input for Mux24 10 : Select .2 input for Mux24 11 : Select .3 input for Mux24 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15-14	MUX23	R/W	0h	Select Bits for OUTPUT8 Mux23: 00 : Select .0 input for Mux23 01 : Select .1 input for Mux23 10 : Select .2 input for Mux23 11 : Select .3 input for Mux23 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13-12	MUX22	R/W	0h	Select Bits for OUTPUT8 Mux22: 00 : Select .0 input for Mux22 01 : Select .1 input for Mux22 10 : Select .2 input for Mux22 11 : Select .3 input for Mux22 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11-10	MUX21	R/W	0h	Select Bits for OUTPUT8 Mux21: 00 : Select .0 input for Mux21 01 : Select .1 input for Mux21 10 : Select .2 input for Mux21 11 : Select .3 input for Mux21 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9-8	MUX20	R/W	0h	Select Bits for OUTPUT8 Mux20: 00 : Select .0 input for Mux20 01 : Select .1 input for Mux20 10 : Select .2 input for Mux20 11 : Select .3 input for Mux20 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7-6	MUX19	R/W	0h	Select Bits for OUTPUT8 Mux19: 00 : Select .0 input for Mux19 01 : Select .1 input for Mux19 10 : Select .2 input for Mux19 11 : Select .3 input for Mux19 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5-4	MUX18	R/W	0h	Select Bits for OUTPUT8 Mux18: 00 : Select .0 input for Mux18 01 : Select .1 input for Mux18 10 : Select .2 input for Mux18 11 : Select .3 input for Mux18 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-110. OUTPUT8MUX16TO31CFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	MUX17	R/W	0h	Select Bits for OUTPUT8 Mux17: 00 : Select .0 input for Mux17 01 : Select .1 input for Mux17 10 : Select .2 input for Mux17 11 : Select .3 input for Mux17 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1-0	MUX16	R/W	0h	Select Bits for OUTPUT8 Mux16: 00 : Select .0 input for Mux16 01 : Select .1 input for Mux16 10 : Select .2 input for Mux16 11 : Select .3 input for Mux16 Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.17 OUTPUT1MUXENABLE Register (Offset = 20h) [Reset = 0h]

OUTPUT1MUXENABLE is shown in Figure 11-99 and described in Table 11-111.

Return to the Summary Table.

Output X-BAR Mux Enable for Output 1

Figure 11-99. OUTPUT1MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-111. OUTPUT1MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux31 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux31 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux30 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux30 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux29 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux29 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux28 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux28 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux27 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux27 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-111. OUTPUT1MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux26 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux26 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux25 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux25 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux24 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux24 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux23 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux23 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux22 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux22 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux21 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux21 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux20 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux20 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux19 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux19 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-111. OUTPUT1MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux18 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux18 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux17 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux17 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux16 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux16 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux15 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux15 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux14 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux14 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux13 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux13 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux12 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux12 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux11 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux11 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-111. OUTPUT1MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux10 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux10 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux9 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux9 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux8 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux8 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux7 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux7 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux6 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux6 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux5 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux5 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux4 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux4 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux3 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux3 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-111. OUTPUT1MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux2 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux2 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux1 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux1 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive OUTPUT1 of OUTPUT-XBAR 0: Respective output of Mux0 is disabled to drive the OUTPUT1 of OUTPUT-XBAR 1: Respective output of Mux0 is enabled to drive the OUTPUT1 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.18 OUTPUT2MUXENABLE Register (Offset = 22h) [Reset = 0h]

OUTPUT2MUXENABLE is shown in Figure 11-100 and described in Table 11-112.

Return to the Summary Table.

Output X-BAR Mux Enable for Output 2

Figure 11-100. OUTPUT2MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-112. OUTPUT2MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux31 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux31 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux30 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux30 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux29 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux29 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux28 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux28 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux27 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux27 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-112. OUTPUT2MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux26 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux26 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux25 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux25 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux24 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux24 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux23 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux23 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux22 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux22 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux21 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux21 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux20 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux20 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux19 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux19 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-112. OUTPUT2MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux18 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux18 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux17 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux17 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux16 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux16 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux15 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux15 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux14 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux14 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux13 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux13 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux12 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux12 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux11 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux11 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-112. OUTPUT2MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux10 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux10 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux9 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux9 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux8 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux8 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux7 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux7 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux6 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux6 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux5 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux5 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux4 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux4 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux3 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux3 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-112. OUTPUT2MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux2 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux2 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux1 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux1 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive OUTPUT2 of OUTPUT-XBAR 0: Respective output of Mux0 is disabled to drive the OUTPUT2 of OUTPUT-XBAR 1: Respective output of Mux0 is enabled to drive the OUTPUT2 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.19 OUTPUT3MUXENABLE Register (Offset = 24h) [Reset = 0h]

OUTPUT3MUXENABLE is shown in Figure 11-101 and described in Table 11-113.

Return to the Summary Table.

Output X-BAR Mux Enable for Output 3

Figure 11-101. OUTPUT3MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-113. OUTPUT3MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux31 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux31 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux30 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux30 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux29 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux29 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux28 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux28 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux27 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux27 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-113. OUTPUT3MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux26 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux26 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux25 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux25 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux24 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux24 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux23 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux23 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux22 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux22 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux21 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux21 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux20 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux20 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux19 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux19 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-113. OUTPUT3MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux18 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux18 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux17 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux17 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux16 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux16 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux15 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux15 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux14 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux14 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux13 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux13 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux12 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux12 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux11 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux11 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-113. OUTPUT3MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux10 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux10 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux9 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux9 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux8 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux8 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux7 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux7 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux6 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux6 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux5 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux5 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux4 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux4 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux3 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux3 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-113. OUTPUT3MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux2 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux2 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux1 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux1 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive OUTPUT3 of OUTPUT-XBAR 0: Respective output of Mux0 is disabled to drive the OUTPUT3 of OUTPUT-XBAR 1: Respective output of Mux0 is enabled to drive the OUTPUT3 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.20 OUTPUT4MUXENABLE Register (Offset = 26h) [Reset = 0h]

OUTPUT4MUXENABLE is shown in Figure 11-102 and described in Table 11-114.

Return to the Summary Table.

Output X-BAR Mux Enable for Output 4

Figure 11-102. OUTPUT4MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-114. OUTPUT4MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux31 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux31 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux30 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux30 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux29 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux29 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux28 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux28 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux27 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux27 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-114. OUTPUT4MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux26 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux26 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux25 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux25 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux24 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux24 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux23 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux23 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux22 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux22 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux21 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux21 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux20 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux20 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux19 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux19 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-114. OUTPUT4MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux18 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux18 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux17 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux17 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux16 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux16 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux15 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux15 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux14 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux14 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux13 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux13 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux12 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux12 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux11 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux11 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-114. OUTPUT4MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux10 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux10 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux9 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux9 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux8 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux8 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux7 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux7 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux6 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux6 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux5 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux5 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux4 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux4 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux3 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux3 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-114. OUTPUT4MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux2 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux2 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux1 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux1 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive OUTPUT4 of OUTPUT-XBAR 0: Respective output of Mux0 is disabled to drive the OUTPUT4 of OUTPUT-XBAR 1: Respective output of Mux0 is enabled to drive the OUTPUT4 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.21 OUTPUT5MUXENABLE Register (Offset = 28h) [Reset = 0h]

OUTPUT5MUXENABLE is shown in Figure 11-103 and described in Table 11-115.

Return to the Summary Table.

Output X-BAR Mux Enable for Output 5

Figure 11-103. OUTPUT5MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-115. OUTPUT5MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux31 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux31 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux30 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux30 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux29 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux29 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux28 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux28 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux27 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux27 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-115. OUTPUT5MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux26 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux26 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux25 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux25 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux24 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux24 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux23 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux23 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux22 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux22 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux21 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux21 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux20 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux20 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux19 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux19 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-115. OUTPUT5MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux18 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux18 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux17 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux17 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux16 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux16 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux15 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux15 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux14 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux14 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux13 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux13 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux12 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux12 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux11 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux11 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-115. OUTPUT5MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux10 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux10 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux9 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux9 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux8 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux8 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux7 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux7 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux6 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux6 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux5 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux5 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux4 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux4 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux3 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux3 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-115. OUTPUT5MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux2 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux2 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux1 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux1 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive OUTPUT5 of OUTPUT-XBAR 0: Respective output of Mux0 is disabled to drive the OUTPUT5 of OUTPUT-XBAR 1: Respective output of Mux0 is enabled to drive the OUTPUT5 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.22 OUTPUT6MUXENABLE Register (Offset = 2Ah) [Reset = 0h]

OUTPUT6MUXENABLE is shown in Figure 11-104 and described in Table 11-116.

Return to the Summary Table.

Output X-BAR Mux Enable for Output 6

Figure 11-104. OUTPUT6MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-116. OUTPUT6MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux31 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux31 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux30 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux30 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux29 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux29 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux28 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux28 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux27 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux27 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-116. OUTPUT6MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux26 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux26 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux25 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux25 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux24 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux24 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux23 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux23 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux22 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux22 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux21 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux21 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux20 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux20 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux19 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux19 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-116. OUTPUT6MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux18 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux18 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux17 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux17 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux16 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux16 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux15 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux15 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux14 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux14 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux13 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux13 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux12 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux12 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux11 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux11 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-116. OUTPUT6MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux10 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux10 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux9 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux9 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux8 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux8 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux7 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux7 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux6 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux6 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux5 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux5 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux4 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux4 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux3 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux3 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-116. OUTPUT6MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux2 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux2 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux1 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux1 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive OUTPUT6 of OUTPUT-XBAR 0: Respective output of Mux0 is disabled to drive the OUTPUT6 of OUTPUT-XBAR 1: Respective output of Mux0 is enabled to drive the OUTPUT6 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.23 OUTPUT7MUXENABLE Register (Offset = 2Ch) [Reset = 0h]

OUTPUT7MUXENABLE is shown in Figure 11-105 and described in Table 11-117.

Return to the Summary Table.

Output X-BAR Mux Enable for Output 7

Figure 11-105. OUTPUT7MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-117. OUTPUT7MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux31 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux31 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux30 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux30 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux29 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux29 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux28 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux28 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux27 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux27 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-117. OUTPUT7MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux26 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux26 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux25 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux25 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux24 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux24 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux23 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux23 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux22 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux22 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux21 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux21 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux20 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux20 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux19 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux19 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-117. OUTPUT7MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux18 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux18 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux17 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux17 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux16 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux16 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux15 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux15 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux14 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux14 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux13 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux13 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux12 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux12 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux11 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux11 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-117. OUTPUT7MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux10 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux10 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux9 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux9 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux8 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux8 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux7 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux7 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux6 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux6 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux5 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux5 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux4 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux4 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux3 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux3 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-117. OUTPUT7MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux2 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux2 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux1 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux1 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive OUTPUT7 of OUTPUT-XBAR 0: Respective output of Mux0 is disabled to drive the OUTPUT7 of OUTPUT-XBAR 1: Respective output of Mux0 is enabled to drive the OUTPUT7 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.24 OUTPUT8MUXENABLE Register (Offset = 2Eh) [Reset = 0h]

OUTPUT8MUXENABLE is shown in Figure 11-106 and described in Table 11-118.

Return to the Summary Table.

Output X-BAR Mux Enable for Output 8

Figure 11-106. OUTPUT8MUXENABLE Register

31	30	29	28	27	26	25	24
MUX31	MUX30	MUX29	MUX28	MUX27	MUX26	MUX25	MUX24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
MUX23	MUX22	MUX21	MUX20	MUX19	MUX18	MUX17	MUX16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
MUX15	MUX14	MUX13	MUX12	MUX11	MUX10	MUX9	MUX8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
MUX7	MUX6	MUX5	MUX4	MUX3	MUX2	MUX1	MUX0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-118. OUTPUT8MUXENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MUX31	R/W	0h	Selects the output of Mux31 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux31 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux31 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
30	MUX30	R/W	0h	Selects the output of Mux30 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux30 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux30 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
29	MUX29	R/W	0h	Selects the output of Mux29 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux29 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux29 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
28	MUX28	R/W	0h	Selects the output of Mux28 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux28 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux28 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
27	MUX27	R/W	0h	Selects the output of Mux27 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux27 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux27 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-118. OUTPUT8MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MUX26	R/W	0h	Selects the output of Mux26 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux26 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux26 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
25	MUX25	R/W	0h	Selects the output of Mux25 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux25 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux25 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
24	MUX24	R/W	0h	Selects the output of Mux24 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux24 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux24 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
23	MUX23	R/W	0h	Selects the output of Mux23 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux23 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux23 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
22	MUX22	R/W	0h	Selects the output of Mux22 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux22 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux22 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
21	MUX21	R/W	0h	Selects the output of Mux21 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux21 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux21 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
20	MUX20	R/W	0h	Selects the output of Mux20 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux20 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux20 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
19	MUX19	R/W	0h	Selects the output of Mux19 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux19 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux19 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-118. OUTPUT8MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18	MUX18	R/W	0h	Selects the output of Mux18 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux18 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux18 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
17	MUX17	R/W	0h	Selects the output of Mux17 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux17 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux17 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
16	MUX16	R/W	0h	Selects the output of Mux16 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux16 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux16 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
15	MUX15	R/W	0h	Selects the output of Mux15 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux15 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux15 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
14	MUX14	R/W	0h	Selects the output of Mux14 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux14 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux14 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
13	MUX13	R/W	0h	Selects the output of Mux13 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux13 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux13 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
12	MUX12	R/W	0h	Selects the output of Mux12 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux12 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux12 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
11	MUX11	R/W	0h	Selects the output of Mux11 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux11 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux11 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-118. OUTPUT8MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	MUX10	R/W	0h	Selects the output of Mux10 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux10 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux10 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
9	MUX9	R/W	0h	Selects the output of Mux9 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux9 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux9 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
8	MUX8	R/W	0h	Selects the output of Mux8 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux8 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux8 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
7	MUX7	R/W	0h	Selects the output of Mux7 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux7 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux7 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	MUX6	R/W	0h	Selects the output of Mux6 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux6 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux6 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	MUX5	R/W	0h	Selects the output of Mux5 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux5 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux5 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	MUX4	R/W	0h	Selects the output of Mux4 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux4 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux4 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	MUX3	R/W	0h	Selects the output of Mux3 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux3 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux3 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-118. OUTPUT8MUXENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	MUX2	R/W	0h	Selects the output of Mux2 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux2 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux2 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	MUX1	R/W	0h	Selects the output of Mux1 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux1 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux1 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	MUX0	R/W	0h	Selects the output of mux0 to drive OUTPUT8 of OUTPUT-XBAR 0: Respective output of Mux0 is disabled to drive the OUTPUT8 of OUTPUT-XBAR 1: Respective output of Mux0 is enabled to drive the OUTPUT8 of OUTPUT-XBAR Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.25 OUTPUTLATCH Register (Offset = 30h) [Reset = 0h]

OUTPUTLATCH is shown in Figure 11-107 and described in Table 11-119.

Return to the Summary Table.

Output X-BAR Output Latch

Figure 11-107. OUTPUTLATCH Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
OUTPUT8	OUTPUT7	OUTPUT6	OUTPUT5	OUTPUT4	OUTPUT3	OUTPUT2	OUTPUT1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 11-119. OUTPUTLATCH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7	OUTPUT8	R	0h	Records the OUTPUT8 of OUTPUT-XBAR. 0: Respective output has not been triggered 1: Respective output is triggered Refer to the Output X-BAR section of this chapter for more details. Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
6	OUTPUT7	R	0h	Records the OUTPUT7 of OUTPUT-XBAR. 0: Respective output has not been triggered 1: Respective output is triggered Refer to the Output X-BAR section of this chapter for more details. Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
5	OUTPUT6	R	0h	Records the OUTPUT6 of OUTPUT-XBAR. 0: Respective output has not been triggered 1: Respective output is triggered Refer to the Output X-BAR section of this chapter for more details. Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
4	OUTPUT5	R	0h	Records the OUTPUT5 of OUTPUT-XBAR. 0: Respective output has not been triggered 1: Respective output is triggered Refer to the Output X-BAR section of this chapter for more details. Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

Table 11-119. OUTPUTLATCH Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	OUTPUT4	R	0h	Records the OUTPUT4 of OUTPUT-XBAR. 0: Respective output has not been triggered 1: Respective output is triggered Refer to the Output X-BAR section of this chapter for more details. Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
2	OUTPUT3	R	0h	Records the OUTPUT3 of OUTPUT-XBAR. 0: Respective output has not been triggered 1: Respective output is triggered Refer to the Output X-BAR section of this chapter for more details. Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
1	OUTPUT2	R	0h	Records the OUTPUT2 of OUTPUT-XBAR. 0: Respective output has not been triggered 1: Respective output is triggered Refer to the Output X-BAR section of this chapter for more details. Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn
0	OUTPUT1	R	0h	Records the OUTPUT1 of OUTPUT-XBAR. 0: Respective output has not been triggered 1: Respective output is triggered Refer to the Output X-BAR section of this chapter for more details. Note: [1] setting of this bit has priority over clear by software Reset type: CPU1.SYSRSn

11.3.6.26 OUTPUTLATCHCLR Register (Offset = 32h) [Reset = 0h]

OUTPUTLATCHCLR is shown in Figure 11-108 and described in Table 11-120.

Return to the Summary Table.

Output X-BAR Output Latch Clear

Figure 11-108. OUTPUTLATCHCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
OUTPUT8	OUTPUT7	OUTPUT6	OUTPUT5	OUTPUT4	OUTPUT3	OUTPUT2	OUTPUT1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 11-120. OUTPUTLATCHCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7	OUTPUT8	R-0/W1S	0h	Clears the Output-Latch for OUTPUT8 of OUTPUT-XBAR Writing 1 clears the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	OUTPUT7	R-0/W1S	0h	Clears the Output-Latch for OUTPUT7 of OUTPUT-XBAR Writing 1 clears the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	OUTPUT6	R-0/W1S	0h	Clears the Output-Latch for OUTPUT6 of OUTPUT-XBAR Writing 1 clears the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	OUTPUT5	R-0/W1S	0h	Clears the Output-Latch for OUTPUT5 of OUTPUT-XBAR Writing 1 clears the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	OUTPUT4	R-0/W1S	0h	Clears the Output-Latch for OUTPUT4 of OUTPUT-XBAR Writing 1 clears the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-120. OUTPUTLATCHCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	OUTPUT3	R-0/W1S	0h	Clears the Output-Latch for OUTPUT3 of OUTPUT-XBAR Writing 1 clears the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	OUTPUT2	R-0/W1S	0h	Clears the Output-Latch for OUTPUT2 of OUTPUT-XBAR Writing 1 clears the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	OUTPUT1	R-0/W1S	0h	Clears the Output-Latch for OUTPUT1 of OUTPUT-XBAR Writing 1 clears the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.27 OUTPUTLATCHFRC Register (Offset = 34h) [Reset = 0h]

OUTPUTLATCHFRC is shown in Figure 11-109 and described in Table 11-121.

Return to the Summary Table.

Output X-BAR Output Latch Clear

Figure 11-109. OUTPUTLATCHFRC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
OUTPUT8	OUTPUT7	OUTPUT6	OUTPUT5	OUTPUT4	OUTPUT3	OUTPUT2	OUTPUT1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 11-121. OUTPUTLATCHFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7	OUTPUT8	R-0/W1S	0h	Sets the Output-Latch for OUTPUT8 of OUTPUT-XBAR Writing 1 sets the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	OUTPUT7	R-0/W1S	0h	Sets the Output-Latch for OUTPUT7 of OUTPUT-XBAR Writing 1 sets the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	OUTPUT6	R-0/W1S	0h	Sets the Output-Latch for OUTPUT6 of OUTPUT-XBAR Writing 1 sets the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	OUTPUT5	R-0/W1S	0h	Sets the Output-Latch for OUTPUT5 of OUTPUT-XBAR Writing 1 sets the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	OUTPUT4	R-0/W1S	0h	Sets the Output-Latch for OUTPUT4 of OUTPUT-XBAR Writing 1 sets the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-121. OUTPUTLATCHFRC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	OUTPUT3	R-0/W1S	0h	Sets the Output-Latch for OUTPUT3 of OUTPUT-XBAR Writing 1 sets the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
1	OUTPUT2	R-0/W1S	0h	Sets the Output-Latch for OUTPUT2 of OUTPUT-XBAR Writing 1 sets the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	OUTPUT1	R-0/W1S	0h	Sets the Output-Latch for OUTPUT1 of OUTPUT-XBAR Writing 1 sets the corresponding output latch bit in the OUTPUTLATCH register Write of 0 has no effect Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.28 OUTPUTLATCHENABLE Register (Offset = 36h) [Reset = 0h]

OUTPUTLATCHENABLE is shown in Figure 11-110 and described in Table 11-122.

Return to the Summary Table.

Output X-BAR Output Latch Enable

Figure 11-110. OUTPUTLATCHENABLE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
OUTPUT8	OUTPUT7	OUTPUT6	OUTPUT5	OUTPUT4	OUTPUT3	OUTPUT2	OUTPUT1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-122. OUTPUTLATCHENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7	OUTPUT8	R/W	0h	Selects the output latch to drive OUTPUT8 for OUTPUT-XBAR 0: Output Latch is not selected to driven the respective output 1: Output Latch is selected to drive the respective output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	OUTPUT7	R/W	0h	Selects the output latch to drive OUTPUT7 for OUTPUT-XBAR 0: Output Latch is not selected to driven the respective output 1: Output Latch is selected to drive the respective output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	OUTPUT6	R/W	0h	Selects the output latch to drive OUTPUT6 for OUTPUT-XBAR 0: Output Latch is not selected to driven the respective output 1: Output Latch is selected to drive the respective output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	OUTPUT5	R/W	0h	Selects the output latch to drive OUTPUT5 for OUTPUT-XBAR 0: Output Latch is not selected to driven the respective output 1: Output Latch is selected to drive the respective output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	OUTPUT4	R/W	0h	Selects the output latch to drive OUTPUT4 for OUTPUT-XBAR 0: Output Latch is not selected to driven the respective output 1: Output Latch is selected to drive the respective output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
2	OUTPUT3	R/W	0h	Selects the output latch to drive OUTPUT3 for OUTPUT-XBAR 0: Output Latch is not selected to driven the respective output 1: Output Latch is selected to drive the respective output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-122. OUTPUTLATCHENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	OUTPUT2	R/W	0h	Selects the output latch to drive OUTPUT2 for OUTPUT-XBAR 0: Output Latch is not selected to driven the respective output 1: Output Latch is selected to drive the respective output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	OUTPUT1	R/W	0h	Selects the output latch to drive OUTPUT1 for OUTPUT-XBAR 0: Output Latch is not selected to driven the respective output 1: Output Latch is selected to drive the respective output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.29 OUTPUTINV Register (Offset = 38h) [Reset = 0h]

OUTPUTINV is shown in Figure 11-111 and described in Table 11-123.

Return to the Summary Table.

Output X-BAR Output Inversion

Figure 11-111. OUTPUTINV Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
OUTPUT8	OUTPUT7	OUTPUT6	OUTPUT5	OUTPUT4	OUTPUT3	OUTPUT2	OUTPUT1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 11-123. OUTPUTINV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	RESERVED	R	0h	Reserved
7	OUTPUT8	R/W	0h	Selects polarity for OUTPUT8 of OUTPUT-XBAR 0: drives active high output 1: drives active-low output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
6	OUTPUT7	R/W	0h	Selects polarity for OUTPUT7 of OUTPUT-XBAR 0: drives active high output 1: drives active-low output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
5	OUTPUT6	R/W	0h	Selects polarity for OUTPUT6 of OUTPUT-XBAR 0: drives active high output 1: drives active-low output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
4	OUTPUT5	R/W	0h	Selects polarity for OUTPUT5 of OUTPUT-XBAR 0: drives active high output 1: drives active-low output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
3	OUTPUT4	R/W	0h	Selects polarity for OUTPUT4 of OUTPUT-XBAR 0: drives active high output 1: drives active-low output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
2	OUTPUT3	R/W	0h	Selects polarity for OUTPUT3 of OUTPUT-XBAR 0: drives active high output 1: drives active-low output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

Table 11-123. OUTPUTINV Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	OUTPUT2	R/W	0h	Selects polarity for OUTPUT2 of OUTPUT-XBAR 0: drives active high output 1: drives active-low output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn
0	OUTPUT1	R/W	0h	Selects polarity for OUTPUT1 of OUTPUT-XBAR 0: drives active high output 1: drives active-low output Refer to the Output X-BAR section of this chapter for more details. Reset type: CPU1.SYSRSn

11.3.6.30 OUTPUTLOCK Register (Offset = 3Eh) [Reset = 0h]

OUTPUTLOCK is shown in Figure 11-112 and described in Table 11-124.

Return to the Summary Table.

Output X-BAR Configuration Lock register

Figure 11-112. OUTPUTLOCK Register

31	30	29	28	27	26	25	24
KEY							
R-0/W1S-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W1S-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							LOCK
R-0h							R/WSonce-0h

Table 11-124. OUTPUTLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W1S	0h	Bit-0 of this register can be set only if KEY= 0x5a5a Reset type: CPU1.SYSRSn
15-1	RESERVED	R	0h	Reserved
0	LOCK	R/WSonce	0h	Locks the configuration for OUTPUT-XBAR. Once the configuration is locked, writes to the below registers for OUTPUT-XBAR is blocked. Registers Affected by the LOCK mechanism: OUTPUT-XBAROUTyMUX0TO15CFG OUTPUT-XBAROUTyMUX16TO31CFG OUTPUT-XBAROUTyMUXENABLE OUTPUT-XBAROUTLATENABLE OUTPUT-XBAROUTINV 0: Writes to the above registers are allowed 1: Writes to the above registers are blocked Note: [1] LOCK mechanism only apples to writes. Reads are never blocked. Reset type: CPU1.SYSRSn

11.3.7 Register to Driverlib Function Mapping

11.3.7.1 INPUTXBAR Registers to Driverlib Functions

Table 11-125. INPUTXBAR Registers to Driverlib Functions

File	Driverlib Function
INPUT1SELECT	
xbar.h	XBAR_setInputPin
INPUT2SELECT	
-	See INPUT1SELECT
INPUT3SELECT	
-	See INPUT1SELECT
INPUT4SELECT	
-	See INPUT1SELECT
INPUT5SELECT	
-	See INPUT1SELECT
INPUT6SELECT	
-	See INPUT1SELECT
INPUT7SELECT	
-	See INPUT1SELECT
INPUT8SELECT	
-	See INPUT1SELECT
INPUT9SELECT	
-	See INPUT1SELECT
INPUT10SELECT	
-	See INPUT1SELECT
INPUT11SELECT	
-	See INPUT1SELECT
INPUT12SELECT	
-	See INPUT1SELECT
INPUT13SELECT	
-	See INPUT1SELECT
INPUT14SELECT	
-	See INPUT1SELECT
INPUT15SELECT	
-	See INPUT1SELECT
INPUT16SELECT	
-	See INPUT1SELECT
INPUTSELECTLOCK	
xbar.h	XBAR_lockInput

11.3.7.2 XBAR Registers to Driverlib Functions

Table 11-126. XBAR Registers to Driverlib Functions

File	Driverlib Function
FLG1	
xbar.c	XBAR_getInputFlagStatus
FLG2	
xbar.c	XBAR_getInputFlagStatus

Table 11-126. XBAR Registers to Driverlib Functions (continued)

File	Driverlib Function
FLG3	
xbar.c	XBAR_getInputFlagStatus
FLG4	
xbar.c	XBAR_getInputFlagStatus
CLR1	
xbar.c	XBAR_clearInputFlag
CLR2	
xbar.c	XBAR_clearInputFlag
CLR3	
xbar.c	XBAR_clearInputFlag
CLR4	
xbar.c	XBAR_clearInputFlag

11.3.7.3 EPWMXBAR Registers to Driverlib Functions

Table 11-127. EPWMXBAR Registers to Driverlib Functions

File	Driverlib Function
TRIP4MUX0TO15CFG	
xbar.c	XBAR_setEPWMMuxConfig
TRIP4MUX16TO31CFG	
xbar.c	XBAR_setEPWMMuxConfig
TRIP5MUX0TO15CFG	
-	See TRIP4MUX0TO15CFG
TRIP5MUX16TO31CFG	
-	See TRIP4MUX0TO15CFG
TRIP7MUX0TO15CFG	
-	See TRIP4MUX0TO15CFG
TRIP7MUX16TO31CFG	
-	See TRIP4MUX0TO15CFG
TRIP8MUX0TO15CFG	
-	See TRIP4MUX0TO15CFG
TRIP8MUX16TO31CFG	
-	See TRIP4MUX0TO15CFG
TRIP9MUX0TO15CFG	
-	See TRIP4MUX0TO15CFG
TRIP9MUX16TO31CFG	
-	See TRIP4MUX0TO15CFG
TRIP10MUX0TO15CFG	
-	See TRIP4MUX0TO15CFG
TRIP10MUX16TO31CFG	
-	See TRIP4MUX0TO15CFG
TRIP11MUX0TO15CFG	
-	See TRIP4MUX0TO15CFG
TRIP11MUX16TO31CFG	
-	See TRIP4MUX0TO15CFG
TRIP12MUX0TO15CFG	

Table 11-127. EPWMXBAR Registers to Driverlib Functions (continued)

File	Driverlib Function
-	See TRIP4MUX0TO15CFG
TRIP12MUX16TO31CFG	
-	See TRIP4MUX0TO15CFG
TRIP4MUXENABLE	
xbar.h	XBAR_enableEPWMMux
xbar.h	XBAR_disableEPWMMux
TRIP5MUXENABLE	
-	See TRIP4MUXENABLE
TRIP7MUXENABLE	
-	See TRIP4MUXENABLE
TRIP8MUXENABLE	
-	See TRIP4MUXENABLE
TRIP9MUXENABLE	
-	See TRIP4MUXENABLE
TRIP10MUXENABLE	
-	See TRIP4MUXENABLE
TRIP11MUXENABLE	
-	See TRIP4MUXENABLE
TRIP12MUXENABLE	
-	See TRIP4MUXENABLE
TRIPOUTINV	
xbar.h	XBAR_invertEPWMSignal
TRIPLOCK	
xbar.h	XBAR_lockEPWM

11.3.7.4 CLBXBAR Registers to Driverlib Functions

Table 11-128. CLBXBAR Registers to Driverlib Functions

File	Driverlib Function
AUXSIG0MUX0TO15CFG	
xbar.c	XBAR_setCLBMuxConfig
AUXSIG0MUX16TO31CFG	
xbar.c	XBAR_setCLBMuxConfig
AUXSIG1MUX0TO15CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG1MUX16TO31CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG2MUX0TO15CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG2MUX16TO31CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG3MUX0TO15CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG3MUX16TO31CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG4MUX0TO15CFG	

Table 11-128. CLBXBAR Registers to Driverlib Functions (continued)

File	Driverlib Function
-	See AUXSIG0MUX0TO15CFG
AUXSIG4MUX16TO31CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG5MUX0TO15CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG5MUX16TO31CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG6MUX0TO15CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG6MUX16TO31CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG7MUX0TO15CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG7MUX16TO31CFG	
-	See AUXSIG0MUX0TO15CFG
AUXSIG0MUXENABLE	
xbar.h	XBAR_enableCLBMux
xbar.h	XBAR_disableCLBMux
AUXSIG1MUXENABLE	
-	See AUXSIG0MUXENABLE
AUXSIG2MUXENABLE	
-	See AUXSIG0MUXENABLE
AUXSIG3MUXENABLE	
-	See AUXSIG0MUXENABLE
AUXSIG4MUXENABLE	
-	See AUXSIG0MUXENABLE
AUXSIG5MUXENABLE	
-	See AUXSIG0MUXENABLE
AUXSIG6MUXENABLE	
-	See AUXSIG0MUXENABLE
AUXSIG7MUXENABLE	
-	See AUXSIG0MUXENABLE
AUXSIGOUTINV	
xbar.h	XBAR_invertCLBSignal
AUXSIGLOCK	
-	

11.3.7.5 OUTPUTXBAR Registers to Driverlib Functions

Table 11-129. OUTPUTXBAR Registers to Driverlib Functions

File	Driverlib Function
OUTPUT1MUX0TO15CFG	
xbar.c	XBAR_setOutputMuxConfig
OUTPUT1MUX16TO31CFG	
-	
OUTPUT2MUX0TO15CFG	

Table 11-129. OUTPUTXBAR Registers to Driverlib Functions (continued)

File	Driverlib Function
-	See OUTPUT1MUX0TO15CFG
OUTPUT2MUX16TO31CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT3MUX0TO15CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT3MUX16TO31CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT4MUX0TO15CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT4MUX16TO31CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT5MUX0TO15CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT5MUX16TO31CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT6MUX0TO15CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT6MUX16TO31CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT7MUX0TO15CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT7MUX16TO31CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT8MUX0TO15CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT8MUX16TO31CFG	
-	See OUTPUT1MUX0TO15CFG
OUTPUT1MUXENABLE	
xbar.h	XBAR_enableOutputMux
xbar.h	XBAR_disableOutputMux
OUTPUT2MUXENABLE	
-	See OUTPUT1MUXENABLE
OUTPUT3MUXENABLE	
-	See OUTPUT1MUXENABLE
OUTPUT4MUXENABLE	
-	See OUTPUT1MUXENABLE
OUTPUT5MUXENABLE	
-	See OUTPUT1MUXENABLE
OUTPUT6MUXENABLE	
-	See OUTPUT1MUXENABLE
OUTPUT7MUXENABLE	
-	See OUTPUT1MUXENABLE
OUTPUT8MUXENABLE	
-	See OUTPUT1MUXENABLE
OUTPUTLATCH	

Table 11-129. OUTPUTXBAR Registers to Driverlib Functions (continued)

File	Driverlib Function
xbar.h	XBAR_setOutputLatchMode
xbar.h	XBAR_getOutputLatchStatus
xbar.h	XBAR_clearOutputLatch
xbar.h	XBAR_forceOutputLatch
OUTPUTLATCHCLR	
xbar.h	XBAR_clearOutputLatch
OUTPUTLATCHFRC	
xbar.h	XBAR_forceOutputLatch
OUTPUTLATCHENABLE	
xbar.h	XBAR_setOutputLatchMode
OUTPUTINV	
xbar.h	XBAR_invertOutputSignal
OUTPUTLOCK	
xbar.h	XBAR_lockOutput

This page intentionally left blank.

Chapter 12

Direct Memory Access (DMA)

The direct memory access (DMA) module provides a hardware method of transferring data between peripherals and/or memory without intervention from the CPU, thereby freeing up bandwidth for other system functions. Additionally, the DMA has the capability to orthogonally rearrange the data as it is transferred as well as “ping-pong” data between buffers. These features are useful for structuring data into blocks for optimal CPU processing.

12.1 Introduction

The strength of a controller is not measured purely in processor speed, but in total system capabilities. As a part of the equation, any time the CPU bandwidth for a given function can be reduced, the greater the system capabilities. Many times applications spend a significant amount of their bandwidth moving data, whether it is from off-chip memory to on-chip memory, or from a peripheral such as an analog-to-digital converter (ADC) to RAM, or even from one peripheral to another. Furthermore, many times this data comes in a format that is not conducive to the optimum processing powers of the CPU. The DMA module described in this reference guide has the ability to free up CPU bandwidth and rearrange the data into a pattern for more streamlined processing.

The DMA module is an event-based machine, meaning it requires a peripheral or software trigger to start a DMA transfer. Although it can be made into a periodic time-driven machine by configuring a timer as the DMA trigger source, there is no mechanism within the module itself to start memory transfers periodically. The DMA module has six independent DMA channels which can be configured separately and each channel contains its own independent PIE interrupt to let the CPU know when a DMA transfer has either started or completed. Five of the six channels are exactly the same, while Channel 1 has the ability to be configured at a higher priority than the others. At the heart of the DMA is a state machine and tightly coupled address control logic. It is this address control logic that allows for rearrangement of the block of data during the transfer as well as the process of ping-ponging data between buffers. Each of these features, along with others, will be discussed in detail in this chapter.

12.1.1 Features

DMA features include:

- Six channels with independent PIE interrupts
- Each DMA channel can be triggered from multiple peripheral trigger sources independently
- Word Size: 16-bit or 32-bit (SPI limited to 16-bit)
- Throughput: 3 cycles/word without arbitration

12.1.2 Block Diagram

Figure 12-1 shows a device-level block diagram of the DMA.

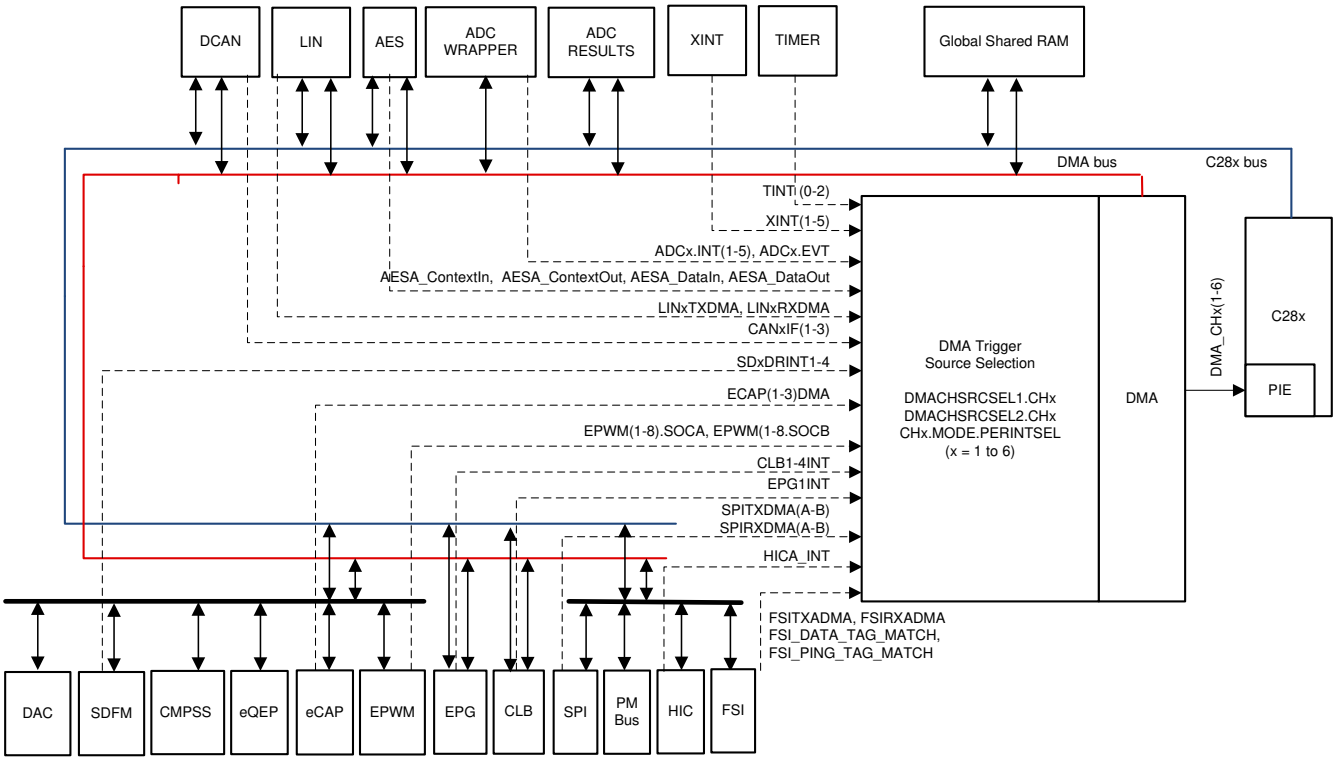


Figure 12-1. DMA Block Diagram

12.2 Architecture

12.2.1 Peripheral Interrupt Event Trigger Sources

Each DMA Channel can be configured to trigger by software and other peripheral triggers events. DMACHSRCSELx register can be used to configure DMA Trigger sources for each DMA channel. CHx.MODE.PERINTSEL register bitfield should be set to channel number (CHx.MODE.PERINTSEL = x) as shown in Figure 12-2. Included in these DMA Trigger sources are five external interrupt signals which can be connected to most of the general-purpose input/output (GPIO) pins on the device. This adds significant flexibility to the event trigger capabilities. Upon receipt of a peripheral interrupt event signal, the DMA will automatically send a clear signal to the interrupt source so that subsequent interrupt events will occur.

Note

To use the system level DMA Trigger source selection, the DMA internal trigger source selection configuration for each channel should be done using the DMACHSRCSELx register, and the CHx.MODE.PERINTSEL register as shown here. See Table 12-1 or the DMACHSRCSELx register definition for a complete list of DMA trigger sources.

Regardless of the value of the MODE.CHx[PERINTSEL] bit field, software can always force a trigger by using the CONTROL.CHx[PERINTFRC] bit. Likewise, software can always clear a pending DMA trigger using the CONTROL.CHx[PERINTCLR] bit.

Once a particular peripheral trigger event sets a channel's PERINTFLG bit, the bit remains pending until the priority logic of the state machine starts the burst transfer for that channel. Once the burst transfer starts, the flag is cleared. If a new peripheral trigger event is generated while a burst is in progress, the burst will complete before responding to the new peripheral trigger event (after proper prioritization). If a third peripheral trigger event occurs before the pending event is serviced, an error flag is set in the CONTROL.CHx[OVRFLG] bit. If a peripheral trigger event occurs at the same time as the latched flag is being cleared, the trigger event has priority and the PERINTFLG will remain set.

Figure 12-3 shows a diagram of the trigger select circuit.

Table 12-1 shows the peripheral trigger source options that are available for each channel.

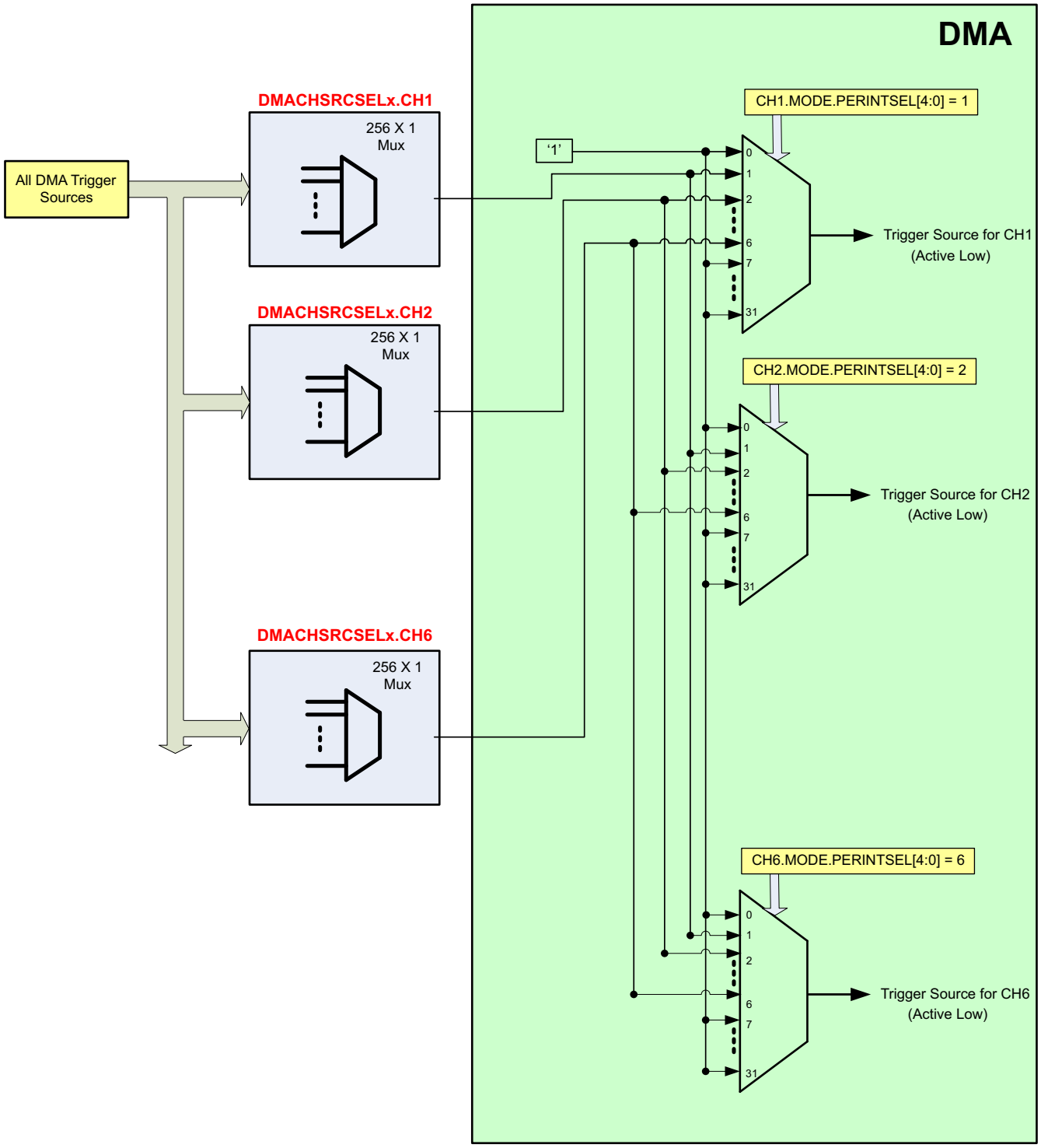
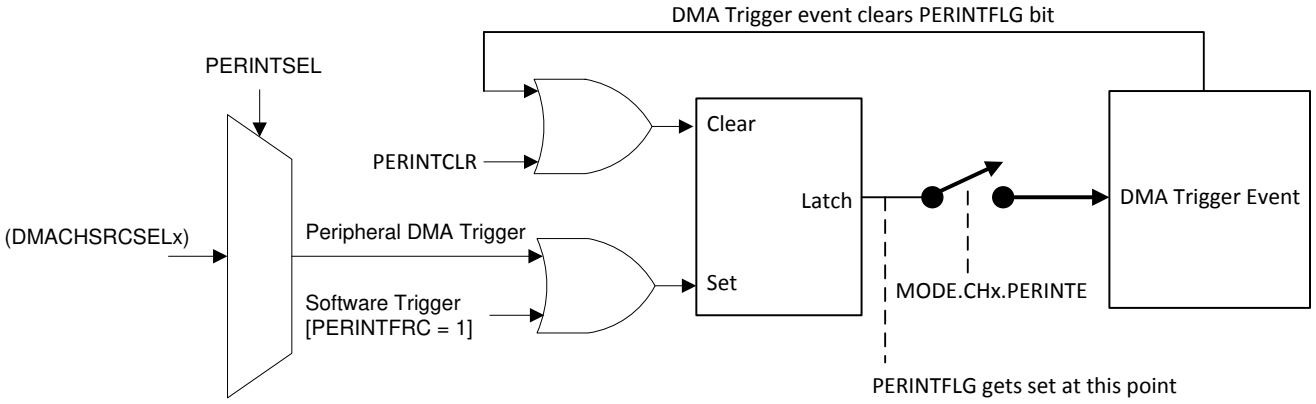


Figure 12-2. DMA Trigger Architecture



Note: See Figure 12-2.

Figure 12-3. Peripheral Interrupt Trigger Input Diagram

Table 12-1. DMA Trigger Source Options

Select Value (8-bit)	DMA ChTrigger Source
0	No Peripheral
1	ADCA.1
2	ADCA.2
3	ADCA.3
4	ADCA.4
5	ADCAEVT
6	ADCB.1
7	ADCB.2
8	ADCB.3
9	ADCB.4
10	ADCBEVT
11	ADCC.1
12	ADCC.2
13	ADCC.3
14	ADCC.4
15	ADCCEVT
16 - 28	No Peripheral
29	XINT1
30	XINT2
31	XINT3
32	XINT4
33	XINT5
34 - 35	No Peripheral
36	EPWM1.SOCA
37	EPWM1.SOCB
38	EPWM2.SOCA
39	EPWM2.SOCB
40	EPWM3.SOCA
41	EPWM3.SOCB

Table 12-1. DMA Trigger Source Options (continued)

Select Value (8-bit)	DMA ChTrigger Source
42	EPWM4.SOCA
43	EPWM4.SOCB
44	EPWM5.SOCA
45	EPWM5.SOCB
46	EPWM6.SOCA
47	EPWM6.SOCB
48	EPWM7.SOCA
49	EPWM7.SOCB
50	EPWM8.SOCA
51	EPWM8.SOCB
52-67	No Peripheral
68	TINT0
69	TINT1
70	TINT2
71 - 74	No Peripheral
75	ECAP1DMA
76	ECAP2DMA
77	ECAP3DMA
78 - 95	No Peripheral
96	SD1DRINT1
97	SD1DRINT2
98	SD1DRINT3
99	SD1DRINT4
100	No Peripheral
101	SD2DRINT1
102	SD2DRINT2
103	SD2DRINT3
104	SD2DRINT4
105-108	No Peripheral
109	SPITXDMAA
110	SPIRXDMAA
111	SPITXDMAB
112	SPIRXDMAB
113 - 116	No Peripheral
117	LINATXDMA
118	LINARXDMA
119	LINBTXDMA
120	LINBRXDMA
121 - 122	No Peripheral
123	FSITXADMA
124	FSI_DATA_TAG_MATCH
125	FSIRXADMA
126	FSI_PING_TAG_MATCH
127	CLB1INT
128	CLB2INT

Table 12-1. DMA Trigger Source Options (continued)

Select Value (8-bit)	DMA ChTrigger Source
129	CLB3INT
130	CLB4INT
131 - 166	No Peripheral
167	CANAIF1
168	CANAIF2
169	CANAIF3
170 - 178	No Peripheral
179	HICA_INT
180	AESA_ContextIn
181	AESA_DataIn
182	AESA_ContextOut
183	AESA_DataOut
184	EPG1INT
185-255	No peripheral

12.2.2 DMA Bus

The DMA bus architecture consists of a 32-bit address bus, a 32-bit data read bus, and a 32-bit data write bus. Memories and register locations connected to the DMA bus are via interfaces that sometimes share resources with the CPU memory or peripheral bus.

12.3 Address Pointer and Transfer Control

The DMA state machine is, at its most basic level, two nested loops.

Burst (Inner) Loop:

The burst (inner) loop transfers programmable number of words set by (BURST_SIZE + 1) register when a DMA channel trigger (Peripheral / Software trigger) is received. The BURST_SIZE register allows a maximum of 32 sixteen-bit words to be transferred in one burst. Each DMA channel supports both 16-bit (or) 32-bit word burst which can be controlled by MODE.DATASIZE bit field. Each DMA channel contains a shadowed address pointer for the source (SRC_ADDR_SHADOW) and the destination (DST_ADDR_SHADOW) address. At the beginning of each transfer, the shadowed version of each pointer is copied into its respective active (SRC_ADDR_ACTIVE / DST_ADDR_ACTIVE) register. During the burst loop, after each word is transferred, the signed value contained in the appropriate source or destination BURST_STEP register is added to the active register:

SRC_ADDR_ACTIVE = SRC_ADDR_ACTIVE + SRC_BURST_STEP

DST_ADDR_ACTIVE = DST_ADDR_ACTIVE + DST_BURST_STEP

The burst (inner) loop transfers a burst of data when a DMA Channel Trigger (Peripheral / Software trigger) is received.

Transfer (Outer) Loop:

The Transfer (outer) loop transfers programmable number of bursts set by (TRANSFER_SIZE + 1) register for each channel. Since TRANSFER_SIZE is a 16-bit register, the total size of a transfer allowed is well beyond any practical requirement. During the transfer loop, after each burst is complete, there are two methods that can be used to modify the active address pointer.

Method 1 (Default): When address wrapping is disabled (SRC_WRAP_SIZE / DST_WRAP_SIZE is greater than TRANSFER_SIZE), active address pointer is updated as shown below

SRC_ADDR_ACTIVE = SRC_ADDR_ACTIVE + SRC_TRANSFER_STEP

DST_ADDR_ACTIVE = DST_ADDR_ACTIVE + DST_TRANSFER_STEP

Method 2: Address wrapping gets enabled when SRC_WRAP_SIZE / DST_WRAP_SIZE is less than TRANSFER_SIZE. This allows the channel to wrap multiple times within a single transfer. When the number of bursts is equal to (SRC/DST_WRAP_SIZE + 1) register, the state machine modifies the active address pointers as:

SRC_BEG_ADDR_ACTIVE = SRC_BEG_ADDR_ACTIVE + SRC_WRAP_STEP

DST_BEG_ADDR_ACTIVE = DST_BEG_ADDR_ACTIVE + DST_WRAP_STEP

SRC_ADDR_ACTIVE = SRC_BEG_ADDR_ACTIVE

DST_ADDR_ACTIVE = DST_BEG_ADDR_ACTIVE

At the end of DMA transfer, DMA would have transferred (BURST_SIZE + 1) x (TRANSFER_SIZE + 1) words.

OneShot Mode:

OneShot mode is disabled by default.

When OneShot mode is disabled (MODE.CHx[ONESHOT] = 0), DMA transfers one burst [(BURST_SIZE + 1) words] of data each time a DMA Channel Trigger is received. After the burst is completed, the state machine moves on to the next pending channel in the priority scheme, even if another trigger for the channel just completed is pending. This feature keeps any single channel from monopolizing the DMA bus.

When OneShot mode is enabled (MODE.CHx[ONESHOT] = 1), DMA transfers all the bursts [(BURST_SIZE + 1) x (TRANSFER_SIZE + 1) words] on a single DMA channel trigger. Be careful when using this mode, since this can create a condition where one trigger uses up the majority of the DMA bandwidth.

Continuous Mode:

Continuous mode is disabled by default.

When Continuous mode is disabled (MODE.CHx[CONTINUOUS] = 0), DMA state machine disables channel after all bursts in a transfer loop (TRANSFER_COUNT = 0) are complete. The channel must be re-enabled by setting the RUN bit in the CONTROL register before another transfer can be started on that channel.

When Continuous mode is enabled (MODE.CHx[CONTINUOUS] = 1), DMA state machine keep channel active even after all bursts in a transfer loop (TRANSFER_COUNT = 0) are complete.

Each DMA channel can trigger its own EPIE interrupt for each DMA transfer either at start of DMA transfer (or) end of DMA transfer using MODE.CHx[CHINTMODE] bit.

Source/Destination Address Pointers (SRC/DST_ADDR)

The value written into the shadow register is the start address of the first location where data is read or written to.

At the beginning of a transfer the shadow register (SRC/DST_ADDR_SHADOW) is copied into the active register (SRC/DST_ADDR_ACTIVE). The active register performs as the current address pointer.

Source/Destination Begin Address Pointers (SRC/DST_BEG_ADDR)

This is the wrap pointer.

The value written into the shadow register (SRC/DST_BEG_ADDR_SHADOW) will be loaded into the active register (SRC/DST_BEG_ADDR_ACTIVE) at the start of a transfer. On a wrap condition, the active register (SRC/DST_BEG_ADDR_ACTIVE) will be incremented by the signed value in the appropriate SRC/DST_WRAP_STEP register prior to being loaded into the active register ((SRC/DST_ADDR_ACTIVE).

For each channel, the transfer process can be controlled with the following size values:

Source and Destination Burst Size (BURST_SIZE):

This specifies the number of words to be transferred in a burst.

This value is loaded into the BURST_COUNT register at the beginning of each burst. The BURST_COUNT decrements each word that is transferred and when it reaches a zero value, the burst is complete, indicating that the next channel can be serviced. The behavior of the current channel is defined by the ONE_SHOT bit in the MODE register. The maximum size of the burst is dictated by the type of peripheral. For the ADC, the burst size could be all 16 registers (if all 16 registers are used). For RAM the burst size can be up to the maximum allowed by the BURST_SIZE register, which is 32. See Table 12-2 to understand how BURST_SIZE register affects the number of 16-bit words transferred with respect to DATASIZE.

Table 12-2. BURSTSIZE vs DATASIZE Behavior

BURSTSIZE	Number of 16-bit words transferred in	
	DataSize = 16-bit data	DataSize = 32-bit data
0	1	2
1	2	2
2	3	4
3	4	4
4	5	6
5	6	6
6	7	8
7	8	8
8	9	10
9	10	10

Table 12-2. BURSTSIZE vs DATASIZE Behavior (continued)

BURSTSIZE	Number of 16-bit words transferred in	
	DataSize = 16-bit data	DataSize = 32-bit data
10	11	12
11	12	12
*	*	*
*	*	*
*	*	*
30	31	32
31	32	32

Source and Destination Transfer Size (TRANSFER_SIZE):

This specifies the number of bursts to be transferred per CPU interrupt (if enabled).

Whether this interrupt is generated at the beginning or the end of the transfer is defined in the CHINTMODE bit in the MODE register. Whether the channel remains enabled or not after the transfer is completed is defined by the CONTINUOUS bit in the MODE register. The TRANSFER_SIZE register is loaded into the TRANSFER_COUNT register at the beginning of each transfer. The TRANSFER_COUNT register keeps track of how many bursts of data the channel has transferred and when it reaches zero, the DMA transfer is complete.

Source/Destination Wrap Size (SRC/DST_WRAP_SIZE)

This specifies the number of bursts to be transferred before the current address pointer wraps around to the beginning.

This feature is used to implement a circular addressing type function. This value is loaded into the appropriate SRC/DST_WRAP_COUNT register at the beginning of each transfer. The SRC/DST_WRAP_COUNT registers keep track of how many bursts of data the channel has transferred and when they reach zero, the wrap procedure is performed on the appropriate source or destination address pointer. A separate size and count register is allocated for source and destination pointers. To *disable* the wrap function, assign the value of these registers to be larger than the TRANSFER_SIZE.

Note

The value written to the SIZE registers is one less than the intended size. So, to transfer three 16-bit words, the value 2 should be placed in the SIZE register.

Regardless of the state of the DATASIZE bit, the value specified in the SIZE registers are for 16-bit addresses. So, to transfer three 32-bit words, the value 5 should be placed in the SIZE register.

For each source/destination pointer, the address changes can be controlled with the following step values:

Source/Destination Burst Step (SRC/DST_BURST_STEP)

Within each burst transfer, the address source and destination step sizes are specified by these registers.

This value is a signed 2's compliment number so that the address pointer can be incremented or decremented as required. If no increment is desired, such as when accessing the data receive or transmit registers in a communication peripheral, the value of these registers should be set to zero.

Source/Destination Transfer Step (SRC/DST_TRANSFER_STEP)

This specifies the address offset to start the next burst transfer after completing the current burst transfer.

This is used in cases where registers or data memory locations are spaced at constant intervals. This value is a signed 2's compliment number so that the address pointer can be incremented or decremented as required.

Source/Destination Wrap Step (SRC/DST_WRAP_STEP):

When the wrap counter reaches zero, this value specifies the number of words to add/subtract from the SRC/DST_BEG_ADDR pointer and hence sets the new start address.

This implements a circular type of addressing mode, useful in many applications. This value is a signed 2's compliment number so that the address pointer can be incremented or decremented as required.

Note

Regardless of the state of the DATASIZE bit, the value specified in the STEP registers are for 16-bit addresses. So, to increment one 32-bit address, a value of 2 should be placed in these registers.

Channel Interrupt Mode (CHINTMODE)

This mode bit selects whether the DMA interrupt from the respective channel is generated at the beginning of a new transfer or at the end of the transfer.

If implementing a ping-pong buffer scheme with continuous mode of operation, then the interrupt would be generated at the beginning, just after the working registers are copied to the shadow set. If the DMA does not operate in continuous mode, then the interrupt is typically generated at the end when the transfer is complete.

All of the above features and modes are shown in Figure 12-4. The following items are in reference to Figure 12-4.

- The *HALT* points represent where the channel halts operation when interrupted by a high priority channel 1 trigger, or when the HALT command is set, or when an emulation halt is issued and the FREE bit is cleared to 0.
- The SRC/DST_ADDR_ACTIVE registers are not affected by SRC/DST_BEG_ADDR_ACTIVE at the start of a transfer. SRC/DST_BEG_ADDR_ACTIVE only affects the SRC/DST_ADDR_ACTIVE registers on a wrap. Following is what happens when a transfer first starts:
  - SRC/DST_BEG_ADDR_SHADOW remains unchanged.
  - SRC/DST_ADDR_SHADOW remains unchanged.
  - SRC/DST_BEG_ADDR_ACTIVE = SRC/DST_BEG_ADDR_SHADOW
  - SRC/DST_ADDR_ACTIVE = SRC/DST_ADDR_SHADOW
- The active registers get updated when a wrap occurs. The shadow registers remain unchanged. Specifically:
  - SRC/DST_BEG_ADDR_SHADOW remains unchanged.
  - SRC/DST_ADDR_SHADOW remains unchanged.
  - SRC/DST_BEG_ADDR_ACTIVE += SRC/DST_WRAP_STEP
  - SRC/DST_ADDR_ACTIVE = SRC/DST_BEG_ADDR_ACTIVE
- The best way to remember this is:
  - The shadow registers never change except by software.
  - The active registers never change except by hardware, and a shadow register is only copied into its own active register, never an active register by another name.

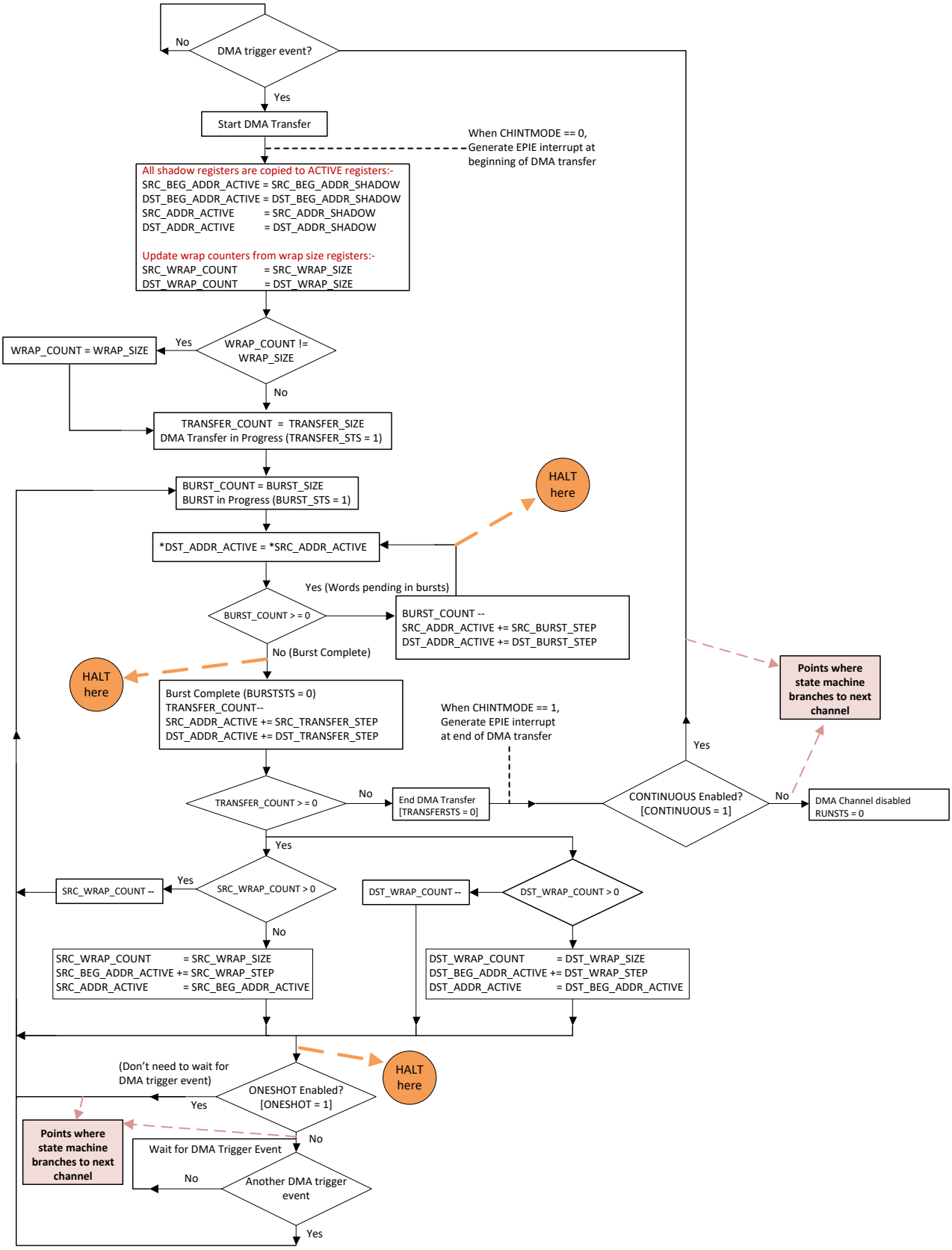


Figure 12-4. DMA State Diagram

12.4 Pipeline Timing and Throughput

In addition to the pipeline there are a few other behaviors of the DMA that affect its total throughput:

- A 1-cycle delay is added at the beginning of each burst
- A 1-cycle delay is added when returning from a CH1 high priority interrupt
- Collisions with the CPU may add delay slots
- 32-bit transfers run at double the speed of a 16-bit transfer (it takes the same amount of time to transfer a 32-bit word as it does a 16-bit word)

For example, to transfer 128 16-bit words from GS0 RAM to GS3 RAM, a channel can be configured to transfer 8 bursts of 16 words/burst. This will give:

8 bursts * [(3 cycles/word * 16 words/burst) + 1] = 392 cycles

If instead the channel were configured to transfer the same amount of data 32 bits at a time (the word size is configured to 32 bits) the transfer would take:

8 bursts * [(3 cycles/word * 8 words/burst) + 1] = 200 cycles

The DMA module consists of a 3-stage pipeline as shown in Figure 12-5 and Figure 12-6.

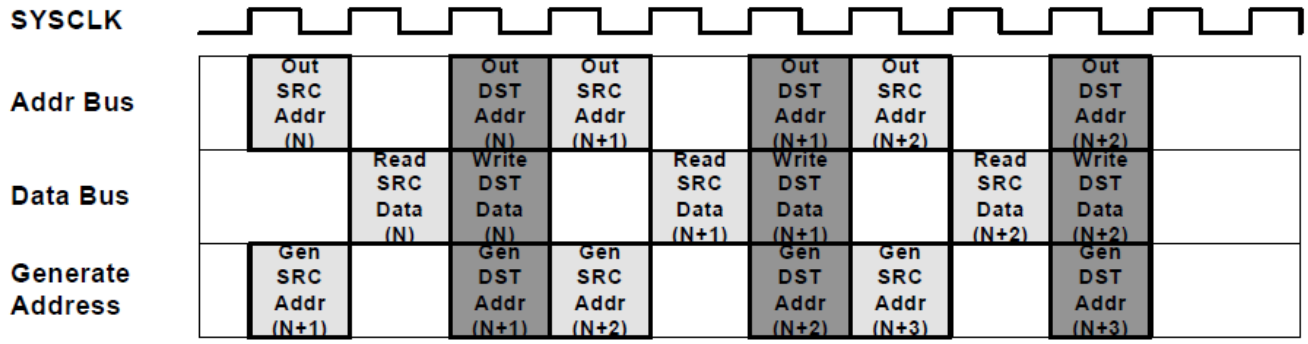


Figure 12-5. 3-Stage Pipeline DMA Transfer

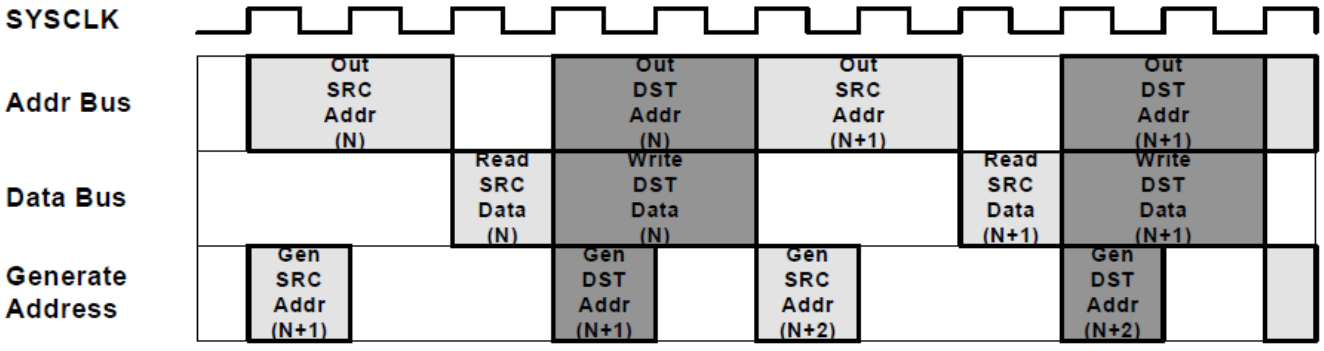


Figure 12-6. 3-stage Pipeline With One Read Stall

12.5 CPU and CLA Arbitration

Typically, DMA activity is independent of CPU and CLA activity. However, when the DMA and CPU (or CLA) try to access the same peripheral at the same time, an arbitration procedure is required to resolve the conflict. All instances of the same peripheral type conflict with each other. For instance, CAN-A and CAN-B conflict, as do the GS0 and GS2 RAMs. Different peripheral types can share a bus interface, which creates further opportunities for conflicts. These bus interfaces are:

- Peripheral frame 1: ePWM, eCAP, eQEP, SDFM, CMPSS, DAC
- Peripheral frame 2: PMBus and SPI

Conflict Example: The CLA is accessing DAC-A while the DMA is simultaneously accessing DAC-B.

Conflict Example: The CPU is accessing an SPI FIFO while the DMA is simultaneously accessing a PMBus register.

Non-conflict Example: The CPU is accessing a shared ePWM while the DMA is accessing an SPI.

The exception to all this is the ADC result registers, which are duplicated for each bus master. The CPU, DMA, and CLA can all simultaneously read these registers with no stalls for any master.

A DMA transfer consists of four phases: send source address, read source data, send destination address, and write destination data (see Section 12.4). Suppose CPU accesses a peripheral / memory causing conflict in middle of a DMA transfer, CPU is stalled till the current DMA access is complete and not until the completion of whole DMA transfer.

The following priority schemes are implemented for the various interfaces on the device.

- The arbitration follows a fixed arbitration scheme with highest priority first:
  - DMA WRITE
  - DMA READ
  - CLA WRITE
  - CLA READ
  - CPU WRITE
  - CPU READ
- The priority scheme for GSx RAM accesses is round-robin.
- All masters can access the ADC result registers simultaneously without delay.

Note

If the CPU is performing a read-modify-write operation and the DMA performs a write to the same location, the DMA write may be lost if the operation occurs in between the CPU read and the CPU write. Avoid mixing CPU writes with DMA writes to the same locations.

Arbitration within DMA channels is based on a round-robin priority (or) Channel 1 high priority scheme described in Section 12.6.

12.6 Channel Priority

Two priority schemes exist when determining channel priority: Round-robin mode and Channel 1 high-priority mode.

12.6.1 Round-Robin Mode

In this mode, all channels have *equal* priority and each enabled channel is serviced in round-robin fashion as follows:

CH1 → CH2 → CH3 → CH4 → CH5 → CH6 → CH1 → CH2 → …

In the case above, after each channel has transferred a burst of words, the next channel is serviced. You can specify the size of the burst for each channel. Once CH6 (or the last enabled channel) has been serviced, and no other channels are pending, the round-robin state machine enters an idle state.

From the idle state, channel 1 (if enabled) is always serviced first. However, if the DMA is currently processing another channel x, all other pending channels between x and the end of the round are serviced before CH1. It is in this sense that all the channels are of *equal* priority. For instance, take an example where CH1, CH4, and CH5 are enabled in round-robin mode and CH4 is currently being processed. Then CH1 and CH5 both receive an interrupt trigger from their respective peripherals before CH4 completes. CH1 and CH5 are now both pending. When CH4 completes its burst, CH5 will be serviced next. Only after CH5 completes will CH1 be serviced. Upon completion of CH1, if there are no more channels pending, the round-robin state machine will enter an idle state.

A more complicated example is shown below:

- Assume all channels are enabled, and the DMA is in an idle state,
- Initially a trigger occurs on CH1, CH3, and CH5 on the same cycle,
- When the CH1 burst transfer starts, requests from CH3 and CH5 are pending,
- Before completion of the CH1 burst, the DMA receives a request from CH2. Now the pending requests are from CH2, CH3, and CH5,
- After completing the CH1 burst, CH2 will be serviced since it is next in the round-robin scheme after CH1.
- After the burst from CH2 is finished, the CH3 burst will be serviced, followed by CH5 burst.
- Now while the CH5 burst is being serviced, the DMA receives a request from CH1, CH3, and CH6.
- The burst from CH6 will start after the completion of the CH5 burst since it is the next channel after CH5 in the round-robin scheme.
- This will be followed by the CH1 burst and then the CH3 burst
- After the CH3 burst finishes, assuming no more triggers have occurred, the round-robin state machine will enter an idle state.

The round-robin state machine may be reset to the idle state via the DMACTRL[PRIORITYRESET] bit.

12.6.2 Channel 1 High Priority Mode

In this mode, Channel 1 has high priority over all the other channels. Channel 2 – 6 have equal priority and each enabled channel is serviced in round-robin fashion.

Higher Priority:	CH1
Lower priority:	CH2 → CH3 → CH4 → CH5 → CH6 → CH2 → …

Given an example where CH1, CH4 and CH5 are enabled in Channel 1 High Priority Mode and CH4 is currently being processed. Then CH1 and CH5 both receive an interrupt trigger from their respective peripherals before CH4 completes. CH1 and CH5 are now both pending. When the current CH4 word transfer is completed, regardless of whether the DMA has completed the entire CH4 burst, CH4 execution will be suspended and CH1 will be serviced. After the CH1 burst completes, CH4 will resume execution.

Upon completion of CH4, CH5 will be serviced. After CH5 completes, if there are no more channels pending, the round-robin state machine will enter an idle state.

Typically Channel 1 would be used in this mode for the ADC, since its data rate is so high. However, Channel 1 High Priority Mode may be used in conjunction with any peripheral.

Note

High-priority mode and ONESHOT mode may not be used at the same time on channel 1. Other channels may use ONESHOT mode when channel 1 is in high-priority mode.

12.7 Overrun Detection Feature

The DMA contains overrun detection logic. When a peripheral event trigger is received by the DMA, the PERINTFLG bit in the CONTROL register is set, pending the channel to the DMA state machine. When the burst for that channel is started, the PERINTFLG is cleared. If however, between the time that the PERINTFLG bit is set by an event trigger and cleared by the start of the burst, an additional event trigger arrives, the second trigger will be lost. This condition will set the OVRFLG bit in the CONTROL register as in Figure 12-7. If the overrun interrupt is enabled, the channel interrupt will be generated to the PIE module.

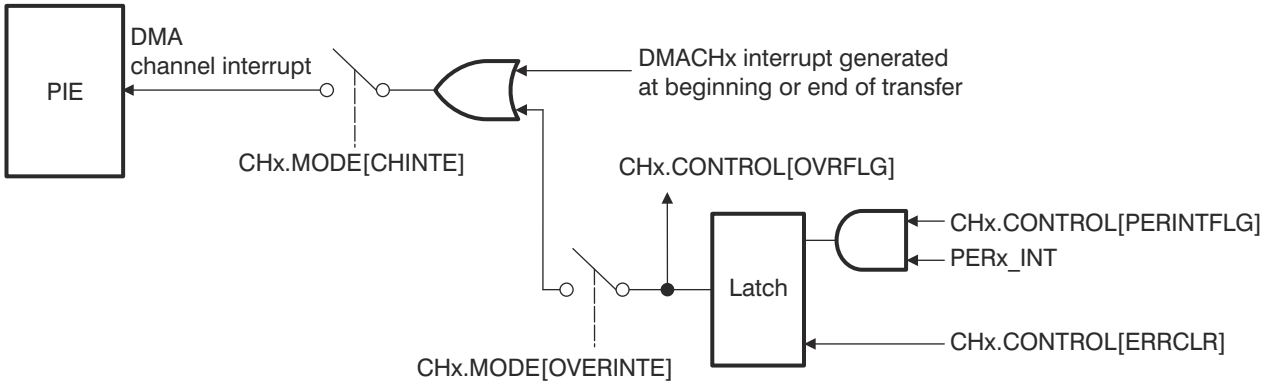


Figure 12-7. Overrun Detection Logic

12.8 Software

12.8.1 DMA Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/dma

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

12.8.1.1 DMA GSRAM Transfer (dma_ex1_gsram_transfer)

FILE: dma_ex1_gsram_transfer.c

This example uses one DMA channel to transfer data from a buffer in RAMGS0 to a buffer in RAMGS1. The example sets the DMA channel PERINTFRC bit repeatedly until the transfer of 16 bursts (where each burst is 8 16-bit words) has been completed. When the whole transfer is complete, it will trigger the DMA interrupt.

Watch Variables

- *sData* - Data to send
- *rData* - Received data

12.9 DMA Registers

This section describes the C28x Direct Memory Access Registers.

12.9.1 DMA Base Address Table

Table 12-3. DMA Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
DmaRegs	DMA_REGS	DMA_BASE	0x0000_1000	YES	-	-	-	-
Dmach1Regs	DMA_CH_REGS	DMA_CH1_BASE	0x0000_1020	YES	-	-	-	-
Dmach2Regs	DMA_CH_REGS	DMA_CH2_BASE	0x0000_1040	YES	-	-	-	-
Dmach3Regs	DMA_CH_REGS	DMA_CH3_BASE	0x0000_1060	YES	-	-	-	-
Dmach4Regs	DMA_CH_REGS	DMA_CH4_BASE	0x0000_1080	YES	-	-	-	-
Dmach5Regs	DMA_CH_REGS	DMA_CH5_BASE	0x0000_10A0	YES	-	-	-	-
Dmach6Regs	DMA_CH_REGS	DMA_CH6_BASE	0x0000_10C0	YES	-	-	-	-

12.9.2 DMA_REGS Registers

Table 12-4 lists the memory-mapped registers for the DMA_REGS registers. All register offset addresses not listed in Table 12-4 should be considered as reserved locations and the register contents should not be modified.

Table 12-4. DMA_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	DMACTRL	DMA Control Register	EALLOW	Go
1h	DEBUGCTRL	Debug Control Register	EALLOW	Go
4h	PRIORITYCTRL1	Priority Control 1 Register	EALLOW	Go
6h	PRIORITYSTAT	Priority Status Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 12-5 shows the codes that are used for access types in this section.

Table 12-5. DMA_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

12.9.2.1 DMACTRL Register (Offset = 0h) [Reset = 0h]

DMACTRL is shown in Figure 12-7 and described in Table 12-6.

Return to the Summary Table.

DMA Control Register

Figure 12-7. DMACTRL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						PRIORITYRESET	HARDRESET
R-0h						R-0/W1S-0h	R-0/W1S-0h

Table 12-6. DMACTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-2	RESERVED	R	0h	Reserved
1	PRIORITYRESET	R-0/W1S	0h	The priority reset bit resets the round-robin state machine when a 1 is written. Service starts from the first enabled channel. Writes of 0 are ignored and this bit always reads back a 0. When a 1 is written to this bit, any pending burst transfer completes before resetting the channel priority machine. If CH1 is configured as a high-priority channel, and this bit is written to while CH1 is servicing a burst, both the CH1 burst and the next pending low-priority burst are completed before the state machine is reset. If CH1 is high-priority, the state machine restarts from CH2 (or the next highest enabled channel). Reset type: SYSRSn
0	HARDRESET	R-0/W1S	0h	Writing a 1 to the hard reset bit resets the whole DMA and aborts any current access (similar to applying a device reset). Writes of 0 are ignored and this bit always reads back a 0. For a soft reset, a bit is provided for each channel to perform a gentler reset. Refer to the channel control registers. When writing to this bit, there is a one cycle delay before it takes effect. Hence, a one-cycle delay (such as a NOP instruction) is required in software before attempting to access any other DMA register. Reset type: SYSRSn

12.9.2.2 DEBUGCTRL Register (Offset = 1h) [Reset = 0h]

DEBUGCTRL is shown in Figure 12-8 and described in Table 12-7.

Return to the Summary Table.

Debug Control Register

Figure 12-8. DEBUGCTRL Register

15	14	13	12	11	10	9	8
FREE	RESERVED						
R/W-0h	R-0h						
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

Table 12-7. DEBUGCTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	FREE	R/W	0h	Emulation Control This bit specifies the action when an emulation halt event occurs. Reset type: SYSRSn 0h (R/W) = The DMA completes the current read-write operation, then halts. 1h (R/W) = The DMA continues running during an emulation halt.
14-0	RESERVED	R	0h	Reserved

12.9.2.3 PRIORITYCTRL1 Register (Offset = 4h) [Reset = 0h]

PRIORITYCTRL1 is shown in Figure 12-9 and described in Table 12-8.

Return to the Summary Table.

Priority Control 1 Register

Figure 12-9. PRIORITYCTRL1 Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							CH1PRIORITY
R-0h							R/W-0h

Table 12-8. PRIORITYCTRL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R	0h	Reserved
0	CH1PRIORITY	R/W	0h	DMA Channel 1 Priority This bit selects whether CH1 has high priority or not. The priority can only be changed when all channels are disabled. A priority reset should be performed before restarting channels after changing priority Reset type: SYSRSn 0h (R/W) = CH1 has the same priority as the other channels 1h (R/W) = CH1 has a higher priority than the other channels

12.9.2.4 PRIORITYSTAT Register (Offset = 6h) [Reset = 0h]

PRIORITYSTAT is shown in Figure 12-10 and described in Table 12-9.

Return to the Summary Table.

Priority Status Register

Figure 12-10. PRIORITYSTAT Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	ACTIVESTS_SHADOW			RESERVED	ACTIVESTS		
R-0h	R-0h			R-0h	R-0h		

Table 12-9. PRIORITYSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R	0h	Reserved
6-4	ACTIVESTS_SHADOW	R	0h	Active Channel Status Shadow These bits are only meaningful when CH1 is in high-priority mode. When CH1 is serviced, the ACTIVESTS bits are copied to the shadow bits and indicate which channel was interrupted by CH1. When CH1 service is completed, the shadow bits are copied back to the ACTIVESTS bits. If this bit field is zero or the same as the ACTIVESTS bit field, then no channel is pending due to a CH1 interrupt. When CH1 is not a higher priority channel, these bits should be ignored. Reset type: SYSRSn 0h (R/W) = No channel is active 1h (R/W) = CH 1 2h (R/W) = CH 2 3h (R/W) = CH 3 4h (R/W) = CH 4 5h (R/W) = CH 5 6h (R/W) = CH 6 7h (R/W) = Reserved
3	RESERVED	R	0h	Reserved
2-0	ACTIVESTS	R	0h	Active Channel Status These bits indicate which channel (if any) is currently active or performing a transfer. Reset type: SYSRSn 0h (R/W) = No channel is active 1h (R/W) = CH 1 2h (R/W) = CH 2 3h (R/W) = CH 3 4h (R/W) = CH 4 5h (R/W) = CH 5 6h (R/W) = CH 6 7h (R/W) = Reserved

12.9.3 DMA_CH_REGS Registers

Table 12-10 lists the memory-mapped registers for the DMA_CH_REGS registers. All register offset addresses not listed in Table 12-10 should be considered as reserved locations and the register contents should not be modified.

Table 12-10. DMA_CH_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	MODE	Mode Register	EALLOW	Go
1h	CONTROL	Control Register	EALLOW	Go
2h	BURST_SIZE	Burst Size Register	EALLOW	Go
3h	BURST_COUNT	Burst Count Register	EALLOW	Go
4h	SRC_BURST_STEP	Source Burst Step Register	EALLOW	Go
5h	DST_BURST_STEP	Destination Burst Step Register	EALLOW	Go
6h	TRANSFER_SIZE	Transfer Size Register	EALLOW	Go
7h	TRANSFER_COUNT	Transfer Count Register	EALLOW	Go
8h	SRC_TRANSFER_STEP	Source Transfer Step Register	EALLOW	Go
9h	DST_TRANSFER_STEP	Destination Transfer Step Register	EALLOW	Go
Ah	SRC_WRAP_SIZE	Source Wrap Size Register	EALLOW	Go
Bh	SRC_WRAP_COUNT	Source Wrap Count Register	EALLOW	Go
Ch	SRC_WRAP_STEP	Source Wrap Step Register	EALLOW	Go
Dh	DST_WRAP_SIZE	Destination Wrap Size Register	EALLOW	Go
Eh	DST_WRAP_COUNT	Destination Wrap Count Register	EALLOW	Go
Fh	DST_WRAP_STEP	Destination Wrap Step Register	EALLOW	Go
10h	SRC_BEG_ADDR_SHADOW	Source Begin Address Shadow Register	EALLOW	Go
12h	SRC_ADDR_SHADOW	Source Address Shadow Register	EALLOW	Go
14h	SRC_BEG_ADDR_ACTIVE	Source Begin Address Active Register	EALLOW	Go
16h	SRC_ADDR_ACTIVE	Source Address Active Register	EALLOW	Go
18h	DST_BEG_ADDR_SHADOW	Destination Begin Address Shadow Register	EALLOW	Go
1Ah	DST_ADDR_SHADOW	Destination Address Shadow Register	EALLOW	Go
1Ch	DST_BEG_ADDR_ACTIVE	Destination Begin Address Active Register	EALLOW	Go
1Eh	DST_ADDR_ACTIVE	Destination Address Active Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 12-11 shows the codes that are used for access types in this section.

Table 12-11. DMA_CH_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 12-11. DMA_CH_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

12.9.3.1 MODE Register (Offset = 0h) [Reset = 0h]

MODE is shown in Figure 12-11 and described in Table 12-12.

Return to the Summary Table.

Mode Register

Figure 12-11. MODE Register

15	14	13	12	11	10	9	8
CHINTE	DATASIZE	RESERVED	RESERVED	CONTINUOUS	ONESHOT	CHINTMODE	PERINTE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
OVRINTE	RESERVED		PERINTSEL				
R/W-0h	R-0h		R/W-0h				

Table 12-12. MODE Register Field Descriptions

Bit	Field	Type	Reset	Description
15	CHINTE	R/W	0h	Channel Interrupt Enable Bit This bit enables the DMA channel's CPU interrupt. Reset type: SYSRSn 0h (R/W) = Interrupt disabled 1h (R/W) = Interrupt enabled
14	DATASIZE	R/W	0h	Data Size Mode Bit This bit determines whether the DMA channel transfers 16 bits or 32 bits of data per read/write operation. Regardless of this setting, all data lengths and offsets in other DMA registers refer to 16- bit words. The pointer step increments must be configured to accomodate 32-bit words. Reset type: SYSRSn 0h (R/W) = 16-bit data transfer size 1h (R/W) = 32-bit data transfer size
13	RESERVED	R/W	0h	Reserved
12	RESERVED	R/W	0h	Reserved
11	CONTINUOUS	R/W	0h	Continuous Mode Bit If this bit is set to 1, then the channel re-initializes when TRANSFER_COUNT is zero and waits for the next event trigger. Otherwise, the DMA stops and clears the RUNSTS bit. Reset type: SYSRSn
10	ONESHOT	R/W	0h	One Shot Mode If this bit is set to 1, each peripheral event trigger causes the channel to perform an entire transfer. Otherwise, the channel only performs one burst per trigger. Reset type: SYSRSn
9	CHINTMODE	R/W	0h	Channel Interrupt Generation Mode This bit specifies when the DMA channel generates a CPU interrupt for a transfer. Reset type: SYSRSn 0h (R/W) = Generate interrupt at beginning of new transfer 1h (R/W) = Generate interrupt at end of transfer.
8	PERINTE	R/W	0h	Peripheral Event Trigger Enable This bit enables peripheral event triggers on the DMA channel. Reset type: SYSRSn 0h (R/W) = Peripheral event trigger disabled. Neither the selected peripheral nor software can start a DMA burst. 1h (R/W) = Peripheral event trigger enabled.

Table 12-12. MODE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	OVRINTE	R/W	0h	Overflow Interrupt Enable The bit determines whether the DMA module generates a CPU interrupt when it detects an overflow event. Reset type: SYSRSn 0h (R/W) = Overflow interrupt disabled 1h (R/W) = Overflow interrupt enabled
6-5	RESERVED	R	0h	Reserved
4-0	PERINTSEL	R/W	0h	Peripheral Event Trigger Source Select These are legacy bits and should be set to the channel number. The actual source selection is done via the DMACHSRCSELn registers, which are part of the DMA_CLA_SRC_SEL_REGS group. Reset type: SYSRSn

12.9.3.2 CONTROL Register (Offset = 1h) [Reset = 0h]

CONTROL is shown in Figure 12-12 and described in Table 12-13.

Return to the Summary Table.

Control Register

Figure 12-12. CONTROL Register

15	14	13	12	11	10	9	8
RESERVED	OVRFLG	RUNSTS	BURSTSTS	TRANSFERSTS	RESERVED	RESERVED	PERINTFLG
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
ERRCLR	RESERVED	RESERVED	PERINTCLR	PERINTFRC	SOFTRESET	HALT	RUN
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 12-13. CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	OVRFLG	R	0h	Overflow Flag This bit indicates that a peripheral event trigger was received while PERINTFLG was already set. It can be cleared by writing to the ERRCLR bit. Reset type: SYSRSn 0h (R/W) = No overflow detected 1h (R/W) = Overflow detected
13	RUNSTS	R	0h	Run Status Flag This bit indicates that the DMA channel is ready to respond to peripheral event triggers. This bit is set when a 1 is written to the RUN bit. It is cleared when a transfer completes (TRANSFER_COUNT = 0) and continuous mode is disabled, or when the HARDRESET, SOFTRESET, or HALT bit is set. Reset type: SYSRSn 0h (R/W) = The channel is disabled 1h (R/W) = The channel is enabled
12	BURSTSTS	R	0h	Burst Status Flag This bit is set when a DMA burst begins. The BURST_COUNT is set to the BURST_SIZE. This bit is cleared when BURST_COUNT reaches zero, or when the HARDRESET or SOFTRESET bit is set. Reset type: SYSRSn 0h (R/W) = No burst activity 1h (R/W) = The DMA is currently servicing or suspending a burst transfer from this channel
11	TRANSFERSTS	R	0h	Transfer Status Flag This bit is set when a DMA transfer begins. The address registers are copied to the shadow set and the TRANSFER_COUNT is set to the TRANSFER_SIZE. This bit is cleared when TRANSFER_COUNT reaches zero, or when the HARDRESET or SOFTRESET bit is set. Reset type: SYSRSn 0h (R/W) = No transfer activity 1h (R/W) = The channel is currently in the middle of a transfer regardless of whether a burst of data is actively being transferred or not
10	RESERVED	R	0h	Reserved
9	RESERVED	R	0h	Reserved

Table 12-13. CONTROL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	PERINTFLG	R	0h	Peripheral Event Trigger Flag This bit indicates whether a peripheral event trigger has arrived. This bit is automatically cleared when the first burst transfer begins. Reset type: SYSRSn 0h (R/W) = Waiting for event trigger 1h (R/W) = Event trigger pending
7	ERRCLR	R-0/W1S	0h	Clear Error Writing a 1 to this bit will clear the OVRFLG bit. This is normally done when initializing the DMA module or if an overflow condition is detected. If an overflow event occurs at the same time this bit is set, the overrun has priority and the OVRFLG bit is set. Reset type: SYSRSn
6	RESERVED	R-0/W1S	0h	Reserved
5	RESERVED	R-0/W1S	0h	Reserved
4	PERINTCLR	R-0/W1S	0h	Clear Peripheral Event Trigger Writing a 1 to this bit clears PERINTFLG, which cancels a pending event trigger. This is normally done when initializing the DMA module. If an event trigger arrives at the same time this bit is set, the trigger has priority and PERINTFLG is set. Reset type: SYSRSn
3	PERINTFRC	R-0/W1S	0h	Force Peripheral Event Trigger If the PERINTE bit of the MODE register is set, writing a 1 to this bit sets PERINTFLG, which triggers a DMA burst. This bit can be used to start a DMA transfer in software. Reset type: SYSRSn
2	SOFTRESET	R-0/W1S	0h	Channel Soft Reset Writing a 1 to this bit places the channel into its default state after the current read/write access has completed: RUNSTS = 0 TRANSFERSTS = 0 BURSTSTS = 0 BURST_COUNT = 0 TRANSFER_COUNT = 0 SRC_WRAP_COUNT = 0 DST_WRAP_COUNT = 0 When writing to this bit, there is a one cycle delay before it takes effect. Hence, a one-cycle delay (such as a NOP instruction) is required in software before attempting to access any other DMA register. Reset type: SYSRSn
1	HALT	R-0/W1S	0h	Halt Channel Writing a 1 to this bit halts the DMA channel in its current state after any ongoing read/write access has completed. Reset type: SYSRSn
0	RUN	R-0/W1S	0h	Run Channel Writing a 1 to this bit enables the DMA channel and sets the RUNSTS bit to 1. This bit is also used to resume after a channel halt. The RUN bit is typically used to start the DMA channel after configuration. The channel will then wait for the first peripheral event trigger (PERINTFLG == 1) to start a burst. Reset type: SYSRSn

12.9.3.3 BURST_SIZE Register (Offset = 2h) [Reset = 0h]

BURST_SIZE is shown in Figure 12-13 and described in Table 12-14.

Return to the Summary Table.

Burst Size Register

Figure 12-13. BURST_SIZE Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			BURSTSIZE				
R-0h			R/W-0h				

Table 12-14. BURST_SIZE Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R	0h	Reserved
4-0	BURSTSIZE	R/W	0h	These bits specify the burst size in 16-bit words. The actual size is equal to BURSTSIZE + 1. Reset type: SYSRSn

12.9.3.4 BURST_COUNT Register (Offset = 3h) [Reset = 0h]

BURST_COUNT is shown in Figure 12-14 and described in Table 12-15.

Return to the Summary Table.

Burst Count Register

Figure 12-14. BURST_COUNT Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			BURSTCOUNT				
R-0h			R-0h				

Table 12-15. BURST_COUNT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R	0h	Reserved
4-0	BURSTCOUNT	R	0h	These bits indicate the number of words left in the current burst. Reset type: SYSRSn 0h (R/W) = 0 word left in a burst 1h (R/W) = 1 word left in a burst 2h (R/W) = 2 word left in a burst 3h (R/W) = 3 word left in a burst 4h (R/W) = 4 word left in a burst 5h (R/W) = 5 word left in a burst 6h (R/W) = 6 word left in a burst 7h (R/W) = 7 word left in a burst 8h (R/W) = 8 word left in a burst 9h (R/W) = 9 word left in a burst Ah (R/W) = 10 word left in a burst Bh (R/W) = 11 word left in a burst Ch (R/W) = 12 word left in a burst Dh (R/W) = 13 word left in a burst Eh (R/W) = 14 word left in a burst Fh (R/W) = 15 word left in a burst 10h (R/W) = 16 word left in a burst 11h (R/W) = 17 word left in a burst 12h (R/W) = 18 word left in a burst 13h (R/W) = 19 word left in a burst 14h (R/W) = 20 word left in a burst 15h (R/W) = 21 word left in a burst 16h (R/W) = 22 word left in a burst 17h (R/W) = 23 word left in a burst 18h (R/W) = 24 word left in a burst 19h (R/W) = 25 word left in a burst 1Ah (R/W) = 26 word left in a burst 1Bh (R/W) = 27 word left in a burst 1Ch (R/W) = 28 word left in a burst 1Dh (R/W) = 29 word left in a burst 1Eh (R/W) = 30 word left in a burst 1Fh (R/W) = 31 word left in a burst

12.9.3.5 SRC_BURST_STEP Register (Offset = 4h) [Reset = 0h]

SRC_BURST_STEP is shown in Figure 12-15 and described in Table 12-16.

Return to the Summary Table.

Source Burst Step Register

Figure 12-15. SRC_BURST_STEP Register

15	14	13	12	11	10	9	8
SRCBURSTSTEP							
R/W-0h							
7	6	5	4	3	2	1	0
SRCBURSTSTEP							
R/W-0h							

Table 12-16. SRC_BURST_STEP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SRCBURSTSTEP	R/W	0h	These bits specify the change in the source address after each word in a burst. The size must be a 16-bit two's complement value between -4096 and 4095 (inclusive). This value is added to the source address after each read/write operation in a burst. Reset type: SYSRSn 0h (R/W) = No address change 1h (R/W) = Add 1 to the address 2h (R/W) = Add 2 to the address FFEh (R/W) = Add 4094 to the address FFFh (R/W) = Add 4095 to the address F000h (R/W) = Subtract 4096 from the address F001h (R/W) = Subtract 4095 from the address FFFEh (R/W) = Subtract 2 from the address FFFFh (R/W) = Subtract 1 from the address

12.9.3.6 DST_BURST_STEP Register (Offset = 5h) [Reset = 0h]

DST_BURST_STEP is shown in Figure 12-16 and described in Table 12-17.

Return to the Summary Table.

Destination Burst Step Register

Figure 12-16. DST_BURST_STEP Register

15	14	13	12	11	10	9	8
DSTBURSTSTEP							
R/W-0h							
7	6	5	4	3	2	1	0
DSTBURSTSTEP							
R/W-0h							

Table 12-17. DST_BURST_STEP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	DSTBURSTSTEP	R/W	0h	These bits specify the change in the destination address after each word in a burst. The size must be a 16-bit two's complement value between -4096 and 4095 (inclusive). This value is added to the destination address after each read/write operation in a burst. Reset type: SYSRSn 0h (R/W) = No address change 1h (R/W) = Add 1 to the address 2h (R/W) = Add 2 to the address FFEh (R/W) = Add 4094 to the address FFFh (R/W) = Add 4095 to the address F000h (R/W) = Subtract 4096 from the address F001h (R/W) = Subtract 4095 from the address FFFEh (R/W) = Subtract 2 from the address FFFFh (R/W) = Subtract 1 from the address

12.9.3.7 TRANSFER_SIZE Register (Offset = 6h) [Reset = 0h]

TRANSFER_SIZE is shown in Figure 12-17 and described in Table 12-18.

Return to the Summary Table.

Transfer Size Register

Figure 12-17. TRANSFER_SIZE Register

15	14	13	12	11	10	9	8
TRANSFERSIZE							
R/W-0h							
7	6	5	4	3	2	1	0
TRANSFERSIZE							
R/W-0h							

Table 12-18. TRANSFER_SIZE Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	TRANSFERSIZE	R/W	0h	These bits specify the transfer size in bursts. The actual size is equal to TRANSFERSIZE + 1. Reset type: SYSRSn

12.9.3.8 TRANSFER_COUNT Register (Offset = 7h) [Reset = 0h]

TRANSFER_COUNT is shown in Figure 12-18 and described in Table 12-19.

Return to the Summary Table.

Transfer Count Register

Figure 12-18. TRANSFER_COUNT Register

15	14	13	12	11	10	9	8
TRANSFERCOUNT							
R-0h							
7	6	5	4	3	2	1	0
TRANSFERCOUNT							
R-0h							

Table 12-19. TRANSFER_COUNT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	TRANSFERCOUNT	R	0h	These bits indicate the number of bursts left in the current transfer. Reset type: SYSRSn

12.9.3.9 SRC_TRANSFER_STEP Register (Offset = 8h) [Reset = 0h]

SRC_TRANSFER_STEP is shown in Figure 12-19 and described in Table 12-20.

Return to the Summary Table.

Source Transfer Step Register

Figure 12-19. SRC_TRANSFER_STEP Register

15	14	13	12	11	10	9	8
SRCTRANSFERSTEP							
R/W-0h							
7	6	5	4	3	2	1	0
SRCTRANSFERSTEP							
R/W-0h							

Table 12-20. SRC_TRANSFER_STEP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SRCTRANSFERSTEP	R/W	0h	These bits specify the change in the source address after a burst completes. The size must be a 16-bit two's complement value between -4096 and 4095 (inclusive). This value is added to the source address after each burst completes. Reset type: SYSRSn 0h (R/W) = No address change 1h (R/W) = Add 1 to the address 2h (R/W) = Add 2 to the address FFEh (R/W) = Add 4094 to the address FFFh (R/W) = Add 4095 to the address F000h (R/W) = Subtract 4096 from the address F001h (R/W) = Subtract 4095 from the address FFFEh (R/W) = Subtract 2 from the address FFFFh (R/W) = Subtract 1 from the address

12.9.3.10 DST_TRANSFER_STEP Register (Offset = 9h) [Reset = 0h]

DST_TRANSFER_STEP is shown in Figure 12-20 and described in Table 12-21.

Return to the Summary Table.

Destination Transfer Step Register

Figure 12-20. DST_TRANSFER_STEP Register

15	14	13	12	11	10	9	8
DSTTRANSFERSTEP							
R/W-0h							
7	6	5	4	3	2	1	0
DSTTRANSFERSTEP							
R/W-0h							

Table 12-21. DST_TRANSFER_STEP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	DSTTRANSFERSTEP	R/W	0h	These bits specify the change in the destination address after a burst completes. The size must be a 16-bit two's complement value between -4096 and 4095 (inclusive). This value is added to the destination address after each burst completes. Reset type: SYSRSn 0h (R/W) = No address change 1h (R/W) = Add 1 to the address 2h (R/W) = Add 2 to the address FFEh (R/W) = Add 4094 to the address FFFh (R/W) = Add 4095 to the address F000h (R/W) = Subtract 4096 from the address F001h (R/W) = Subtract 4095 from the address FFFEh (R/W) = Subtract 2 from the address FFFFh (R/W) = Subtract 1 from the address

12.9.3.11 SRC_WRAP_SIZE Register (Offset = Ah) [Reset = FFFFh]

SRC_WRAP_SIZE is shown in Figure 12-21 and described in Table 12-22.

Return to the Summary Table.

Source Wrap Size Register

Figure 12-21. SRC_WRAP_SIZE Register

15	14	13	12	11	10	9	8
WRAPSIZE							
R/W-FFFFh							
7	6	5	4	3	2	1	0
WRAPSIZE							
R/W-FFFFh							

Table 12-22. SRC_WRAP_SIZE Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	WRAPSIZE	R/W	FFFFh	These bits specify the number of bursts to transfer before the source address wraps around to the beginning address. The actual number is equal to WRAPSIZE + 1. To disable the wrapping function, set WRAPSIZE to a value larger than TRANSFERSIZE. Reset type: SYSRSn

12.9.3.12 SRC_WRAP_COUNT Register (Offset = Bh) [Reset = 0h]

SRC_WRAP_COUNT is shown in Figure 12-22 and described in Table 12-23.

Return to the Summary Table.

Source Wrap Count Register

Figure 12-22. SRC_WRAP_COUNT Register

15	14	13	12	11	10	9	8
WRAPSIZE							
R-0h							
7	6	5	4	3	2	1	0
WRAPSIZE							
R-0h							

Table 12-23. SRC_WRAP_COUNT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	WRAPSIZE	R	0h	These bits indicate the number of bursts left before wrapping the source address. Reset type: SYSRSn

12.9.3.13 SRC_WRAP_STEP Register (Offset = Ch) [Reset = 0h]

SRC_WRAP_STEP is shown in Figure 12-23 and described in Table 12-24.

Return to the Summary Table.

Source Wrap Step Register

Figure 12-23. SRC_WRAP_STEP Register

15	14	13	12	11	10	9	8
WRAPSTEP							
R/W-0h							
7	6	5	4	3	2	1	0
WRAPSTEP							
R/W-0h							

Table 12-24. SRC_WRAP_STEP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	WRAPSTEP	R/W	0h	These bits specify the change in the source beginning address when the wrap counter reaches zero. The size must be a 16-bit two's complement value between -4096 and 4095 (inclusive). This value is added to the source address when wrapping occurs. Reset type: SYSRSn 0h (R/W) = No address change 1h (R/W) = Add 1 to the address 2h (R/W) = Add 2 to the address FFEh (R/W) = Add 4094 to the address FFFh (R/W) = Add 4095 to the address F000h (R/W) = Subtract 4096 from the address F001h (R/W) = Subtract 4095 from the address FFFEh (R/W) = Subtract 2 from the address FFFFh (R/W) = Subtract 1 from the address

12.9.3.14 DST_WRAP_SIZE Register (Offset = Dh) [Reset = FFFFh]

DST_WRAP_SIZE is shown in Figure 12-24 and described in Table 12-25.

Return to the Summary Table.

Destination Wrap Size Register

Figure 12-24. DST_WRAP_SIZE Register

15	14	13	12	11	10	9	8
WRAPSIZE							
R/W-FFFFh							
7	6	5	4	3	2	1	0
WRAPSIZE							
R/W-FFFFh							

Table 12-25. DST_WRAP_SIZE Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	WRAPSIZE	R/W	FFFFh	These bits specify the number of bursts to transfer before the destination address wraps around to the beginning address. The actual number is equal to WRAPSIZE + 1. To disable the wrapping function, set WRAPSIZE to a value larger than TRANSFERSIZE. Reset type: SYSRSn

12.9.3.15 DST_WRAP_COUNT Register (Offset = Eh) [Reset = 0h]

DST_WRAP_COUNT is shown in Figure 12-25 and described in Table 12-26.

Return to the Summary Table.

Destination Wrap Count Register

Figure 12-25. DST_WRAP_COUNT Register

15	14	13	12	11	10	9	8
WRAPSIZE							
R-0h							
7	6	5	4	3	2	1	0
WRAPSIZE							
R-0h							

Table 12-26. DST_WRAP_COUNT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	WRAPSIZE	R	0h	These bits indicate the number of bursts left before wrapping the destination address. Reset type: SYSRSn

12.9.3.16 DST_WRAP_STEP Register (Offset = Fh) [Reset = 0h]

DST_WRAP_STEP is shown in Figure 12-26 and described in Table 12-27.

Return to the Summary Table.

Destination Wrap Step Register

Figure 12-26. DST_WRAP_STEP Register

15	14	13	12	11	10	9	8
WRAPSTEP							
R/W-0h							
7	6	5	4	3	2	1	0
WRAPSTEP							
R/W-0h							

Table 12-27. DST_WRAP_STEP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	WRAPSTEP	R/W	0h	These bits specify the change in the destination beginning address when the wrap counter reaches zero. The size must be a 16-bit two's complement value between -4096 and 4095 (inclusive). This value is added to the destination address when wrapping occurs. Reset type: SYSRSn 0h (R/W) = No address change 1h (R/W) = Add 1 to the address 2h (R/W) = Add 2 to the address FFEh (R/W) = Add 4094 to the address FFFh (R/W) = Add 4095 to the address F000h (R/W) = Subtract 4096 from the address F001h (R/W) = Subtract 4095 from the address FFFEh (R/W) = Subtract 2 from the address FFFFh (R/W) = Subtract 1 from the address

12.9.3.17 SRC_BEG_ADDR_SHADOW Register (Offset = 10h) [Reset = 0h]

SRC_BEG_ADDR_SHADOW is shown in Figure 12-27 and described in Table 12-28.

Return to the Summary Table.

Source Begin Address Shadow Register

Figure 12-27. SRC_BEG_ADDR_SHADOW Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BEGADDR																															
R/W-0h																															

Table 12-28. SRC_BEG_ADDR_SHADOW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BEGADDR	R/W	0h	Shadow Source Beginning Address At the start of a transfer, the value in this register is loaded into the SRC_BEG_ADDR_ACTIVE register and used as the beginning value for the source address. This register can be safely updated during a transfer. Reset type: SYSRSn

12.9.3.18 SRC_ADDR_SHADOW Register (Offset = 12h) [Reset = 0h]

SRC_ADDR_SHADOW is shown in Figure 12-28 and described in Table 12-29.

Return to the Summary Table.

Source Address Shadow Register

Figure 12-28. SRC_ADDR_SHADOW Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															
R/W-0h																															

Table 12-29. SRC_ADDR_SHADOW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR	R/W	0h	Shadow Source Address At the start of a transfer, the value in this register is loaded into the SRC_ADDR_ACTIVE register and used as the value of the source address. This register can be safely updated during a transfer. Reset type: SYSRSn

12.9.3.19 SRC_BEG_ADDR_ACTIVE Register (Offset = 14h) [Reset = 0h]

SRC_BEG_ADDR_ACTIVE is shown in Figure 12-29 and described in Table 12-30.

Return to the Summary Table.

Source Begin Address Active Register

Figure 12-29. SRC_BEG_ADDR_ACTIVE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BEGADDR																															
R-0h																															

Table 12-30. SRC_BEG_ADDR_ACTIVE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BEGADDR	R	0h	Active Source Beginning Address If a transfer is ongoing, this register holds the current beginning value for the source address. This address may be updated after wrapping. When a transfer starts, this register is loaded with the shadow address from the SRC_BEG_ADDR_SHADOW register. Reset type: SYSRSn

12.9.3.20 SRC_ADDR_ACTIVE Register (Offset = 16h) [Reset = 0h]

SRC_ADDR_ACTIVE is shown in Figure 12-30 and described in Table 12-31.

Return to the Summary Table.

Source Address Active Register

Figure 12-30. SRC_ADDR_ACTIVE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															
R-0h																															

Table 12-31. SRC_ADDR_ACTIVE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR	R	0h	Active Source Address If a transfer is ongoing, this register holds the current value of the source address. This address may change after a write, a burst, or wrapping. Reset type: SYSRSn

12.9.3.21 DST_BEG_ADDR_SHADOW Register (Offset = 18h) [Reset = 0h]

DST_BEG_ADDR_SHADOW is shown in Figure 12-31 and described in Table 12-32.

Return to the Summary Table.

Destination Begin Address Shadow Register

Figure 12-31. DST_BEG_ADDR_SHADOW Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BEGADDR																															
R/W-0h																															

Table 12-32. DST_BEG_ADDR_SHADOW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BEGADDR	R/W	0h	Shadow Destination Beginning Address At the start of a transfer, the value in this register is loaded into the DST_BEG_ADDR_ACTIVE register and used as the beginning value for the destination address. This register can be safely updated during a transfer. Reset type: SYSRSn

12.9.3.22 DST_ADDR_SHADOW Register (Offset = 1Ah) [Reset = 0h]

DST_ADDR_SHADOW is shown in Figure 12-32 and described in Table 12-33.

Return to the Summary Table.

Destination Address Shadow Register

Figure 12-32. DST_ADDR_SHADOW Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															
R/W-0h																															

Table 12-33. DST_ADDR_SHADOW Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR	R/W	0h	Shadow Destination Address At the start of a transfer, the value in this register is loaded into the DST_ADDR_ACTIVE register and used as the value of the destination address. This register can be safely updated during a transfer. Reset type: SYSRSn

12.9.3.23 DST_BEG_ADDR_ACTIVE Register (Offset = 1Ch) [Reset = 0h]

DST_BEG_ADDR_ACTIVE is shown in Figure 12-33 and described in Table 12-34.

Return to the Summary Table.

Destination Begin Address Active Register

Figure 12-33. DST_BEG_ADDR_ACTIVE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BEGADDR																															
R-0h																															

Table 12-34. DST_BEG_ADDR_ACTIVE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	BEGADDR	R	0h	Active Destination Beginning Address If a transfer is ongoing, this register holds the current destination value for the source address. This address may be updated after wrapping. When a transfer starts, this register is loaded with the shadow address from the DST_BEG_ADDR_SHADOW register. Reset type: SYSRSn

12.9.3.24 DST_ADDR_ACTIVE Register (Offset = 1Eh) [Reset = 0h]

DST_ADDR_ACTIVE is shown in Figure 12-34 and described in Table 12-35.

Return to the Summary Table.

Destination Address Active Register

Figure 12-34. DST_ADDR_ACTIVE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ADDR																															
R-0h																															

Table 12-35. DST_ADDR_ACTIVE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ADDR	R	0h	Active Destination Address If a transfer is ongoing, this register holds the current value of the destination address. This address may change after a write, a burst, or wrapping. Reset type: SYSRSn

12.9.4 DMA Registers to Driverlib Functions

Table 12-36. DMA Registers to Driverlib Functions

File	Driverlib Function
CTRL	
dma.h	DMA_initController
DEBUGCTRL	
dma.h	DMA_setEmulationMode
PRIORITYCTRL1	
dma.h	DMA_setPriorityMode
PRIORITYSTAT	
-	
MODE	
dma.c	DMA_configMode
dma.h	DMA_enableTrigger
dma.h	DMA_disableTrigger
dma.h	DMA_enableInterrupt
dma.h	DMA_disableInterrupt
dma.h	DMA_enableOverrunInterrupt
dma.h	DMA_disableOverrunInterrupt
dma.h	DMA_setInterruptMode
CONTROL	
dma.h	DMA_triggerSoftReset
dma.h	DMA_forceTrigger
dma.h	DMA_clearTriggerFlag
dma.h	DMA_getTransferStatusFlag
dma.h	DMA_getBurstStatusFlag
dma.h	DMA_getRunStatusFlag
dma.h	DMA_getOverflowFlag
dma.h	DMA_getTriggerFlagStatus
dma.h	DMA_startChannel

Table 12-36. DMA Registers to Driverlib Functions (continued)

File	Driverlib Function
dma.h	DMA_stopChannel
dma.h	DMA_clearErrorFlag
BURST_SIZE	
dma.c	DMA_configBurst
BURST_COUNT	
-	
SRC_BURST_STEP	
dma.c	DMA_configBurst
DST_BURST_STEP	
dma.c	DMA_configBurst
TRANSFER_SIZE	
dma.c	DMA_configTransfer
TRANSFER_COUNT	
-	
SRC_TRANSFER_STEP	
dma.c	DMA_configTransfer
DST_TRANSFER_STEP	
dma.c	DMA_configTransfer
SRC_WRAP_SIZE	
dma.c	DMA_configWrap
SRC_WRAP_COUNT	
-	
SRC_WRAP_STEP	
dma.c	DMA_configWrap
DST_WRAP_SIZE	
dma.c	DMA_configWrap
DST_WRAP_COUNT	
-	
DST_WRAP_STEP	
dma.c	DMA_configWrap
SRC_BEG_ADDR_SHADOW	
dma.c	DMA_configAddresses
dma.h	DMA_configSourceAddress
SRC_ADDR_SHADOW	
dma.c	DMA_configAddresses
dma.h	DMA_configSourceAddress
SRC_BEG_ADDR_ACTIVE	
-	
SRC_ADDR_ACTIVE	
-	
DST_BEG_ADDR_SHADOW	
dma.c	DMA_configAddresses
dma.h	DMA_configDestAddress
DST_ADDR_SHADOW	
dma.c	DMA_configAddresses

Table 12-36. DMA Registers to Driverlib Functions (continued)

File	Driverlib Function
dma.h	DMA_configDestAddress
DST_BEG_ADDR_ACTIVE	
-	
DST_ADDR_ACTIVE	
-	

This page intentionally left blank.

Chapter 13

Embedded Real-time Analysis and Diagnostic (ERAD)

This chapter describes the features and operation of the embedded real-time analysis and diagnostic (ERAD) module. The ERAD module enhances the debug and system analysis capabilities of the device. The debug and system analysis enhancements provided by the ERAD module are implemented external to the CPU. . The EBC units are used to generate hardware breakpoints, hardware watch points, and other output events. The SEC units are used to analyze and profile the system. The ERAD module is accessible both by the debugger and the application software, which significantly increases the debug capabilities of many real-time systems, especially in situations where the debugger is not connected.

13.1 Introduction

The ERAD module is shown in Figure 13-1.

The ERAD enhances the debug and system analysis capabilities of the device external to the CPU. The C28x CPU alone has two analysis resources; Analysis Unit 1 (AU1) and Analysis Unit 2 (AU2). The first analysis unit counts events or monitors address buses. The second analysis unit monitors address and data buses. The two analysis units can be configured for hardware breakpoints or hardware watch points, and additionally the first analysis unit can be configured as a benchmark counter or event counter. The ERAD module further expands this capability to provide additional hardware breakpoints, hardware watch points, and counters for profiling, as well as other advanced features. The ERAD module can be utilized by the debugger, and also by the application software. For many real-time systems, it is not always possible to connect a debugger and perform an intrusive debug. Under these situations, the user's code has the ability to set up and control the ERAD module in order to debug and profile the system without disturbing the end application.

The ERAD module consists of eight enhanced bus comparator (EBC) units and four system event counter (SEC) units. The EBC units monitor buses and generate output events. The SEC units can be used with EBC units to profile and analyze the system. These units are described in detail in the following sections.

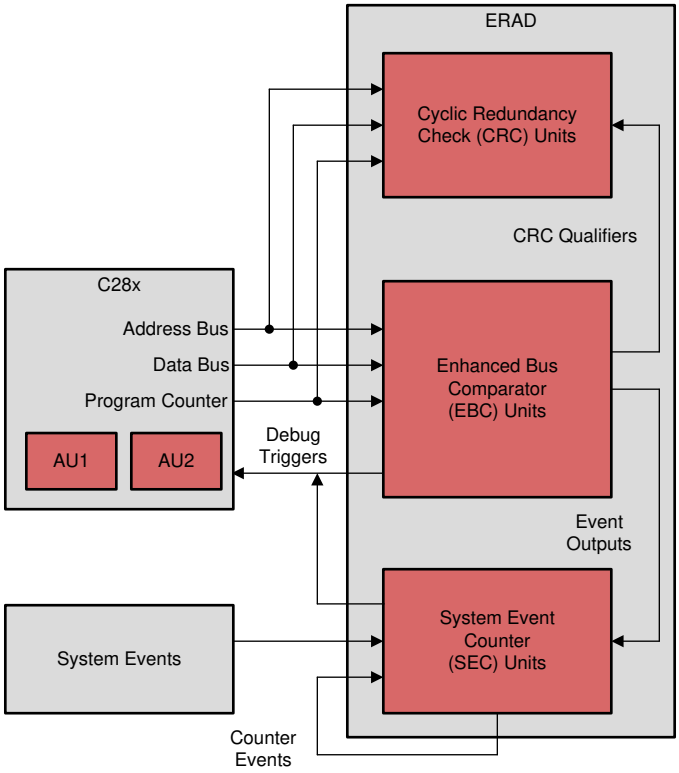


Figure 13-1. ERAD Overview

13.1.1 ERAD Related Collateral

Foundational Materials

- C2000 Academy - System Design
- Embedded Real-Time Analysis & Diagnostics (ERAD) on C2000 Devices (Video)

Getting Started Materials

- Embedded Real-Time Analysis & Diagnostics (ERAD) on C2000 MCUs (Video)
- *Embedded Real-Time Analysis and Response for Control Applications Application Report*

13.2 Enhanced Bus Comparator Unit

The EBC units connect to the CPU similar to any standard direct memory interface connection. The program space buses, data space buses, debug qualifiers for memory access, and debug-related strobes necessary for breakpoints, watch points and trace points, are connected between the CPU and these units. The EBC units are usually controlled by the debug software. However, a CPU application code can also configure and use these units for interrupt and event generation. The ERAD module can be configured and used by either the application software or debug software. It is not possible for an EBC unit to be used by both the application software and the debug software simultaneously. The ERAD module will be owned by either the debugger or the application software through the memory-mapped registers.

The EBC unit has the following capabilities:

- Generate hardware breakpoints
- Generate hardware watch points
- Generate trace tags for instruction fetch matches and generate RTOSINT
- Monitor data read address buses, data write address buses, data write data bus, and generate RTOSINT
- Generate an event output which can be used by other modules. This is done through monitoring any of the program address buses, Virtual Program Counter (VPC), or the Program Counter of the CPU

The following features are not supported by the EBC units:

- Chain breakpoints
- Ability to monitor DMA transfers
- Ability to monitor CLA buses

The EBC units have the capability to monitor a range of addresses by defining masks and generating outputs based on greater than, less than, or equal events.

The EBC units can be used with the SEC units for profiling and analysis purposes.

13.2.1 Enhanced Bus Comparator Unit Operations

The following operations are supported by the EBC unit:

- Operation of a hardware breakpoint: The EBC unit will generate the appropriate tags for an access made to a particular address. When the instruction is going to enter the DECODE 2 (D2) phase of the pipeline, the CPU is halted.
- Operation of a watch point: Unlike breakpoints which halt the CPU before a designated address is executed, watch points will detect the occurrence of a data memory read or write, and will halt the CPU. There is no precise definition as to when the CPU will halt, since it is entirely dependant on the current state of the CPU pipeline. The CPU will halt at the next interruptible boundary.
- Operation of program trace: This operation is very similar to the hardware breakpoint operation. The difference between the CPU behavior is that it will internally generate an RTOSINT when the tagged instruction reaches the DECODE 2 (D2) phase of the pipeline. If this instruction is discarded in the fetch buffer due to discontinuity, then no RTOSINT will be generated.
- Operation of RTOSINTn: This is an alternative to a watch point where an interrupt can be generated to the CPU on detecting the designated read and write access.

It is important to note that a hardware breakpoint will only halt the CPU if a debugger is connected.

13.2.2 Event Masking and Exporting

The events generated by different EBC units can be combined using OR and AND logics to generate new events as required. There are four AND and four OR combinations that could be exported using masks to suppress unnecessary events. These events may also be configured to generate an RTOSINT.

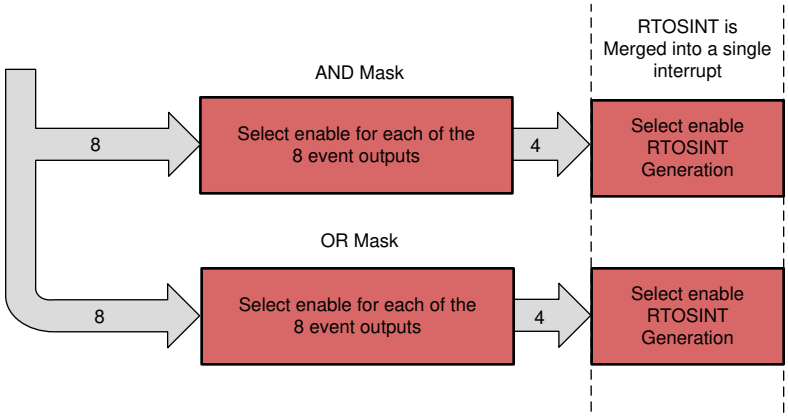


Figure 13-2. EBC Units Event Masking

13.3 System Event Counter Unit

The SEC units provide better system profiling, analysis, and debug capability. The SEC units contain counters which can enhance the debug and profiling process in various types of system scenarios such as:

- Profiling code segments
- Counting duration between specified memory reads and writes
- Counting system events (such as interrupts)
- Counting duration between system events
- System timer
- Measuring the number of wait states in code segments
- Measuring the maximum amount of time spent in between a pair of events, measured over multiple iterations
- Chaining counters in order to link events or create larger counters

Furthermore, the SEC unit has the capability to:

- Function as a counter capable of counting:
  - Any of the match events generated by the EBC units
  - Events generated by the EBC units. These events can be used to start and stop the counting.
  - System events including the PIE interrupts, timer interrupts, and CLA task interrupts. These system events can be used to start and stop the counting.
  - More information on the sources of the SEC units can be found at the Event Selector Mux Signals Table.
- Generate an interrupt or a watch point if the count reaches a reference value.
- Operate in counting mode in one of the following two modes:
  - Duration mode: The counter will count the CPU cycles as long as the event is active.
  - Event mode: The counter will count only the positive edge of the event signal. This is effectively counting the number of times the event transitions from inactive to active.

13.3.1 System Event Counter Modes

The following are the key features of the SEC unit. The counters are initialized to zero and will always count up.

- Continuous Count: In this mode the counter continues to count as specified by the input selector. If the counter reaches the maximum value, it resets to zero and will continue to count up. A sticky overflow bit will be set to indicate that this counter had overflowed. The counter can count the CPU cycles without any events selected. In this mode the module can be used as a software controlled SYSCLK counter.
- Timer Mode Count: In this mode, the counter counts up to a set reference value and after reaching the reference value, it resets to zero. The counter will generate an event that can send an interrupt to the CPU or generate a watch point. The counter can be set up to either continue incrementing or reset when a match event occurs.
- Start-Stop Count: In this mode, two events are allowed to act as start and stop indicators to the counter. The counter will commence counting only when the defined start event occurs. The counter will then continue to count up until the stop event occurs. Once the first start event has occurred, further start events will be ignored till the stop event occurs.

13.3.1.1 Counting Active Levels Versus Edges

The SEC units can be configured to either count active levels or edges of the selected inputs.

Each SEC unit has eight inputs from the EBC units and many inputs from other events in the device. Each SEC unit can be configured to count any of the input events or just count up on every cycle. For example, if an input event occurs and is active for 25 cycles, the SEC unit's counter increments only by 1 in event mode, whereas in the duration mode, it increments by 25.

13.3.1.2 Max Mode

Max mode is also supported by the SEC units. This mode allows the user to detect the maximum count that has occurred during various count iterations in start-stop mode. For example, a user can set up the counter in the start-stop count mode to count the duration of a critical code loop. Every time the stop event occurs and the counter stops, the counter value is checked against the current MAX_COUNT present in the register. If the new value is greater, then the MAX_COUNT register is updated. The counter will always reset to zero at the stop event and will be ready to start counting on the next start event. Therefore the MAX_COUNT will contain the maximum number of cycles that occurred between the start and stop condition over many iterations.

13.3.1.3 Cumulative Mode

The SEC units can be used to yield the cumulative count over several start and stop events. In this mode the, unlike Max mode, the counter does not reset due to a stop event. Instead it stops counting and resumes counting when a start event occurs. in cumulative count mode, the MAX_COUNT is not valid.

13.3.1.4 Input Signal Selection

The SEC inputs can be selected from various signals from in the system to enable debug and system analysis. Figure 13-3 shows the SEC inputs. Each event selector MUX can select from various signals on in the system. These signals are show in Table 13-1.

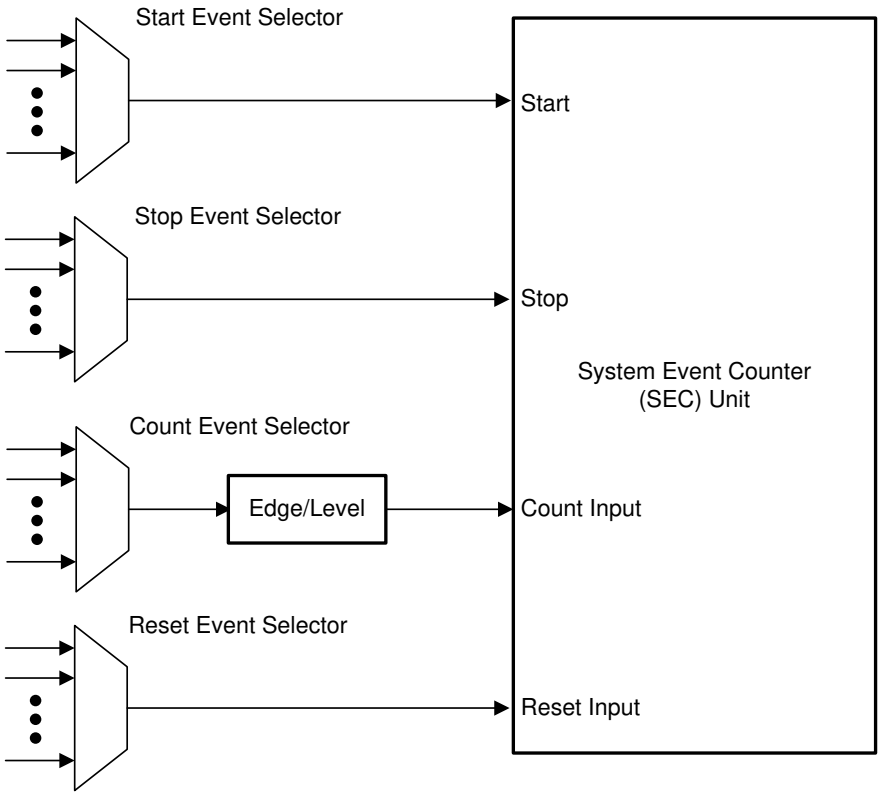


Figure 13-3. System Event Counter Inputs

Table 13-1. Event Selector Mux Signals

CTM_INP_SEL, STA_INP_SEL, STO_INP_SEL, RST_INP_SEL	EVENT_INPUT_SELECTED
INP_SEL[0]	EBC1
INP_SEL[1]	EBC2
INP_SEL[2]	EBC3
INP_SEL[3]	EBC4
INP_SEL[4]	EBC5
INP_SEL[5]	EBC6
INP_SEL[6]	EBC7
INP_SEL[7]	EBC8
INP_SEL[8]	COUNTER1_EVENT
INP_SEL[9]	COUNTER2_EVENT
INP_SEL[10]	COUNTER3_EVENT
INP_SEL[11]	COUNTER4_EVENT
INP_SEL[12]	ERAD_OR_MASK[0]
INP_SEL[13]	ERAD_OR_MASK[1]
INP_SEL[14]	ERAD_OR_MASK[2]
INP_SEL[15]	ERAD_OR_MASK[3]
INP_SEL[16]	ERAD_AND_MASK[0]
INP_SEL[17]	ERAD_AND_MASK[1]
INP_SEL[18]	ERAD_AND_MASK[2]
INP_SEL[19]	ERAD_AND_MASK[3]
INP_SEL[20]	PIE_INT1

Table 13-1. Event Selector Mux Signals (continued)

CTM_INP_SEL, STA_INP_SEL, STO_INP_SEL, RST_INP_SEL	EVENT_INPUT_SELECTED
INP_SEL[21]	PIE_INT2
INP_SEL[22]	PIE_INT3
INP_SEL[23]	PIE_INT4
INP_SEL[24]	PIE_INT5
INP_SEL[25]	PIE_INT6
INP_SEL[26]	PIE_INT7
INP_SEL[27]	PIE_INT8
INP_SEL[28]	PIE_INT9
INP_SEL[29]	PIE_INT10
INP_SEL[30]	PIE_INT11
INP_SEL[31]	PIE_INT12
INP_SEL[32]	TIMER0INT
INP_SEL[33]	TIMER1INT
INP_SEL[34]	TIMER2INT
INP_SEL[35]	DMACH1-INT
INP_SEL[36]	DMACH2-INT
INP_SEL[37]	DMACH3-INT
INP_SEL[38]	DMACH4-INT
INP_SEL[39]	DMACH5-INT
INP_SEL[40]	DMACH6-INT
INP_SEL[41]	FSIRXA DATA_PKT_RCVD
INP_SEL[42]	FSIRXA ERROR_PKT_RCVD
INP_SEL[43]	FSIRXA PING_PKT_RCVD
INP_SEL[44]	FSIRXA PING_FRAME_TAG_MATCH
INP_SEL[45]	FSIRXA DATA_FRAME_TAG_MATCH
INP_SEL[46]	FSIRXA ERROR_FRAME_TAG_MATCH
INP_SEL[47]	FSIRXA FRAME_DONE
INP_SEL[48]	ADCA_EVT_INT
INP_SEL[49]	ADCC_EVT_INT
INP_SEL[50]	MCANA_EVT0
INP_SEL[51]	MCANA_EVT1
INP_SEL[52]	MCANA_EVT2
INP_SEL[53]	ADCSOCA
INP_SEL[54]	ADCSOCB
INP_SEL[55]	CLATASKRUN1
INP_SEL[56]	CLATASKRUN2
INP_SEL[57]	CLATASKRUN3
INP_SEL[58]	CLATASKRUN4
INP_SEL[59]	CLATASKRUN5
INP_SEL[60]	CLATASKRUN6
INP_SEL[61]	CLATASKRUN7
INP_SEL[62]	CLATASKRUN8
INP_SEL[63]	(Async, Polarity = LOW) ePWM XBAR 0
INP_SEL[64]	(Async, Polarity = LOW) ePWM XBAR 1
INP_SEL[65]	(Async, Polarity = LOW) ePWM XBAR 2

Table 13-1. Event Selector Mux Signals (continued)

CTM_INP_SEL, STA_INP_SEL, STO_INP_SEL, RST_INP_SEL	EVENT_INPUT_SELECTED
INP_SEL[66]	(Async, Polarity = LOW) ePWM XBAR 3
INP_SEL[67]	(Async, Polarity = LOW) ePWM XBAR 4
INP_SEL[68]	(Async, Polarity = LOW) ePWM XBAR 5
INP_SEL[69]	(Async, Polarity = LOW) ePWM XBAR 6
INP_SEL[70]	(Async, Polarity = LOW) ePWM XBAR 7
INP_SEL[71]	(Async, Polarity = LOW) OUTPUT XBAR0
INP_SEL[72]	(Async, Polarity = LOW) OUTPUT XBAR1
INP_SEL[73]	(Async, Polarity = LOW) OUTPUT XBAR2
INP_SEL[74]	(Async, Polarity = LOW) OUTPUT XBAR3
INP_SEL[75]	(Async, Polarity = LOW) OUTPUT XBAR4
INP_SEL[76]	(Async, Polarity = LOW) OUTPUT XBAR5
INP_SEL[77]	(Async, Polarity = LOW) OUTPUT XBAR6
INP_SEL[78]	(Async, Polarity = LOW) OUTPUT XBAR7
INP_SEL[79]	(Async, Polarity = LOW) OUTPUT XBAR8
INP_SEL[80]	(Async, Polarity = LOW) OUTPUT XBAR9
INP_SEL[81]	(Async, Polarity = LOW) OUTPUT XBAR10
INP_SEL[82]	(Async, Polarity = LOW) OUTPUT XBAR11
INP_SEL[83]	(Async, Polarity = LOW) OUTPUT XBAR12
INP_SEL[84]	(Async, Polarity = LOW) OUTPUT XBAR13
INP_SEL[85]	(Async, Polarity = LOW) OUTPUT XBAR14
INP_SEL[86]	(Async, Polarity = LOW) OUTPUT XBAR15
INP_SEL[87]	(Polarity = LOW) CPUx.CPUSTAT
INP_SEL[88]	CPUx.DBGACK
INP_SEL[89]	CPUx.NMI
INP_SEL[90]	(Async) CMPSS1.CTRIPH_OR_CTRIPL
INP_SEL[91]	(Async) CMPSS2.CTRIPH_OR_CTRIPL
INP_SEL[92]	(Async) CMPSS3.CTRIPH_OR_CTRIPL
INP_SEL[93]	(Async) CMPSS4.CTRIPH_OR_CTRIPL
INP_SEL[94]	Reserved
INP_SEL[95]	Reserved
INP_SEL[96]	Reserved
INP_SEL[97]	Reserved
INP_SEL[98]	SD1FLT1.COMPH_OR_COMPL
INP_SEL[99]	SD1FLT2.COMPH_OR_COMPL
INP_SEL[100]	SD1FLT3.COMPH_OR_COMPL
INP_SEL[101]	SD1FLT4.COMPH_OR_COMPL
INP_SEL[102]	SD2FLT1.COMPH_OR_COMPL
INP_SEL[103]	SD2FLT2.COMPH_OR_COMPL
INP_SEL[104]	SD2FLT3.COMPH_OR_COMPL
INP_SEL[105]	SD2FLT4.COMPH_OR_COMPL
INP_SEL[106]	ADCAINT1
INP_SEL[107]	ADCAINT2
INP_SEL[108]	ADCAINT3
INP_SEL[109]	ADCAINT4
INP_SEL[110]	ADCBINT1

Table 13-1. Event Selector Mux Signals (continued)

CTM_INP_SEL, STA_INP_SEL, STO_INP_SEL, RST_INP_SEL	EVENT_INPUT_SELECTED
INP_SEL[111]	ADCBINT2
INP_SEL[112]	ADCBINT3
INP_SEL[113]	ADCBINT4
INP_SEL[114]	ADCCINT1
INP_SEL[115]	ADCCINT2
INP_SEL[116]	ADCCINT3
INP_SEL[117]	ADCCINT4
INP_SEL[118:121]	Reserved
INP_SEL[122]	HIC_nOE
INP_SEL[123]	HIC_nWE
INP_SEL[124]	(Async) HIC_nRDY
INP_SEL[125]	ADCB_EVT_INT
INP_SEL[126:127]	Reserved

13.3.2 Reset on Event

It is also possible to reset the counters on external events. Additionally all the counter event outputs are fed back to each of the counter's input MUX which will select the event that can be used as a reset input. When enabled, an active high on the reset input will cause the counter to reset. This gives a powerful feature which allows the user to set up threshold monitors. This can be used to flag an interrupt or a watch point if the distance between two events crosses a certain threshold.

13.3.3 Operation Conditions

The SEC units will count accurately only when the CPU is operating in normal conditions. If the counters are running and capturing the CPU cycles while the CPU is controlled through the debugger in order to single-step through the code, then the result may differ from when the CPU was executing the code in normal conditions.

It is also important to note that if the counters are set up to use the value of the raw program counter register (VPC) as the source for the start and stop events, the value of the CPU cycles measured may be off by a few cycles when the CALL instruction is executed.

13.4 ERAD Ownership, Initialization and Reset

Even though the features of this module are meant for the debugger to use, the user application may also need to take advantage of the capabilities to monitor the buses and generate interrupts and events. It is an option to completely hand over the ownership of the ERAD module to the application software or the debugger. Another option is for both the application code and the debugger to use the ERAD module. Only the owner of the module (application code or debugger) is allowed to use the module at that time. The responsibility of resolving any ownership conflict, is on the software. The last option is to have no owner. In this mode, both the application code and the debugger have the capabillity to access the module at the same time. It is critical for the software, both on the application side and the debugger side, to resolve any conflicts. In this mode, it is possible for the debugger to use some of the EBC units and the SEC units, while the application software uses the remaining units.

The ERAD module will initialize its internal states and all the registers to their initialized states under the following conditions:

- At power-on-reset (POR)
- With DCON and SYSRSN
  - Debug logic disconnected when the debugger owns the module
  - Functional reset when application owns the module

13.5 ERAD Programming Sequence

The ERAD module can be used to set hardware breakpoints and hardware watch points. The programming sequence to set hardware breakpoints, hardware watch points, or to use the timers to profile and analyze the system are described in this section. The same sequence can be used through the debugger software or the application code.

Refer to the Driverlib example projects in C2000Ware for JavaScript files to configure the ERAD module. Example projects are also available to showcase the usage of these script files. These examples are included in the latest Driverlib install: <C2000Ware installation>\driverlib\f28004x\examples\erad

13.5.1 Hardware Breakpoint and Hardware Watch Point Programming Sequence

The programming sequence is identical when using the EBC units, regardless of whether it is a debug software or the application programing the units. A typical programming sequence for a unit is described below.

- Read and make sure the ownership is set as expected; if not, acquire the ownership before proceeding further if required.
- Ensure the unit is in IDLE mode.
- Set up the address reference, mask, bus select and stop bits.
- Enable the corresponding module bit in the global enable register.
- Once the usage is completed, write ‘1’ to Clear EVENT_FIRED sticky bit; This will take the module back to the enabled state.

The example programming sequences for hardware breakpoints and hardware watch points are shown below.

Set a hardware breakpoint on address 0x201000:

- Read GLBL_OWNER to confirm ownership.
- Read HWBP_STATUS to confirm the module is in IDLE state.
- Write 0x0 to HWBP_MASK.
- Write 0x201000 to HWBP_REF.
- Enable the corresponding module bit in the global enable register.

Set a hardware watch point on read of addresses from 0x121010 to 0x12101f:

- Read GLBL_OWNER to confirm ownership.
- Read HWBP_STATUS to confirm the module is in IDLE state.
- Write 0xf to HWBP_MASK.
- Write 0x121010 to HWBP_REF.
- Enable the corresponding module bit in the global enable register.

Set a hardware watch point on write to address 0xff10101a:

- Read GLBL_OWNER to confirm ownership
- Read HWBP_STATUS to confirm the module is in IDLE state
- Write 0x0 to HWBP_MASK
- Write 0xff10101a to HWBP_REF
- Enable the corresponding module bit in the global enable register.

13.5.2 Timer and Counter Programming Sequence

The programming sequence is identical when using the SEC units, regardless of whether it is a debug software, or the application, programing the units. Typical programming sequence for a unit is described below.

- Read and make sure the ownership is set as expected. If not, acquire the ownership before proceeding further if required.
- Ensure the unit is in IDLE mode.
- Set up the counter reference, counter registers (clear/reset if a clean start is required) and CTM_CNTL.
- Enable the corresponding module bit in the global enable register.
- Once the usage is completed, write ‘1’ to Clear EVENT_FIRED sticky bit. This will take the module back to the enabled state.

Set up a free running counter:

- Read and make sure the ownership is set as expected. If not, acquire the ownership before proceeding further, if required.
- Read CTM_STATUS to confirm that the module is in IDLE state.
- Write 0x0 to CTM_INPUT_SEL.
- Write 0x0 to CTM_CNTL.
- Enable the module in the GLBL_ENABLE register.

Set up the counter to count the duration spent between addresses 0x1000 and 0x1210:

- Read and make sure the ownership is set as expected. If not, acquire the ownership before proceeding further, if required.
- Read CTM_STATUS to confirm that the module is in IDLE state.
- Set up the EBC unit 1 to generate an event for VPC = 0x1000.
- Set up the EBC unit 2 to generate an event for VPC = 0x1210.
- Enable the module in the GLBL_ENABLE register.

Set up the counter to count the number of times a function at address 0x2010 is called and fire an RTOSINT if this count reaches 0x300:

- Read and make sure the ownership is set as expected. If not, acquire the ownership before proceeding further, if required.
- Read CTM_STATUS to confirm that the module is in IDLE state.
- Set up the EBC unit 1 to generate an event for VPC = 0x2010.
- Write 0x300 CTM_REF.
- Enable the counter in the EVENT_MODE, and also allow it to generate an RTOSINT when the count matches the reference. This is achieved by writing 0x88 to CTM_CNTL (bit 3 = 1 and bit 7 = 1).
- Enable the module in the GLBL_ENABLE register.

13.6 Cyclic Redundancy Check Unit

The cyclic redundancy check (CRC) units monitor CPU buses and compute CRC when the self-test code is executed. This capability aids in achieving simpler, non-intrusive and interruptible self-test mechanisms with Software Test Library (STL). Each CRC unit is used to monitor a different CPU interface. For example, CRC unit 1 is used to monitor the Program Counter, while CRC unit 2 is used to monitor the data read address bus. Table 13-2 identifies the CPU interface monitored by each of the CRC units.

Table 13-2. CPU Interfaces Monitored by CRC Units

CRC Unit	CPU Interface
CRC Unit 1	Program Counter Register
CRC Unit 2	Data Read Address Bus
CRC Unit 3	Data Read Data Bus
CRC Unit 4	Data Write Address Bus
CRC Unit 5	Data Write Data Bus
CRC Unit 6	Instruction Register Value (Unsecured)
CRC Unit 7	Instruction Register Value (Secure-Zone 1)
CRC Unit 8	Instruction Register Value (Secure-Zone 2)

The main purpose of the CRC units is to ensure that the CPU functionally remains intact when it is executing the same software test library over multiple iterations. This is done by comparing the computed CRC values after each iteration, with a pre-computed golden value.

CRC units 7 and 8 are intended to compute the Instruction Register values for secure-zone 1 and secure-zone 2 instruction execution. Computed CRC values for the a given secure-zone is available only for accesses originating from that zone.

13.6.1 CRC Unit Qualifier

By default, all the valid events on a given interface enables a CRC unit for computation. However there are optional qualifiers which can be used to gate the CRC computation when valid events occur. If required, these qualifiers can be generated by masking the EBC units events. The AND/OR masks discussed in Section 13.2.2 are used to generate these qualifiers. Refer to the CRC_QUALIFIER register for a full list of available qualifiers. Figure 13-4 shows the connection between EBC event masking and exporting with the CRC qualifiers.

EBC units have the capability to monitor the Program Counter, data writes and data reads. CRC units can use the EBC units to decide when the check values should be calculated. For example, if it is required to calculate the check values only when the CPU is executing a specific function, the user can set up an EBC unit to monitor the PC and generate a CRC qualifier when that function is executed. This allows the CRC unit to calculated the check value only when desired.

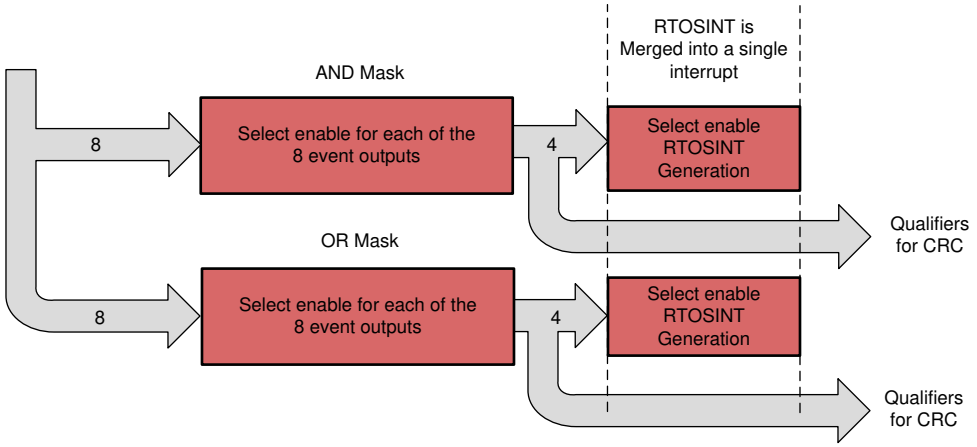


Figure 13-4. Event Masking and Exporting for CRC Qualifiers

13.6.2 CRC Unit Programming Sequence

The following sequence can be used to initialize a CRC unit to calculate the check value for the corresponding CPU interface.

1. Initialize the CRC unit by writing a '1' to the corresponding CRC_INIT field in the CRC_GLOBAL_CTRL register.
2. Configure the seed value (if required) using CRC_SEED register (CRC_EN should be ‘0’ for when modifying the CRC_SEED register).
3. Configure the CRC_QUALIFIER register if additional qualification from EBC units is required; If not additional qualification is not required, set the CRC_QUALIFIER register to '0'.
4. Enable the CRC unit by setting the CRC_EN to '1' at the beginning of the software for which the CRC calculation is done.
5. Diable the CRC unit by setting the CRC_EN to '0' at the end of the software for which the CRC calculation is done.
6. Read the CRC_CURRENT register to record the computed CRC; This check value can be compared with previous check values to ensure no changes has occurred.
7. Repeat steps 1-7 periodically as needed.

Note

Sufficient NOP's must be added immediately after the CRC submodule is enabled in order to make sure the pipeline contains predictable code as soon as the CRC is enabled. Similarly, sufficient NOP's must be added before the CRC submodule is disabled. Disabling the CRC write access will take a few cycles, therefore there must be predictable code in the pipeline stages until the write takes place.

13.7 ERAD Registers

This section describes the Embedded Real-Time Analysis and Diagnostic Registers.

13.7.1 ERAD Base Address Table

Table 13-3. ERAD Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
EradGlobalRegs	ERAD_GLOBAL_REGS	ERAD_GLOBAL_BASE	0x0005_E800	YES	-	-	-	YES
EradHWBP1Regs	ERAD_HWBP_REGS	ERAD_HWBP1_BASE	0x0005_E900	YES	-	-	-	YES
EradHWBP2Regs	ERAD_HWBP_REGS	ERAD_HWBP2_BASE	0x0005_E908	YES	-	-	-	YES
EradHWBP3Regs	ERAD_HWBP_REGS	ERAD_HWBP3_BASE	0x0005_E910	YES	-	-	-	YES
EradHWBP4Regs	ERAD_HWBP_REGS	ERAD_HWBP4_BASE	0x0005_E918	YES	-	-	-	YES
EradHWBP5Regs	ERAD_HWBP_REGS	ERAD_HWBP5_BASE	0x0005_E920	YES	-	-	-	YES
EradHWBP6Regs	ERAD_HWBP_REGS	ERAD_HWBP6_BASE	0x0005_E928	YES	-	-	-	YES
EradHWBP7Regs	ERAD_HWBP_REGS	ERAD_HWBP7_BASE	0x0005_E930	YES	-	-	-	YES
EradHWBP8Regs	ERAD_HWBP_REGS	ERAD_HWBP8_BASE	0x0005_E938	YES	-	-	-	YES
EradCounter1Regs	ERAD_COUNTER_REGS	ERAD_COUNTER1_BASE	0x0005_E980	YES	-	-	-	YES
EradCounter2Regs	ERAD_COUNTER_REGS	ERAD_COUNTER2_BASE	0x0005_E990	YES	-	-	-	YES
EradCounter3Regs	ERAD_COUNTER_REGS	ERAD_COUNTER3_BASE	0x0005_E9A0	YES	-	-	-	YES
EradCounter4Regs	ERAD_COUNTER_REGS	ERAD_COUNTER4_BASE	0x0005_E9B0	YES	-	-	-	YES
EradCRCGlobalRegs	ERAD_CRC_GLOBAL_REGS	ERAD_CRC_GLOBAL_BASE	0x0005_EA00	YES	-	-	-	YES
EradCRC1Regs	ERAD_CRC_REGS	ERAD_CRC1_BASE	0x0005_EA10	YES	-	-	-	YES
EradCRC2Regs	ERAD_CRC_REGS	ERAD_CRC2_BASE	0x0005_EA20	YES	-	-	-	YES
EradCRC3Regs	ERAD_CRC_REGS	ERAD_CRC3_BASE	0x0005_EA30	YES	-	-	-	YES
EradCRC4Regs	ERAD_CRC_REGS	ERAD_CRC4_BASE	0x0005_EA40	YES	-	-	-	YES
EradCRC5Regs	ERAD_CRC_REGS	ERAD_CRC5_BASE	0x0005_EA50	YES	-	-	-	YES
EradCRC6Regs	ERAD_CRC_REGS	ERAD_CRC6_BASE	0x0005_EA60	YES	-	-	-	YES
EradCRC7Regs	ERAD_CRC_REGS	ERAD_CRC7_BASE	0x0005_EA70	YES	-	-	-	YES
EradCRC8Regs	ERAD_CRC_REGS	ERAD_CRC8_BASE	0x0005_EA80	YES	-	-	-	YES

13.7.2 ERAD_GLOBAL_REGS Registers

Table 13-4 lists the memory-mapped registers for the ERAD_GLOBAL_REGS registers. All register offset addresses not listed in Table 13-4 should be considered as reserved locations and the register contents should not be modified.

Table 13-4. ERAD_GLOBAL_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	GLBL_EVENT_STAT	Global Event Status Register		Go
2h	GLBL_HALT_STAT	Global Halt Status Register		Go
4h	GLBL_ENABLE	Global Enable Register	EALLOW	Go
6h	GLBL_CTM_RESET	Global Counter Reset	EALLOW	Go
8h	GLBL_NMI_CTL	Global Debug NMI control	EALLOW	Go
Ah	GLBL_OWNER	Global Ownership	EALLOW	Go
Ch	GLBL_EVENT_AND_MASK	Global Bus Comparator Event AND Mask Register	EALLOW	Go
Eh	GLBL_EVENT_OR_MASK	Global Bus Comparator Event OR Mask Register	EALLOW	Go
10h	GLBL_AND_EVENT_INT_MASK	Global AND Event Interrupt Mask Register	EALLOW	Go
12h	GLBL_OR_EVENT_INT_MASK	Global OR Event Interrupt Mask Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 13-5 shows the codes that are used for access types in this section.

Table 13-5. ERAD_GLOBAL_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

13.7.2.1 GLBL_EVENT_STAT Register (Offset = 0h) [Reset = 0h]

GLBL_EVENT_STAT is shown in Figure 13-5 and described in Table 13-6.

Return to the Summary Table.

This register contains one bit for each of the bus comparator modules and the counter modules that are present in a device. Each bit directly reflects the state of the EVENT_FIRED bit of the corresponding module. This facilitates software to just read one register and find out if any of the debug modules had fired.

Figure 13-5. GLBL_EVENT_STAT Register

15	14	13	12	11	10	9	8
RESERVED				CTM4	CTM3	CTM2	CTM1
R-0h				R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
HWBP8	HWBP7	HWBP6	HWBP5	HWBP4	HWBP3	HWBP2	HWBP1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 13-6. GLBL_EVENT_STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11	CTM4	R	0h	This bit directly reflects the state of the EVENT_FIRED bit of the Counter unit 4. 0 No Event 1 Event Fired Reset type: ERAD_RESET
10	CTM3	R	0h	This bit directly reflects the state of the EVENT_FIRED bit of the Counter unit 3. 0 No Event 1 Event Fired Reset type: ERAD_RESET
9	CTM2	R	0h	This bit directly reflects the state of the EVENT_FIRED bit of the Counter unit 2. 0 No Event 1 Event Fired Reset type: ERAD_RESET
8	CTM1	R	0h	This bit directly reflects the state of the EVENT_FIRED bit of the Counter unit 1. 0 No Event 1 Event Fired Reset type: ERAD_RESET
7	HWBP8	R	0h	This bit directly reflects the state of the EVENT_FIRED bit of the Enhanced Bus Comparator (EBC) unit 8. 0 No Event 1 Event Fired Reset type: ERAD_RESET
6	HWBP7	R	0h	This bit directly reflects the state of the EVENT_FIRED bit of the Enhanced Bus Comparator (EBC) unit 7. 0 No Event 1 Event Fired Reset type: ERAD_RESET
5	HWBP6	R	0h	This bit directly reflects the state of the EVENT_FIRED bit of the Enhanced Bus Comparator (EBC) unit 6. 0 No Event 1 Event Fired Reset type: ERAD_RESET

Table 13-6. GLBL_EVENT_STAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	HWBP5	R	0h	This bit directly reflects the state of the EVENT_FIRED bit of the Enhanced Bus Comparator (EBC) unit 5. 0 No Event 1 Event Fired Reset type: ERAD_RESET
3	HWBP4	R	0h	This bit directly reflects the state of the EVENT_FIRED bit of the Enhanced Bus Comparator (EBC) unit 4. 0 No Event 1 Event Fired Reset type: ERAD_RESET
2	HWBP3	R	0h	This bit directly reflects the state of the EVENT_FIRED bit of the Enhanced Bus Comparator (EBC) unit 3. 0 No Event 1 Event Fired Reset type: ERAD_RESET
1	HWBP2	R	0h	This bit directly reflects the state of the EVENT_FIRED bit of the Enhanced Bus Comparator (EBC) unit 2. 0 No Event 1 Event Fired Reset type: ERAD_RESET
0	HWBP1	R	0h	This bit directly reflects the state of the EVENT_FIRED bit of the Enhanced Bus Comparator (EBC) unit 1. 0 No Event 1 Event Fired Reset type: ERAD_RESET

13.7.2.2 GLBL_HALT_STAT Register (Offset = 2h) [Reset = 0h]

GLBL_HALT_STAT is shown in Figure 13-6 and described in Table 13-7.

Return to the Summary Table.

This register contains one bit for each of the bus comparator modules and the counter modules that are present in a device. Each bit directly reflects the state of the EVENT_FIRED status bit. This facilitates software to just read one register and find out if any of the debug modules have fired.

Figure 13-6. GLBL_HALT_STAT Register

15	14	13	12	11	10	9	8
RESERVED				CTM4	CTM3	CTM2	CTM1
R-0h				R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
HWBP8	HWBP7	HWBP6	HWBP5	HWBP4	HWBP3	HWBP2	HWBP1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 13-7. GLBL_HALT_STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11	CTM4	R	0h	This bit directly reflects the state of the completed bit of the Counter unit 4. 0 No Event 1 Event Fired Reset type: ERAD_RESET
10	CTM3	R	0h	This bit directly reflects the state of the completed bit of the Counter unit 3. 0 No Event 1 Event Fired Reset type: ERAD_RESET
9	CTM2	R	0h	This bit directly reflects the state of the completed bit of the Counter unit 2. 0 No Event 1 Event Fired Reset type: ERAD_RESET
8	CTM1	R	0h	This bit directly reflects the state of the completed bit of the Counter unit 1. 0 No Event 1 Event Fired Reset type: ERAD_RESET
7	HWBP8	R	0h	This bit directly reflects the state of the completed bit of the Enhanced Bus Comparator (EBC) unit 8. 0 Not Completed 1 Completed Reset type: ERAD_RESET
6	HWBP7	R	0h	This bit directly reflects the state of the completed bit of the Enhanced Bus Comparator (EBC) unit 7. 0 Not Completed 1 Completed Reset type: ERAD_RESET
5	HWBP6	R	0h	This bit directly reflects the state of the completed bit of the Enhanced Bus Comparator (EBC) unit 6. 0 Not Completed 1 Completed Reset type: ERAD_RESET

Table 13-7. GLBL_HALT_STAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	HWBP5	R	0h	This bit directly reflects the state of the completed bit of the Enhanced Bus Comparator (EBC) unit 5. 0 Not Completed 1 Completed Reset type: ERAD_RESET
3	HWBP4	R	0h	This bit directly reflects the state of the completed bit of the Enhanced Bus Comparator (EBC) unit 4. 0 Not Completed 1 Completed Reset type: ERAD_RESET
2	HWBP3	R	0h	This bit directly reflects the state of the completed bit of the Enhanced Bus Comparator (EBC) unit 3. 0 Not Completed 1 Completed Reset type: ERAD_RESET
1	HWBP2	R	0h	This bit directly reflects the state of the completed bit of the Enhanced Bus Comparator (EBC) unit 2. 0 Not Completed 1 Completed Reset type: ERAD_RESET
0	HWBP1	R	0h	This bit directly reflects the state of the completed bit of the Enhanced Bus Comparator (EBC) unit 1. 0 Not Completed 1 Completed Reset type: ERAD_RESET

13.7.2.3 GLBL_ENABLE Register (Offset = 4h) [Reset = 0h]

GLBL_ENABLE is shown in Figure 13-7 and described in Table 13-8.

Return to the Summary Table.

This register contains one bit for each of the bus comparator modules and the counter modules that are present in a device. Each bit directly acts as a global enable for the corresponding module. This bit has to be set to 1 for the module to be functional.

Figure 13-7. GLBL_ENABLE Register

15	14	13	12	11	10	9	8
RESERVED				CTM4	CTM3	CTM2	CTM1
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
HWBP8	HWBP7	HWBP6	HWBP5	HWBP4	HWBP3	HWBP2	HWBP1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 13-8. GLBL_ENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11	CTM4	R/W	0h	This bit directly reflects the state of the ENABLE bit of the Counter unit 4. 0 Disabled 1 Enabled Reset type: ERAD_RESET
10	CTM3	R/W	0h	This bit directly reflects the state of the ENABLE bit of the Counter unit 3. 0 Disabled 1 Enabled Reset type: ERAD_RESET
9	CTM2	R/W	0h	This bit directly reflects the state of the ENABLE bit of the Counter unit 2. 0 Disabled 1 Enabled Reset type: ERAD_RESET
8	CTM1	R/W	0h	This bit directly reflects the state of the ENABLE bit of the Counter unit 1. 0 Disabled 1 Enabled Reset type: ERAD_RESET
7	HWBP8	R/W	0h	This bit directly reflects the state of the ENABLE bit of the Enhanced Bus Comparator (EBC) unit 8. 0 Disabled 1 Enabled Reset type: ERAD_RESET
6	HWBP7	R/W	0h	This bit directly reflects the state of the ENABLE bit of the Enhanced Bus Comparator (EBC) unit 7. 0 Disabled 1 Enabled Reset type: ERAD_RESET
5	HWBP6	R/W	0h	This bit directly reflects the state of the ENABLE bit of the Enhanced Bus Comparator (EBC) unit 6. 0 Disabled 1 Enabled Reset type: ERAD_RESET

Table 13-8. GLBL_ENABLE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	HWBP5	R/W	0h	This bit directly reflects the state of the ENABLE bit of the Enhanced Bus Comparator (EBC) unit 5. 0 Disabled 1 Enabled Reset type: ERAD_RESET
3	HWBP4	R/W	0h	This bit directly reflects the state of the ENABLE bit of the Enhanced Bus Comparator (EBC) unit 4. 0 Disabled 1 Enabled Reset type: ERAD_RESET
2	HWBP3	R/W	0h	This bit directly reflects the state of the ENABLE bit of the Enhanced Bus Comparator (EBC) unit 3. 0 Disabled 1 Enabled Reset type: ERAD_RESET
1	HWBP2	R/W	0h	This bit directly reflects the state of the ENABLE bit of the Enhanced Bus Comparator (EBC) unit 2. 0 Disabled 1 Enabled Reset type: ERAD_RESET
0	HWBP1	R/W	0h	This bit directly reflects the state of the ENABLE bit of the Enhanced Bus Comparator (EBC) unit 1. 0 Disabled 1 Enabled Reset type: ERAD_RESET

13.7.2.4 GLBL_CTM_RESET Register (Offset = 6h) [Reset = 0h]

GLBL_CTM_RESET is shown in Figure 13-8 and described in Table 13-9.

Return to the Summary Table.

This register contains one bit for each of the counter modules that are present in a device. Each bit directly acts as a reset for the counters for the corresponding module.(It does not affect anything else except resetting the counter.
Example: If the counter was previously incrementing before reset, then on a reset event the counter gets reset and continues to increment again).

Figure 13-8. GLBL_CTM_RESET Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				CTM4	CTM3	CTM2	CTM1
R-0h				R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h

Table 13-9. GLBL_CTM_RESET Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	CTM4	R-0/W	0h	This bit directly resets the state the Counter unit 4. 0 No Effect 1 Reset Reset type: ERAD_RESET
2	CTM3	R-0/W	0h	This bit directly resets the state the Counter unit 3. 0 No Effect 1 Reset Reset type: ERAD_RESET
1	CTM2	R-0/W	0h	This bit directly resets the state the Counter unit 2. 0 No Effect 1 Reset Reset type: ERAD_RESET
0	CTM1	R-0/W	0h	This bit directly resets the state the Counter unit 1. 0 No Effect 1 Reset Reset type: ERAD_RESET

13.7.2.5 GLBL_NMI_CTL Register (Offset = 8h) [Reset = 0h]

GLBL_NMI_CTL is shown in Figure 13-9 and described in Table 13-10.

Return to the Summary Table.

This register contains one bit for each of the bus comparator modules anf the counter modules that can cause an NMI to the CPU. When the corresponding bit of a unit is set to 1, then if an event occurs from that module, then an NMI will be generated.

Figure 13-9. GLBL_NMI_CTL Register

15	14	13	12	11	10	9	8
RESERVED				CTM4	CTM3	CTM2	CTM1
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
HWBP8	HWBP7	HWBP6	HWBP5	HWBP4	HWBP3	HWBP2	HWBP1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 13-10. GLBL_NMI_CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11	CTM4	R/W	0h	This bit enables the genration of an NMI to the CPU by Counter unit 4 0 Do NOT Generate NMI 1 Generate NMI Reset type: ERAD_RESET
10	CTM3	R/W	0h	This bit enables the genration of an NMI to the CPU by Counter unit 3. 0 Do NOT Generate NMI 1 Generate NMI Reset type: ERAD_RESET
9	CTM2	R/W	0h	This bit enables the genration of an NMI to the CPU by Counter unit 2. 0 Do NOT Generate NMI 1 Generate NMI Reset type: ERAD_RESET
8	CTM1	R/W	0h	This bit enables the genration of an NMI to the CPU by Counter unit 1. 0 Do NOT Generate NMI 1 Generate NMI Reset type: ERAD_RESET
7	HWBP8	R/W	0h	This bit enables the genration of an NMI to the CPU by Enhanced Bus Comparator (EBC) unit 8. 0 Do NOT Generate NMI 1 Generate NMI Reset type: ERAD_RESET
6	HWBP7	R/W	0h	This bit enables the genration of an NMI to the CPU by Enhanced Bus Comparator (EBC) unit 7. 0 Do NOT Generate NMI 1 Generate NMI Reset type: ERAD_RESET
5	HWBP6	R/W	0h	This bit enables the genration of an NMI to the CPU by Enhanced Bus Comparator (EBC) unit 6. 0 Do NOT Generate NMI 1 Generate NMI Reset type: ERAD_RESET

Table 13-10. GLBL_NMI_CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	HWBP5	R/W	0h	This bit enables the genration of an NMI to the CPU by Enhanced Bus Comparator (EBC) unit 5. 0 Do NOT Generate NMI 1 Generate NMI Reset type: ERAD_RESET
3	HWBP4	R/W	0h	This bit enables the genration of an NMI to the CPU by Enhanced Bus Comparator (EBC) unit 4. 0 Do NOT Generate NMI 1 Generate NMI Reset type: ERAD_RESET
2	HWBP3	R/W	0h	This bit enables the genration of an NMI to the CPU by Enhanced Bus Comparator (EBC) unit 3. 0 Do NOT Generate NMI 1 Generate NMI Reset type: ERAD_RESET
1	HWBP2	R/W	0h	This bit enables the genration of an NMI to the CPU by Enhanced Bus Comparator (EBC) unit 2. 0 Do NOT Generate NMI 1 Generate NMI Reset type: ERAD_RESET
0	HWBP1	R/W	0h	This bit enables the genration of an NMI to the CPU by Enhanced Bus Comparator (EBC) unit 1. 0 Do NOT Generate NMI 1 Generate NMI Reset type: ERAD_RESET

13.7.2.6 GLBL_OWNER Register (Offset = Ah) [Reset = 0h]

GLBL_OWNER is shown in Figure 13-10 and described in Table 13-11.

Return to the Summary Table.

Global Ownership

Figure 13-10. GLBL_OWNER Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						OWNER	
R-0h						R/W-0h	

Table 13-11. GLBL_OWNER Register Field Descriptions

Bit	Field	Type	Reset	Description
15-2	RESERVED	R	0h	Reserved
1-0	OWNER	R/W	0h	This register determines whether Application Code or Debugger owns this module or it's kept in No Owner state where debugger or application can access the module. 00 No Owner 01 Application owned 10 Debugger owned 11 Reserved Reset type: ERAD_RESET

13.7.2.7 GLBL_EVENT_AND_MASK Register (Offset = Ch) [Reset = FFFFFFFFh]

GLBL_EVENT_AND_MASK is shown in Figure 13-11 and described in Table 13-12.

Return to the Summary Table.

Global Bus Comparator Event AND Mask Register

Figure 13-11. GLBL_EVENT_AND_MASK Register

31	30	29	28	27	26	25	24
MASK4_HWBP8	MASK4_HWBP7	MASK4_HWBP6	MASK4_HWBP5	MASK4_HWBP4	MASK4_HWBP3	MASK4_HWBP2	MASK4_HWBP1
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
23	22	21	20	19	18	17	16
MASK3_HWBP8	MASK3_HWBP7	MASK3_HWBP6	MASK3_HWBP5	MASK3_HWBP4	MASK3_HWBP3	MASK3_HWBP2	MASK3_HWBP1
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
15	14	13	12	11	10	9	8
MASK2_HWBP8	MASK2_HWBP7	MASK2_HWBP6	MASK2_HWBP5	MASK2_HWBP4	MASK2_HWBP3	MASK2_HWBP2	MASK2_HWBP1
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
7	6	5	4	3	2	1	0
MASK1_HWBP8	MASK1_HWBP7	MASK1_HWBP6	MASK1_HWBP5	MASK1_HWBP4	MASK1_HWBP3	MASK1_HWBP2	MASK1_HWBP1
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h

Table 13-12. GLBL_EVENT_AND_MASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MASK4_HWBP8	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 8: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND4 output Reset type: ERAD_RESET
30	MASK4_HWBP7	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 7: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND4 output Reset type: ERAD_RESET
29	MASK4_HWBP6	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 6: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND4 output Reset type: ERAD_RESET
28	MASK4_HWBP5	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 5: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND4 output Reset type: ERAD_RESET
27	MASK4_HWBP4	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 4: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND4 output Reset type: ERAD_RESET

Table 13-12. GLBL_EVENT_AND_MASK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MASK4_HWBP3	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 3: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND4 output Reset type: ERAD_RESET
25	MASK4_HWBP2	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 2: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND4 output Reset type: ERAD_RESET
24	MASK4_HWBP1	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 1: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND4 output Reset type: ERAD_RESET
23	MASK3_HWBP8	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 8: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND3 output Reset type: ERAD_RESET
22	MASK3_HWBP7	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 7: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND3 output Reset type: ERAD_RESET
21	MASK3_HWBP6	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 6: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND3 output Reset type: ERAD_RESET
20	MASK3_HWBP5	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 5: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND3 output Reset type: ERAD_RESET
19	MASK3_HWBP4	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 4: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND3 output Reset type: ERAD_RESET
18	MASK3_HWBP3	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 3: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND3 output Reset type: ERAD_RESET
17	MASK3_HWBP2	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 2: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND3 output Reset type: ERAD_RESET

Table 13-12. GLBL_EVENT_AND_MASK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	MASK3_HWBP1	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 1: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND3 output Reset type: ERAD_RESET
15	MASK2_HWBP8	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 8: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND2 output Reset type: ERAD_RESET
14	MASK2_HWBP7	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 7: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND2 output Reset type: ERAD_RESET
13	MASK2_HWBP6	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 6: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND2 output Reset type: ERAD_RESET
12	MASK2_HWBP5	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 5: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND2 output Reset type: ERAD_RESET
11	MASK2_HWBP4	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 4: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND2 output Reset type: ERAD_RESET
10	MASK2_HWBP3	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 3: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND2 output Reset type: ERAD_RESET
9	MASK2_HWBP2	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 2: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND2 output Reset type: ERAD_RESET
8	MASK2_HWBP1	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 1: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND2 output Reset type: ERAD_RESET
7	MASK1_HWBP8	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 8: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND1 output Reset type: ERAD_RESET

Table 13-12. GLBL_EVENT_AND_MASK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	MASK1_HWBP7	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 7: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND1 output Reset type: ERAD_RESET
5	MASK1_HWBP6	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 6: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND1 output Reset type: ERAD_RESET
4	MASK1_HWBP5	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 5: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND1 output Reset type: ERAD_RESET
3	MASK1_HWBP4	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 4: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND1 output Reset type: ERAD_RESET
2	MASK1_HWBP3	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 3: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND1 output Reset type: ERAD_RESET
1	MASK1_HWBP2	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 2: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND1 output Reset type: ERAD_RESET
0	MASK1_HWBP1	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 1: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_AND1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_AND1 output Reset type: ERAD_RESET

13.7.2.8 GLBL_EVENT_OR_MASK Register (Offset = Eh) [Reset = FFFFFFFFh]

GLBL_EVENT_OR_MASK is shown in Figure 13-12 and described in Table 13-13.

Return to the Summary Table.

Global Bus Comparator Event OR Mask Register

Figure 13-12. GLBL_EVENT_OR_MASK Register

31	30	29	28	27	26	25	24
MASK4_HWBP8	MASK4_HWBP7	MASK4_HWBP6	MASK4_HWBP5	MASK4_HWBP4	MASK4_HWBP3	MASK4_HWBP2	MASK4_HWBP1
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
23	22	21	20	19	18	17	16
MASK3_HWBP8	MASK3_HWBP7	MASK3_HWBP6	MASK3_HWBP5	MASK3_HWBP4	MASK3_HWBP3	MASK3_HWBP2	MASK3_HWBP1
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
15	14	13	12	11	10	9	8
MASK2_HWBP8	MASK2_HWBP7	MASK2_HWBP6	MASK2_HWBP5	MASK2_HWBP4	MASK2_HWBP3	MASK2_HWBP2	MASK2_HWBP1
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h
7	6	5	4	3	2	1	0
MASK1_HWBP8	MASK1_HWBP7	MASK1_HWBP6	MASK1_HWBP5	MASK1_HWBP4	MASK1_HWBP3	MASK1_HWBP2	MASK1_HWBP1
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h

Table 13-13. GLBL_EVENT_OR_MASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MASK4_HWBP8	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 8: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR4 output Reset type: ERAD_RESET
30	MASK4_HWBP7	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 7: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR4 output Reset type: ERAD_RESET
29	MASK4_HWBP6	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 6: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR4 output Reset type: ERAD_RESET
28	MASK4_HWBP5	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 5: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR4 output Reset type: ERAD_RESET
27	MASK4_HWBP4	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 4: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR4 output Reset type: ERAD_RESET

Table 13-13. GLBL_EVENT_OR_MASK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	MASK4_HWBP3	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 3: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR4 output Reset type: ERAD_RESET
25	MASK4_HWBP2	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 2: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR4 output Reset type: ERAD_RESET
24	MASK4_HWBP1	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 1: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR4 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR4 output Reset type: ERAD_RESET
23	MASK3_HWBP8	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 8: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR3 output Reset type: ERAD_RESET
22	MASK3_HWBP7	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 7: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR3 output Reset type: ERAD_RESET
21	MASK3_HWBP6	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 6: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR3 output Reset type: ERAD_RESET
20	MASK3_HWBP5	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 5: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR3 output Reset type: ERAD_RESET
19	MASK3_HWBP4	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 4: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR3 output Reset type: ERAD_RESET
18	MASK3_HWBP3	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 3: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR3 output Reset type: ERAD_RESET
17	MASK3_HWBP2	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 2: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR3 output Reset type: ERAD_RESET

Table 13-13. GLBL_EVENT_OR_MASK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	MASK3_HWBP1	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 1: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR3 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR3 output Reset type: ERAD_RESET
15	MASK2_HWBP8	R/W	1h	AND event mask for Enhanced Bus Comparator (EBC) unit 8: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR2 output Reset type: ERAD_RESET
14	MASK2_HWBP7	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 7: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR2 output Reset type: ERAD_RESET
13	MASK2_HWBP6	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 6: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR2 output Reset type: ERAD_RESET
12	MASK2_HWBP5	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 5: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR2 output Reset type: ERAD_RESET
11	MASK2_HWBP4	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 4: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR2 output Reset type: ERAD_RESET
10	MASK2_HWBP3	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 3: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR2 output Reset type: ERAD_RESET
9	MASK2_HWBP2	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 2: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR2 output Reset type: ERAD_RESET
8	MASK2_HWBP1	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 1: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR2 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR2 output Reset type: ERAD_RESET
7	MASK1_HWBP8	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 8: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR1 output Reset type: ERAD_RESET

Table 13-13. GLBL_EVENT_OR_MASK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	MASK1_HWBP7	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 7: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR1 output Reset type: ERAD_RESET
5	MASK1_HWBP6	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 6: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR1 output Reset type: ERAD_RESET
4	MASK1_HWBP5	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 5: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR1 output Reset type: ERAD_RESET
3	MASK1_HWBP4	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 4: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR1 output Reset type: ERAD_RESET
2	MASK1_HWBP3	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 3: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR1 output Reset type: ERAD_RESET
1	MASK1_HWBP2	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 2: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR1 output Reset type: ERAD_RESET
0	MASK1_HWBP1	R/W	1h	OR event mask for Enhanced Bus Comparator (EBC) unit 1: 0 Corresponding HWBP_EVENT is enabled for HWBP_EVENT_OR1 output 1 Corresponding HWBP_EVENT is masked for HWBP_EVENT_OR1 output Reset type: ERAD_RESET

13.7.2.9 GLBL_AND_EVENT_INT_MASK Register (Offset = 10h) [Reset = 0h]

GLBL_AND_EVENT_INT_MASK is shown in Figure 13-13 and described in Table 13-14.

Return to the Summary Table.

Global AND Event Interrupt Mask Register

Figure 13-13. GLBL_AND_EVENT_INT_MASK Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RTOSINT_MASK4	RTOSINT_MASK3	RTOSINT_MASK2	RTOSINT_MASK1
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 13-14. GLBL_AND_EVENT_INT_MASK Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	RTOSINT_MASK4	R/W	0h	RTOSINT generation mask for global AND events MASK4: 0 Corresponding GLOBAL_EVENT_AND is enabled for RTOSINT generation 1 Corresponding GLOBAL_EVENT_AND is disabled for RTOSINT generation Reset type: ERAD_RESET
2	RTOSINT_MASK3	R/W	0h	RTOSINT generation mask for global AND events MASK3: 0 Corresponding GLOBAL_EVENT_AND is enabled for RTOSINT generation 1 Corresponding GLOBAL_EVENT_AND is disabled for RTOSINT generation Reset type: ERAD_RESET
1	RTOSINT_MASK2	R/W	0h	RTOSINT generation mask for global AND events MASK2: 0 Corresponding GLOBAL_EVENT_AND is enabled for RTOSINT generation 1 Corresponding GLOBAL_EVENT_AND is disabled for RTOSINT generation Reset type: ERAD_RESET
0	RTOSINT_MASK1	R/W	0h	RTOSINT generation mask for global AND events MASK1: 0 Corresponding GLOBAL_EVENT_AND is enabled for RTOSINT generation 1 Corresponding GLOBAL_EVENT_AND is disabled for RTOSINT generation Reset type: ERAD_RESET

13.7.2.10 GLBL_OR_EVENT_INT_MASK Register (Offset = 12h) [Reset = 0h]

GLBL_OR_EVENT_INT_MASK is shown in Figure 13-14 and described in Table 13-15.

Return to the Summary Table.

Global OR Event Interrupt Mask Register

Figure 13-14. GLBL_OR_EVENT_INT_MASK Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RTOSINT_MASK4	RTOSINT_MASK3	RTOSINT_MASK2	RTOSINT_MASK1
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 13-15. GLBL_OR_EVENT_INT_MASK Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	RTOSINT_MASK4	R/W	0h	RTOSINT generation mask for global OR events MASK3: 0 Corresponding GLOBAL_EVENT_OR is enabled for RTOSINT generation 1 Corresponding GLOBAL_EVENT_OR is disabled for RTOSINT generation Reset type: ERAD_RESET
2	RTOSINT_MASK3	R/W	0h	RTOSINT generation mask for global OR events MASK2: 0 Corresponding GLOBAL_EVENT_OR is enabled for RTOSINT generation 1 Corresponding GLOBAL_EVENT_OR is disabled for RTOSINT generation Reset type: ERAD_RESET
1	RTOSINT_MASK2	R/W	0h	RTOSINT generation mask for global OR events MASK2: 0 Corresponding GLOBAL_EVENT_OR is enabled for RTOSINT generation 1 Corresponding GLOBAL_EVENT_OR is disabled for RTOSINT generation Reset type: ERAD_RESET
0	RTOSINT_MASK1	R/W	0h	RTOSINT generation mask for global OR events MASK1: 0 Corresponding GLOBAL_EVENT_OR is enabled for RTOSINT generation 1 Corresponding GLOBAL_EVENT_OR is disabled for RTOSINT generation Reset type: ERAD_RESET

13.7.3 ERAD_HWBP_REGS Registers

Table 13-16 lists the memory-mapped registers for the ERAD_HWBP_REGS registers. All register offset addresses not listed in Table 13-16 should be considered as reserved locations and the register contents should not be modified.

Table 13-16. ERAD_HWBP_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	HWBP_MASK	HWBP (EBC) Mask Register	EALLOW	Go
2h	HWBP_REF	HWBP (EBC) Reference Register	EALLOW	Go
4h	HWBP_CLEAR	HWBP (EBC) Clear Register	EALLOW	Go
6h	HWBP_CNTL	HWBP (EBC) Control Register	EALLOW	Go
7h	HWBP_STATUS	HWBP (EBC) Status Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 13-17 shows the codes that are used for access types in this section.

Table 13-17. ERAD_HWBP_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

13.7.3.1 HWBP_MASK Register (Offset = 0h) [Reset = 0h]

HWBP_MASK is shown in Figure 13-15 and described in Table 13-18.

Return to the Summary Table.

HWBP (EBC) Mask Register

Figure 13-15. HWBP_MASK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MASK																															
R/W-0h																															

Table 13-18. HWBP_MASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MASK	R/W	0h	This register contains address mask for comparison. The contents of this register are used along with the reference register to determine the address match. The equation used to determine a match is as follows. Match is true if, (address | mask) == (ref | mask) This register is writable by CPU only if application owns the unit and if EALLOW is set. Otherwise, the writes are ignored. The register is writable by the debugger only if the debugger owns this unit. Otherwise, the writes are ignored. Reset type: ERAD_RESET

13.7.3.2 HWBP_REF Register (Offset = 2h) [Reset = 0h]

HWBP_REF is shown in Figure 13-16 and described in Table 13-19.

Return to the Summary Table.

HWBP (EBC) Reference Register

Figure 13-16. HWBP_REF Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REF																															
R/W-0h																															

Table 13-19. HWBP_REF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REF	R/W	0h	This register contains the reference address for comparison. The contents of this register are used along with the mask register to determine the address match. The equation used to determine a match is as follows. Match is true if, (address | mask) == (ref | mask) This register is writable by CPU only if application owns the unit and if EALLOW is set. Otherwise, the writes are ignored. The register is writable by the debugger only if the debugger owns this unit. Otherwise, the writes are ignored. Reset type: ERAD_RESET

13.7.3.3 HWBP_CLEAR Register (Offset = 4h) [Reset = 0h]

HWBP_CLEAR is shown in Figure 13-17 and described in Table 13-20.

Return to the Summary Table.

HWBP (EBC) Clear Register

Figure 13-17. HWBP_CLEAR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EVENT_CLR
R-0h							R-0/W-0h

Table 13-20. HWBP_CLEAR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R	0h	Reserved
0	EVENT_CLR	R-0/W	0h	Event Clear register: 0 No action. 1 A write with this bit set to 1 will clear the sticky EVENT_FIRED bit in the HWBP_STATUS register and bring the Breakpoint Module statemachine status back to IDLE. Reads of this bit position will always return a 0. Reset type: ERAD_RESET

13.7.3.4 HWBP_CNTL Register (Offset = 6h) [Reset = 0h]

HWBP_CNTL is shown in Figure 13-18 and described in Table 13-21.

Return to the Summary Table.

HWBP (EBC) Control Register

Figure 13-18. HWBP_CNTL Register

15	14	13	12	11	10	9	8
RESERVED				RESERVED	RESERVED	COMP_MODE	
R-0h				R-0h	R-0h	R/W-0h	
7	6	5	4	3	2	1	0
COMP_MODE	RTOSINT	STOP	BUS_SEL				RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h				R-0h

Table 13-21. HWBP_CNTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11	RESERVED	R	0h	Reserved
10	RESERVED	R	0h	Reserved
9-7	COMP_MODE	R/W	0h	Enhanced Bus Comparator (EBC) compare modes: 000 Regular masked compare HWBP_MSK will be ignored for the following modes: 100 Bus value GT HWBP_REF 101 Bus value GE HWBP_REF 110 Bus value LT HWBP_REF 111 Bus value LE HWBP_REF GT means Greater Than GE means Greater or Equal LT means Less Than LE means Lesser or Equal Reset type: ERAD_RESET
6	RTOSINT	R/W	0h	This bit decides whether the Enhanced Bus Comparator (EBC) unit will generate RTOSINTn interrupt when event matches occur. Note that the event outputs will always be generated regardless of the state of this bit. 0 The Enhanced Bus Comparator (EBC) unit will not cause any action towards the CPU. 1 The Enhanced Bus Comparator (EBC) unit will assert RTOSINTn for matching data accesses and trace tags for matching program fetches. Reset type: ERAD_RESET
5	STOP	R/W	0h	This bit decides whether the Enhanced Bus Comparator (EBC) unit will generate CPU halting signals when event matches occur. Note that the event outputs will always be generated regardless of the state of this bit. 0 The Enhanced Bus Comparator (EBC) unit will not cause any action towards halting the CPU. 1 The Enhanced Bus Comparator (EBC) unit will assert ANASTOP for matching data accesses and break tags for matching program fetches. These can cause the CPU to HALT Reset type: ERAD_RESET

Table 13-21. HWBP_CNTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-1	BUS_SEL	R/W	0h	These bits are used to select which CPU buses will be used for comparison to generate the match events. For each bus selected, the corresponding strobes will automatically be selected to determine valid accesses. 0000 PAB for instruction fetches 0001 VPC 0010 DWAB for data write accesses 0011 DRAB for data read accesses 0100 DWDB for write data match 0101 DRDB for read data match 0110 VPC Instruction aligned match 0111 VPC R1 aligned match 1000 VPC R2 aligned match 1001 VPC W aligned match All other combinations are RESERVED. Reset type: ERAD_RESET
0	RESERVED	R	0h	Reserved

13.7.3.5 HWBP_STATUS Register (Offset = 7h) [Reset = 400h]

HWBP_STATUS is shown in Figure 13-19 and described in Table 13-22.

Return to the Summary Table.

HWBP (EBC) Status Register

Figure 13-19. HWBP_STATUS Register

15	14	13	12	11	10	9	8
STATUS		MODULE_ID					
R-0h		R-4h					
7	6	5	4	3	2	1	0
RESERVED							EVENT_FIRED
R-0h							R-0h

Table 13-22. HWBP_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	STATUS	R	0h	Bus comaparator status: 00 Idle 10 Enabled 11 Completed Reset type: ERAD_RESET
13-8	MODULE_ID	R	4h	These bits are always a constant representing a unique identification for the Enhanced Bus Comparator (EBC) unit. Reset type: ERAD_RESET
7-1	RESERVED	R	0h	Reserved
0	EVENT_FIRED	R	0h	This is a sticky bit which gets set every time the HWBP (EBC) unit generates a match event. This will be used by software to figure out whether this HWBP module fired an event or not. This bit will get cleared by writing a '1' to bit 0 of the HWBP_CLEAR register. Reset type: ERAD_RESET

13.7.4 ERAD_COUNTER_REGS Registers

Table 13-23 lists the memory-mapped registers for the ERAD_COUNTER_REGS registers. All register offset addresses not listed in Table 13-23 should be considered as reserved locations and the register contents should not be modified.

Table 13-23. ERAD_COUNTER_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CTM_CNTL	Counter Control Register	EALLOW	Go
1h	CTM_STATUS	Counter Status Register	EALLOW	Go
2h	CTM_REF	Counter Reference Register	EALLOW	Go
4h	CTM_COUNT	Counter Current Value Register	EALLOW	Go
6h	CTM_MAX_COUNT	Counter Max Count Value Register	EALLOW	Go
8h	CTM_INPUT_SEL	Counter Input Select Register	EALLOW	Go
9h	CTM_CLEAR	Counter Clear Register	EALLOW	Go
Ah	CTM_INPUT_SEL_2	Counter Input Select Extension Register	EALLOW	Go
Bh	CTM_INPUT_COND	Counter Input Conditioning Register		Go

Complex bit access types are encoded to fit into small table cells. Table 13-24 shows the codes that are used for access types in this section.

Table 13-24. ERAD_COUNTER_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

13.7.4.1 CTM_CNTL Register (Offset = 0h) [Reset = 0h]

CTM_CNTL is shown in Figure 13-20 and described in Table 13-25.

Return to the Summary Table.

Counter Control Register

Figure 13-20. CTM_CNTL Register

15	14	13	12	11	10	9	8
RESERVED				CNT_INP_SEL_EN	RST_EN	RESERVED	START_STOP_CUMULATIVE
R-0h				R/W-0h	R/W-0h	R-0h	R/W-0h
7	6	5	4	3	2	1	0
RTOSINT	STOP	RESERVED	RST_ON_MATCH	EVENT_MODE	START_STOP_MODE	RESERVED	
R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	

Table 13-25. CTM_CNTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11	CNT_INP_SEL_EN	R/W	0h	0 = Disable using the input_select register for the count input. The counter will always count CPU cycles. 1 = Enable using the input_select register for the count input. The counter will count the event selected by the count input register. Reset type: ERAD_RESET
10	RST_EN	R/W	0h	This bit decides if the reset input is enabled or not. Setting this to 1 will cause the counter to reset to zero whenever the selected reset input goes active high. No event will be generated when the counter is reset. Setting this bit to 0 will cause the counter to ignore the reset inputs. Reset type: ERAD_RESET
9	RESERVED	R	0h	Reserved
8	START_STOP_CUMULATIVE	R/W	0h	This bit decides whether the counter counts to give the cumulative cycle count for 'n' number of successive start stop events or clears the counter on very stop event to record the MAX_COUNT across successive start stop sequences. 0 When in START_STOP mode counter gets cleared on every stop event and MAX_COUNT records the max value 1 When in START_STOP mode counter keeps counting between successive start stop events to generate a cumulative count w/o clearing the counter on any stop events. MAX_COUNT register is invalid when this bit is set. Reset type: ERAD_RESET
7	RTOSINT	R/W	0h	This bit decides whether the counter module will generate RTOSINTn interrupt when count value matches the reference. Note that the event outputs will always be generated regardless of the state of this bit. 0 The counter unit will not cause any action towards the CPU. 1 The counter unit will assert RTOSINTn when the count value matches the reference value. Reset type: ERAD_RESET
6	STOP	R/W	0h	This bit decides whether the counter module will generate a watchpoint to the CPU when the count value matches the reference. Note that the event outputs will always be generated regardless of the state of this bit. 0 The counter unit will not generate a watchpoint. 1 The counter unit will assert ANASTOP when the count value matches the reference. Reset type: ERAD_RESET

Table 13-25. CTM_CNTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	RESERVED	R	0h	Reserved
4	RST_ON_MATCH	R/W	0h	This bit is used to decide whether the counter will reset to zero once it reaches the reference value. 0 Counter will stay at the reference value and the counter will go to COMPLETED state and further counting will be stopped. 1 The counter will reset to zero once it reaches the match value and will stay enabled. Reset type: ERAD_RESET
3	EVENT_MODE	R/W	0h	This bit is used to decide whether the counter will count the level of the event or the edge of the event. 0 Counter will increment the count as long as the count input is active high. 1 The counter will count only on the rising edge of the count input. Reset type: ERAD_RESET
2	START_STOP_MODE	R/W	0h	This bit is used to decide whether the counter will count in the START_STOP mode or not. 0 Normal count mode. The counter will not depend on the START and STOP events 1 This is the START-STOP mode of the counter. The counter will start counting only after the START input has been asserted. It will continue to count the selected event till the STOP event is seen. Reset type: ERAD_RESET
1-0	RESERVED	R	0h	Reserved

13.7.4.2 CTM_STATUS Register (Offset = 1h) [Reset = 10h]

CTM_STATUS is shown in Figure 13-21 and described in Table 13-26.

Return to the Summary Table.

Counter Status Register

Figure 13-21. CTM_STATUS Register

15	14	13	12	11	10	9	8
STATUS				MODULE_ID			
R-0h				R-4h			
7	6	5	4	3	2	1	0
MODULE_ID						OVERFLOW	EVENT_FIRED
R-4h						R-0h	R-0h

Table 13-26. CTM_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	STATUS	R	0h	Counter unit status, 00 Idle 10 Enabled 11 Completed Reset type: ERAD_RESET
11-2	MODULE_ID	R	4h	These bits are always a constant representing a unique identification for the trigger unit. Reset type: ERAD_RESET
1	OVERFLOW	R	0h	This is a sticky bit which gets set every time the counter overflows and wraps around after reaching 0xffffffff.This bit will get cleared by writing a '1' to bit 9 of the CTM_CNTL register. Reset type: ERAD_RESET
0	EVENT_FIRED	R	0h	This is a sticky bit which gets set every time the CTM unit generates a match event. This will be used by software to figure out whether this CTM module fired an event or not. This bit will get cleared by writing a '1' to bit 9 of the CTM_CNTL register. Reset type: ERAD_RESET

13.7.4.3 CTM_REF Register (Offset = 2h) [Reset = 0h]

CTM_REF is shown in Figure 13-22 and described in Table 13-27.

Return to the Summary Table.

Counter Reference Register

Figure 13-22. CTM_REF Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REF																															
R/W-0h																															

Table 13-27. CTM_REF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REF	R/W	0h	This register contains the counter reference value for comparison. The counter will generate an event if the count value matches the reference register (considering both upper and lower half of the register). This register is writable by CPU only if application owns the unit and if EALLOW is set. Otherwise, the writes are ignored. The register is writable by the debugger only if the debugger owns this unit. Otherwise, the writes are ignored. Refrence match is enabled only when a non zero value is programmed on one of the REF register. Reset type: ERAD_RESET

13.7.4.4 CTM_COUNT Register (Offset = 4h) [Reset = 0h]

CTM_COUNT is shown in Figure 13-23 and described in Table 13-28.

Return to the Summary Table.

Counter Current Value Register

Figure 13-23. CTM_COUNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
COUNT																															
R/W-0h																															

Table 13-28. CTM_COUNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	COUNT	R/W	0h	This register contains the current count value. The counter will generate an event if the count value matches the reference register (considering both upper and lower half of the register). This register is writable by CPU only if application owns the unit and if EALLOW is set. Otherwise, the writes are ignored. The register is writable by the debugger only if the debugger owns this unit. Otherwise, the writes are ignored. Reset type: ERAD_RESET

13.7.4.5 CTM_MAX_COUNT Register (Offset = 6h) [Reset = 0h]

CTM_MAX_COUNT is shown in Figure 13-24 and described in Table 13-29.

Return to the Summary Table.

Counter Max Count Value Register

Figure 13-24. CTM_MAX_COUNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAX_COUNT																															
R/W-0h																															

Table 13-29. CTM_MAX_COUNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MAX_COUNT	R/W	0h	This register contains the maximum recorded counter value. This is relevant only in the Start Stop mode of operation. This register is writable by CPU only if application owns the unit and if EALLOW is set. Otherwise, the writes are ignored. The register is writable by the debugger only if the debugger owns this unit. Otherwise, the writes are ignored. Reset type: ERAD_RESET

13.7.4.6 CTM_INPUT_SEL Register (Offset = 8h) [Reset = 0h]

CTM_INPUT_SEL is shown in Figure 13-25 and described in Table 13-30.

Return to the Summary Table.

Counter Input Select Register

Figure 13-25. CTM_INPUT_SEL Register

15	14	13	12	11	10	9	8
RESERVED	STA_INP_SEL						
R-0h	R/W-0h						
7	6	5	4	3	2	1	0
RESERVED	CNT_INP_SEL						
R-0h	R/W-0h						

Table 13-30. CTM_INPUT_SEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14-8	STA_INP_SEL	R/W	0h	These bits decide which of the inputs will be selected as the START event for the counter. These inputs will be hooked up to the event outputs from the breakpoint module, counter module and to other system events. The usage of these bits are relevant only in the START_STOP mode of counting. Reset type: ERAD_RESET
7	RESERVED	R	0h	Reserved
6-0	CNT_INP_SEL	R/W	0h	These bits decide which of the inputs will be selected to enable counting. These inputs will be hooked up to the event outputs from the breakpoint module, counter module and to other system events Reset type: ERAD_RESET

13.7.4.7 CTM_CLEAR Register (Offset = 9h) [Reset = 0h]

CTM_CLEAR is shown in Figure 13-26 and described in Table 13-31.

Return to the Summary Table.

Counter Clear Register

Figure 13-26. CTM_CLEAR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						OVERFLOW_CLEAR	EVENT_CLEAR
R-0h						R-0/W-0h	R-0/W-0h

Table 13-31. CTM_CLEAR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-2	RESERVED	R	0h	Reserved
1	OVERFLOW_CLEAR	R-0/W	0h	Clear OVERFLOW: 0 No action. 1 A write with this bit set to 1 will clear the sticky OVERFLOW bit in the CTM_STATUS register. Reads of this bit position will always return a 0. Reset type: ERAD_RESET
0	EVENT_CLEAR	R-0/W	0h	Clear EVENT_FIRED: 0 No action. 1 A write with this bit set to 1 will clear the sticky EVENT_FIRED bit in the CTM_STATUS register and bring the Breakpoint Module statemachine status back to IDLE. Reads of this bit position will always return a 0. Reset type: ERAD_RESET

13.7.4.8 CTM_INPUT_SEL_2 Register (Offset = Ah) [Reset = 0h]

CTM_INPUT_SEL_2 is shown in Figure 13-27 and described in Table 13-32.

Return to the Summary Table.

Counter Input Select Extension Register

Figure 13-27. CTM_INPUT_SEL_2 Register

15	14	13	12	11	10	9	8
RESERVED	RST_INP_SEL						
R-0h	R/W-0h						
7	6	5	4	3	2	1	0
RESERVED	STO_INP_SEL						
R-0h	R/W-0h						

Table 13-32. CTM_INPUT_SEL_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14-8	RST_INP_SEL	R/W	0h	These bits decide are used to select the event input that will be used as the reset input. These bits matter only if the Enable Reset bit is set to 1. Reset type: ERAD_RESET
7	RESERVED	R	0h	Reserved
6-0	STO_INP_SEL	R/W	0h	These bits decide which of the inputs will be selected as the STOP event for the counter. These inputs will be hooked up to the event outputs from the breakpoint module, counter module and to other system events. The usage of these bits are relevant only in the START_STOP mode of counting. Reset type: ERAD_RESET

13.7.4.9 CTM_INPUT_COND Register (Offset = Bh) [Reset = 0h]

CTM_INPUT_COND is shown in Figure 13-28 and described in Table 13-33.

Return to the Summary Table.

Counter Input Conditioning Register

Figure 13-28. CTM_INPUT_COND Register

15	14	13	12	11	10	9	8
RESERVED		RST_INP_SYNCH	RST_INP_INV	RESERVED		STO_INP_SYNCH	STO_INP_INV
R-0h		R/W-0h	R/W-0h	R-0h		R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED		STA_INP_SYNCH	STA_INP_INV	RESERVED		CTM_INP_SYNCH	CTM_INP_INV
R-0h		R/W-0h	R/W-0h	R-0h		R/W-0h	R/W-0h

Table 13-33. CTM_INPUT_COND Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	RESERVED	R	0h	Reserved
13	RST_INP_SYNCH	R/W	0h	Enable the 2-stage synchronizer for the selected Reset input Reset type: ERAD_RESET
12	RST_INP_INV	R/W	0h	Invert the Selected Reset input Reset type: ERAD_RESET
11-10	RESERVED	R	0h	Reserved
9	STO_INP_SYNCH	R/W	0h	Enable the 2-stage synchronizer for the selected Stop input Reset type: ERAD_RESET
8	STO_INP_INV	R/W	0h	Invert the Selected Stop input Reset type: ERAD_RESET
7-6	RESERVED	R	0h	Reserved
5	STA_INP_SYNCH	R/W	0h	Enable the 2-stage synchronizer for the selected Start input Reset type: ERAD_RESET
4	STA_INP_INV	R/W	0h	Invert the Selected Start input Reset type: ERAD_RESET
3-2	RESERVED	R	0h	Reserved
1	CTM_INP_SYNCH	R/W	0h	Enable the 2-stage synchronizer for the selected Counter input Reset type: ERAD_RESET
0	CTM_INP_INV	R/W	0h	Invert the Selected Counter input Reset type: ERAD_RESET

13.7.5 ERAD_CRC_GLOBAL_REGS Registers

Table 13-34 lists the memory-mapped registers for the ERAD_CRC_GLOBAL_REGS registers. All register offset addresses not listed in Table 13-34 should be considered as reserved locations and the register contents should not be modified.

Table 13-34. ERAD_CRC_GLOBAL_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CRC_GLOBAL_CTRL	CRC_GLOBAL_CRTL		Go

Complex bit access types are encoded to fit into small table cells. Table 13-35 shows the codes that are used for access types in this section.

Table 13-35. ERAD_CRC_GLOBAL_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

13.7.5.1 CRC_GLOBAL_CTRL Register (Offset = 0h) [Reset = 0h]

CRC_GLOBAL_CTRL is shown in Figure 13-29 and described in Table 13-36.

Return to the Summary Table.

CRC Global Control Register

Figure 13-29. CRC_GLOBAL_CTRL Register

15	14	13	12	11	10	9	8
CRC8_EN	CRC7_EN	CRC6_EN	CRC5_EN	CRC4_EN	CRC3_EN	CRC2_EN	CRC1_EN
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CRC8_INIT	CRC7_INIT	CRC6_INIT	CRC5_INIT	CRC4_INIT	CRC3_INIT	CRC2_INIT	CRC1_INIT
R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h	R-0/W-0h

Table 13-36. CRC_GLOBAL_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	CRC8_EN	R/W	0h	0 = No action. 1 = Corresponding CRC Module is enabled and a valid event on this interface starts CRC computation Reset type: SYSRSn
14	CRC7_EN	R/W	0h	0 = No action. 1 = Corresponding CRC Module is enabled and a valid event on this interface starts CRC computation Reset type: SYSRSn
13	CRC6_EN	R/W	0h	0 = No action. 1 = Corresponding CRC Module is enabled and a valid event on this interface starts CRC computation Reset type: SYSRSn
12	CRC5_EN	R/W	0h	0 = No action. 1 = Corresponding CRC Module is enabled and a valid event on this interface starts CRC computation Reset type: SYSRSn
11	CRC4_EN	R/W	0h	0 = No action. 1 = Corresponding CRC Module is enabled and a valid event on this interface starts CRC computation Reset type: SYSRSn
10	CRC3_EN	R/W	0h	0 = No action. 1 = Corresponding CRC Module is enabled and a valid event on this interface starts CRC computation Reset type: SYSRSn
9	CRC2_EN	R/W	0h	0 = No action. 1 = Corresponding CRC Module is enabled and a valid event on this interface starts CRC computation Reset type: SYSRSn
8	CRC1_EN	R/W	0h	0 = No action. 1 = Corresponding CRC Module is enabled and a valid event on this interface starts CRC computation Reset type: SYSRSn
7	CRC8_INIT	R-0/W	0h	0 = No action. 1 = Initialize the CRC Module (Module registers/statemachine gets cleared for a fresh start) Reads of this bit position always returns zero Reset type: SYSRSn
6	CRC7_INIT	R-0/W	0h	0 = No action. 1 = Initialize the CRC Module (Module registers/statemachine gets cleared for a fresh start) Reads of this bit position always returns zero Reset type: SYSRSn

Table 13-36. CRC_GLOBAL_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	CRC6_INIT	R-0/W	0h	0 = No action. 1 = Initialize the CRC Module (Module registers/statemachine gets cleared for a fresh start) Reads of this bit position always returns zero Reset type: SYSRSn
4	CRC5_INIT	R-0/W	0h	0 = No action. 1 = Initialize the CRC Module (Module registers/statemachine gets cleared for a fresh start) Reads of this bit position always returns zero Reset type: SYSRSn
3	CRC4_INIT	R-0/W	0h	0 = No action. 1 = Initialize the CRC Module (Module registers/statemachine gets cleared for a fresh start) Reads of this bit position always returns zero Reset type: SYSRSn
2	CRC3_INIT	R-0/W	0h	0 = No action. 1 = Initialize the CRC Module (Module registers/statemachine gets cleared for a fresh start) Reads of this bit position always returns zero Reset type: SYSRSn
1	CRC2_INIT	R-0/W	0h	0 = No action. 1 = Initialize the CRC Module (Module registers/statemachine gets cleared for a fresh start) Reads of this bit position always returns zero Reset type: SYSRSn
0	CRC1_INIT	R-0/W	0h	0 = No action. 1 = Initialize the CRC Module (Module registers/statemachine gets cleared for a fresh start) Reads of this bit position always returns zero Reset type: SYSRSn

13.7.6 ERAD_CRC_REGS Registers

Table 13-37 lists the memory-mapped registers for the ERAD_CRC_REGS registers. All register offset addresses not listed in Table 13-37 should be considered as reserved locations and the register contents should not be modified.

Table 13-37. ERAD_CRC_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CRC_CURRENT	CRC_CURRENT		Go
2h	CRC_SEED	CRC SEED value		Go
4h	CRC_QUALIFIER	CRC_QUALIFIER		Go

Complex bit access types are encoded to fit into small table cells. Table 13-38 shows the codes that are used for access types in this section.

Table 13-38. ERAD_CRC_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

13.7.6.1 CRC_CURRENT Register (Offset = 0h) [Reset = 0h]

CRC_CURRENT is shown in Figure 13-30 and described in Table 13-39.

Return to the Summary Table.

Current computed CRC value

Figure 13-30. CRC_CURRENT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC_CURRENT																															
R-0h																															

Table 13-39. CRC_CURRENT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CRC_CURRENT	R	0h	Reads the current CRC value Reset type: SYSRSn

13.7.6.2 CRC_SEED Register (Offset = 2h) [Reset = 0h]

CRC_SEED is shown in Figure 13-31 and described in Table 13-40.

Return to the Summary Table.

CRC SEED value

Figure 13-31. CRC_SEED Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CRC_SEED																															
R/W-0h																															

Table 13-40. CRC_SEED Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CRC_SEED	R/W	0h	CRC Seed Register Reset type: SYSRSn

13.7.6.3 CRC_QUALIFIER Register (Offset = 4h) [Reset = 0h]

CRC_QUALIFIER is shown in Figure 13-32 and described in Table 13-41.

Return to the Summary Table.

CRC compute enable register

Figure 13-32. CRC_QUALIFIER Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			CRC_QUALIFIER				
R-0h			R/W-0h				

Table 13-41. CRC_QUALIFIER Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R	0h	Reserved
4-0	CRC_QUALIFIER	R/W	0h	0000 = No Qualifier, every valid event is qualified for CRC computation 00001 = CRC Compute Qualified by HWBP_EVENT1 00010 = CRC Compute Qualified by HWBP_EVENT2 00011 = CRC Compute Qualified by HWBP_EVENT3 00100 = CRC Compute Qualified by HWBP_EVENT4 00101 = CRC Compute Qualified by HWBP_EVENT5 00110 = CRC Compute Qualified by HWBP_EVENT6 00111 = CRC Compute Qualified by HWBP_EVENT7 01000 = CRC Compute Qualified by HWBP_EVENT8 01001 = CRC Compute Qualified by HWBP_EVENT_OR1 01010 = CRC Compute Qualified by HWBP_EVENT_OR2 01011 = CRC Compute Qualified by HWBP_EVENT_OR3 01100 = CRC Compute Qualified by HWBP_EVENT_OR4 01101 = CRC Compute Qualified by HWBP_EVENT_AND1 01110 = CRC Compute Qualified by HWBP_EVENT_AND2 01111 = CRC Compute Qualified by HWBP_EVENT_AND3 10000 = CRC Compute Qualified by HWBP_EVENT_AND4 Others = No Qualifier, every valid event is qualified for CRC computation Reset type: SYSRSn

13.7.7 ERAD Registers to Driverlib Functions

Table 13-42. ERAD Registers to Driverlib Functions

File	Driverlib Function
GLBL_EVENT_STAT	
erad.h	ERAD_getEventStatus
GLBL_HALT_STAT	
erad.h	ERAD_getHaltStatus
GLBL_ENABLE	
erad.h	ERAD_enableModules
erad.h	ERAD_disableModules
GLBL_CTM_RESET	
erad.h	ERAD_resetCounter
GLBL_NMI_CTL	
erad.h	ERAD_enableNMI
erad.h	ERAD_disableNMI

Table 13-42. ERAD Registers to Driverlib Functions (continued)

File	Driverlib Function
GLBL_OWNER	
erad.h	ERAD_getOwnership
erad.h	ERAD_setOwnership
GLBL_EVENT_AND_MASK	
erad.c	ERAD_configMask
GLBL_EVENT_OR_MASK	
erad.c	ERAD_configMask
GLBL_AND_EVENT_INT_MASK	
erad.c	ERAD_configMask
GLBL_OR_EVENT_INT_MASK	
erad.c	ERAD_configMask
HWBP_MASK	
erad.c	ERAD_configBusComp
HWBP_REF	
erad.c	ERAD_configBusComp
HWBP_CLEAR	
erad.h	ERAD_clearBusCompEvent
HWBP_CNTL	
erad.c	ERAD_configBusComp
HWBP_STATUS	
erad.h	ERAD_getBusCompStatus
CTM_CNTL	
erad.c	ERAD_configCounterInCountingMode
erad.c	ERAD_configCounterInStartStopMode
erad.c	ERAD_configCounterInCumulativeMode
erad.h	ERAD_enableCounterResetInput
erad.h	ERAD_disableCounterResetInput
CTM_STATUS	
erad.h	ERAD_getCounterStatus
CTM_REF	
erad.c	ERAD_configCounterInCountingMode
erad.c	ERAD_configCounterInStartStopMode
erad.c	ERAD_configCounterInCumulativeMode
CTM_COUNT	
erad.h	ERAD_getCurrentCount
erad.h	ERAD_setCurrentCount
CTM_MAX_COUNT	
erad.h	ERAD_getMaxCount
erad.h	ERAD_setMaxCount
CTM_INPUT_SEL	
erad.c	ERAD_configCounterInCountingMode
erad.c	ERAD_configCounterInStartStopMode
erad.c	ERAD_configCounterInCumulativeMode
erad.h	ERAD_enableCounterResetInput
CTM_CLEAR	

Table 13-42. ERAD Registers to Driverlib Functions (continued)

File	Driverlib Function
erad.h	ERAD_clearCounterEvent
erad.h	ERAD_clearCounterOverflow
CTM_INPUT_SEL_2	
erad.c	ERAD_configCounterInStartStopMode
erad.c	ERAD_configCounterInCumulativeMode
erad.h	ERAD_enableCounterResetInput
CTM_INPUT_COND	
erad.h	ERAD_setCounterInputConditioning
CRC_GLOBAL_CTRL	
erad.h	ERAD_initCRC
erad.h	ERAD_enableCRC
erad.h	ERAD_disableCRC
erad.h	ERAD_setSeed
erad.h	ERAD_setCRCQualifier
CRC_CURRENT	
erad.h	ERAD_getCurrentCRC
CRC_SEED	
erad.h	ERAD_setSeed
CRC_QUALIFIER	
erad.h	ERAD_setCRCQualifier

Chapter 14

Host Interface Controller (HIC)

The HIC module allows an external host controller to directly access resources of the device using the ASRAM protocol.

14.1 Overview

The HIC module allows an external host controller to directly access resources of the device by emulating the ASRAM protocol. It has two modes of operation: direct access and mailbox access. In direct access mode, device resources are written to and read from directly by the external host. In mailbox access mode, external host and device write to and read from a buffer and notify each other when the buffer write/read is complete. For security reasons, the HIC has to be enabled by the device before the external host can access it by using the HICGCR.HICEN and HICMODECR.EN_DEVACC registers.

14.1.1 HIC Related Collateral

Getting Started Materials

- *Design Guide for Enabling Peripheral Expansion Applications Using the HIC Application Report*

14.1.2 Features

The HIC module has the following features:

- Configurable I/O data lines of 8 and 16 bit
- Direct and mailbox access modes
- 8 address lines and 8 configurable base addresses for a total of 2048 possible addressable regions
- Two 64-byte buffers for external host and device when using mailbox access mode
- Interrupt generation on buffer full/empty
- High throughput
- Trigger HIC activity from other peripherals
- Error indicators to the system or interface
- Commit feature that blocks writes to configuration registers

Note

The C28x processor is not byte addressable. The minimum addressable region in memory is 16-bits.

14.1.3 Block Diagram

Figure 14-1 shows the block diagram of the HIC.

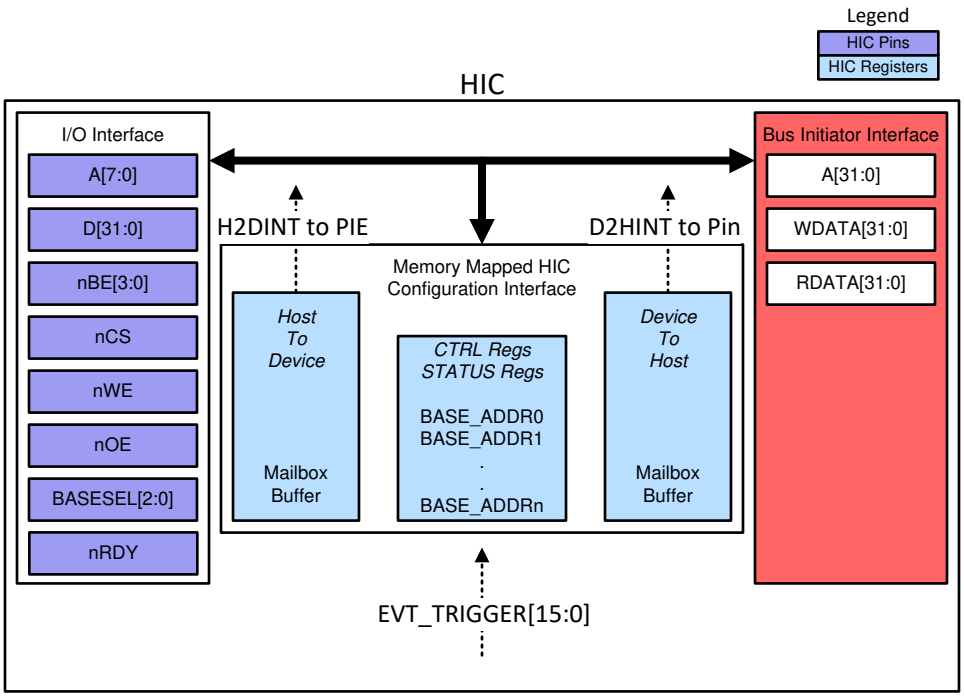


Figure 14-1. HIC Block Diagram

14.2 Functional Description

The HIC consists of 3 main components:

- **I/O Interface:** The Input and Output (I/O) Interface emulates the ASRAM protocol to perform read and write operations between the external host and the device.
- **Configuration Interface:** The configuration interface consists of the registers that make up the HIC module. These registers include the control, data and status registers of the HIC and are accessible by both the external host and processors within the device.
- **Bus Initiator Interface:** The bus initiator interface is the port that accesses the device memory and peripheral registers. Reads from and writes to the device from the external host are handled by the bus initiator interface.

14.2.1 Memory Map

Figure 14-2 shows the memory and peripherals accessible by the HIC module and by extension, the external host.

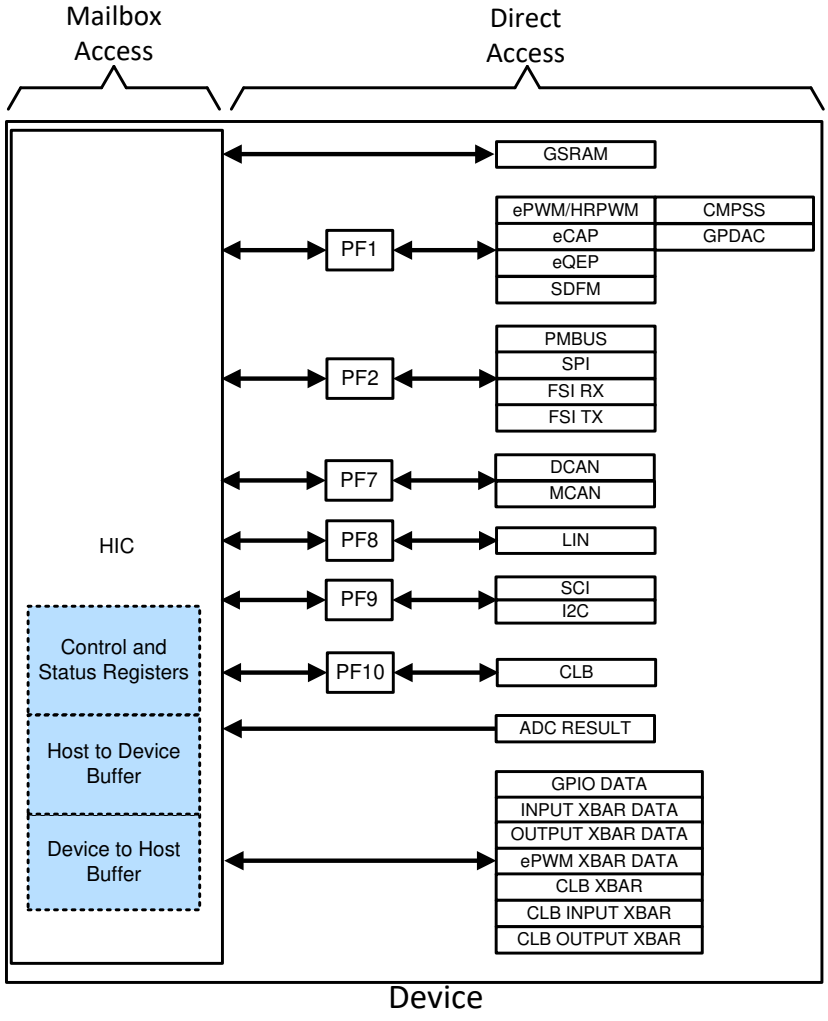


Figure 14-2. HIC Memory Map

14.2.2 Connections

Figure 14-3 shows the possible connections that can be made between the external host and the HIC module. Not all connections are available on all devices. Check the device specific data manual to see the connections supported on your device.

Table 14-1 is a brief description of what each connection means.

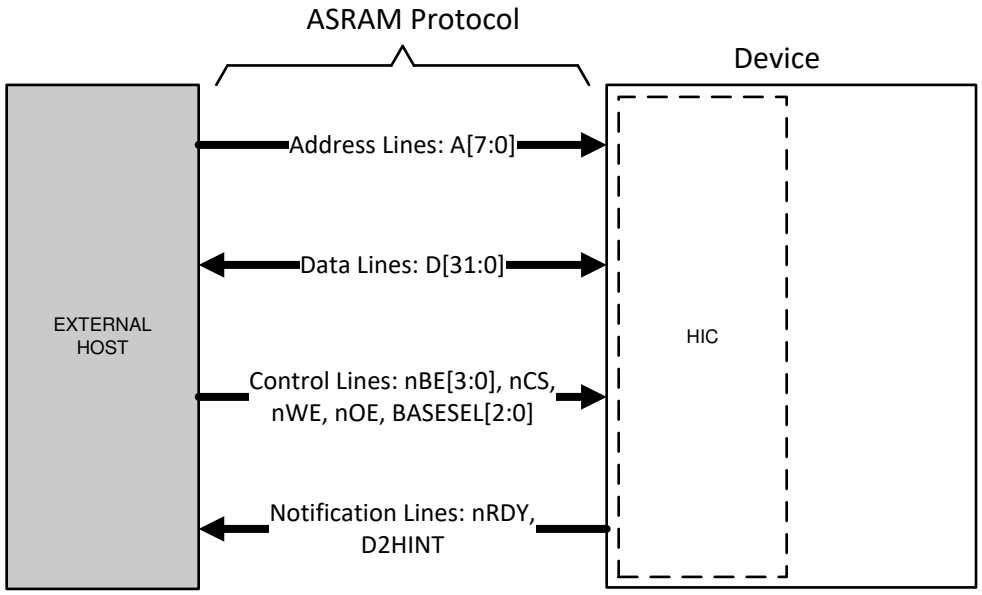


Figure 14-3. HIC Connections

Table 14-1. HIC Connections Table

Connection Group	Connection Name	Direction	Description
Address Lines	A[7:0]	Input	Address Lines Bit0 to Bit7
Data Lines	D[31:0]	Input/Output	Data Lines Bit0 to Bit31
Control Lines	nBE[3:0]	Input	Byte Enable Bit0 to Bit3
	nCS	Input	Chip Select
	nWE	Input	Write Enable
	nOE/RnW	Input	Output Enable/Read not Write
	BASESEL[2:0]	Input	Base Select Bit0 to Bit2
Notification Lines	nRDY	Output	Ready/Wait
	D2HINT	Output	Device to Host Interrupt

14.2.2.1 Functions of the Connections

- **Address Lines:** The address lines are part of the signals the external hosts uses to select which region of the device to read from or write to.
- **Data Lines:** The data lines carry the actual data read from or written to the device.
- **Control Lines**
  - **nBE:** The HIC supports 8-bit and 16-bit data transfers. The Byte Enable signal is used to select which of these modes data transfer will happen in.
  - **nCS:** The Chip Select signal enables or disables the HIC interface.
  - **nWE:** The Write Enable signal enables or disables writes from the external host to the device.
  - **nOE:** The Output Enable signal enables or disables reads from the device to the external host.
  - **BASESEL:** The HIC supports access to device memory and peripherals or the HIC registers. The Base Select signal is used to select which region the external host will access.
- **Notification Lines**
  - **nRDY:** The Ready signal is used by the HIC to indicate to the external host that it is ready for a new read or write access.
  - **D2HINT:** The Device to Host Interrupt signal is used by the HIC to indicate to the external host that data transaction is complete.

Note

By default, nBE, nCS, nWE, nOE and nRDY are active-low signals. Their polarity can be changed by configuring the HICPINPOLCR register.

14.2.3 Interrupts and Triggers

The external host interrupts the device using H2DINT that goes to the PIE. A write to the HICH2DTOKEN register triggers H2DINT. The H2DINT sources are shown in Figure 14-4.

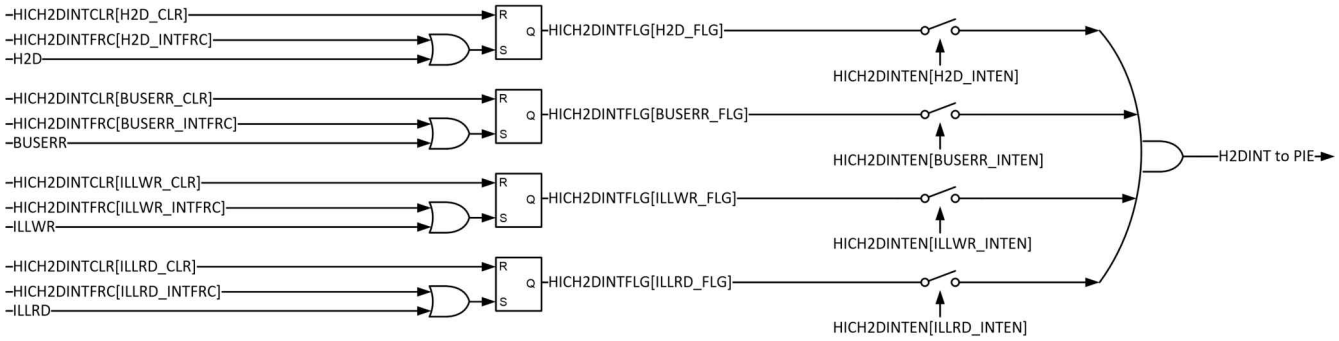


Figure 14-4. HIC Host to Device Interrupt Sources

The device interrupts the external host using D2HINT that comes out on a pin. A write to the HICD2HTOKEN register triggers D2HINT. The D2HINT sources are shown in Figure 14-5.

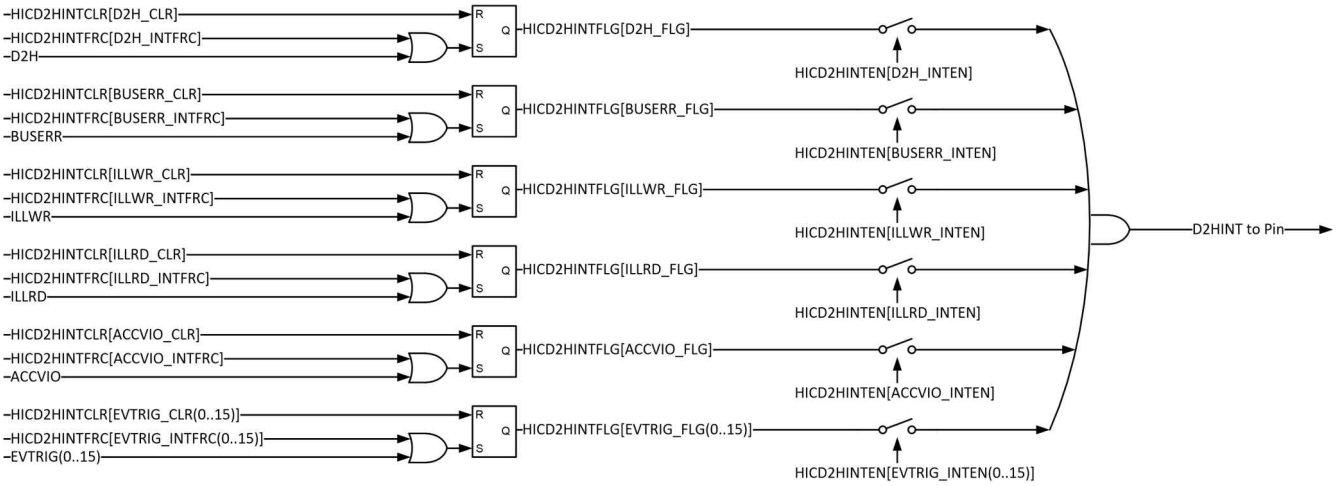


Figure 14-5. HIC Device to Host Interrupt Sources

D2HINT can also be triggered by peripherals in the device (EVTRIG) as indicated in Figure 14-5. The sources for EVTRIG are listed in Table 14-2.

Table 14-2. Event Trigger (EVTRIG) Sources

Event Trigger	Peripheral Interrupt Source
EVTRIG[0]	FSIRX_INT1, FSIRX_INT2
EVTRIG[1]	FSITX_INT1, FSITX_INT2
EVTRIG[2]	SPIB_TX, SPIB_RX
EVTRIG[3]	SPIA_TX, SPIA_RX
EVTRIG[4]	EPWM2, EPWM2_TZ
EVTRIG[5]	EPWM1, EPWM1_TZ
EVTRIG[6]	CLB2
EVTRIG[7]	CLB1
EVTRIG[8]	CLB4
EVTRIG[9]	CLB3
EVTRIG[10]	DMA Transfer
EVTRIG[11..15]	Reserved

14.3 Operation

14.3.1 Mailbox Access Mode Overview

In Mailbox Access Mode, the external host only sees the HIC memory mapped region and not the device resources and peripherals. This memory mapped region of the HIC includes the HIC control, status and data registers.

128 bytes of this region is reserved for HIC control and status registers. The remaining 128 bytes is used as a data/buffer space and is split 64 bytes each as Host to Device (H2D) buffer and Device to Host (D2H) buffer. Note that this distinction between H2D and D2H buffer is only in name. Device and external host can each access the combined H2D and D2H buffers of 128 bytes by configuring the H2DBUF_DEVWREN and D2HBUF_HOSTWREN fields in the HICMODECR register. Figure 14-6 shows the Mailbox Access Mode and the minimum external connections required for this mode.

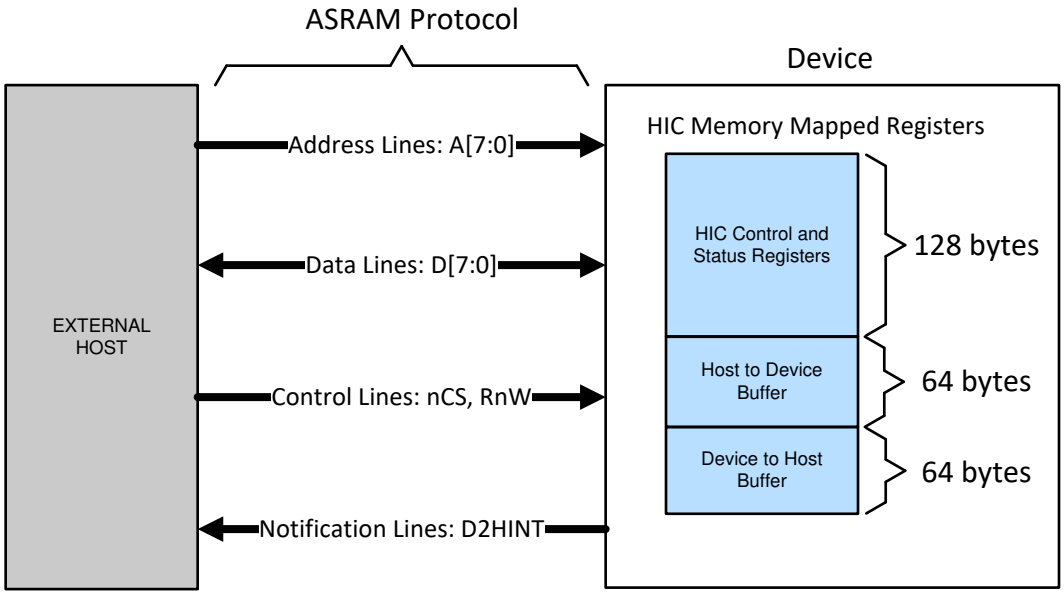


Figure 14-6. HIC Mailbox Access Mode Diagram

14.3.1.1 Mailbox Access Mode Operation

To place the HIC in Mailbox Access Mode, BASESEL[2:0] pins should be set to '000'. Address lines A[7:0] can then be used to select which of the 256 bytes of the HIC memory mapped registers to access.

14.3.1.2 Configuring HIC Registers With External Host

The external host can only configure the HIC registers and read its status in Mailbox Access Mode. As mentioned in the prior section, the HIC control and status registers reside in the first 128 bytes of the HIC memory mapped region.

14.3.1.3 Mailbox Access Mode Read/Write

The remaining 128 bytes of the HIC memory-mapped region can be used as a general-purpose buffer. To write to or read from this region:

- The device enables the HIC by writing 0xA to the HICGCR.HICEN register
- The device configures the availability and polarity of the HIC pins using the HICMODECR.RW_MODE, HICMODECR.BEN_PRESENT, HICMODECR.RDY_PRESENT, and HICPINPOLCR registers

To write data to the device:

- The external host writes to the H2D buffer, that is, H2D_BUF register
- After completing the write, the external host triggers an interrupt (H2DINT) to the device by writing a pattern to the HICH2DTOKEN register. This pattern can be any value and a useful value to write to the HICH2DTOKEN register is the number of bytes written by the external host. Any write to the HICH2DTOKEN register will trigger H2DINT.
- The device acknowledges this interrupt by transferring the data from the H2D buffer (H2D_BUF register) to the required location.

To read data from the device:

- The device writes to the D2H buffer i.e D2H_BUF register
- After completing the write, the device triggers an interrupt (D2HINT) to the external host by writing a pattern to the HICD2HTOKEN register. This pattern can be any value and a useful value to write to the HICD2HTOKEN register is the number of bytes written by the device.
- The external host acknowledges this interrupt by reading the contents of the D2H buffer (D2H_BUF register).

Note

An error will not be generated if writes to the D2H_BUF and H2D_BUF exceed their memory region, that is, 128 bytes. Application needs to make sure writes to the above regions are appropriately bounded.

14.3.2 Direct Access Mode Overview

In Direct Access Mode, the external host through the HIC can directly read from and write to device locations specified in the memory map. The device location to read is a combination of the 8-bit address lines A[7:0] and the 24-bit HICDBADDR{#} register.

These two address pieces are concatenated as:

Full 32-bit Address = {HICDBADDR{#}.BASE_ADDR[31:8],A[7:0]}

There are 8 HICDBADDR{#} registers for a total of 2048 address locations in combination with A[7:0] address lines. Which HICDBADDR{#} register to concatenate with A[7:0] is selectable by the HICBASESEL register or BASESEL[2:0] pins. Figure 14-7 shows the Direct Access Mode and the minimum external connections required for this mode.

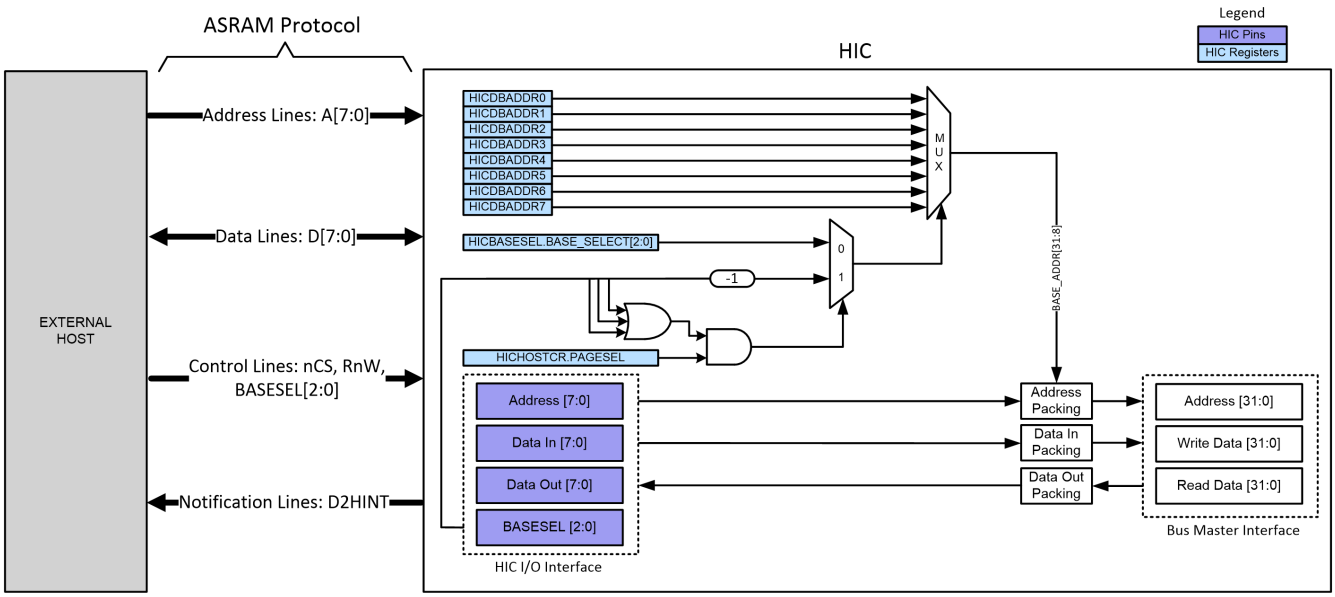


Figure 14-7. HIC Direct Access Mode Diagram

14.3.2.1 Direct Access Mode Operation

To place the HIC in Direct Access Mode, BASESEL[2:0] pins should be greater than '000'. Address lines A[7:0] are used in conjunction with the HICDBADDR{#} register to read from or write to a location in the device. Table 14-3 shows the device address space view for the external host.

Note

BASESEL[2:0] pins cannot select HICDBADDR7 register, BASESEL[2:0] pins can only select HICDBADDR0-6 registers. HICDBADDR7 can only be selected through software using the HICBASESEL.BASE_SELECT[2:0] register.

Table 14-3. Address Space View for External Host

BASESEL[2:0] Pins(1)	HICBASESEL.BASE_SELECT[2:0] Register(1)	HICHOSTCR.PAGESEL Register(1)	Address Space	Comment
000	x	x	**HIC Memory Map Registers - 256 Bytes**	HIC Control, Status, D2H_BUF and H2D_BUF Registers
001	x	1	**{HICDBADDR0.BASE_ADDR[31:8], A[7:0]} - 256 Locations**	See HIC Memory Map
010	x	1	**{HICDBADDR1.BASE_ADDR[31:8], A[7:0]} - 256 Locations**	See HIC Memory Map
011	x	1	**{HICDBADDR2.BASE_ADDR[31:8], A[7:0]} - 256 Locations**	See HIC Memory Map
100	x	1	**{HICDBADDR3.BASE_ADDR[31:8], A[7:0]} - 256 Locations**	See HIC Memory Map
101	x	1	**{HICDBADDR4.BASE_ADDR[31:8], A[7:0]} - 256 Locations**	See HIC Memory Map
110	x	1	**{HICDBADDR5.BASE_ADDR[31:8], A[7:0]} - 256 Locations**	See HIC Memory Map
111	x	1	**{HICDBADDR6.BASE_ADDR[31:8], A[7:0]} - 256 Locations**	See HIC Memory Map
x	000	0	{HICDBADDR0.BASE_ADDR[31:8], A[7:0]} - 256 Locations	See HIC Memory Map
x	001	0	{HICDBADDR1.BASE_ADDR[31:8], A[7:0]} - 256 Locations	See HIC Memory Map
x	010	0	{HICDBADDR2.BASE_ADDR[31:8], A[7:0]} - 256 Locations	See HIC Memory Map
x	011	0	{HICDBADDR3.BASE_ADDR[31:8], A[7:0]} - 256 Locations	See HIC Memory Map
x	100	0	{HICDBADDR4.BASE_ADDR[31:8], A[7:0]} - 256 Locations	See HIC Memory Map
x	101	0	{HICDBADDR5.BASE_ADDR[31:8], A[7:0]} - 256 Locations	See HIC Memory Map
x	110	0	{HICDBADDR6.BASE_ADDR[31:8], A[7:0]} - 256 Locations	See HIC Memory Map
x	111	0	{HICDBADDR7.BASE_ADDR[31:8], A[7:0]} - 256 Locations	See HIC Memory Map

(1) "x" indicates don't care.

14.3.2.2 Direct Access Mode Read/Write

The steps below assume all HIC pins are available. To write to or read from the device:

- The device enables the HIC by writing 0xA to the HICGCR.HICEN register
- The device enables direct access mode using HICMODECR.EN_DEVACC register
- The device configures the availability and polarity of the HIC pins using the HICMODECR.RW_MODE, HICMODECR.BEN_PRESENT, HICMODECR.RDY_PRESENT and HICPINPOLCR registers
- The device configures the HICDBADDR{#} registers
- To write data to the device, the external host selects a HICDBADDR{#} using the BASESEL[2:0] pins, drives the 8-bit portion of the address on the address lines A[7:0] and the data to write on the data lines D[15:0]
- To read data from the device, the external host selects a HICDBADDR{#} using the BASESEL[2:0] pins, drives the 8-bit portion of the address on the address lines A[7:0] and reads the data on the data lines D[15:0]

14.3.3 Controlling Reads and Writes

The HIC has two modes of controlling reads and writes: single-pin mode and dual-pin mode.

14.3.3.1 Single-Pin Read/Write Mode (nOE/RnW Pin)

In single-pin read/write access (Figure 14-8), nOE(RnW) determines whether the transaction being performed is a read or a write. If nOE is low, data is written from the host to the device; when nOE is high, data is read from the device by the host.

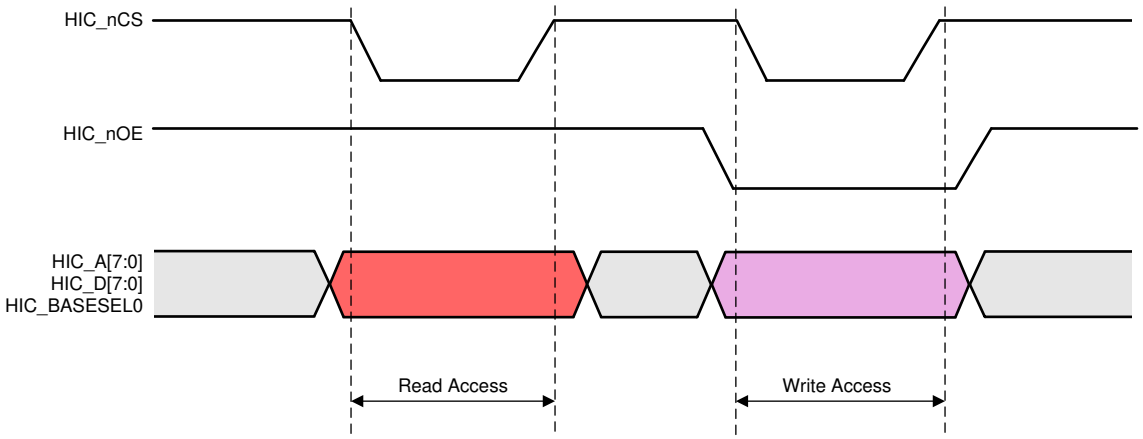


Figure 14-8. Read/Write Access in Single-Pin Mode

14.3.3.2 Dual-Pin Read/Write Mode (nOE and nWE Pins)

In dual-pin read/write access (Figure 14-9), nOE and nWE pins determine whether the transaction being performed is a read or a write. In the default polarity state, nOE going low indicates a data read from the device to the host and nWE going low indicates a data write to the device by the host.

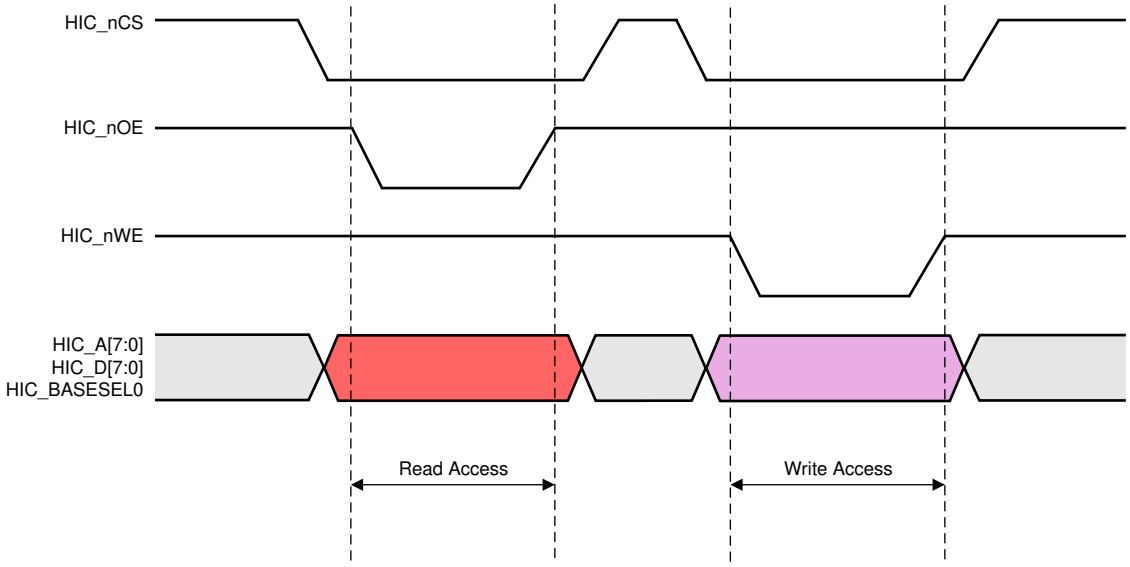


Figure 14-9. Read/Write Access in Dual-Pin Mode

14.3.4 Data Lines, Data Width, Data Packing and Unpacking

The HIC supports 8 and 16 data lines and host data transfers can be made in 8-bit and 16-bit modes. The host data transfer mode is selected using the byte enable (nBE) pins. In the absence of the nBE pins, host data transfer mode can be selected using the HICMODECR.DW_MODE register.

At the initiator port level, the HIC internal access is restricted to 16-bit and 32-bit modes selectable using the HICHOSTCR.ACCSIZE register. If there is a mismatch between host access mode and initiator port access mode, the HIC will internally pack and unpack the data automatically as needed.

For instance, if host access mode is set to 16-bit using the nBE pins or the HICMODECR.DW_MODE register and initiator port access mode is set to 16-bit using the HICHOSTCR.ACCSIZE register, no data packing/ unpacking will be done. If host access mode is set to 8-bit and initiator port access mode is set to 16-bit, a single 8-bit access will wait for another 8-bit access to pack the two 8-bit accesses into one 16-bit access.

Table 14-4 shows the combinations for write access.

Table 14-4. Data Packing and Unpacking for Writes

HICMODECR. DW_MODE	Host Port Write			Initiator Port Write (16-bit Aligned)		Error	Comment
	A[7:0]	D[7:0] for 8-bit DW_MODE D[15:0] for 16-bit DW_MODE	nBE[1:0]	DWAB[31:0]	DWDB[31:0]		
8-bit ("0")	0x0	B0	-			No	8-bit write to even address, waits for packing.
	0x1	B1	-	BADDRx + 0x0	[15:0]=B1B0	No	8-bit write to odd address, packed into 16-bit.
	0x2	B2	-			No	8-bit write to even address, waits for packing.
	0x3	B3	-	BADDRx + 0x1	[31:16]=B3B2	No	8-bit write to odd address, packed into 16-bit.
	0x1	B1	-	BADDRx + 0x0	[15:0]=B1B0	No	8-bit write to odd address, packed into 16-bit with previous even byte data.
	0x3	B3	-	BADDRx + 0x1	[31:16]=B3XX	No	8-bit write to odd address, packed into 16-bit with previous even byte data.
	0xA	B0	-			No	8-bit write to even address, waits for packing.
	0xA+n (n > 1)	XX	-			Yes	Consecutive byte writes beyond 16-bit boundary are illegal, previous byte discarded.

Table 14-4. Data Packing and Unpacking for Writes (continued)

HICMODECR. DW_MODE	Host Port Write			Initiator Port Write (16-bit Aligned)		Error	Comment
	A[7:0]	D[7:0] for 8-bit DW_MODE D[15:0] for 16-bit DW_MODE	nBE[1:0]	DWAB[31:0]	DWDB[31:0]		
16-bit ("1")	0x0	B1B0	2'b00	BADDRx + 0x0	[15:0]=B1B0	No	Full 16-bit write sent through.
	0x1	B3B2	2'b00	BADDRx + 0x1	[31:16]=B3B2	No	Full 16-bit write sent through.
	0x2	B5B4	2'b00	BADDRx + 0x2	[15:0]=B5B4	No	Full 16-bit write sent through.
	0x3	B7B6	2'b00	BADDRx + 0x3	[31:16]=B7B6	No	Full 16-bit write sent through.
	0x0	XXB0	2'b10			No	8-bit write to even address, waits for packing.
	0x0	B1XX	2'b01	BADDRx + 0x0	[15:0]=B1B0	No	8-bit write to odd address, packed into 16-bit.
	0x1	XXB2	2'b10			No	8-bit write to even address, waits for packing.
	0x1	B3XX	2'b01	BADDRx + 0x1	[31:16]=B3B2	No	8-bit write to odd address, packed into 16-bit.
	0x0	B1XX	2'b01	BADDRx + 0x0	[15:0]=B1BXX	No	Any odd 8-bit write triggers 16-bit write packed with previous even byte data.
	0x1	B3XX	2'b01	BADDRx + 0x1	[31:16]=B3XX	No	Any odd 8-bit write triggers 16-bit write packed with previous even byte data.
	0xA	XXB0	2'b10			No	8-bit write to even address, waits for packing.
	0xA+n (n > 1)	XXB0	2'b10			Yes	Consecutive byte writes beyond 16-bit boundary are illegal, previous byte discarded.

Table 14-5 shows the combinations for read access.

Table 14-5. Data Packing and Unpacking for Reads

HICMODECR. DW_MODE	Host Port Read		Initiator Port Read		Comment
	A[7:0]	D[7:0] for 8-bit DW_MODE D[15:0] for 16-bit DW_MODE	DRAB[31:0]	DRDB[31:0]	
8-bit ("0")	0x0	B0	BADDRx + 0x0	[31:0]=B3B2B1B0	A random read from the host will trigger a 32-bit read on the initiator port.
	0x1	B1			A sequential byte read within the aligned 32-bit word will be returned from the internal data buffer.
	0x2	B2			
	0x3	B3			
	0x4	B4	BADDRx + 0x2	[31:0]=B7B6B5B4	A byte read outside of an aligned 32-bit word boundary will trigger a 32-bit read on the initiator port.
16-bit ("1")	0x0	B1B0	BADDRx + 0x0	[31:0]=B7B6B5B4	A random read from the host will trigger a 32-bit read on the initiator port.
	0x1	B3B2			A sequential 16-bit read within an aligned 32-bit word will be returned from the internal data buffer.
	0x2	B5B4	BADDRx + 0x2	[31:0]=B3B2B1B0	A 16-bit read from the host port outside of an aligned 32-bit word triggers a 32-bit read on the initiator port.
	0x3	B7B6			A sequential 16-bit read within an aligned 32-bit word will be returned from the internal data buffer.

14.3.5 Address Translation

Since the initiator port uses 16-bit aligned addressing while the host port can operate in 8 and 16-bit modes, address translation is needed for correct operation. Table 14-6 shows an example of address translation performed in 8-bit data width mode while Table 14-7 shows the same for 16-bit data width mode.

Table 14-6. Address Translation for 8-bit Data Width Mode

HICMODECR. DW_MODE	Host Port Address	Initiator Port Address {HICDBADDR.BASE_ADDR[31:7],A[6:1]}
8-bit ("0")	0x0	0x0
	0x1	0x0
	0x2	0x1
	0x3	0x1
	0x4	0x2
	0x5	0x2
	0x6	0x3
	0x7	0x3
	0x7F	0x3F
	0x80	0x40
	0x81	0x40
	0xFE	0x7F
	0xFF	0x7F

Table 14-7. Address Translation for 16-bit Data Width Mode

HICMODECR. DW_MODE	Host Port Address	Initiator Port Address{HICDBADDR.BASE_ADDR[31 :8],A[7:0]}
16-bit ("1")	0x0	0x0
	0x1	0x1
	0x2	0x2
	0x7F	0x7F
	0x80	0x80
	0x81	0x81
	0xFE	0xFE
	0xFF	0xFF

14.3.6 Access Errors

The HIC has the HICH2DINTFLG, HICD2HINTFLG, HICERRADDR and HICACCVIOADDR registers that capture the state of the HIC when an access error occurs. The errors captured are:

Illegal Write Error

This error occurs during a write when host mode bit size is greater than initiator port bit size. For instance if the host is configured in 32-bit mode using either nBE or HICMODECR.DW_MODE, a write error will occur if HICHOSTCR.ACCSIZE is set to 16-bit. This error will also occur if incorrect address range is specified for write access.

See Table 14-4.

Illegal Read Error

This error occurs during a read when host mode bit size is greater than initiator port bit size. For instance if the host is configured in 32-bit mode using either nBE or HICMODECR.DW_MODE, a read error will occur if HICHOSTCR.ACCSIZE is set to 16-bit. This error will also occur if a read is made from a location which was incompletely written in a previous transaction. For instance, the HIC is waiting on another byte to complete a 16-bit write transaction but gets a read access before this second byte.

Bus Error

This error occurs when a memory location is simultaneously written by both host and device. The host data will be discarded and device data retained.

Initiator Port Access Violation Error

This error occurs when the host tries to access a memory region that has access protection enabled, is reserved or a Parity/ECC occurred while performing a read access.

Other Errors

The previous errors do not capture all the possible errors that can occur with the HIC. The following errors either hang the HIC module or fail silently:

- The HIC module or device is under reset and an access is initiated.
- The HIC clock is disabled or the device is in a low-power mode and an access is initiated.
- The HIC clock is disabled in the middle of a transaction by the host.

It is recommended that the application perform data integrity checks, for instance, read back written data in order to catch these silent errors.

14.3.7 Security

There are multiple security levels implemented in the device and HIC module to prevent unauthorized access:

- **Level 1:** The device must turn on the clock for HIC and enable the HIC using HICGCR.HICEN before any access to the device region can be made. The configuration registers are read-only by the host.
- **Level 2:** MEM_CFG_REGS.GSxACCPROT0.HICWRPROT_GSy enables and disables write access to the GSRAM by the HIC.
- **Level 3:** HICMODECR.EN_HOSTWREALLOW, HICMODECR.EN_DEVACC, and HICHOSTCR.EALLOW_EN further restricts access by enabling/disabling EALLOW and device region access by host.
- **Level 4:** HICLOCK register can be used to lock the configuration registers from further writes. This register is read-only by the host.
- **Level 5:** The state of the lock register can be commited with the HICCOMMIT register which will prevent further changes until device is reset.

14.3.8 HIC Usage

To get started using the HIC, it is recommend to first map out the pins needed for the application. See the device data sheet as it lists the set of HIC pins available on the device.

In addition, it is recommend to read the *Enabling Peripheral Expansion Applications Using the HIC Application Report* before getting started with using the HIC in an application. The application report contains examples of use-cases and it is written from an application point of view. Some of the content in this chapter is repeated in the application report. This is done to ensure the application report is a standalone document.

14.4 Usage Scenarious for Reduced Number of Pins

The HIC does not require all pins for operation. The following describes use-cases for a reduced set of pins.

Read/Write Control Pins

Controlling read/write can be achieved using just one pin, nOE/RnW pin, instead of the traditional two pins: nOE and nWE pins. Configuration is achieved through the HICMODECR.RW_MODE register. See Section 14.3.3 for details.

BASESEL Pins

If no BASESEL pins are available, the HIC is restricted to mailbox access mode where the external host can only access the HIC memory mapped registers. However to access device registers through direct access mode, only one BASESEL pin is required.

In this configuration, BASESEL0 pin determines whether the access being made is mailbox access or direct access. BASESEL0 set to 0 will be a mailbox access and BASESEL0 set to 1 will be direct access. This configuration restricts base address selection to only HICDBADDR0. Configuration is achieved through the HICHOSTCR.PAGESEL register. See Section 14.3.2.1 for details.

Data Pins

The HIC supports 8-bit and 16-bit transfers. 16-bit data transfers can still be achieved with 8 data lines. This is handled internally by the HIC through data packing and unpacking. See Section 14.3.4 for details.

Note

8 data lines is the minimum supported.

Byte-Enable Pins

If no byte-enable pins are available, the HIC defaults to 16-bit data transfers but can be re-configured using the HICMODECR.DW_MODE register. Configuration is achieved through the HICMODECR.BEN_PRESENT register. See Section 14.3.4 for details.

Ready Pin

If no nRDY pin is available, the host must add sufficiently large setup/hold times during data transmission to prevent clashes. Configuration is achieved through the HICMODECR.RDY_PRESENT register.

14.5 Software

14.5.1 HIC Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/f28003x/examples/hic

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

14.5.1.1 HIC 16-bit Memory Access Example

FILE: hic_ex1_config_16bit.c

Example sets up Host Interface Controller Module in 16-bit mode the data memory of device is accessed by the *External* Host.

This example is validated on an TI Internal Validation Platform will not run in F28002x Control Card The example initializes the GPIOs for 16 bit Memory access mode with separate Read and Write Control Pins, Extended Wait, and nRDY pin The example demonstrates the following sequences

1. Sending a message to the Host using Device to Host buffer,Passing a token which triggers an interrupt at the Host.
2. Waits for the Host to clear the interrupt.
3. Then waits for a message from Host which contains the code 0x1 in buffer index 0 and index 1 contains the Base address to be configured
4. Configures the base address for Page 0
5. Sends a message to Host after that is configured
6. After this step the external host can use the address 0-0xFF to access the Base address region in System memory

External Connections
This is tested on TI Internal validation platform hence no connection needed on Control card. This has been tested with F2838x EMIF interface.

Watch Variables
None

14.5.1.2 HIC 8-bit Memory Access Example

FILE: hic_ex2_config_8bit_adc.c

Example sets up Host Interface Controller Module in 8-bit mode the data memory of device is accessed by the External Host.

- This follows example configuration for pin constrained applications described in the *Design Guide for Enabling Peripheral Expansion Applications Using the HIC Application Report*.

  This example is validated on an TI Internal Validation Platform will not run in F28002x Control Card. This is to be run with a corresponding Host side code emif_ex8_8bit_asram_hic_adc is run on F2838x based host first and then this example is run.
- The example initializes the HIC for 8 bit Memory access mode with Single Read and Write Control Pin
- DMA is configured to do transfer of the ADC result from ADCRESULT0-15 to HIC D2H buffer.
- The example configures CPUTIMER0 to generate an event every second on which the ADC SOC0-15 start conversion.
- The timer is enabled on receiving HIC_START_TOKEN on the H2D Token from the host.
- The end of ADC_SOC15 will generate a DMA event
- The DMA ISR signals the host with HIC_DATA_TOKEN to signal data write.
- The Host is interrupted with HIC_INT on ADC End of conversion event
- The DMA Channel is stopped after HIC_TEST_NUM_SAMPLES samples are collected This example can be used as a reference to implement Communication between Host and HIC with minimal number of pins. Refer to the *Design Guide for Enabling Peripheral Expansion Applications Using the HIC Application Report* for further details about Host interface Controller applications.

External Connections

This is tested on TI Internal validation platform hence no connection needed on Control card. This has been tested with F2838x EMIF interface with emif_ex8_8bit_asram_hic_adc example Analog channel A7 has been used as Analog Input to Sample

Watch Variables

- numDMAInterrupts

14.5.1.3 HIC 16-bit Memory Access FSI Example

FILE: hic_ex3_config_16bit_fsi.c

Example sets up Host Interface Controller Module in 16-bit mode the FSI Memory region of device is accessed by the External Host. This follows example configuration for performance critical applications described in the Application note titled "Application guide for peripheral expansion using HIC"(SPRACR2).

emif_ex7_16bit_asram_hic_fsi is to be run on the host before running this example on F28002x side.

This example is validated on an TI Internal Validation Platform will not run in F28002x Control Card

1. The example initializes the GPIOs for 16 bit HIC Memory access mode with separate Read and Write Control Pins, Extended Wait, and nRDY pin
2. Configures the base address for Region 0 to access FSI Tx Memory region
3. Sets up the FSI module in Loopback mode
4. The EVTTRIG interface of HIC is enabled to route FSI Rx events to the Host through the HIC_INT
5. Sending a message to the Host using Device to Host buffer,Passing a token
6. This sets up the FSI direct bridge over HIC. The host can then access the FSI registers over HIC

External Connections
This is tested on TI Internal validation platform hence no connection needed on Control card. This has been tested with F2838x EMIF interface.

Watch Variables
None - Use the Host to access FSI region

14.6 HIC Registers

This section describes the Host Interface Controller Registers.

14.6.1 HIC Base Address Table

Table 14-8. HIC Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
HicRegs	HIC_CFG_REGS	HIC_BASE	0x0000_6500	YES	YES	-	-	YES

14.6.2 HIC_CFG_REGS Registers

Table 14-9 lists the memory-mapped registers for the HIC_CFG_REGS registers. All register offset addresses not listed in Table 14-9 should be considered as reserved locations and the register contents should not be modified.

Table 14-9. HIC_CFG_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	HICREV	Module Revision Register	No	Go
2h	HICGCR	Global Control Register	EALLOW	Go
4h	HICLOCK	Lock Register	EALLOW	Go
6h	HICMODECR	Mode Control Register	EALLOW and LOCK	Go
8h	HICPINPOLCR	Pin Polarity Control Register	EALLOW and LOCK	Go
Ah	HICBASESEL	Base Select Register	No	Go
Ch	HICHOSTCR	HIC Host Control Register	No	Go
Eh	HICERRADDR	Host Error Address register	No	Go
10h	HICH2DTOKEN	Host to Device Token Register	No	Go
12h	HICD2HTOKEN	Devie to Host Token Register	No	Go
14h	HICDBADDR0	Device Base Address Register 0	EALLOW and LOCK	Go
16h	HICDBADDR1	Device Base Address Register 1	EALLOW and LOCK	Go
18h	HICDBADDR2	Device Base Address Register 2	EALLOW and LOCK	Go
1Ah	HICDBADDR3	Device Base Address Register 3	EALLOW and LOCK	Go
1Ch	HICDBADDR4	Device Base Address Register 4	EALLOW and LOCK	Go
1Eh	HICDBADDR5	Device Base Address Register 5	EALLOW and LOCK	Go
20h	HICDBADDR6	Device Base Address Register 6	EALLOW and LOCK	Go
22h	HICDBADDR7	Device Base Address Register 7	EALLOW and LOCK	Go
28h	HICH2DINTEN	H2D Interrupt Enable	No	Go
2Ah	HICH2DINTFLG	H2D Interrupt status Flag	No	Go
2Ch	HICH2DINTCLR	H2D Interrupt status Clear	No	Go
2Eh	HICH2DINTFRC	H2D Interrupt Set Force	No	Go
30h	HICD2HINTEN	D2H Interrupt Enable	No	Go
32h	HICD2HINTFLG	D2H Interrupt status Flag	No	Go
34h	HICD2HINTCLR	D2H Interrupt status Clear	No	Go
36h	HICD2HINTFRC	D2H Interrupt Set Force	No	Go
38h	HICACCVIOADDR	Access Violation Address		Go
3Ah	HICCOMMIT	Commit Register	EALLOW	Go
40h	H2D_BUF0	Host to Device Buffer 0	No	Go
42h	H2D_BUF1	Host to Device Buffer 1	No	Go
44h	H2D_BUF2	Host to Device Buffer 2	No	Go
46h	H2D_BUF3	Host to Device Buffer 3	No	Go
48h	H2D_BUF4	Host to Device Buffer 4	No	Go
4Ah	H2D_BUF5	Host to Device Buffer 5	No	Go

Table 14-9. HIC_CFG_REGS Registers (continued)

Offset	Acronym	Register Name	Write Protection	Section
4Ch	H2D_BUF6	Host to Device Buffer 6	No	Go
4Eh	H2D_BUF7	Host to Device Buffer 7	No	Go
50h	H2D_BUF8	Host to Device Buffer 8	No	Go
52h	H2D_BUF9	Host to Device Buffer 9	No	Go
54h	H2D_BUF10	Host to Device Buffer 10	No	Go
56h	H2D_BUF11	Host to Device Buffer 11	No	Go
58h	H2D_BUF12	Host to Device Buffer 12	No	Go
5Ah	H2D_BUF13	Host to Device Buffer 13	No	Go
5Ch	H2D_BUF14	Host to Device Buffer 14	No	Go
5Eh	H2D_BUF15	Host to Device Buffer 15	No	Go
60h	D2H_BUF0	Device to Host Buffer 0	No	Go
62h	D2H_BUF1	Device to Host Buffer 1	No	Go
64h	D2H_BUF2	Device to Host Buffer 2	No	Go
66h	D2H_BUF3	Device to Host Buffer 3	No	Go
68h	D2H_BUF4	Device to Host Buffer 4	No	Go
6Ah	D2H_BUF5	Device to Host Buffer 5	No	Go
6Ch	D2H_BUF6	Device to Host Buffer 6	No	Go
6Eh	D2H_BUF7	Device to Host Buffer 7	No	Go
70h	D2H_BUF8	Device to Host Buffer 8	No	Go
72h	D2H_BUF9	Device to Host Buffer 9	No	Go
74h	D2H_BUF10	Device to Host Buffer 10	No	Go
76h	D2H_BUF11	Device to Host Buffer 11	No	Go
78h	D2H_BUF12	Device to Host Buffer 12	No	Go
7Ah	D2H_BUF13	Device to Host Buffer 13	No	Go
7Ch	D2H_BUF14	Device to Host Buffer 14	No	Go
7Eh	D2H_BUF15	Device to Host Buffer 15	No	Go

Complex bit access types are encoded to fit into small table cells. Table 14-10 shows the codes that are used for access types in this section.

Table 14-10. HIC_CFG_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 14-10. HIC_CFG_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

14.6.2.1 HICREV Register (Offset = 0h) [Reset = 40000001h]

HICREV is shown in Figure 14-10 and described in Table 14-11.

Return to the Summary Table.

Module Revision Register

Figure 14-10. HICREV Register

31	30	29	28	27	26	25	24
SCHEME		RESERVED		FUNC			
R-1h		R-0h		R-0h			
23	22	21	20	19	18	17	16
FUNC							
R-0h							
15	14	13	12	11	10	9	8
RTL					MAJOR		
R-0h					R-0h		
7	6	5	4	3	2	1	0
CUSTOM		MINOR					
R-0h		R-1h					

Table 14-11. HICREV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	SCHEME	R	1h	This identifies the scheme of the module. Returns 01. Read-only for Host Reset type: SYSRSn
29-28	RESERVED	R	0h	Reserved
27-16	FUNC	R	0h	Functional Release Number Reflects software-compatability. If there is no level of software compatability, a unique func number is assigned for compatible modules, the same number is maintained. Read-only for Host Reset type: SYSRSn
15-11	RTL	R	0h	Design Release Number Incremented for releases due to spec changes or post-release design changes. Reset to 0 when either MAJOR or MINOR is incremented. Read-only for Host Reset type: SYSRSn
10-8	MAJOR	R	0h	Major Revision Number Represents major changes to the module (e.g. entirely new features are added/changed). The major revision number for this module. Read-only for Host Reset type: SYSRSn
7-6	CUSTOM	R	0h	Custom Module Number Indicates a special version of the module. May not be supported by standard software. Read-only for Host Reset type: SYSRSn
5-0	MINOR	R	1h	Minor Revision Number Represents minor changes to the module (e.g. enhancements to existing features). The minor revision number for this module. Read-only for Host Reset type: SYSRSn

14.6.2.2 HICGCR Register (Offset = 2h) [Reset = 0h]

HICGCR is shown in Figure 14-11 and described in Table 14-12.

Return to the Summary Table.

Global Control Register

Figure 14-11. HICGCR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												HICEN			
R-0h												R/W-0h			

Table 14-12. HICGCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3-0	HICEN	R/W	0h	Host Interface Enable. Controls the operation of the Host Interface. 0xA : Host Interface is enabled. Access to MBOX and Device region are enabled for Host. Others : Host Interface is disabled. Access to MBOX and Device region are disabled for Host. Host can still read all the MMRs and write to writable MMRs of HIC module. Read-only for Host Reset type: SYSRSn

14.6.2.3 HICLOCK Register (Offset = 4h) [Reset = 0h]

HICLOCK is shown in Figure 14-12 and described in Table 14-13.

Return to the Summary Table.

Lock Register

Figure 14-12. HICLOCK Register

31	30	29	28	27	26	25	24
WRITE_ENABLE_KEY							
W-0h							
23	22	21	20	19	18	17	16
WRITE_ENABLE_KEY							
W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							LOCK
R-0h							R/W-0h

Table 14-13. HICLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	WRITE_ENABLE_KEY	W	0h	These 16 bits act as a key to enable writes to Bit 0 of this register. The only time a '1' can be written to Bit 0 is by a single 32-bit write where bits 31:16 equal 0x5a5a. All other writes are ignored including separate 16-bit writes. Read returns 0 for this field always. Read-only for Host Reset type: SYSRSn
15-1	RESERVED	R	0h	Reserved
0	LOCK	R/W	0h	Enable for LOCK feature that blocks writes to certain config registers. Writing '1' to this field along with a valid WRITE_ENABLE_KEY value while HICCOMMIT.COMMIT=0 will set this bit to '1' and enables the LOCK feature. Writing '0' to this field along with a valid WRITE_ENABLE_KEY value while HICCOMMIT.COMMIT=0 will set this bit to '0' and disables the LOCK feature. 0 : LOCK is disabled 1 : LOCK is enabled Read-only for Host Reset type: SYSRSn

14.6.2.4 HICMODECR Register (Offset = 6h) [Reset = 0h]

HICMODECR is shown in Figure 14-13 and described in Table 14-14.

Return to the Summary Table.

Mode Control Register

Figure 14-13. HICMODECR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED				EN_HOSTWREALLOW	EN_DEVACC	D2HBUF_HOSTWREN	H2DBUF_DEVWREN
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	RDY_PRESENT	BEN_PRESENT	RW_MODE	RESERVED		DW_MODE	
R-0h	R/W-0h	R/W-0h	R/W-0h	R-0h		R/W-0h	

Table 14-14. HICMODECR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-12	RESERVED	R	0h	Reserved
11	EN_HOSTWREALLOW	R/W	0h	Enables Host write to HOSTCR.EALLOW register field 0 : Host write to HOSTCR.EALLOW register field will be ignored 1 : Host can write and reprogram HOSTCR.EALLOW register field Read-only for Host Reset type: SYSRSn
10	EN_DEVACC	R/W	0h	Enables Host accesses to Device Region 0 : Host cannot access Device region - only HIC internal MMR and MBOX regions accessible 1 : Host can access both HIC Internal address space and Device address space selected by BASEx+ registers Read-only for Host Reset type: SYSRSn
9	D2HBUF_HOSTWREN	R/W	0h	D2H Buffer Write Enable for Host 0 : D2H Buffer can only be written by Device 1 : D2H Buffer can also be written by Host For the cases where user wants larger D2H Buffer region (and there is limited or no need for H2D Buffer region), this feature helps. Read-only for Host Reset type: SYSRSn
8	H2DBUF_DEVWREN	R/W	0h	H2D Buffer Write Enable for Device 0 : H2D Buffer can only be written by Host 1 : H2D Buffer can also be written by Device For the cases where user wants larger H2D Buffer region (and there is limited or no need for D2H Buffer region), this feature helps. Read-only for Host Reset type: SYSRSn
7	RESERVED	R	0h	Reserved

Table 14-14. HICMODECR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	RDY_PRESENT	R/W	0h	Ready pin Present Defines the presence of nRDY pin. 0 : "nRDY" pin is not present. Host must ensure to retain the control pins active for the period required by the datasheet. 1 : "nRDY" pin is present. Host must hold the control/data signals as long as nRDY signal is asserted. Read-only for Host Reset type: SYSRSn
5	BEN_PRESENT	R/W	0h	ByteEnable pin Present. Defines the presence of Byte Enable pins. 0 : "nBE" pins are not present. DW_MODE field defines the fixed data width. 1 : "nBE" pins are present. Data bus for each access is qualified with nBE pin status. Read-only for Host Reset type: SYSRSn
4	RW_MODE	R/W	0h	Read-Write Mode. Defines pins to control read-write operation. 0 : Both "nOE" and "nWE" pins are available to control read and write operations. 1 : The "nOE" pin will act as a single "RnW" pin to control both read and write operations. Read-only for Host Reset type: SYSRSn
3-2	RESERVED	R	0h	Reserved
1-0	DW_MODE	R/W	0h	Data Width Mode. Data width of host access 0x0 : 8-bit Data Port 0x1 : 16-bit Data Port others : default to 16-bit mode Read-only for Host Reset type: SYSRSn

14.6.2.5 HICPINPOLCR Register (Offset = 8h) [Reset = 0h]

HICPINPOLCR is shown in Figure 14-14 and described in Table 14-15.

Return to the Summary Table.

Pin Polarity Control Register

Figure 14-14. HICPINPOLCR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			RDY_POL	WE_POL	OE_POL	BEN_POL	CS_POL
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 14-15. HICPINPOLCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reserved
4	RDY_POL	R/W	0h	Defines the Polarity of Ready (nRDY) output pin. 0 : RDY pin is active low 1 : RDY pin is active high Read-only for Host Reset type: SYSRSn
3	WE_POL	R/W	0h	Defines the Polarity of Write Enable (nWE) input pin. 0 : Write Enable pin is active low 1 : Write Enable pin is active high. If HIC is configured to use a single RnW pin, then this bit configuration is ignored. Read-only for Host Reset type: SYSRSn
2	OE_POL	R/W	0h	Defines the Polarity of Output Enable (nOE) input pin. 0 : Output Enable pin is active low 1 : Output Enable pin is active high. If HIC is configured to use a single RnW pin, then this bit configuration is ignored. Read-only for Host Reset type: SYSRSn
1	BEN_POL	R/W	0h	Defines Polarity of Byte Enable (nBE) input pins. 0 : Byte Enable is active low 1 : Byte Enable is active high Read-only for Host Reset type: SYSRSn
0	CS_POL	R/W	0h	Defines Polarity of Chip Select (nCS) input pin. 0 : Chip Select pin is active low 1 : Chip Select is active high. Read-only for Host Reset type: SYSRSn

14.6.2.6 HICBASESEL Register (Offset = Ah) [Reset = 0h]

HICBASESEL is shown in Figure 14-15 and described in Table 14-16.

Return to the Summary Table.

Base Select Register

Figure 14-15. HICBASESEL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					BASE_SELECT		
R-0h					R/W-0h		

Table 14-16. HICBASESEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reserved
2-0	BASE_SELECT	R/W	0h	Selection value for the collection of Base Address registers 000 : Select Base Address 0 register 001 : Select Base Address 1 register 010 : Select Base Address 2 register 011 : Select Base Address 3 register 100 : Select Base Address 4 register 101 : Select Base Address 5 register 110 : Select Base Address 6 register 111 : Select Base Address 7 register Read-Write for Host. Reset type: SYSRSn

14.6.2.7 HICHOSTCR Register (Offset = Ch) [Reset = 0h]

HICHOSTCR is shown in Figure 14-16 and described in Table 14-17.

Return to the Summary Table.

HIC Host Control Register

Figure 14-16. HICHOSTCR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
HKEY							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED					PAGESEL	ACCSIZE	EALLOW_EN
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 14-17. HICHOSTCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	HKEY	R/W	0h	KEY that enables/disables the writes to HOSTCR[7:0] fields. The HOSTCR[7:0] fields can be written only if the HKEY is set to 0xA5 during the write. Writes with any other pattern on HKEY field will be ignored. Read from this field returns 0 always. Read-Write for Host. Reset type: SYSRSn
7-3	RESERVED	R	0h	Reserved
2	PAGESEL	R/W	0h	Selection between BASE_SEL pins and BASE_SELECT register 0 : BASE_SELECT register value selects the required HICDBADDRx register for a Host access to the Device region 1 : BASE_SEL pin values select the required HICDBADDRx register for a Host access to the Device region This field can be written only in conjunction with HKEY field. Read-Write for Host. Reset type: SYSRSn
1	ACCSIZE	R/W	0h	Selection between 16 or 32 bit accesses to the destinations of HIC. 0 : Writes on the Host Port - MMR or Initiator port writes are of 16-bit size. For 8-bit Host, data will be packed to 16-bits before triggering a write to the MMR or the Initiator port. Reads on the Host Port - 16-bit reads to MMR or Initiator Port 1 : Writes on the Host Port - MMR or Initiator port writes are of 32-bit size. For 8/16-bit Host, data will be packed to 32-bit before initiating an access to the MMR or Initiator port Reads on the Host Port - 32-bit reads to MMR or Initiator Port This field can be written only in conjunction with HKEY field. Read-Write for Host. Reset type: SYSRSn

Table 14-17. HICHOSTCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	EALLOW_EN	R/W	0h	Defines the EALLOW signal for the HIC Initiator port. 0 : HIC cannot write to peripheral registers protected by EALLOW. 1 : HIC can access peripheral registers protected by EALLOW. This field can be written only in conjuction with HKEY field if HICMODECR.EN_HOSTWREALLOW=1 Read-Write for Host. Reset type: SYSRSn

14.6.2.8 HICERRADDR Register (Offset = Eh) [Reset = 0h]

HICERRADDR is shown in Figure 14-17 and described in Table 14-18.

Return to the Summary Table.

Host Error Address register

Figure 14-17. HICERRADDR Register

31	30	29	28	27	26	25	24
RESERVED	D2H_BASE_SEL			RESERVED			
R-0h	R-0h			R-0h			
23	22	21	20	19	18	17	16
D2H_ERR_ADDR							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	H2D_BASE_SEL			RESERVED			
R-0h	R-0h			R-0h			
7	6	5	4	3	2	1	0
H2D_ERR_ADDR							
R-0h							

Table 14-18. HICERRADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved
30-28	D2H_BASE_SEL	R	0h	BASE_SEL value at the time of Error capturing D2H_ERR_ADDR The value will freeze upon the first capture until the flags - BUSERR_FLG, ILLWR_FLG and ILLRD_FLG of HICD2HINTFLG register are cleared by the user. Upon an error event, the HIC_BASESEL pin values will be captured into this register field while setting corresponding error flag in HICD2HINTFLG register. For the case where PAGESEL=0, user must read BASE_SELECT register to find out the correct value of BASE_SEL. Read-only for Host. Reset type: SYSRSn
27-24	RESERVED	R	0h	Reserved
23-16	D2H_ERR_ADDR	R	0h	Host address captured upon an erroneous transaction captured for Host to service (as a response to D2HINT interrupt) The value will freeze upon the first capture until the flags - BUSERR_FLG, ILLWR_FLG and ILLRD_FLG of HICD2HINTFLG register are cleared by the user. Upon an error event, the Host address will be captured into this register field while setting corresponding error flag in HICD2HINTFLG register. Read-Write for Host. Reset type: SYSRSn
15	RESERVED	R	0h	Reserved

Table 14-18. HICERRADDR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
14-12	H2D_BASE_SEL	R	0h	BASE_SEL value at the time of Error capturing H2D_ERR_ADDR. The value will freeze upon the first capture until the flags - BUSERR_FLG, ILLWR_FLG and ILLRD_FLG of HICH2DINTFLG register are cleared by the user. Upon an error event, the HIC_BASESEL pin values will be captured into this register field while setting corresponding error flag in HICH2DINTFLG register. For the case where PAGESEL=0, user must read BASE_SELECT register to find out the correct value of BASE_SEL. Read-only for Host. Reset type: SYSRSn
11-8	RESERVED	R	0h	Reserved
7-0	H2D_ERR_ADDR	R	0h	Host address captured upon an erroneous transaction captured for Device to service (as a response to H2DINT interrupt) The value will freeze upon the first capture until the flags - BUSERR_FLG, ILLWR_FLG and ILLRD_FLG of HICH2DINTFLG register are cleared by the user. Upon an error event, the Host address will be captured into this register field while setting corresponding error flag in HICH2DINTFLG register. Read-only for Host. Reset type: SYSRSn

14.6.2.9 HICH2DTOKEN Register (Offset = 10h) [Reset = 0h]

HICH2DTOKEN is shown in Figure 14-18 and described in Table 14-19.

Return to the Summary Table.

Host to Device Token Register

Figure 14-18. HICH2DTOKEN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
H2D_TOKEN																															
R/W-0h																															

Table 14-19. HICH2DTOKEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	H2D_TOKEN	R/W	0h	Host To Device Token is a general purpose register that can be used to send any information from Host to the Device software. Typical usage is to send the Buffer Count of the H2DBUF region. Any write to the lower half-word of this register will automatically trigger an interrupt to H2DINT Read-Write for Host. Reset type: SYSRSn

14.6.2.10 HICD2HTOKEN Register (Offset = 12h) [Reset = 0h]

HICD2HTOKEN is shown in Figure 14-19 and described in Table 14-20.

Return to the Summary Table.

Devie to Host Token Register

Figure 14-19. HICD2HTOKEN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D2H_TOKEN																															
R/W-0h																															

Table 14-20. HICD2HTOKEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	D2H_TOKEN	R/W	0h	Device to Host Token is a general purpose register that can be used to send any information from Device to the Host software. Typical usage is to send the Buffer Count of the D2HBUF region. Any write to the lower half-word of this register will automatically trigger an interrupt to D2HINT Read-only for Host. Reset type: SYSRSn

14.6.2.11 HICDBADDR0 Register (Offset = 14h) [Reset = 0h]

HICDBADDR0 is shown in Figure 14-20 and described in Table 14-21.

Return to the Summary Table.

Device Base Address Register 0

Figure 14-20. HICDBADDR0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BASE_ADDR															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE_ADDR									RESERVED						
R/W-0h									R-0h						

Table 14-21. HICDBADDR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	BASE_ADDR	R/W	0h	Base address of the region inside Device address space that Host intends to access. The valide bits from register are concatenated to the incoming Host address appropriately to form the full 32-bit address on the Initiator port of HIC. When DW_MODE = 8-bit, BASE_ADDR[31..7] should be programmed. In this case, Host can access 256 bytes of Device region through the Initiator port. When DW_MODE = 16-bit, BASE_ADDR[31..8] should be programmed. In this case, Host can access 512 bytes of Device region through the Initiator port. Note: Exact HICDBADDR0 register being used for an access is determined by the HICBASESEL.BASE_SELECT field value. Read-only for Host. Reset type: SYSRSn
6-0	RESERVED	R	0h	Reserved

14.6.2.12 HICDBADDR1 Register (Offset = 16h) [Reset = 0h]

HICDBADDR1 is shown in Figure 14-21 and described in Table 14-22.

Return to the Summary Table.

Device Base Address Register 1

Figure 14-21. HICDBADDR1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BASE_ADDR															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE_ADDR									RESERVED						
R/W-0h									R-0h						

Table 14-22. HICDBADDR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	BASE_ADDR	R/W	0h	Base address of the region inside Device address space that Host intends to access. The valide bits from register are concatenated to the incoming Host address appropriately to form the full 32-bit address on the Initiator port of HIC. When DW_MODE = 8-bit, BASE_ADDR[31..7] should be programmed. In this case, Host can access 256 bytes of Device region through the Initiator port. When DW_MODE = 16-bit, BASE_ADDR[31..8] should be programmed. In this case, Host can access 512 bytes of Device region through the Initiator port. Note: Exact HICDBADDR1 register being used for an access is determined by the HICBASESEL.BASE_SELECT field value. Read-only for Host. Reset type: SYSRSn
6-0	RESERVED	R	0h	Reserved

14.6.2.13 HICDBADDR2 Register (Offset = 18h) [Reset = 0h]

HICDBADDR2 is shown in Figure 14-22 and described in Table 14-23.

Return to the Summary Table.

Device Base Address Register 2

Figure 14-22. HICDBADDR2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BASE_ADDR															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE_ADDR									RESERVED						
R/W-0h									R-0h						

Table 14-23. HICDBADDR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	BASE_ADDR	R/W	0h	Base address of the region inside Device address space that Host intends to access. The valide bits from register are concatenated to the incoming Host address appropriately to form the full 32-bit address on the Initiator port of HIC. When DW_MODE = 8-bit, BASE_ADDR[31..7] should be programmed. In this case, Host can access 256 bytes of Device region through the Initiator port. When DW_MODE = 16-bit, BASE_ADDR[31..8] should be programmed. In this case, Host can access 512 bytes of Device region through the Initiator port. Note: Exact HICDBADDR2 register being used for an access is determined by the HICBASESEL.BASE_SELECT field value. Read-only for Host. Reset type: SYSRSn
6-0	RESERVED	R	0h	Reserved

14.6.2.14 HICDBADDR3 Register (Offset = 1Ah) [Reset = 0h]

HICDBADDR3 is shown in Figure 14-23 and described in Table 14-24.

Return to the Summary Table.

Device Base Address Register 3

Figure 14-23. HICDBADDR3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BASE_ADDR															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE_ADDR									RESERVED						
R/W-0h									R-0h						

Table 14-24. HICDBADDR3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	BASE_ADDR	R/W	0h	Base address of the region inside Device address space that Host intends to access. The valide bits from register are concatenated to the incoming Host address appropriately to form the full 32-bit address on the Initiator port of HIC. When DW_MODE = 8-bit, BASE_ADDR[31..7] should be programmed. In this case, Host can access 256 bytes of Device region through the Initiator port. When DW_MODE = 16-bit, BASE_ADDR[31..8] should be programmed. In this case, Host can access 512 bytes of Device region through the Initiator port. Note: Exact HICDBADDR3 register being used for an access is determined by the HICBASESEL.BASE_SELECT field value. Read-only for Host. Reset type: SYSRSn
6-0	RESERVED	R	0h	Reserved

14.6.2.15 HICDBADDR4 Register (Offset = 1Ch) [Reset = 0h]

HICDBADDR4 is shown in Figure 14-24 and described in Table 14-25.

Return to the Summary Table.

Device Base Address Register 4

Figure 14-24. HICDBADDR4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BASE_ADDR															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE_ADDR									RESERVED						
R/W-0h									R-0h						

Table 14-25. HICDBADDR4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	BASE_ADDR	R/W	0h	Base address of the region inside Device address space that Host intends to access. The valide bits from register are concatenated to the incoming Host address appropriately to form the full 32-bit address on the Initiator port of HIC. When DW_MODE = 8-bit, BASE_ADDR[31..7] should be programmed. In this case, Host can access 256 bytes of Device region through the Initiator port. When DW_MODE = 16-bit, BASE_ADDR[31..8] should be programmed. In this case, Host can access 512 bytes of Device region through the Initiator port. Note: Exact HICDBADDR4 register being used for an access is determined by the HICBASESEL.BASE_SELECT field value. Read-only for Host. Reset type: SYSRSn
6-0	RESERVED	R	0h	Reserved

14.6.2.16 HICDBADDR5 Register (Offset = 1Eh) [Reset = 0h]

HICDBADDR5 is shown in Figure 14-25 and described in Table 14-26.

Return to the Summary Table.

Device Base Address Register 5

Figure 14-25. HICDBADDR5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BASE_ADDR															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE_ADDR									RESERVED						
R/W-0h									R-0h						

Table 14-26. HICDBADDR5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	BASE_ADDR	R/W	0h	Base address of the region inside Device address space that Host intends to access. The valide bits from register are concatenated to the incoming Host address appropriately to form the full 32-bit address on the Initiator port of HIC. When DW_MODE = 8-bit, BASE_ADDR[31..7] should be programmed. In this case, Host can access 256 bytes of Device region through the Initiator port. When DW_MODE = 16-bit, BASE_ADDR[31..8] should be programmed. In this case, Host can access 512 bytes of Device region through the Initiator port. Note: Exact HICDBADDR5 register being used for an access is determined by the HICBASESEL.BASE_SELECT field value. Read-only for Host. Reset type: SYSRSn
6-0	RESERVED	R	0h	Reserved

14.6.2.17 HICDBADDR6 Register (Offset = 20h) [Reset = 0h]

HICDBADDR6 is shown in Figure 14-26 and described in Table 14-27.

Return to the Summary Table.

Device Base Address Register 6

Figure 14-26. HICDBADDR6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BASE_ADDR															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE_ADDR									RESERVED						
R/W-0h									R-0h						

Table 14-27. HICDBADDR6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	BASE_ADDR	R/W	0h	Base address of the region inside Device address space that Host intends to access. The valide bits from register are concatenated to the incoming Host address appropriately to form the full 32-bit address on the Initiator port of HIC. When DW_MODE = 8-bit, BASE_ADDR[31..7] should be programmed. In this case, Host can access 256 bytes of Device region through the Initiator port. When DW_MODE = 16-bit, BASE_ADDR[31..8] should be programmed. In this case, Host can access 512 bytes of Device region through the Initiator port. Note: Exact HICDBADDR6 register being used for an access is determined by the HICBASESEL.BASE_SELECT field value. Read-only for Host. Reset type: SYSRSn
6-0	RESERVED	R	0h	Reserved

14.6.2.18 HICDBADDR7 Register (Offset = 22h) [Reset = 0h]

HICDBADDR7 is shown in Figure 14-27 and described in Table 14-28.

Return to the Summary Table.

Device Base Address Register 7

Figure 14-27. HICDBADDR7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BASE_ADDR															
R/W-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BASE_ADDR									RESERVED						
R/W-0h									R-0h						

Table 14-28. HICDBADDR7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	BASE_ADDR	R/W	0h	Base address of the region inside Device address space that Host intends to access. The valide bits from register are concatenated to the incoming Host address appropriately to form the full 32-bit address on the Initiator port of HIC. When DW_MODE = 8-bit, BASE_ADDR[31..7] should be programmed. In this case, Host can access 256 bytes of Device region through the Initiator port. When DW_MODE = 16-bit, BASE_ADDR[31..8] should be programmed. In this case, Host can access 512 bytes of Device region through the Initiator port. Note: Exact HICDBADDR7 register being used for an access is determined by the HICBASESEL.BASE_SELECT field value. Read-only for Host. Reset type: SYSRSn
6-0	RESERVED	R	0h	Reserved

14.6.2.19 HICH2DINTEN Register (Offset = 28h) [Reset = 0h]

HICH2DINTEN is shown in Figure 14-28 and described in Table 14-29.

Return to the Summary Table.

H2D Interrupt Enable

Figure 14-28. HICH2DINTEN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				ILLRD_INTEN	ILLWR_INTEN	BUSERR_INTEN	H2D_INTEN
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 14-29. HICH2DINTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	ILLRD_INTEN	R/W	0h	Enables Illegal Read Error Interrupt 0 : Illegal Read event does not result in an interrupt 1 : Illegal Read event will trigger an interrupt on H2DINT port Read-Write for Host. Reset type: SYSRSn
2	ILLWR_INTEN	R/W	0h	Enables Illegal Write Error Interrupt 0 : Illegal Write event does not result in an interrupt 1 : Illegal Write event will trigger an interrupt on H2DINT port Read-Write for Host. Reset type: SYSRSn
1	BUSERR_INTEN	R/W	0h	Enables Bus Error Interrupt 0 : BusError does not result in an interrupt 1 : BusError event will trigger an interrupt on H2DINT port Read-Write for Host. Reset type: SYSRSn
0	H2D_INTEN	R/W	0h	Enables Host-to-Device data ready Interrupt 0 : Host-to-Device data ready does not result in an interrupt 1 : Host-to-Device data ready will trigger an interrupt on H2DINT port Read-Write for Host. Reset type: SYSRSn

14.6.2.20 HICH2DINTFLG Register (Offset = 2Ah) [Reset = 0h]

HICH2DINTFLG is shown in Figure 14-29 and described in Table 14-30.

Return to the Summary Table.

H2D Interrupt status Flag

Figure 14-29. HICH2DINTFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				ILLRD_FLG	ILLWR_FLG	BUSERR_FLG	H2D_FLG
R-0h				R-0h	R-0h	R-0h	R-0h

Table 14-30. HICH2DINTFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	ILLRD_FLG	R	0h	Status of Illegal Read Error Interrupt 0 : Illegal Read event is not detected 1 : Illegal Read event is detected Read-only for Host. Reset type: SYSRSn
2	ILLWR_FLG	R	0h	Status of Illegal Write Error Interrupt 0 : Illegal Write event is not detected 1 : Illegal Write event is detected Read-only for Host. Reset type: SYSRSn
1	BUSERR_FLG	R	0h	Status of Bus Error event 0 : BusError event is not detected 1 : BusError event has occurred BusError is generated when there is a loss of data due to simultaneous write access by Host as well as Device to a single register. Read-only for Host. Reset type: SYSRSn
0	H2D_FLG	R	0h	Status of Host-to-Device data ready event 0 : Host-to-Device data ready event is not detected 1 : Host-to-Device data ready event is detected Read-only for Host. Reset type: SYSRSn

14.6.2.21 HICH2DINTCLR Register (Offset = 2Ch) [Reset = 0h]

HICH2DINTCLR is shown in Figure 14-30 and described in Table 14-31.

Return to the Summary Table.

H2D Interrupt status Clear

Figure 14-30. HICH2DINTCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				ILLRD_CLR	ILLWR_CLR	BUSERR_CLR	H2D_CLR
R-0h				R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

Table 14-31. HICH2DINTCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	ILLRD_CLR	R/W1C	0h	Clear Illegal Read error flag and hence the interrupt Writing a '1' clears this flag. Writing '0' has no impact. Read returns '0' always. Read-Write-to-Clear for Host. Reset type: SYSRSn
2	ILLWR_CLR	R/W1C	0h	Clear Illegal Write error flag and hence the interrupt Writing a '1' clears this flag. Writing '0' has no impact. Read returns '0' always. Read-Write-to-Clear for Host. Reset type: SYSRSn
1	BUSERR_CLR	R/W1C	0h	Clear Bus Error flag and hence the interrupt. Writing a '1' clears this flag. Writing '0' has no impact. Read returns '0' always. Read-Write-to-Clear for Host. Reset type: SYSRSn
0	H2D_CLR	R/W1C	0h	Clear Host-to-Device data ready flag and hence the interrupt Writing a '1' clears this flag. Writing '0' has no impact. Read returns '0' always. Read-Write-to-Clear for Host. Reset type: SYSRSn

14.6.2.22 HICH2DINTFRC Register (Offset = 2Eh) [Reset = 0h]

HICH2DINTFRC is shown in Figure 14-31 and described in Table 14-32.

Return to the Summary Table.

H2D Interrupt Set Force

Figure 14-31. HICH2DINTFRC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				ILLRD_INTFRC	ILLWR_INTFRC	BUSERR_INTFRC	H2D_INTFRC
R-0h				R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

Table 14-32. HICH2DINTFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	ILLRD_INTFRC	R/W1S	0h	Force the Illegal Read Interrupt Writing a '1' will set this bit. Writing '0' has no impact. Read returns '0' always. Read-Write-to-Set for Host. Reset type: SYSRSn
2	ILLWR_INTFRC	R/W1S	0h	Force the Illegal Write Interrupt Writing a '1' will set this bit. Writing '0' has no impact. Read returns '0' always. Read-Write-to-Set for Host. Reset type: SYSRSn
1	BUSERR_INTFRC	R/W1S	0h	Force the Bus Error Interrupt Writing a '1' to this bit will set BUSERR_FLG Writing '0' has no impact. Read returns '0' always. Read-Write-to-Set for Host. Reset type: SYSRSn
0	H2D_INTFRC	R/W1S	0h	Force the Host-to-Device data ready Interrupt Writing a '1' will set H2D_FLG bit. Writing '0' has no impact. Read returns '0' always. Read-Write-to-Set for Host. Reset type: SYSRSn

14.6.2.23 HICD2HINTEN Register (Offset = 30h) [Reset = 0h]

HICD2HINTEN is shown in Figure 14-32 and described in Table 14-33.

Return to the Summary Table.

D2H Interrupt Enable

Figure 14-32. HICD2HINTEN Register

31	30	29	28	27	26	25	24
EVTRIG_INTEN							
R/W-0h							
23	22	21	20	19	18	17	16
EVTRIG_INTEN							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			ACCVIO_INTEN	ILLRD_INTEN	ILLWR_INTEN	BUSERR_INTEN	D2H_INTEN
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 14-33. HICD2HINTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	EVTRIG_INTEN	R/W	0h	Enables Interrupts upon Event Triggers 0 : Upon an event on EVT_TRIG[n] (n=0..15), no interrupts will be triggered on D2HINT port 1 : An Interrupt is triggered on D2HINT port when an active high event is detected on EVT_TRIG[n] (n=0..15) Read-Write for Host. Reset type: SYSRSn
15-5	RESERVED	R	0h	Reserved
4	ACCVIO_INTEN	R/W	0h	Enables Access Violation (for Device accesses) Interrupt 0 : Access Violation does not result in an interrupt 1 : Access Violation will trigger an interrupt on D2HINT port Read-Write for Host. Reset type: SYSRSn
3	ILLRD_INTEN	R/W	0h	Enables Illegal Read Error Interrupt 0 : Illegal Read event does not result in an interrupt 1 : Illegal Read event will trigger an interrupt on D2HINT port Read-Write for Host. Reset type: SYSRSn
2	ILLWR_INTEN	R/W	0h	Enables Illegal Write Error Interrupt 0 : Illegal Write event does not result in an interrupt 1 : Illegal Write event will trigger an interrupt on D2HINT port Read-Write for Host. Reset type: SYSRSn
1	BUSERR_INTEN	R/W	0h	Enables Bus Error Interrupt 0 : BusError does not result in an interrupt 1 : BusError event will trigger an interrupt on D2HINT port Read-Write for Host. Reset type: SYSRSn

Table 14-33. HICD2HINTEN Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	D2H_INTEN	R/W	0h	Enables Device-to-Host data ready Interrupt 0 : Device-to-Host data ready does not result in an interrupt 1 : Device-to-Host data ready will trigger an interrupt on D2HINT port Read-Write for Host. Reset type: SYSRSn

14.6.2.24 HICD2HINTFLG Register (Offset = 32h) [Reset = 0h]

HICD2HINTFLG is shown in Figure 14-33 and described in Table 14-34.

Return to the Summary Table.

D2H Interrupt status Flag

Figure 14-33. HICD2HINTFLG Register

31	30	29	28	27	26	25	24
EVTRIG_FLG							
R-0h							
23	22	21	20	19	18	17	16
EVTRIG_FLG							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			ACCVIO_FLG	ILLRD_FLG	ILLWR_FLG	BUSERR_FLG	D2H_FLG
R-0h			R-0h	R-0h	R-0h	R-0h	R-0h

Table 14-34. HICD2HINTFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	EVTRIG_FLG	R	0h	Status of Event Trigger 0 : No event detected on EVT_TRIG[n] (n=0..15) 1 : Event detected on EVT_TRIG[n] (n=0..15) Read-only for Host. Reset type: SYSRSn
15-5	RESERVED	R	0h	Reserved
4	ACCVIO_FLG	R	0h	Status of Access Violation Error 0 : Maste Port Access Violation event is not detected 1 : Initiator Port Access Violation event is detected Read-only for Host. Reset type: SYSRSn
3	ILLRD_FLG	R	0h	Status of Illegal Read Error 0 : Illegal Read event is not detected 1 : Illegal Read event is detected Read-only for Host. Reset type: SYSRSn
2	ILLWR_FLG	R	0h	Status of Illegal Write Error 0 : Illegal Write event is not detected 1 : Illegal Write event is detected Read-only for Host. Reset type: SYSRSn
1	BUSERR_FLG	R	0h	Status of Bus Error event 0 : BusError event is not detected 1 : BusError event has occurred BusError is generated when there is a loss of data due to simultaneous write access by Host as well as Device to a single register. Read-only for Host. Reset type: SYSRSn

Table 14-34. HICD2HINTFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	D2H_FLG	R	0h	Status of Device-to-Host data ready event 0 : Device-to-Host data ready event is not detected 1 : Device-to-Host data ready event is detected Read-only for Host. Reset type: SYSRSn

14.6.2.25 HICD2HINTCLR Register (Offset = 34h) [Reset = 0h]

HICD2HINTCLR is shown in Figure 14-34 and described in Table 14-35.

Return to the Summary Table.

D2H Interrupt status Clear

Figure 14-34. HICD2HINTCLR Register

31	30	29	28	27	26	25	24
EVTRIG_CLR							
R/W1C-0h							
23	22	21	20	19	18	17	16
EVTRIG_CLR							
R/W1C-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			ACCVIO_CLR	ILLRD_CLR	ILLWR_CLR	BUSERR_CLR	D2H_CLR
R-0h			R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

Table 14-35. HICD2HINTCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	EVTRIG_CLR	R/W1C	0h	Clear Event Trigger flag and hence the interrupt Writing a '1' clears this flag EVTRIG_FLG[n] (n=0..15) Writing '0' has no impact. Read returns '0' always. Read-Write-to-Clear for Host. Reset type: SYSRSn
15-5	RESERVED	R	0h	Reserved
4	ACCVIO_CLR	R/W1C	0h	Clear Access Violation error flag and hence the interrupt Writing a '1' clears this flag. Writing '0' has no impact. Read returns '0' always. Read-Write-to-Clear for Host. Reset type: SYSRSn
3	ILLRD_CLR	R/W1C	0h	Clear Illegal Read error flag and hence the interrupt Writing a '1' clears this flag. Writing '0' has no impact. Read returns '0' always. Read-Write-to-Clear for Host. Reset type: SYSRSn
2	ILLWR_CLR	R/W1C	0h	Clear Illegal Write error flag and hence the interrupt Writing a '1' clears this flag. Writing '0' has no impact. Read returns '0' always. Read-Write-to-Clear for Host. Reset type: SYSRSn
1	BUSERR_CLR	R/W1C	0h	Clear Bus Error flag and hence the interrupt. Writing a '1' clears this flag. Writing '0' has no impact. Read returns '0' always. Read-Write-to-Clear for Host. Reset type: SYSRSn

Table 14-35. HICD2HINTCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	D2H_CLR	R/W1C	0h	Clear Device-to-Host data ready flag and hence the interrupt Writing a '1' clears this flag. Writing '0' has no impact. Read returns '0' always. Read-Write-to-Clear for Host. Reset type: SYSRSn

14.6.2.26 HICD2HINTFRC Register (Offset = 36h) [Reset = 0h]

HICD2HINTFRC is shown in Figure 14-35 and described in Table 14-36.

Return to the Summary Table.

D2H Interrupt Set Force

Figure 14-35. HICD2HINTFRC Register

31	30	29	28	27	26	25	24
EVTRIG_INTFRC							
R/W1S-0h							
23	22	21	20	19	18	17	16
EVTRIG_INTFRC							
R/W1S-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			ACCVIO_INTFRC	ILLRD_INTFRC	ILLWR_INTFRC	BUSERR_INTFRC	D2H_INTFRC
R-0h			R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

Table 14-36. HICD2HINTFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	EVTRIG_INTFRC	R/W1S	0h	Force the Event Trigger Interrupt Writing a '1' will set this bit EVTRIG_FLG[n] (n=0..15) Writing '0' has no impact. Read returns '0' always. Read-Write-to-Set for Host. Reset type: SYSRSn
15-5	RESERVED	R	0h	Reserved
4	ACCVIO_INTFRC	R/W1S	0h	Force the Access Violation Interrupt Writing a '1' will set this bit. Writing '0' has no impact. Read returns '0' always. Read-Write-to-Set for Host. Reset type: SYSRSn
3	ILLRD_INTFRC	R/W1S	0h	Force the Illegal Read Interrupt Writing a '1' will set this bit. Writing '0' has no impact. Read returns '0' always. Read-Write-to-Set for Host. Reset type: SYSRSn
2	ILLWR_INTFRC	R/W1S	0h	Force the Illegal Write Interrupt Writing a '1' will set this bit. Writing '0' has no impact. Read returns '0' always. Read-Write-to-Set for Host. Reset type: SYSRSn
1	BUSERR_INTFRC	R/W1S	0h	Force the Bus Error Interrupt Writing a '1' to this bit will set BUSERR_FLG Writing '0' has no impact. Read returns '0' always. Read-Write-to-Set for Host. Reset type: SYSRSn

Table 14-36. HICD2HINTFRC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	D2H_INTFRC	R/W1S	0h	Force the Device-to-Host data ready Interrupt Writing a '1' will set this bit. Writing '0' has no impact. Read returns '0' always. Read-Write-to-Set for Host. Reset type: SYSRSn

14.6.2.27 HICACCVIOADDR Register (Offset = 38h) [Reset = 0h]

HICACCVIOADDR is shown in Figure 14-36 and described in Table 14-37.

Return to the Summary Table.

Access Violation Address

Figure 14-36. HICACCVIOADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ACCVIO_ADDR																															
R-0h																															

Table 14-37. HICACCVIOADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ACCVIO_ADDR	R	0h	Address of the Initiator port accessing the Device region that resulted in violations like unimplemented range or access protection violation. The value will freeze upon the first capture until the ACCVIO_FLG of HICD2HINTFLG register are cleared by the user. When an access is issued by HIC to its Initiator port to allow Host to access the Device region, if any violations are reported, the address on the Initiator port will be captured into this register field while setting corresponding error flag in HICD2HINTFLG register. Read-only for Host. Reset type: SYSRSn

14.6.2.28 HICCOMMIT Register (Offset = 3Ah) [Reset = 0h]

HICCOMMIT is shown in Figure 14-37 and described in Table 14-38.

Return to the Summary Table.

Commit Register

Figure 14-37. HICCOMMIT Register

31	30	29	28	27	26	25	24
WRITE_ENABLE_KEY							
W-0h							
23	22	21	20	19	18	17	16
WRITE_ENABLE_KEY							
W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							COMMIT
R-0h							R/WSonce-0h

Table 14-38. HICCOMMIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	WRITE_ENABLE_KEY	W	0h	These 16 bits act as a key to enable writes to Bit 0 of this register. The only time a '1' can be written to Bit 0 is by a single 32-bit write where bits 31:16 equal 0x5a5a. All other writes are ignored including separate 16-bit writes. Read returns 0 for this field always. Read-only for Host Reset type: SYSRSn
15-1	RESERVED	R	0h	Reserved
0	COMMIT	R/WSonce	0h	Enable for COMMIT feature that blocks writes to certain config registers. This field can be written as '1' only along with a valid WRITE_ENABLE_KEY value. 0: Register lock configuration is not committed. 1: Register lock configuration is committed. Once configuration is committed, only reset can change the configuration. Read-only for Host Reset type: SYSRSn

14.6.2.29 H2D_BUF0 Register (Offset = 40h) [Reset = 0h]

H2D_BUF0 is shown in Figure 14-38 and described in Table 14-39.

Return to the Summary Table.

Host to Device Buffer 0

Figure 14-38. H2D_BUF0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-39. H2D_BUF0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Host to be consumed by the Device. These registers can be read and written by primarily HOST, but can also be written by Device if H2DBUF_DEVWREN bit is set '1'. Reset type: SYSRSn

14.6.2.30 H2D_BUF1 Register (Offset = 42h) [Reset = 0h]

H2D_BUF1 is shown in Figure 14-39 and described in Table 14-40.

Return to the Summary Table.

Host to Device Buffer 1

Figure 14-39. H2D_BUF1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-40. H2D_BUF1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Host to be consumed by the Device. These registers can be read and written by primarily HOST, but can also be written by Device if H2DBUF_DEVWREN bit is set '1'. Reset type: SYSRSn

14.6.2.31 H2D_BUF2 Register (Offset = 44h) [Reset = 0h]

H2D_BUF2 is shown in Figure 14-40 and described in Table 14-41.

Return to the Summary Table.

Host to Device Buffer 2

Figure 14-40. H2D_BUF2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-41. H2D_BUF2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Host to be consumed by the Device. These registers can be read and written by primarily HOST, but can also be written by Device if H2DBUF_DEVWREN bit is set '1'. Reset type: SYSRSn

14.6.2.32 H2D_BUF3 Register (Offset = 46h) [Reset = 0h]

H2D_BUF3 is shown in Figure 14-41 and described in Table 14-42.

Return to the Summary Table.

Host to Device Buffer 3

Figure 14-41. H2D_BUF3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-42. H2D_BUF3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Host to be consumed by the Device. These registers can be read and written by primarily HOST, but can also be written by Device if H2DBUF_DEVWREN bit is set '1'. Reset type: SYSRSn

14.6.2.33 H2D_BUF4 Register (Offset = 48h) [Reset = 0h]

H2D_BUF4 is shown in Figure 14-42 and described in Table 14-43.

Return to the Summary Table.

Host to Device Buffer 4

Figure 14-42. H2D_BUF4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-43. H2D_BUF4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Host to be consumed by the Device. These registers can be read and written by primarily HOST, but can also be written by Device if H2DBUF_DEVWREN bit is set '1'. Reset type: SYSRSn

14.6.2.34 H2D_BUF5 Register (Offset = 4Ah) [Reset = 0h]

H2D_BUF5 is shown in Figure 14-43 and described in Table 14-44.

Return to the Summary Table.

Host to Device Buffer 5

Figure 14-43. H2D_BUF5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-44. H2D_BUF5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Host to be consumed by the Device. These registers can be read and written by primarily HOST, but can also be written by Device if H2DBUF_DEVWREN bit is set '1'. Reset type: SYSRSn

14.6.2.35 H2D_BUF6 Register (Offset = 4Ch) [Reset = 0h]

H2D_BUF6 is shown in Figure 14-44 and described in Table 14-45.

Return to the Summary Table.

Host to Device Buffer 6

Figure 14-44. H2D_BUF6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-45. H2D_BUF6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Host to be consumed by the Device. These registers can be read and written by primarily HOST, but can also be written by Device if H2DBUF_DEVWREN bit is set '1'. Reset type: SYSRSn

14.6.2.36 H2D_BUF7 Register (Offset = 4Eh) [Reset = 0h]

H2D_BUF7 is shown in Figure 14-45 and described in Table 14-46.

Return to the Summary Table.

Host to Device Buffer 7

Figure 14-45. H2D_BUF7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-46. H2D_BUF7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Host to be consumed by the Device. These registers can be read and written by primarily HOST, but can also be written by Device if H2DBUF_DEVWREN bit is set '1'. Reset type: SYSRSn

14.6.2.37 H2D_BUF8 Register (Offset = 50h) [Reset = 0h]

H2D_BUF8 is shown in Figure 14-46 and described in Table 14-47.

Return to the Summary Table.

Host to Device Buffer 8

Figure 14-46. H2D_BUF8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-47. H2D_BUF8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Host to be consumed by the Device. These registers can be read and written by primarily HOST, but can also be written by Device if H2DBUF_DEVWREN bit is set '1'. Reset type: SYSRSn

14.6.2.38 H2D_BUF9 Register (Offset = 52h) [Reset = 0h]

H2D_BUF9 is shown in Figure 14-47 and described in Table 14-48.

Return to the Summary Table.

Host to Device Buffer 9

Figure 14-47. H2D_BUF9 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-48. H2D_BUF9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Host to be consumed by the Device. These registers can be read and written by primarily HOST, but can also be written by Device if H2DBUF_DEVWREN bit is set '1'. Reset type: SYSRSn

14.6.2.39 H2D_BUF10 Register (Offset = 54h) [Reset = 0h]

H2D_BUF10 is shown in Figure 14-48 and described in Table 14-49.

Return to the Summary Table.

Host to Device Buffer 10

Figure 14-48. H2D_BUF10 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-49. H2D_BUF10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Host to be consumed by the Device. These registers can be read and written by primarily HOST, but can also be written by Device if H2DBUF_DEVWREN bit is set '1'. Reset type: SYSRSn

14.6.2.40 H2D_BUF11 Register (Offset = 56h) [Reset = 0h]

H2D_BUF11 is shown in Figure 14-49 and described in Table 14-50.

Return to the Summary Table.

Host to Device Buffer 11

Figure 14-49. H2D_BUF11 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-50. H2D_BUF11 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Host to be consumed by the Device. These registers can be read and written by primarily HOST, but can also be written by Device if H2DBUF_DEVWREN bit is set '1'. Reset type: SYSRSn

14.6.2.41 H2D_BUF12 Register (Offset = 58h) [Reset = 0h]

H2D_BUF12 is shown in Figure 14-50 and described in Table 14-51.

Return to the Summary Table.

Host to Device Buffer 12

Figure 14-50. H2D_BUF12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-51. H2D_BUF12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Host to be consumed by the Device. These registers can be read and written by primarily HOST, but can also be written by Device if H2DBUF_DEVWREN bit is set '1'. Reset type: SYSRSn

14.6.2.42 H2D_BUF13 Register (Offset = 5Ah) [Reset = 0h]

H2D_BUF13 is shown in Figure 14-51 and described in Table 14-52.

Return to the Summary Table.

Host to Device Buffer 13

Figure 14-51. H2D_BUF13 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-52. H2D_BUF13 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Host to be consumed by the Device. These registers can be read and written by primarily HOST, but can also be written by Device if H2DBUF_DEVWREN bit is set '1'. Reset type: SYSRSn

14.6.2.43 H2D_BUF14 Register (Offset = 5Ch) [Reset = 0h]

H2D_BUF14 is shown in Figure 14-52 and described in Table 14-53.

Return to the Summary Table.

Host to Device Buffer 14

Figure 14-52. H2D_BUF14 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-53. H2D_BUF14 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Host to be consumed by the Device. These registers can be read and written by primarily HOST, but can also be written by Device if H2DBUF_DEVWREN bit is set '1'. Reset type: SYSRSn

14.6.2.44 H2D_BUF15 Register (Offset = 5Eh) [Reset = 0h]

H2D_BUF15 is shown in Figure 14-53 and described in Table 14-54.

Return to the Summary Table.

Host to Device Buffer 15

Figure 14-53. H2D_BUF15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-54. H2D_BUF15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Host to be consumed by the Device. These registers can be read and written by primarily HOST, but can also be written by Device if H2DBUF_DEVWREN bit is set '1'. Reset type: SYSRSn

14.6.2.45 D2H_BUF0 Register (Offset = 60h) [Reset = 0h]

D2H_BUF0 is shown in Figure 14-54 and described in Table 14-55.

Return to the Summary Table.

Device to Host Buffer 0

Figure 14-54. D2H_BUF0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-55. D2H_BUF0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Device to be consumed by the Host. These registers are primarily written by Device, but can also be written by Host if the D2HBUF_HOSTWREN bit is set '1'. Reset type: SYSRSn

14.6.2.46 D2H_BUF1 Register (Offset = 62h) [Reset = 0h]

D2H_BUF1 is shown in Figure 14-55 and described in Table 14-56.

Return to the Summary Table.

Device to Host Buffer 1

Figure 14-55. D2H_BUF1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-56. D2H_BUF1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Device to be consumed by the Host. These registers are primarily written by Device, but can also be written by Host if the D2HBUF_HOSTWREN bit is set '1'. Reset type: SYSRSn

14.6.2.47 D2H_BUF2 Register (Offset = 64h) [Reset = 0h]

D2H_BUF2 is shown in Figure 14-56 and described in Table 14-57.

Return to the Summary Table.

Device to Host Buffer 2

Figure 14-56. D2H_BUF2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-57. D2H_BUF2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Device to be consumed by the Host. These registers are primarily written by Device, but can also be written by Host if the D2HBUF_HOSTWREN bit is set '1'. Reset type: SYSRSn

14.6.2.48 D2H_BUF3 Register (Offset = 66h) [Reset = 0h]

D2H_BUF3 is shown in Figure 14-57 and described in Table 14-58.

Return to the Summary Table.

Device to Host Buffer 3

Figure 14-57. D2H_BUF3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-58. D2H_BUF3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Device to be consumed by the Host. These registers are primarily written by Device, but can also be written by Host if the D2HBUF_HOSTWREN bit is set '1'. Reset type: SYSRSn

14.6.2.49 D2H_BUF4 Register (Offset = 68h) [Reset = 0h]

D2H_BUF4 is shown in Figure 14-58 and described in Table 14-59.

Return to the Summary Table.

Device to Host Buffer 4

Figure 14-58. D2H_BUF4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-59. D2H_BUF4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Device to be consumed by the Host. These registers are primarily written by Device, but can also be written by Host if the D2HBUF_HOSTWREN bit is set '1'. Reset type: SYSRSn

14.6.2.50 D2H_BUF5 Register (Offset = 6Ah) [Reset = 0h]

D2H_BUF5 is shown in Figure 14-59 and described in Table 14-60.

Return to the Summary Table.

Device to Host Buffer 5

Figure 14-59. D2H_BUF5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-60. D2H_BUF5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Device to be consumed by the Host. These registers are primarily written by Device, but can also be written by Host if the D2HBUF_HOSTWREN bit is set '1'. Reset type: SYSRSn

14.6.2.51 D2H_BUF6 Register (Offset = 6Ch) [Reset = 0h]

D2H_BUF6 is shown in Figure 14-60 and described in Table 14-61.

Return to the Summary Table.

Device to Host Buffer 6

Figure 14-60. D2H_BUF6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-61. D2H_BUF6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Device to be consumed by the Host. These registers are primarily written by Device, but can also be written by Host if the D2HBUF_HOSTWREN bit is set '1'. Reset type: SYSRSn

14.6.2.52 D2H_BUF7 Register (Offset = 6Eh) [Reset = 0h]

D2H_BUF7 is shown in Figure 14-61 and described in Table 14-62.

Return to the Summary Table.

Device to Host Buffer 7

Figure 14-61. D2H_BUF7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-62. D2H_BUF7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Device to be consumed by the Host. These registers are primarily written by Device, but can also be written by Host if the D2HBUF_HOSTWREN bit is set '1'. Reset type: SYSRSn

14.6.2.53 D2H_BUF8 Register (Offset = 70h) [Reset = 0h]

D2H_BUF8 is shown in Figure 14-62 and described in Table 14-63.

Return to the Summary Table.

Device to Host Buffer 8

Figure 14-62. D2H_BUF8 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-63. D2H_BUF8 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Device to be consumed by the Host. These registers are primarily written by Device, but can also be written by Host if the D2HBUF_HOSTWREN bit is set '1'. Reset type: SYSRSn

14.6.2.54 D2H_BUF9 Register (Offset = 72h) [Reset = 0h]

D2H_BUF9 is shown in Figure 14-63 and described in Table 14-64.

Return to the Summary Table.

Device to Host Buffer 9

Figure 14-63. D2H_BUF9 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-64. D2H_BUF9 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Device to be consumed by the Host. These registers are primarily written by Device, but can also be written by Host if the D2HBUF_HOSTWREN bit is set '1'. Reset type: SYSRSn

14.6.2.55 D2H_BUF10 Register (Offset = 74h) [Reset = 0h]

D2H_BUF10 is shown in Figure 14-64 and described in Table 14-65.

Return to the Summary Table.

Device to Host Buffer 10

Figure 14-64. D2H_BUF10 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-65. D2H_BUF10 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Device to be consumed by the Host. These registers are primarily written by Device, but can also be written by Host if the D2HBUF_HOSTWREN bit is set '1'. Reset type: SYSRSn

14.6.2.56 D2H_BUF11 Register (Offset = 76h) [Reset = 0h]

D2H_BUF11 is shown in Figure 14-65 and described in Table 14-66.

Return to the Summary Table.

Device to Host Buffer 11

Figure 14-65. D2H_BUF11 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-66. D2H_BUF11 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Device to be consumed by the Host. These registers are primarily written by Device, but can also be written by Host if the D2HBUF_HOSTWREN bit is set '1'. Reset type: SYSRSn

14.6.2.57 D2H_BUF12 Register (Offset = 78h) [Reset = 0h]

D2H_BUF12 is shown in Figure 14-66 and described in Table 14-67.

Return to the Summary Table.

Device to Host Buffer 12

Figure 14-66. D2H_BUF12 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-67. D2H_BUF12 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Device to be consumed by the Host. These registers are primarily written by Device, but can also be written by Host if the D2HBUF_HOSTWREN bit is set '1'. Reset type: SYSRSn

14.6.2.58 D2H_BUF13 Register (Offset = 7Ah) [Reset = 0h]

D2H_BUF13 is shown in Figure 14-67 and described in Table 14-68.

Return to the Summary Table.

Device to Host Buffer 13

Figure 14-67. D2H_BUF13 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-68. D2H_BUF13 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Device to be consumed by the Host. These registers are primarily written by Device, but can also be written by Host if the D2HBUF_HOSTWREN bit is set '1'. Reset type: SYSRSn

14.6.2.59 D2H_BUF14 Register (Offset = 7Ch) [Reset = 0h]

D2H_BUF14 is shown in Figure 14-68 and described in Table 14-69.

Return to the Summary Table.

Device to Host Buffer 14

Figure 14-68. D2H_BUF14 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-69. D2H_BUF14 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Device to be consumed by the Host. These registers are primarily written by Device, but can also be written by Host if the D2HBUF_HOSTWREN bit is set '1'. Reset type: SYSRSn

14.6.2.60 D2H_BUF15 Register (Offset = 7Eh) [Reset = 0h]

D2H_BUF15 is shown in Figure 14-69 and described in Table 14-70.

Return to the Summary Table.

Device to Host Buffer 15

Figure 14-69. D2H_BUF15 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data																															
R/W-0h																															

Table 14-70. D2H_BUF15 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	Data	R/W	0h	Data written by the Device to be consumed by the Host. These registers are primarily written by Device, but can also be written by Host if the D2HBUF_HOSTWREN bit is set '1'. Reset type: SYSRSn

14.6.3 HIC Registers to Driverlib Functions

Table 14-71. HIC Registers to Driverlib Functions

File	Driverlib Function
REV	
hic.h	HIC_getRevision
GCR	
hic.h	HIC_enableHostInterface
LOCK	
hic.h	HIC_enableLock
hic.h	HIC_disableLock
MODECR	
hic.h	HIC_setConfig
hic.h	HIC_getConfig
PINPOLCR	
hic.h	HIC_setPinPolarity
hic.h	HIC_getPinPolarity
BASESEL	
hic.h	HIC_selectBaseAddress
hic.h	HIC_getSelectedBaseAddressIndex
HOSTCR	
hic.h	HIC_configureHostAccessMode
hic.h	HIC_getHostAccessMode
ERRADDR	
hic.h	HIC_getErrorAddress
H2DTOKEN	
hic.h	HIC_getH2DToken
D2HTOKEN	
hic.h	HIC_setD2HToken
hic.h	HIC_getD2HToken
DBADDR0	
hic.h	HIC_setBaseAddress

Table 14-71. HIC Registers to Driverlib Functions (continued)

File	Driverlib Function
hic.h	HIC_getBaseAddress
DBADDR1	
-	
DBADDR2	
-	
DBADDR3	
-	
DBADDR4	
-	
DBADDR5	
-	
DBADDR6	
-	
DBADDR7	
-	
H2DINTEN	
hic.h	HIC_enableH2DInterrupt
hic.h	HIC_disableH2DInterrupt
H2DINTFLG	
hic.h	HIC_getH2DInterruptStatus
H2DINTCLR	
hic.h	HIC_clearH2DInterrupt
H2DINTFRC	
hic.h	HIC_forceH2DInterrupt
D2HINTEN	
hic.h	HIC_enableD2HInterrupt
hic.h	HIC_disableD2HInterrupt
D2HINTFLG	
hic.h	HIC_getD2HInterruptStatus
D2HINTCLR	
hic.h	HIC_clearD2HInterrupt
D2HINTFRC	
hic.h	HIC_forceD2HInterrupt
ACCVIOADDR	
hic.h	HIC_getAccessViolationAddress
COMMIT	
-	
H2D_BUF0	
hic.h	HIC_readH2DBuffer
hic.h	HIC_writeH2DBuffer
hic.h	HIC_clearH2DBuffer
H2D_BUF1	
-	
H2D_BUF2	
-	

Table 14-71. HIC Registers to Driverlib Functions (continued)

File	Driverlib Function
H2D_BUF3	
-	
H2D_BUF4	
-	
H2D_BUF5	
-	
H2D_BUF6	
-	
H2D_BUF7	
-	
H2D_BUF8	
-	
H2D_BUF9	
-	
H2D_BUF10	
-	
H2D_BUF11	
-	
H2D_BUF12	
-	
H2D_BUF13	
-	
H2D_BUF14	
-	
H2D_BUF15	
-	
D2H_BUF0	
hic.h	HIC_readD2HBuffer
hic.h	HIC_writeD2HBuffer
hic.h	HIC_clearD2HBuffer
D2H_BUF1	
-	
D2H_BUF2	
-	
D2H_BUF3	
-	
D2H_BUF4	
-	
D2H_BUF5	
-	
D2H_BUF6	
-	
D2H_BUF7	
-	
D2H_BUF8	

Table 14-71. HIC Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
D2H_BUF9	
-	
D2H_BUF10	
-	
D2H_BUF11	
-	
D2H_BUF12	
-	
D2H_BUF13	
-	
D2H_BUF14	
-	
D2H_BUF15	
-	

Chapter 15

Analog Subsystem

The analog subsystem module is described in this chapter.

15.1 Introduction

The analog modules on this device include the Analog-to-Digital Converter (ADC), Temperature Sensor, Buffered Digital-to-Analog Converter (DAC), and Comparator Subsystem (CMPSS).

15.1.1 Features

The analog subsystem has the following features:

- Flexible voltage references
  - The ADCs are referenced to VREFHIx and VSSA pins
    - VREFHIx pin voltage can be driven in externally or can be generated by an internal bandgap voltage reference
    - The internal voltage reference range can be selected to be 0V to 3.3V or 0V to 2.5V
  - The buffered DACs are referenced to VREFHIx and VSSA
    - Alternately, these DACs can be referenced to the VDAC pin and VSSA
  - The comparator DACs are referenced to VDDA and VSSA
    - Alternately, these DACs can be referenced to the VDAC pin and VSSA
- Flexible pin usage
  - Buffered DAC outputs, comparator subsystem inputs, and digital inputs (AIOs)/outputs (AGPIOs) are multiplexed with ADC inputs
  - Internal connection to V_{REFLO} on all ADCs for offset self-calibration

15.1.2 Block Diagram

The following analog subsystem block diagrams show the connections between the different integrated analog modules to the device pins. These pins fall into two categories: analog module inputs/outputs and reference pins.

The analog pins are organized into analog groups around the CMPSS module. The block diagram shows which pins connect to each group. The pins which connect to CMPSS inputs can be used for the CMPSS without further action and without preventing use as an ADC input simultaneously.

The VDAC reference pin can be used to set an alternate range for DAC A, DAC B and for the DACs inside the CMPSS modules (the CMPSS DACs are referenced to VDDA and VSSA by default). Using this pin as a reference prevents the channel from being used as an ADC input (but the ADC can be used to sample the VDAC voltage, if desired). The choice of reference is configurable per-module for each CMPSS or buffered DAC, and the selection is made via the module's configuration registers.

Some analog pins support digital functionality through muxed AIOs and AGPIOs. AIOs support only digital input functionality while AGPIOs support full digital input and output functionality.

The following notes apply to all packages:

- Not all analog pins are available on all devices. See your device-specific datasheet to determine which pins are available.
- See your device-specific datasheet to determine the allowable voltage range for VREFHI and VREFLO
- An external capacitor is required on the VREFHI pins. See your device-specific datasheet for the specific value required.
- For buffered DAC modules, VSSA will be the low reference whether VREFHIx or VDAC is selected as the high reference.
- For CMPSS modules, VSSA will be the low reference whether VDAC or VDDA is selected as the high reference.

The following figures show how each analog group is structured. Table 15-2 lists the analog pins and internal connections.

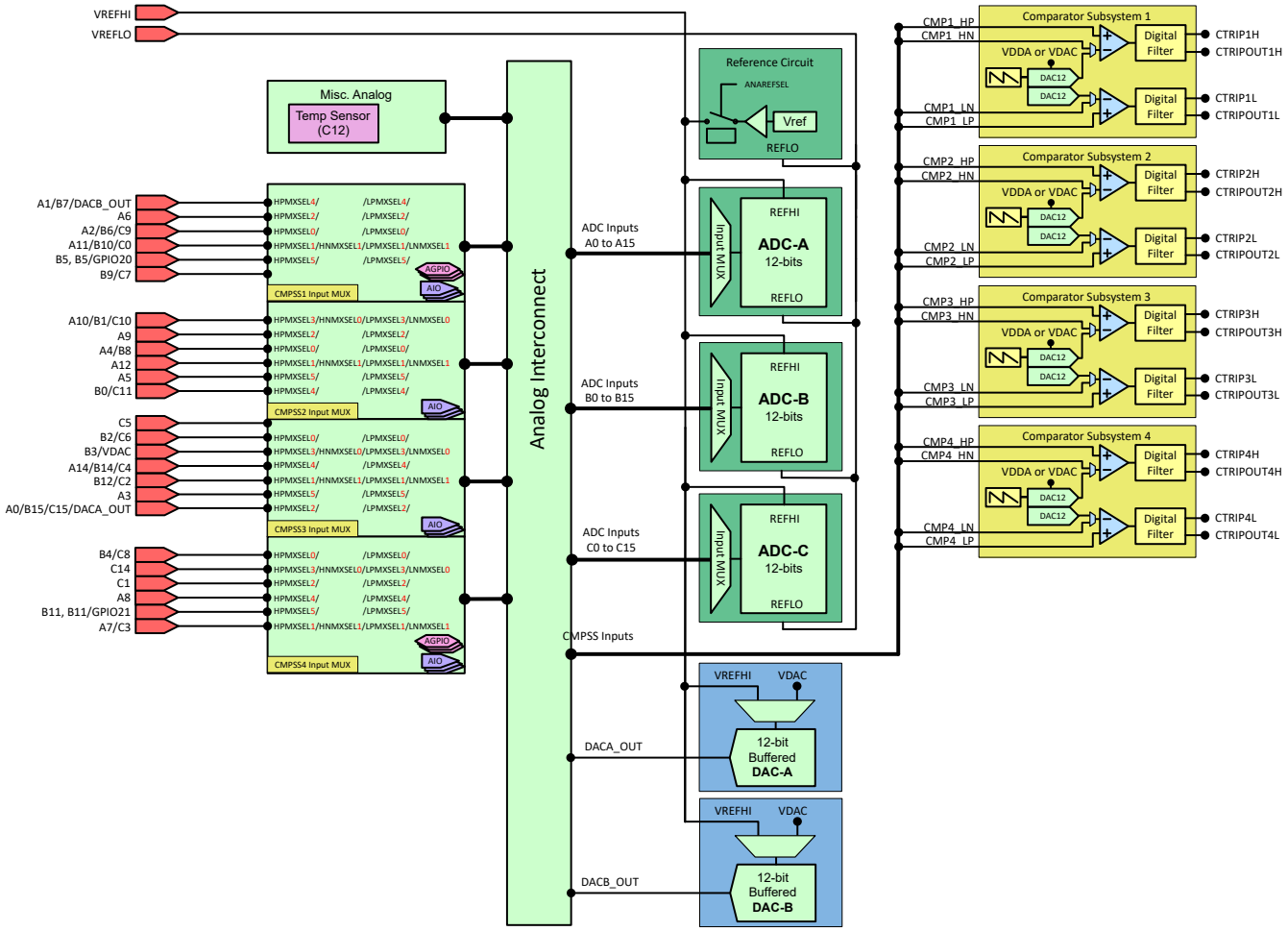


Figure 15-1. Analog Subsystem Block Diagram (100-Pin QFP)

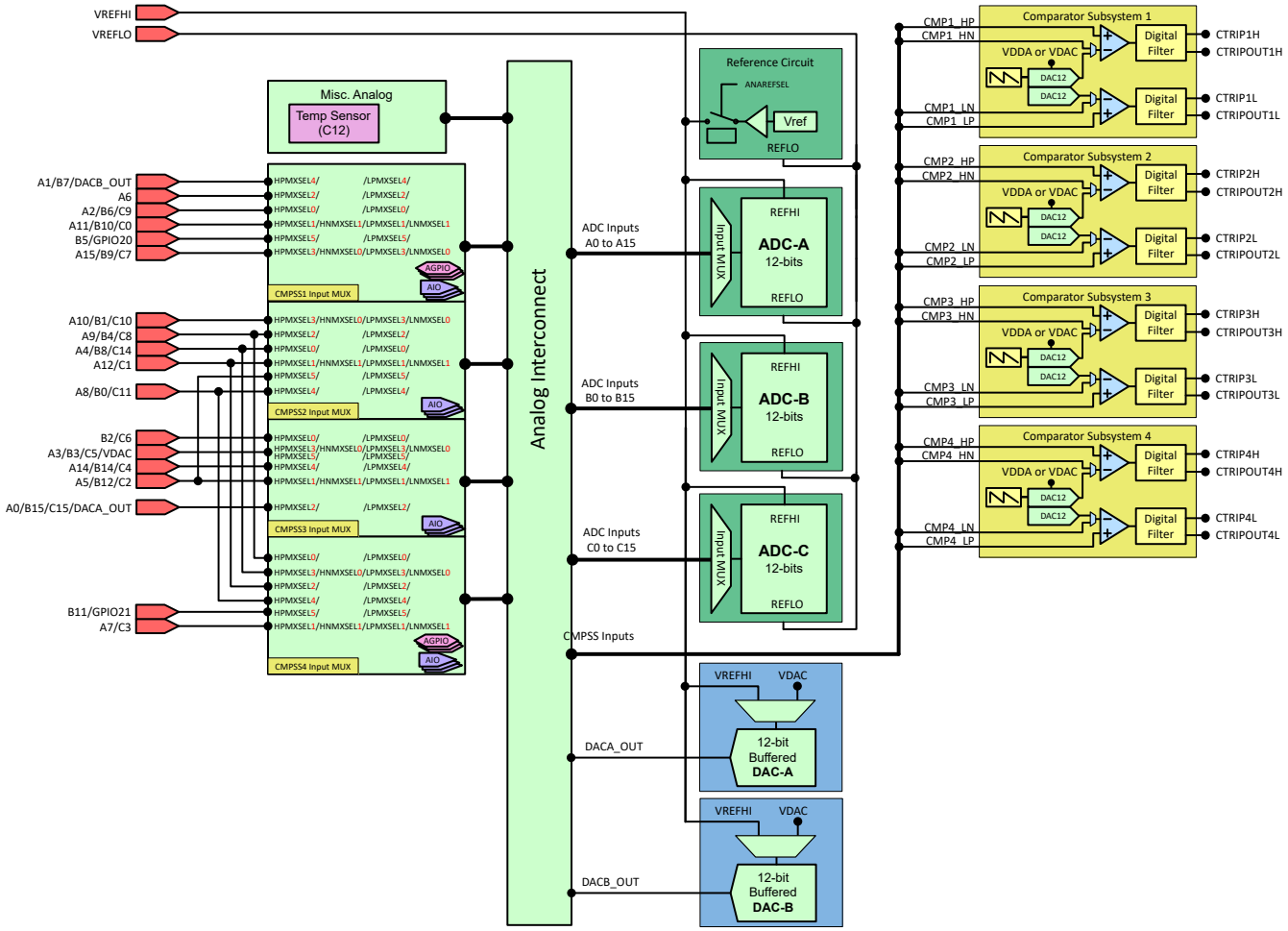


Figure 15-2. Analog Subsystem Block Diagram (80-Pin QFP)

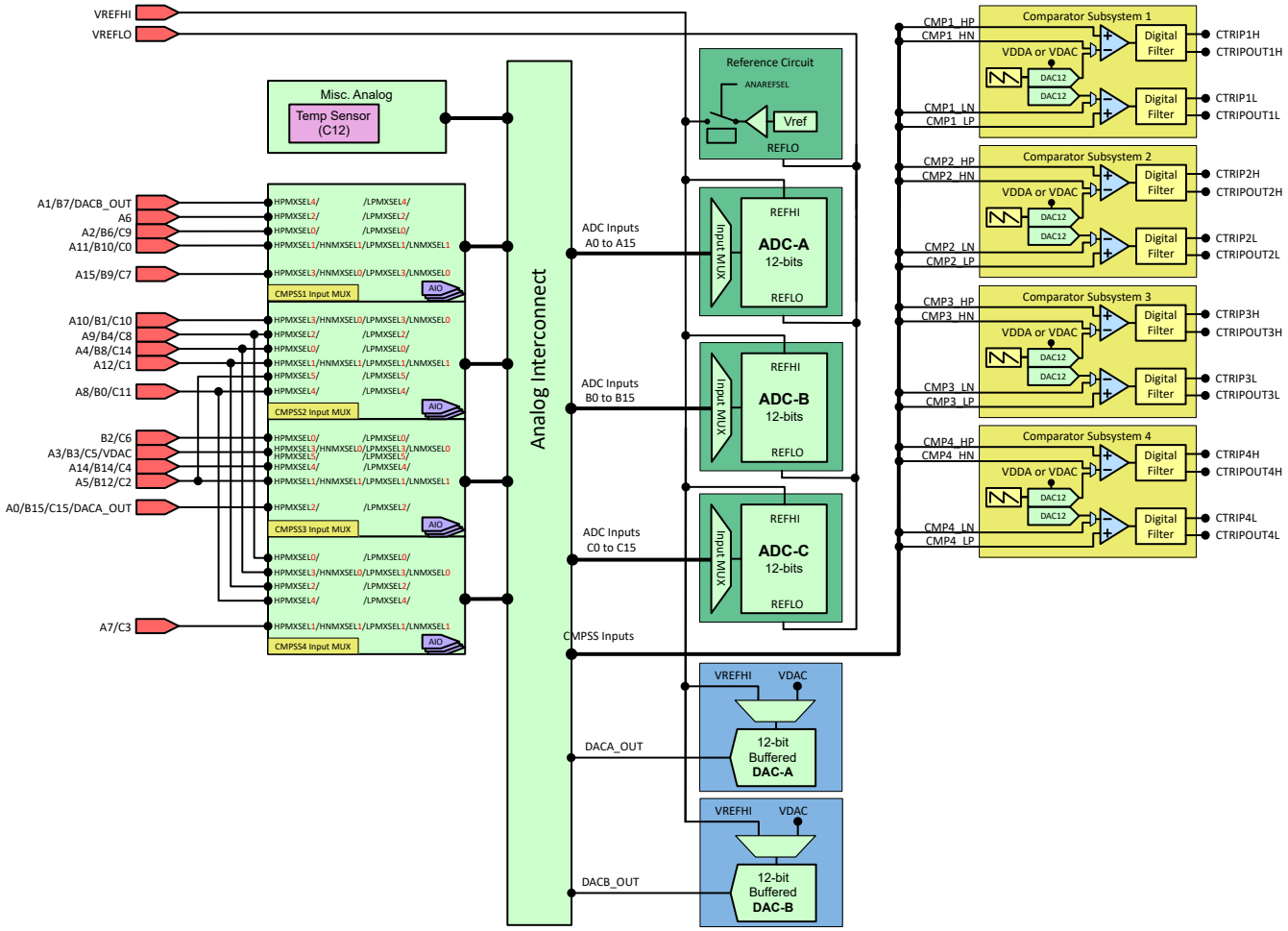


Figure 15-3. Analog Subsystem Block Diagram (64-Pin QFP)

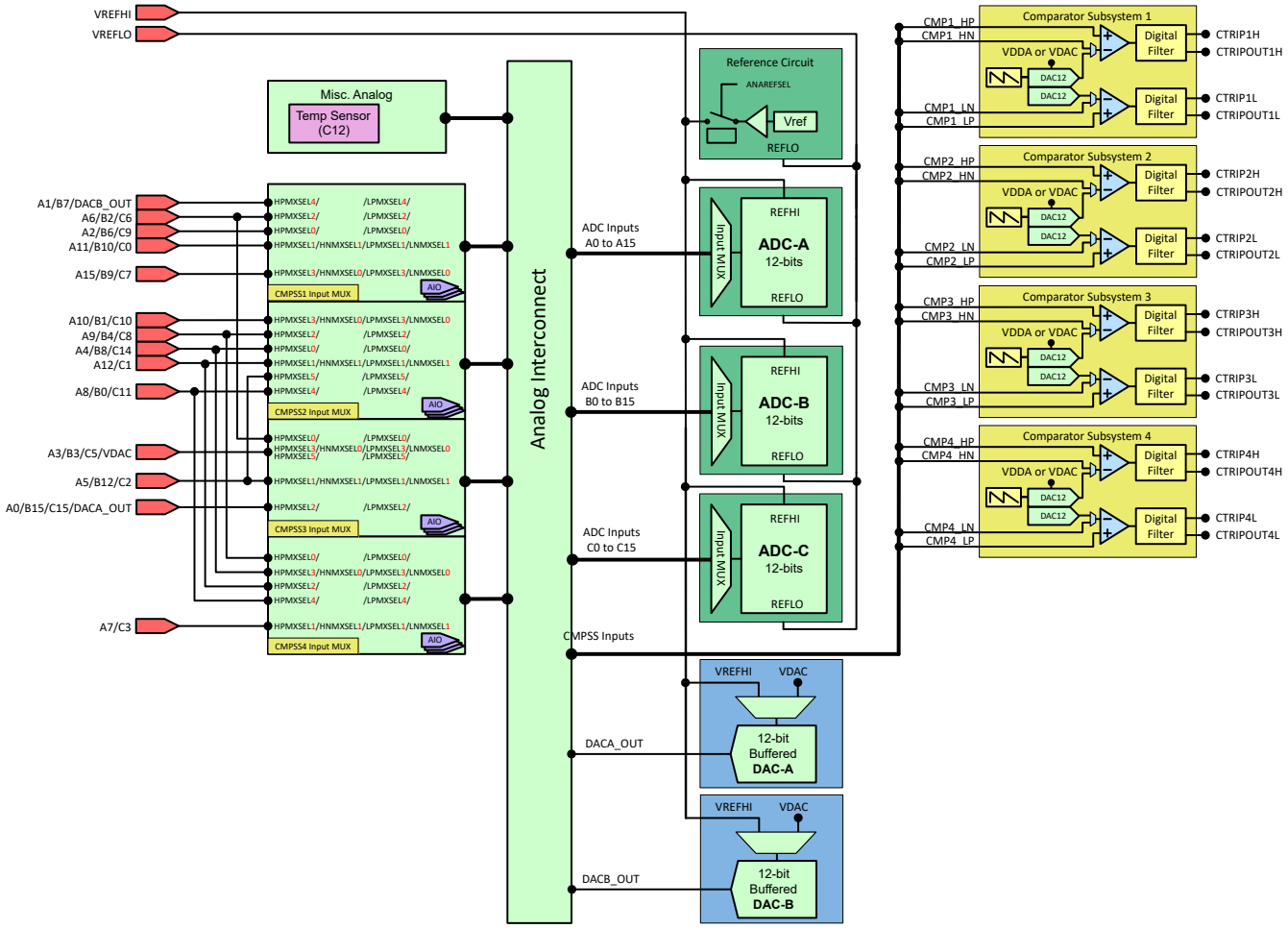
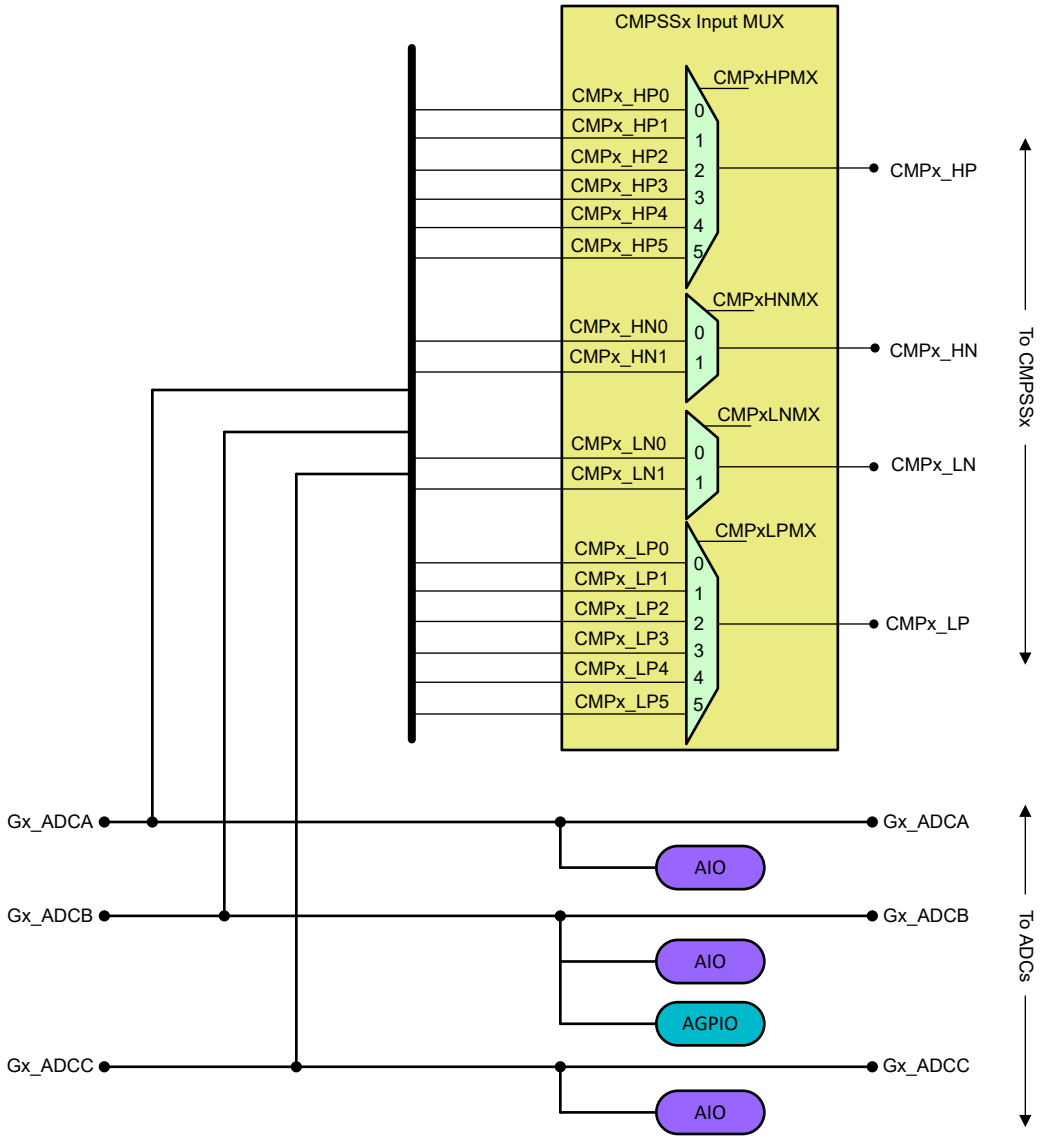


Figure 15-4. Analog Subsystem Block Diagram (48-Pin QFP)



Note: AIOs support digital input mode only.

Figure 15-5. Analog Group Connections

15.1.3 Digital Inputs on ADC Pins (AIOs)

GPIOs on port H (GPIO224–GPIO253) are multiplexed with analog pins. These are also referred to as AIOs. These pins can only function in input mode. By default, these pins will function as analog pins and the GPIOs are in a high-Z state. The GPHAMSEL register is used to configure these pins for digital or analog operation.

Note

If digital signals with sharp edges (high dv/dt) are connected to the AIOs, cross-talk can occur with adjacent analog signals. The user should therefore limit the edge rate of signals connected to AIOs if adjacent channels are being used for analog functions.

15.1.4 Digital Inputs and Outputs on ADC Pins (AGPIOs)

Some GPIOs on this device are multiplexed with analog pins. These are also referred to as AGPIOs. Unlike AIOs, AGPIOs have full input and output capability. This device has two GPIOs (GPIO20 and GPIO21) that offer this feature on the 100-pin PZ and 80-pin PN packages.

100-Pin PZ: On this package, there are dedicated pins for B5 (pin 32) and B11 (pin 30) that also have AIO252 and AIO251 functionality, respectively. In addition, GPIO20 (pin 48) and GPIO21 (pin 49) are also available as B5 and B11, respectively. Since B5 and B11 are dedicated pins on this package, it is recommended to use them instead of the pins on GPIO20/21.

80-Pin PN: On this package, GPIO20 (pin 33) and GPIO21 (pin 34) are also available as B5 and B11, respectively. There are no dedicated pin for B5 and B11.

By default, the AGPIOs are not connected and have to be configured. Table 15-1 shows how to configure the AGPIOs using B5 (pin 32) and GPIO20 (pin 48) on the 100-pin PZ as an example.

Table 15-1. AGPIO Configuration

AGPIOCTRLA.bit. GPIO20	GPAAMSEL.bit. GPIO20	GPHAMSEL.bit. GPIO252	B5 Connected To			GPIO20 Connected To		
			ADC	GPIO20	AIO252	ADC	GPIO20	AIO252
0	0	1	Yes	-	-	-	Yes	-
0	1	1	Yes	-	-	-	-	-
1	0	1	Yes	-	-	-	Yes	-
1	1	1	-	-	-	Yes	-	-
0	0	0	Yes	-	Yes	-	Yes	-
0	1	0	Yes	-	Yes	-	-	-
1	0	0	Yes	-	Yes	-	Yes	-
1	1	0	-	-	Yes	Yes	-	-

Note

If digital signals with sharp edges (high dv/dt) are connected to the AGPIOs, cross-talk can occur with adjacent analog signals. The user should therefore limit the edge rate of signals connected to AGPIOs if adjacent channels are being used for analog functions.

15.1.5 Analog Pins and Internal Connections

Table 15-2. Analog Pins and Internal Connections

Pin Name	Package Pin				ADC			Comparator Subsystem (MUX)				AIO Input
	100 PZ	80 PN	64 PM	48 PT	A	B	C	High Positive	High Negative	Low Positive	Low Negative	
VREFHI	24, 25	20	16	12								
VREFLO	26, 27	21	17	13	A13	B13	C13					
Analog Group 1								**CMP1**				
A6	14	10	6	4(1)	A6	-	-	CMP1 (HPMXSEL=2)		CMP1 (LPMXSEL=2)		AIO228
A2/B6/C9	17	13	9	6	A2	B6	C9	CMP1 (HPMXSEL=0)		CMP1 (LPMXSEL=0)		AIO224
A15	-	14	10	7	A15	-	-	CMP1 (HPMXSEL=3)	CMP1 (HNMXSEL=0)	CMP1 (LPMXSEL=3)	CMP1 (LNMXSEL=0)	AIO233
B9/C7	18				-	B9	C7					
A11/B10/C0	20	16	12	8	A11	B10	C0	CMP1 (HPMXSEL=1)	CMP1 (HNMXSEL=1)	CMP1 (LPMXSEL=1)	CMP1 (LNMXSEL=1)	AIO237
A1/B7/DACB_OUT	22	18	14	10	A1	B7	-	CMP1 (HPMXSEL=4)		CMP1 (LPMXSEL=4)		AIO232
Analog Group 2								**CMP2**				
A10/B1/C10	40	29	25	21	A10	B1	C10	CMP2 (HPMXSEL=3)	CMP2 (HNMXSEL=0)	CMP2 (LPMXSEL=3)	CMP2 (LNMXSEL=0)	AIO230
Analog Group 3								**CMP3**				
B2/C6	15	11	7	4(1)	-	B2	C6	CMP3 (HPMXSEL=0)		CMP3 (LPMXSEL=0)		AIO226
B3/VDAC(2)	16	12	8	5	-	B3	-	CMP3 (HPMXSEL=3)	CMP3 (HNMXSEL=0)	CMP3 (LPMXSEL=3)	CMP3 (LNMXSEL=0)	AIO242
C5	28				-	-	C5					
A3	-				A3	-	-	CMP3 (HPMXSEL=5)		CMP3 (LPMXSEL=5)		
	18	-	-	-		-	-					AIO229
A14/B14/C4	19	15	11	-	A14	B14	C4	CMP3 (HPMXSEL=4)		CMP3 (LPMXSEL=4)		AIO239
A0/B15/C15/ DACA_OUT	23	19	15	11	A0	B15	C15	CMP3 (HPMXSEL=2)		CMP3 (LPMXSEL=2)		AIO231
Analog Group 4								**CMP4**				
A7/C3	31	23	19	15	A7	-	C3	CMP4 (HPMXSEL=1)	CMP4 (HNMXSEL=1)	CMP4 (LPMXSEL=1)	CMP4 (LNMXSEL=1)	AIO245
Combined Analog Group 2/3								**CMP2/3**				
A5	35	-	-	-	A5	-	-	CMP2 (HPMXSEL=5)		CMP2 (LPMXSEL=5)		AIO249
	-	17	13	9		-	-					
B12/C2	21				-	B12	C2	CMP3 (HPMXSEL=1)	CMP3 (HNMXSEL=1)	CMP3 (LPMXSEL=1)	CMP3 (LNMXSEL=1)	AIO244
Combined Analog Group 2/4								**CMP2/4**				
A12	28	22	18	14	A12	-	-	CMP2 (HPMXSEL=1)	CMP2 (HNMXSEL=1)	CMP2 (LPMXSEL=1)	CMP2 (LNMXSEL=1)	AIO238
C1	29				-	-	C1	CMP4 (HPMXSEL=2)		CMP4 (LPMXSEL=2)		AIO248

Table 15-2. Analog Pins and Internal Connections (continued)

Pin Name	Package Pin				ADC			Comparator Subsystem (MUX)				AIO Input
	100 PZ	80 PN	64 PM	48 PT	A	B	C	High Positive	High Negative	Low Positive	Low Negative	
A8	37	-	-	-	A8	-	-	CMP4 (HPMXSEL=4)		CMP4 (LPMXSEL=4)		AIO240
	-	24	20	16		-	-					AIO241
B0/C11	-				-	B0	C11	CMP2 (HPMXSEL=4)		CMP2 (LPMXSEL=4)		
	41	-	-	-	-							AIO253
A4/B8	36	27	23	19	A4	B8	-	CMP2 (HPMXSEL=0)		CMP2 (LPMXSEL=0)		AIO225
C14	-				-	-	C14	CMP4 (HPMXSEL=3)	CMP4 (HNMXSEL=0)	CMP4 (LPMXSEL=3)	CMP4 (LNMXSEL=0)	
	42	-	-	-	-	-						AIO247
A9	38	28	24	20	A9	-	-	CMP2 (HPMXSEL=2)		CMP2 (LPMXSEL=2)		AIO227
B4/C8	39				-	B4	C8	CMP4 (HPMXSEL=0)		CMP4 (LPMXSEL=0)		AIO236
Other Analog												
B5	32	-	-	-	-	B5	-	CMP1 (HPMXSEL=5)		CMP1 (LPMXSEL=5)		AIO252
B5/GPIO20[3]	48	33	-	-	-		-					GPIO20
B11	30	-	-	-	-	B11	-	CMP4 (HPMXSEL=5)		CMP4 (LPMXSEL=5)		AIO251
B11/GPIO21[3]	49	34	-	-	-		-					GPIO21
TempSensor[4]	-	-	-	-	-	-	C12					

(1) A6 and C6 is double bonded as pin # 4.
(2) Optional external reference voltage for on-chip COMPDACs/GPDACs. There is an internal capacitance to VSSA on this pin whether used for ADC input or COMPDAC/GPDAC reference. If used as a VDAC reference, place at least a 1-µF capacitor on this pin.
(3) The GPIOs on these analog pins support full digital input and output functionality and are referred to as AGPIOs. By default, the AGPIOs are unconnected; that is, the analog and digital functions are both disabled. For configuration details, see the *Digital Inputs and Outputs on ADC Pins (AGPIOs)* section.
(4) Internal connection only; does not come to a device pin.

Table 15-3. Analog Signal Descriptions

Signal Name	Description
AIOx	Digital input on ADC pin
GPIOx	Digital input/output pin with ADC functionality
Ax	ADC A Input
Bx	ADC B Input
Cx	ADC C Input
CMPx_DACH	Comparator subsystem high DAC output
CMPx_DACL	Comparator subsystem low DAC output
CMPx_HNy	Comparator subsystem high comparator negative input
CMPx_HPy	Comparator subsystem high comparator positive input

Table 15-3. Analog Signal Descriptions (continued)

Signal Name	Description
CMPx_LNy	Comparator subsystem low comparator negative input
CMPx_LPy	Comparator subsystem low comparator positive input
DACx_OUT	Buffered DAC Output
TempSensor	Internal temperature sensor
VDAC	Optional external reference voltage for on-chip COMPDACs. This pin has a higher capacitance compared to the other analog pins. *See the Per-Channel Parasitic Capacitance* table for details. This capacitance is present whether the pin is being used for ADC input or COMPDAC/GPDAC reference and cannot be disabled. If this pin is being used as a reference for the on-chip COMPDAC/GPDACs, place at least a 1-µF capacitor on this pin.

Table 15-4. Reference Summary

Module	Reference Option	Configured Where?	Register	Driverlib Function	Notes
ADC	Internal or External	Analog System	AnalogSubsysRegs. ANAREFCTL.bit. ANAREFSEL	ADC_setVREF	Both options require use of the VREFHI pin
	3.3-V or 2.5-V Internal Reference Range	Analog System	AnalogSubsysRegs. ANAREFCTL.bit. ANAREF2P5SEL	ADC_setVREF	Only applicable when using internal reference mode
Buffered DAC	VREFHI or VDAC	DAC Module	DacxRegs. DACCTL.bit. DACREFSEL	DAC_setReferenceVoltage	
	Internal or External	Analog System	(follows ADC configuration)	ADC_setVREF	Only applicable when using VREFHI
	3.3-V or 2.5-V Internal Reference Range	Analog System	(follows ADC configuration)	ADC_setVREF	Only applicable when using VREFHI with internal reference
		DAC Module	DacxRegs. DACCTL.bit. MODE	DAC_setGainMode	Gain = 2 should be set when using 3.3V internal reference mode
CMPSS DACs	VDDA or VDAC	CMPSS Module	CmpssxRegs. COMPDACCTL.bit. SELREF	CMPSS_configDAC	

15.1.6 Lock Registers

Setting one of the bits in the lock registers prevents further writes to the associated analog subsystem configuration register. This lock can only be cleared by a device reset.

15.2 Analog Subsystem Registers

This section describes the Analog Subsystem Registers.

15.2.1 ASBSYS Base Address Table

Table 15-5. ASBSYS Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
AnalogSubsysRegs	ANALOG_SUBSYS_REGS	ANALOGSUBSYS_BASE	0x0005_D700	YES	-	-	-	YES

15.2.2 ANALOG_SUBSYS_REGS Registers

Table 15-6 lists the memory-mapped registers for the ANALOG_SUBSYS_REGS registers. All register offset addresses not listed in Table 15-6 should be considered as reserved locations and the register contents should not be modified.

Table 15-6. ANALOG_SUBSYS_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
5Eh	CONFIGLOCK	Lock Register for all the config registers.	EALLOW	Go
60h	TSNSCTL	Temperature Sensor Control Register	EALLOW	Go
68h	ANAREFCTL	Analog Reference Control Register	EALLOW	Go
70h	VMONCTL	Voltage Monitor Control Register	EALLOW	Go
82h	CMPHPMXSEL	Bits to select one of the many sources on CompHP inputs. Refer to Pimux diagram for details.	EALLOW	Go
84h	CMPLPMXSEL	Bits to select one of the many sources on CompLP inputs. Refer to Pimux diagram for details.	EALLOW	Go
86h	CMPHNMXSEL	Bits to select one of the many sources on CompHN inputs. Refer to Pimux diagram for details.	EALLOW	Go
87h	CMPLNMXSEL	Bits to select one of the many sources on CompLN inputs. Refer to Pimux diagram for details.	EALLOW	Go
88h	ADCDACLOOPBACK	Enabble loopback from DAC to ADCs		Go
8Eh	LOCK	Lock Register	EALLOW	Go
102h	AGPIOCTRLA	AGPIO Control Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 15-7 shows the codes that are used for access types in this section.

Table 15-7. ANALOG_SUBSYS_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
WOnce	W Once	Write Write once
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 15-7. ANALOG_SUBSYS_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

15.2.2.1 CONFIGLOCK Register (Offset = 5Eh) [Reset = 0h]

CONFIGLOCK is shown in Figure 15-5 and described in Table 15-8.

Return to the Summary Table.

Lock Register for all the config registers.

Figure 15-5. CONFIGLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			RESERVED	AGPIOCTRL	RESERVED	RESERVED	RESERVED
R-0h			R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 15-8. CONFIGLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reserved
4	RESERVED	R/WSonce	0h	Reserved
3	AGPIOCTRL	R/WSonce	0h	Locks all AGPIOCTRL Register. Setting this bit will disable any future writes to this reigster. This bit can only be cleared by a reset. Reset type: SYSRSn
2	RESERVED	R/WSonce	0h	Reserved
1	RESERVED	R/WSonce	0h	Reserved
0	RESERVED	R/WSonce	0h	Reserved

15.2.2.2 TSNSCTL Register (Offset = 60h) [Reset = 0h]

TSNSCTL is shown in Figure 15-6 and described in Table 15-9.

Return to the Summary Table.

Temperature Sensor Control Register

Figure 15-6. TSNSCTL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							ENABLE
R-0h							R/W-0h

Table 15-9. TSNSCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R	0h	Reserved
0	ENABLE	R/W	0h	Temperature Sensor Enable. This bit enables the temperature sensor output to the ADC. 0 Disabled 1 Enabled Reset type: SYSRSn

15.2.2.3 ANAREFCTL Register (Offset = 68h) [Reset = 7h]

ANAREFCTL is shown in Figure 15-7 and described in Table 15-10.

Return to the Summary Table.

Analog Reference Control Register

Figure 15-7. ANAREFCTL Register

15	14	13	12	11	10	9	8
RESERVED					RESERVED	RESERVED	ANAREF2P5SEL
R-0h					R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED					RESERVED	RESERVED	ANAREFSEL
R-0h					R/W-1h	R/W-1h	R/W-1h

Table 15-10. ANAREFCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	RESERVED	R/W	0h	Reserved
8	ANAREF2P5SEL	R/W	0h	Analog reference A 2.5V source select. In internal reference mode, this bit selects which voltage the internal reference buffer drives onto the VREFHI pin. The buffer can drive either 1.65V onto the pin, resulting in a reference range of 0 to 3.3V, or the buffer can drive 2.5V onto the pin, resulting in a reference range of 0 to 2.5V. If switching between these two modes, the user must allow adequate time for the external capacitor to charge to the new voltage before using the ADC or buffered DAC. 0 Internal 1.65V reference mode (3.3V reference range) 1 Internal 2.5V reference mode (2.5V reference range) Reset type: XRSn
7-3	RESERVED	R	0h	Reserved
2	RESERVED	R/W	1h	Reserved
1	RESERVED	R/W	1h	Reserved
0	ANAREFSEL	R/W	1h	Analog reference mode select. This bit selects whether the VREFHI pin uses internal reference mode (the device drives a voltage onto the VREFHI pin) or external reference mode (the system is expected to drive a voltage into the VREFHI pin). 0 Internal reference mode 1 External reference mode Reset type: XRSn

15.2.2.4 VMONCTL Register (Offset = 70h) [Reset = 0h]

VMONCTL is shown in Figure 15-8 and described in Table 15-11.

Return to the Summary Table.

Voltage Monitor Control Register

Figure 15-8. VMONCTL Register

15	14	13	12	11	10	9	8
RESERVED							BORLVMONDIS
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED						RESERVED	RESERVED
R-0h						R/W-0h	R/W-0h

Table 15-11. VMONCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-9	RESERVED	R	0h	Reserved
8	BORLVMONDIS	R/W	0h	BORL disable on VDDIO. 0 BORL is enabled on VDDIO, i.e BOR circuit will be triggered if VDDIO goes lower than the lower BOR threshold of VDDIO. 1 BORL is disabled on VDDIO, i.e BOR circuit will not be triggered if VDDIO goes lower than the lower BOR threshold of VDDIO. Reset type: SYSRSn
7-2	RESERVED	R	0h	Reserved
1	RESERVED	R/W	0h	Reserved
0	RESERVED	R/W	0h	Reserved

15.2.2.5 CMPHPMXSEL Register (Offset = 82h) [Reset = 0h]

CMPHPMXSEL is shown in Figure 15-9 and described in Table 15-12.

Return to the Summary Table.

Bits to select one of the many sources on CompHP inputs. Refer to Pimux diagram for details.

Figure 15-9. CMPHPMXSEL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED			RESERVED			RESERVED	
R-0h			R/W-0h			R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	RESERVED			CMP4HPMXSEL			CMP3HPMXSEL
R-0h	R/W-0h			R/W-0h			R/W-0h
7	6	5	4	3	2	1	0
CMP3HPMXSEL			CMP2HPMXSEL			CMP1HPMXSEL	
R/W-0h			R/W-0h			R/W-0h	

Table 15-12. CMPHPMXSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R	0h	Reserved
21-19	RESERVED	R/W	0h	Reserved
18-16	RESERVED	R/W	0h	Reserved
15	RESERVED	R	0h	Reserved
14-12	RESERVED	R/W	0h	Reserved
11-9	CMP4HPMXSEL	R/W	0h	CMP4HPMXSEL bits, Refer to the Analog Subsystem chapter Note: Only values 0 to 5 are valid, rest are reserved Reset type: XRSn
8-6	CMP3HPMXSEL	R/W	0h	CMP3HPMXSEL bits, Refer to the Analog Subsystem chapter Note: Only values 0 to 5 are valid, rest are reserved Reset type: XRSn
5-3	CMP2HPMXSEL	R/W	0h	CMP2HPMXSEL bits, Refer to the Analog Subsystem chapter Note: Only values 0 to 5 are valid, rest are reserved Reset type: XRSn
2-0	CMP1HPMXSEL	R/W	0h	CMP1HPMXSEL bits, Refer to the Analog Subsystem chapter Note: Only values 0 to 5 are valid, rest are reserved Reset type: XRSn

15.2.2.6 CMPLPMXSEL Register (Offset = 84h) [Reset = 0h]

CMPLPMXSEL is shown in Figure 15-10 and described in Table 15-13.

Return to the Summary Table.

Bits to select one of the many sources on CompLP inputs. Refer to Pimux diagram for details.

Figure 15-10. CMPLPMXSEL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED			RESERVED			RESERVED	
R-0h			R/W-0h			R/W-0h	
15	14	13	12	11	10	9	8
RESERVED	RESERVED			CMP4LPMXSEL			CMP3LPMXSEL
R-0h	R/W-0h			R/W-0h			R/W-0h
7	6	5	4	3	2	1	0
CMP3LPMXSEL			CMP2LPMXSEL			CMP1LPMXSEL	
R/W-0h			R/W-0h			R/W-0h	

Table 15-13. CMPLPMXSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R	0h	Reserved
21-19	RESERVED	R/W	0h	Reserved
18-16	RESERVED	R/W	0h	Reserved
15	RESERVED	R	0h	Reserved
14-12	RESERVED	R/W	0h	Reserved
11-9	CMP4LPMXSEL	R/W	0h	CMP4LPMXSEL bits, Refer to the Analog Subsystem chapter Note: Only values 0 to 5 are valid, rest are reserved Reset type: XRSn
8-6	CMP3LPMXSEL	R/W	0h	CMP3LPMXSEL bits, Refer to the Analog Subsystem chapter Note: Only values 0 to 5 are valid, rest are reserved Reset type: XRSn
5-3	CMP2LPMXSEL	R/W	0h	CMP2LPMXSEL bits, Refer to the Analog Subsystem chapter Note: Only values 0 to 5 are valid, rest are reserved Reset type: XRSn
2-0	CMP1LPMXSEL	R/W	0h	CMP1LPMXSEL bits, Refer to the Analog Subsystem chapter Note: Only values 0 to 5 are valid, rest are reserved Reset type: XRSn

15.2.2.7 CMPHNMXSEL Register (Offset = 86h) [Reset = 0h]

CMPHNMXSEL is shown in Figure 15-11 and described in Table 15-14.

Return to the Summary Table.

Bits to select one of the many sources on CompHN inputs. Refer to Pimux diagram for details.

Figure 15-11. CMPHNMXSEL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	CMP4HNMXSEL	CMP3HNMXSEL	CMP2HNMXSEL	CMP1HNMXSEL
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-14. CMPHNMXSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R	0h	Reserved
6	RESERVED	R/W	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	CMP4HNMXSEL	R/W	0h	CMP4HNMXSEL bits, Refer to the Analog Subsystem chapter Reset type: XRSn
2	CMP3HNMXSEL	R/W	0h	CMP3HNMXSEL bits, Refer to the Analog Subsystem chapter Reset type: XRSn
1	CMP2HNMXSEL	R/W	0h	CMP2HNMXSEL bits, Refer to the Analog Subsystem chapter Reset type: XRSn
0	CMP1HNMXSEL	R/W	0h	CMP1HNMXSEL bits, Refer to the Analog Subsystem chapter Reset type: XRSn

15.2.2.8 CMPLNMXSEL Register (Offset = 87h) [Reset = 0h]

CMPLNMXSEL is shown in Figure 15-12 and described in Table 15-15.

Return to the Summary Table.

Bits to select one of the many sources on CompLN inputs. Refer to Pimux diagram for details.

Figure 15-12. CMPLNMXSEL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	CMP4LNMXSEL	CMP3LNMXSEL	CMP2LNMXSEL	CMP1LNMXSEL
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-15. CMPLNMXSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R	0h	Reserved
6	RESERVED	R/W	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	CMP4LNMXSEL	R/W	0h	CMP4LNMXSEL bits, Refer to the Analog Subsystem chapter Reset type: XRSn
2	CMP3LNMXSEL	R/W	0h	CMP3LNMXSEL bits, Refer to the Analog Subsystem chapter Reset type: XRSn
1	CMP2LNMXSEL	R/W	0h	CMP2LNMXSEL bits, Refer to the Analog Subsystem chapter Reset type: XRSn
0	CMP1LNMXSEL	R/W	0h	CMP1LNMXSEL bits, Refer to the Analog Subsystem chapter Reset type: XRSn

15.2.2.9 ADCDACLOOPBACK Register (Offset = 88h) [Reset = 0h]

ADCDACLOOPBACK is shown in Figure 15-13 and described in Table 15-16.

Return to the Summary Table.

Enabble loopback from DAC to ADCs

Figure 15-13. ADCDACLOOPBACK Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					ENLB2ADCC	ENLB2ADCB	ENLB2ADCA
R-0h					R/W-0h	R/W-0h	R/W-0h

Table 15-16. ADCDACLOOPBACK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Write Key. Writes to this register must include the value 0xA5A5 in the KEY bit field to take effect. Otherwise the register will remain as it was prior to the write attempt. Reads will return a 0. Reset type: XRSn
15-3	RESERVED	R	0h	Reserved
2	ENLB2ADCC	R/W	0h	1 Loops back CMPSS1 DACL output to ADCC. 0 Loop back is broken. Note: Setting this bit to 1, will override the CHSEL specification for the ADC. ADC would sample CMPSS1 DACL output irrespective of the value of CHSEL. Reset type: XRSn
1	ENLB2ADCB	R/W	0h	1 Loops back CMPSS1 DACL output to ADCB. 0 Loop back is broken. Note: Setting this bit to 1, will override the CHSEL specification for the ADC. ADC would sample CMPSS1 DACL output irrespective of the value of CHSEL. Reset type: XRSn
0	ENLB2ADCA	R/W	0h	1 Loops back CMPSS1 DACL output to ADCA. 0 Loop back is broken. Note: Setting this bit to 1, will override the CHSEL specification for the ADC. ADC would sample CMPSS1 DACL output irrespective of the value of CHSEL. Reset type: XRSn

15.2.2.10 LOCK Register (Offset = 8Eh) [Reset = 0h]

LOCK is shown in Figure 15-14 and described in Table 15-17.

Return to the Summary Table.

Lock Register

Figure 15-14. LOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						VREGCTL	CMPLNMXSEL
R-0h						R/WSonce-0h	R/WSonce-0h
7	6	5	4	3	2	1	0
CMPHNMXSEL	CMPLPMXSEL	CMPHPMXSEL	RESERVED	RESERVED	VMONCTL	ANAREFCTL	TSNSCTL
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 15-17. LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9	VREGCTL	R/WSonce	0h	VREGCTL Register Lock. Setting this bit will disable any future write to the respective register. This bit can only be cleared by a reset. Reset type: SYSRSn
8	CMPLNMXSEL	R/WSonce	0h	CMPLNMXSEL Register Lock. Setting this bit will disable any future write to the respective register. This bit can only be cleared by a reset. Reset type: SYSRSn
7	CMPHNMXSEL	R/WSonce	0h	CMPHNMXSEL Register Lock. Setting this bit will disable any future write to the respective register. This bit can only be cleared by a reset. Reset type: SYSRSn
6	CMPLPMXSEL	R/WSonce	0h	CMPLPMXSEL Register Lock. Setting this bit will disable any future write to the respective register. This bit can only be cleared by a reset. Reset type: SYSRSn
5	CMPHPMXSEL	R/WSonce	0h	CMPHPMXSEL Register Lock. Setting this bit will disable any future write to the respective register. This bit can only be cleared by a reset. Reset type: SYSRSn
4	RESERVED	R/WSonce	0h	Reserved
3	RESERVED	R/WSonce	0h	Reserved
2	VMONCTL	R/WSonce	0h	VMONCTL Register Lock. Setting this bit will disable any future write to the respective register. This bit can only be cleared by a reset. Reset type: SYSRSn
1	ANAREFCTL	R/WSonce	0h	ANAREFCTL Register Lock. Setting this bit will disable any future write to the respective register. This bit can only be cleared by a reset. Reset type: SYSRSn

Table 15-17. LOCK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	TSNSCTL	R/WSonce	0h	TSNSCTL Register Lock. Setting this bit will disable any future write to the respective register. This bit can only be cleared by a reset. Reset type: SYSRSn

15.2.2.11 AGPIOCTRLA Register (Offset = 102h) [Reset = 0h]

AGPIOCTRLA is shown in Figure 15-15 and described in Table 15-18.

Return to the Summary Table.

AGPIO Control Register

Figure 15-15. AGPIOCTRLA Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED	RESERVED	GPIO21	GPIO20	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 15-18. AGPIOCTRLA Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30	RESERVED	R/W	0h	Reserved
29	RESERVED	R/W	0h	Reserved
28	RESERVED	R/W	0h	Reserved
27	RESERVED	R/W	0h	Reserved
26	RESERVED	R/W	0h	Reserved
25	RESERVED	R/W	0h	Reserved
24	RESERVED	R/W	0h	Reserved
23	RESERVED	R/W	0h	Reserved
22	RESERVED	R/W	0h	Reserved
21	GPIO21	R/W	0h	One time configuration for GPIO21 to decide whether AGPIO PAD or Analog PAD is pinned out Reset type: XRSn
20	GPIO20	R/W	0h	One time configuration for GPIO20 to decide whether AGPIO PAD or Analog PAD is pinned out Reset type: XRSn
19	RESERVED	R/W	0h	Reserved
18	RESERVED	R/W	0h	Reserved
17	RESERVED	R/W	0h	Reserved
16	RESERVED	R/W	0h	Reserved
15	RESERVED	R/W	0h	Reserved
14	RESERVED	R/W	0h	Reserved
13	RESERVED	R/W	0h	Reserved
12	RESERVED	R/W	0h	Reserved
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	RESERVED	R/W	0h	Reserved

Table 15-18. AGPIOCTRLA Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	RESERVED	R/W	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	RESERVED	R/W	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1	RESERVED	R/W	0h	Reserved
0	RESERVED	R/W	0h	Reserved

15.2.3 ASYSCTL Registers to Driverlib Functions

Table 15-19. ASYSCTL Registers to Driverlib Functions

File	Driverlib Function
CONFIGLOCK	
-	
TSNSCTL	
asysctl.h	ASysCtl_enableTemperatureSensor
asysctl.h	ASysCtl_disableTemperatureSensor
ANAREFCTL	
adc.c	ADC_setVREF
adc.c	ADC_setOffsetTrim
asysctl.h	ASysCtl_setAnalogReferenceInternal
asysctl.h	ASysCtl_setAnalogReferenceExternal
asysctl.h	ASysCtl_setAnalogReference2P5
asysctl.h	ASysCtl_setAnalogReference1P65
VMONCTL	
-	
CMPHPMXSEL	
asysctl.h	ASysCtl_selectCMPHPMux
CMPLPMXSEL	
asysctl.h	ASysCtl_selectCMPLPMux
CMPHNMXSEL	
asysctl.h	ASysCtl_selectCMPHNMux
asysctl.h	ASysCtl_selectCMPHNMuxValue
CMPLNMXSEL	
asysctl.h	ASysCtl_selectCMPLNMux
asysctl.h	ASysCtl_selectCMPLNMuxValue
ADCDACLOOPBACK	
-	
LOCK	
asysctl.h	ASysCtl_lockTemperatureSensor
asysctl.h	ASysCtl_lockANAREF
asysctl.h	ASysCtl_lockVMON
asysctl.h	ASysCtl_lockCMPHPMux
asysctl.h	ASysCtl_lockCMPLPMux

Table 15-19. ASYSCTL Registers to Driverlib Functions (continued)

File	Driverlib Function
asysctl.h	ASysCtl_lockCMPHNMux
asysctl.h	ASysCtl_lockCMPLNMux
asysctl.h	ASysCtl_lockVREG
AGPIOCTRLA	
gpio.c	GPIO_setAnalogMode

Chapter 16

Analog-to-Digital Converter (ADC)

The analog-to-digital converter (ADC) module described in this chapter is a Type 5 ADC. See the *C2000 Real-Time Control Peripheral Reference Guide* for a list of all devices with modules of the same type, to determine the differences between the types, and for a list of device-specific differences within a type.

16.1 Introduction

The ADC module is a successive approximation (SAR) style ADC. The ADC is composed of a core and a wrapper. The core is composed of the analog circuits which include the channel select MUX, the sample-and-hold (S/H) circuit, the successive approximation circuits, voltage reference circuits, and other analog support circuits. The wrapper is composed of the digital circuits that configure and control the ADC. These circuits include the logic for programmable conversions, result registers, interfaces to analog circuits, interfaces to the peripheral buses, post-processing circuits, and interfaces to other on-chip modules.

Each ADC module consists of a single sample-and-hold (s/h) circuit. The ADC module is designed to be duplicated multiple times on the same chip, allowing simultaneous sampling or independent operation of multiple ADCs. The ADC wrapper is start-of-conversion (SOC) based (see Section 16.3).

16.1.1 ADC Related Collateral

Foundational Materials

- *ADC Input Circuit Evaluation for C2000 MCUs (TINA-TI) Application Report*
- C2000 Academy - Analog Subsystem
- PSpice for TI design and simulation tool
- *Real-Time Control Reference Guide*
  - Refer to the ADC section
- TI Precision Labs - ADCs (Video)
  - Start with the "Introduction to analog-to-digital converters (ADCs)" section.
- TI Precision Labs: Driving the reference input on a SAR ADC (Video)
- TI Precision Labs: Introduction to analog-to-digital converters (ADCs) (Video)
- TI Precision Labs: SAR ADC input driver design (Video)
- TI e2e: Connecting VDDA to VREFHI
- TI e2e: Topologies for ADC Input Protection
- TI e2e: Why does the ADC Input Voltage drop with sampling?
  - Sampling a high impedance voltage divider with ADC
- TINA-TI SPICE-based Analog Simulation Program
- *Understanding Data Converters Application Report*

Getting Started Materials

- ADC-PWM Synchronization Using ADC Interrupt
  - NOTE: This is a non-TI (third party) site.
- *Hardware Design Guide for F2800x C2000 Real-Time MCU Series*

Expert Materials

- Analog Engineer's Calculator
- Analog Engineer's Pocket Reference
- *Charge-Sharing Driving Circuits for C2000 ADCs (PSPICE for TI) Application Report*
- *Charge-Sharing Driving Circuits for C2000 ADCs (using PSPICE-FOR-TI) Application Report*
- Debugging an integrated ADC in a microcontroller using an oscilloscope
- *Methods for Mitigating ADC Memory Cross-Talk Application Report*
- TI Precision Labs: ADC AC specifications (Video)
- TI Precision Labs: ADC Error sources (Video)
- TI Precision Labs: ADC Noise (Video)
- TI Precision Labs: Analog-to-digital converter (ADC) drive topologies (Video)
- TI Precision Labs: Electrical overstress on data converters (Video)
- TI Precision Labs: High-speed ADC fundamentals (Video)
- TI Precision Labs: SAR & Delta-Sigma: Understanding the Difference (Video)
- TI e2e: ADC Bandwidth Clarification
- TI e2e: ADC Calibration and Total Unadjusted Error

- TI e2e: ADC Reference Driver Options
- TI e2e: ADC Resolution with Oversampling
- TI e2e: ADC configuration for interleaved mode
- TI e2e: Simultaneous Sampling with Single ADC

16.1.2 Features

Each ADC has the following features:

- Ratiometric external reference set by VREFHI and VREFLO pins
- Single-ended signal conversions
- Input multiplexer with up to 16 channels
- 16 configurable SOCs
- 16 individually addressable result registers
- Multiple trigger sources
  - S/W - software immediate start
  - All ePWMs - ADCSOC A or B
  - GPIO XINT2
  - CPU Timers 0/1/2
  - ADCINT1/2
- Four flexible PIE interrupts
- Burst mode
- Four post-processing blocks, each with:
  - Saturating offset calibration
  - Error from setpoint calculation
  - High, low, and zero-crossing compare, with interrupt and ePWM trip capability
  - Trigger-to-sample delay capture

Note

Not every channel may be pinned out from all ADCs. Check the datasheet for your device to determine which channels are available.

16.1.3 Block Diagram

Figure 16-1 shows the block diagram for the ADC core and ADC wrapper.

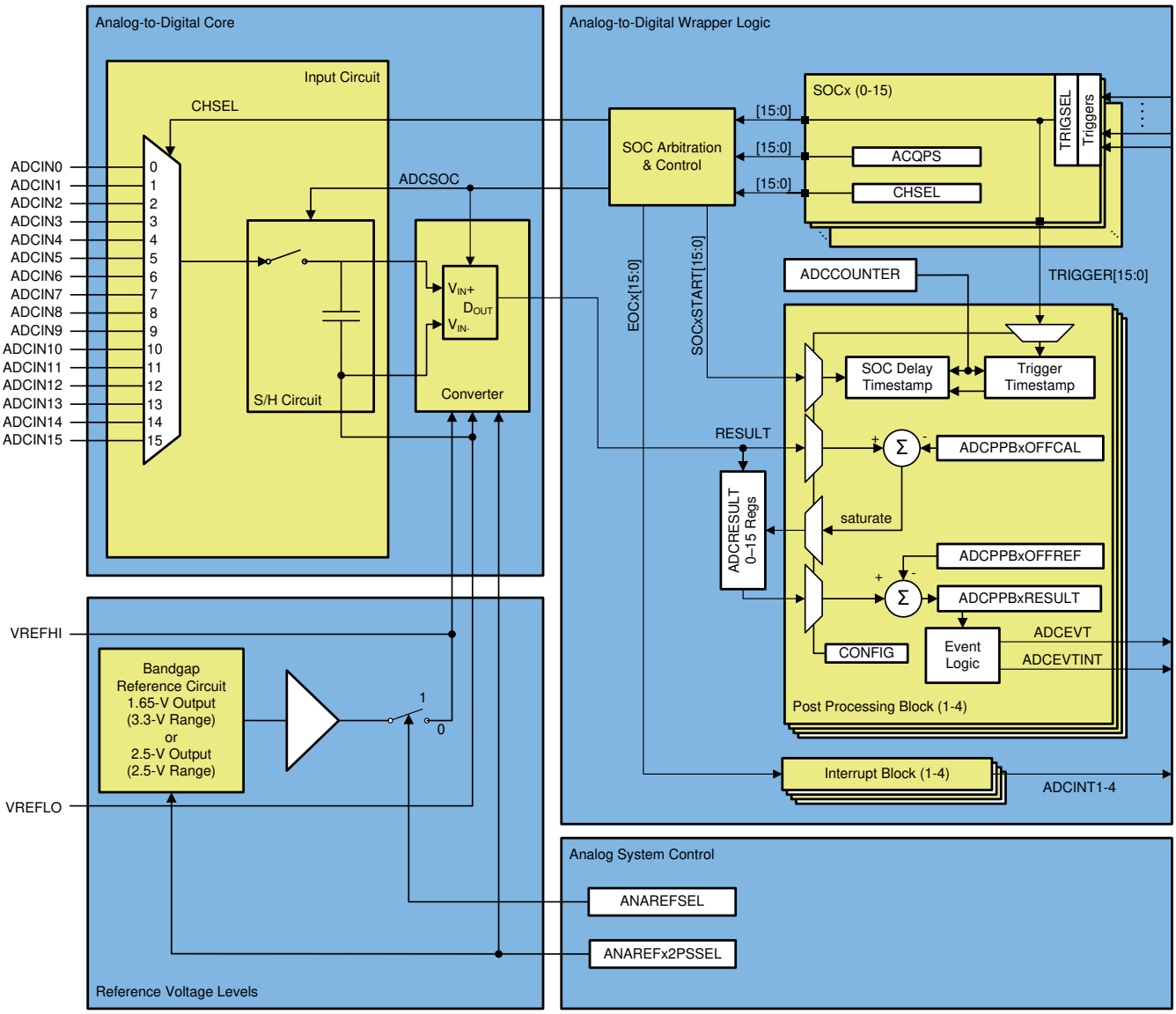


Figure 16-1. ADC Module Block Diagram

16.2 ADC Configurability

Some ADC configurations are individually controlled by the SOCs, while others are globally controlled per ADC module. Table 16-1 summarizes the basic ADC options and their level of configurability. The subsequent sections discuss these configurations.

Table 16-1. ADC Options and Configuration Levels

Options	Configurability
Clock	Per module(1)
Resolution	Not configurable (12-bit only)
Signal mode	Not configurable (single-ended only)
Reference voltage source	Either external or internal for all modules
Trigger source	Per SOC(1)
Converted channel	Per SOC
Acquisition window duration	Per SOC(1)
EOC location	Per module
Burst Mode	Per module(1)

(1) Writing these values differently to different ADC modules could cause the ADCs to operate asynchronously. See Section 16.13.1 for guidance on when the ADCs are operating synchronously or asynchronously.

16.2.1 Clock Configuration

The base ADC clock is provided directly by the system clock (SYSCLK). SYSCLK is used to generate the ADC acquisition window. The register ADCCTL2 has a PRESCALE field that determines the ADCCLK. The ADCCLK is used to clock the converter.

The core requires approximately 10.5 ADCCLK cycles to process a voltage into a conversion result. It is necessary for the user to determine the required duration of the acquisition window, see Section 16.13.2.

Note

To determine an appropriate value for ADCCTL2.PRESCALE, see your device-specific datasheet to determine the maximum SYSCLK and ADCCLK frequency.

16.2.2 Resolution

The resolution of the ADC determines how finely the analog range is quantized into digital values. This ADC supports a resolution of 12 bits.

16.2.3 Voltage Reference

16.2.3.1 External Reference Mode

The ADC modules share VREFHI and VRELO inputs. In external reference mode these pins are used as a ratiometric reference to determine the ADC conversion input range.

See Section 16.13.6 for information on how to supply the reference voltage.

Note

- On devices with no external VREFLO signals, VREFLO is internally connected to the device analog ground, VSSA.
- See your device-specific datasheet to determine the allowable voltage range for VREFHI and VREFLO.
- The external reference mode requires an external capacitor on the VREFHI pin. See your device-specific datasheet for the specific value required.

16.2.3.2 Internal Reference Mode

In internal reference mode the device drives a voltage out onto the VREFHI pin. The VREFHI and VREFLO pins then set the ADC conversion range.

The internal reference voltage can be configured to be either 2.5V or 1.65V. When a 1.65V internal reference voltage is selected, the effective ADC conversion input range will be VREFLO to 3.3V.

Note

The internal reference mode also requires an external capacitor on the VREFHI pin. See your device-specific datasheet for the specific value required.

16.2.3.3 Selecting Reference Mode

The voltage reference mode should be configured by using either SetVREF or ADC_setVREF function provided in C2000Ware. Using any of these functions ensures that the correct trim is loaded in the ADC trim registers. This function must be called at least once after a device reset. Do not configure the voltage reference mode by directly writing to the ANAREFCTL register.

16.2.4 Signal Mode

The ADC supports single-ended signaling.

In single-ended mode, the input voltage to the converter is sampled through a single pin (ADCINx), referenced to VREFLO.

16.2.5 Expected Conversion Results

Based on a given analog input voltage, the ideal expected digital conversion is given in Table 16-2. Fractional values are truncated.

Table 16-2. Analog to 12-bit Digital Formulas

Analog Input	Digital Result
when ADCINy ≤ VREFLO	ADCRESULTx = 0
when VREFLO < ADCINy < VREFHI	$\text{ADCRESULTx} = 4096 \left(\frac{\text{ADCINy - VREFLO}}{\text{VREFHI - VREFLO}} \right)$
when ADCINy ≥ VREFHI	ADCRESULTx = 4095

16.2.6 Interpreting Conversion Results

Based on a given ADC conversion result, the ideal corresponding analog input is given in Table 16-3. This corresponds to the center of the possible range of analog voltages that could have produced this conversion result.

Table 16-3. 12-Bit Digital-to-Analog Formulas

Digital Value	Analog Equivalent
when ADCRESULTy = 0	ADCINx ≤ VREFLO
when 0 < ADCRESULTy < 4095	$\text{ADCINx} = (\text{VREFHI - VREFLO}) \left(\frac{\text{ADCRESULTy}}{4096} \right) + \text{VREFLO}$
when ADCRESULTy = 4095	ADCINx ≥ VREFHI

16.3 SOC Principle of Operation

The ADC triggering and conversion sequencing is accomplished through configurable start-of-conversions (SOCs). Each SOC is a configuration set defining the single conversion of a single channel. In that set there are three configurations: the trigger source that starts the conversion, the channel to convert, and the acquisition (sample) window duration. Upon receiving the trigger configured for a SOC, the wrapper will ensure that the specified channel is captured using the specified acquisition window duration.

Multiple SOCs can be configured for the same trigger, channel, and/or acquisition window as desired. Configuring multiple SOCs to use the same trigger will allow the trigger to generate a sequence of conversions. Configuring multiple SOCs to use the same trigger and channel will allow for oversampling.

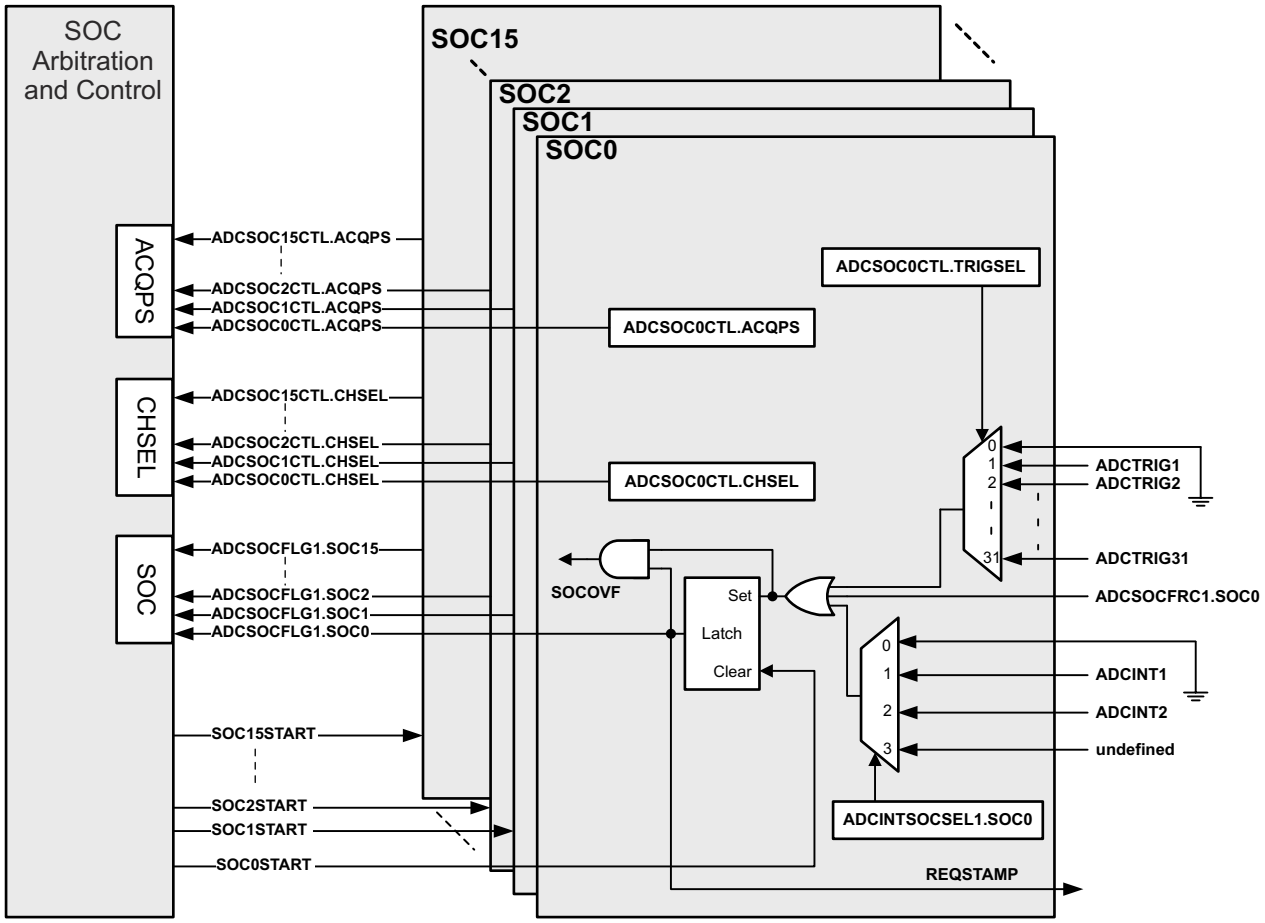


Figure 16-2. SOC Block Diagram

16.3.1 SOC Configuration

Each SOC has its own configuration register, ADCSOCxCTL. Within this register, SOCx can be configured for trigger source, channel to convert, and acquisition (sample) window duration.

16.3.2 Trigger Operation

Each SOC can be configured to start on one of many input triggers. The primary trigger select for SOCx is in the ADCSOCxCTL.TRIGSEL register, which can select between:

- Disabled (software only)
- CPU Timers 0/1/2
- GPIO: Input X-Bar INPUT5
- ADCSOCA or ADCSOCB from each ePWM module

In addition, each SOC can also be triggered when the ADCINT1 flag or ADCINT2 flag is set. This is achieved by configuring the ADCINTSOCSEL1 register (for SOC0 to SOC7) or the ADCINTSOCSEL2 register (for SOC8 to SOC15). This is useful for creating continuous conversions.

16.3.3 ADC Acquisition (Sample and Hold) Window

External signal sources vary in their ability to drive an analog signal quickly and effectively. In order to achieve rated resolution, the signal source needs to charge the sampling capacitor in the ADC core to within 0.5LSBs of the signal voltage. The acquisition window is the amount of time the sampling capacitor is allowed to charge and is configurable for SOCx by the ADCSOCxCTL.ACQPS register.

ACQPS is a 9-bit register that can be set to a value between 0 and 511, resulting in an acquisition window duration of:

Acquisition window = (ACQPS + 1)·(System Clock (SYSCLK) cycle time)

- The acquisition window duration is based on the System Clock (SYSCLK), not the ADC clock (ADCCLK).
- The selected acquisition window duration must be at least as long as one ADCCLK cycle.
- The datasheet will specify a minimum acquisition window duration (in nanoseconds). The user is responsible for selecting an acquisition window duration that meets this requirement.

16.3.4 ADC Input Models

For single-ended operation, the ADC input characteristics for values in the single-ended input model (see Figure 16-3) can be found in the device-specific datasheet.

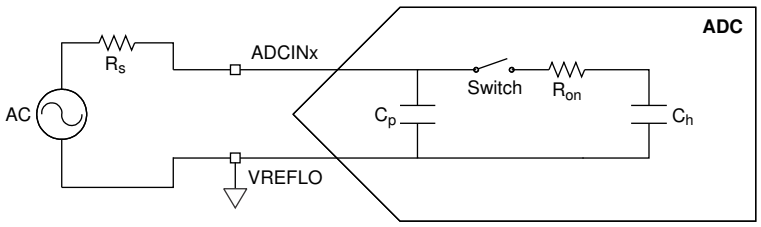


Figure 16-3. Single-Ended Input Model

These input models should be used along with actual signal source impedance to determine the acquisition window duration. See Section 16.13.2 for more information.

16.3.5 Channel Selection

Each SOC can be configured to convert any of the ADC channels. This behavior is selected for SOCx by the ADCSOCxCTL.CHSEL register. This is summarized in Table 16-4.

Table 16-4. Channel Selection of Input Pins

Input Mode	CHSEL	Input
Single-Ended	0	ADCIN0
	1	ADCIN1
	2	ADCIN2
	3	ADCIN3
	4	ADCIN4
	5	ADCIN5
	6	ADCIN6
	7	ADCIN7
	8	ADCIN8
	9	ADCIN9
	10	ADCIN10
	11	ADCIN11
	12	ADCIN12
	13	ADCIN13
	14	ADCIN14
	15	ADCIN15

16.4 SOC Configuration Examples

The following sections provide some specific examples of how to configure the SOCs to produce some conversions.

16.4.1 Single Conversion from ePWM Trigger

To configure ADCA to perform a single conversion on channel ADCIN1 when the ePWM timer reaches its period match, a few things are necessary. First, ePWM3 must be configured to generate an SOCA or SOCB signal (in this statement, SOC refers to a signal in the ePWM module). See the *Enhanced Pulse Width Modulator Module (ePWM)* chapter on how to do this. Assume that SOCB was chosen.

SOC5 is chosen arbitrarily. Any of the 16 SOCs could be used.

Assuming a 100 ns sample window is desired with a SYSCLK frequency of 100 MHz, then the acquisition window duration should be 100 ns/10 ns = 10 cycles. The ACQPS field should therefore be set to 10 - 1 = 9.

```
AdcaRegs.ADCSOC5CTL.bit.CHSEL = 1;      //SOC5 will convert ADCINA1
AdcaRegs.ADCSOC5CTL.bit.ACQPS = 9;     //SOC5 will use sample duration of 10 SYSCLK cycles
AdcaRegs.ADCSOC5CTL.bit.TRIGSEL = 10;   //SOC5 will begin conversion on ePWM3 SOCB
```

As configured, when ePWM3 matches its period and generates the SOCB signal, the ADC will begin sampling channel ADCINA1 (SOC5) immediately if the ADC is idle. If the ADC is busy, ADCINA1 will begin sampling when SOC5 gains priority (see Section 16.5). The ADC control logic will sample ADCINA1 with the specified acquisition window width of 100 ns. Immediately after the acquisition is complete, the ADC will begin converting the sampled voltage to a digital value. When the ADC conversion is complete, the results will be available in the ADCRESULT5 register (see Section 16.12 for exact sample, conversion, and result latch timings).

16.4.2 Oversampled Conversion from ePWM Trigger

To configure the ADC to oversample ADCINA1 4 times, we use the same configurations as the previous example, but apply them to SOC5, SOC6, SOC7, and SOC8.

As configured, when ePWM3 matches its period and generates the SOCB signal, the ADC will begin sampling channel ADCINA1 (SOC5) immediately if the ADC is idle. If the ADC is busy, ADCINA1 will begin sampling when SOC5 gains priority (see Section 16.5). Once the conversion is complete for SOC5, SOC6 will begin converting ADCINA1 and the results for SOC5 will be placed in the ADCRESULT5 register. All four conversions will eventually be completed sequentially, with the results in ADCRESULT5, ADCRESULT6, ADCRESULT7, and ADCRESULT8 for SOC5, SOC6, SOC7, and SOC8, respectively.

Note

It is possible, but unlikely, that the ADC could begin converting SOC6, SOC7, or SOC8 before SOC5 depending on the position of the round-robin pointer when the ePWM trigger is received. See Section 16.5 to understand how the next SOC to be converted is chosen.

16.4.3 Multiple Conversions from CPU Timer Trigger

This example will show how to sample multiple signals with different acquisition window requirements. CPU1 Timer 2 will be used to generate the trigger. To see how to configure the CPU timer, see the *System Control and Interrupts* chapter.

A good first step when designing a sampling scheme with many signals is to list out the signals and their required acquisition window. From this, calculate the necessary number of SYSCLK cycles for each signal, then the ACQPS register setting. This is shown in Table 16-5, where a SYCLK of 100 MHz is assumed (10 ns cycle time).

Table 16-5. Example Requirements for Multiple Signal Sampling

Signal Name	Acquisition Window Requirement (ns)	Acquisition Window SYSCLK Cycles	ACQPS Register Value
Signal 1	>240 ns	240 ns/10 ns = 24	24 – 1 = 23
Signal 2	>883.333 ns	883.333 ns/10 ns = 89 (round up)	89 – 1 = 88
Signal 3	>220 ns	220 ns/10 ns = 22	22 – 1 = 21
Signal 4	>585 ns	585 ns/10 ns = 59 (round up)	59 – 1 = 58

Next decide which ADC pins to connect to each signal. This will be highly dependent on application board layout. Once the pins are selected, determining the value of CHSEL is straightforward (see Table 16-6).

Table 16-6. Example Connections for Multiple Signal Sampling

Signal Name	ADC Pin	CHSEL Register Value
Signal 1	ADCINA5	5
Signal 2	ADCINA0	0
Signal 3	ADCINA3	3
Signal 4	ADCINA2	2

With the information tabulated, it is easy to generate the SOC configurations:

```
AdcaRegs.ADCSOC0CTL.bit.CHSEL = 5;        //SOC0 will convert ADCINA5
AdcaRegs.ADCSOC0CTL.bit.ACQPS = 23;       //SOC0 will use sample duration of 24 SYSCLK cycles
AdcaRegs.ADCSOC0CTL.bit.TRIGSEL = 3;      //SOC0 will begin conversion on CPU1 Timer 2
AdcaRegs.ADCSOC1CTL.bit.CHSEL = 0;        //SOC1 will convert ADCINA0
AdcaRegs.ADCSOC1CTL.bit.ACQPS = 88;       //SOC1 will use sample duration of 89 SYSCLK cycles
AdcaRegs.ADCSOC1CTL.bit.TRIGSEL = 3;      //SOC1 will begin conversion on CPU1 Timer 2
AdcaRegs.ADCSOC2CTL.bit.CHSEL = 3;        //SOC2 will convert ADCINA3
AdcaRegs.ADCSOC2CTL.bit.ACQPS = 21;       //SOC2 will use sample duration of 22 SYSCLK cycles
AdcaRegs.ADCSOC2CTL.bit.TRIGSEL = 3;      //SOC2 will begin conversion on CPU1 Timer 2
AdcaRegs.ADCSOC3CTL.bit.CHSEL = 2;        //SOC3 will convert ADCINA2
AdcaRegs.ADCSOC3CTL.bit.ACQPS = 58;       //SOC3 will use sample duration of 59 SYSCLK cycles
AdcaRegs.ADCSOC3CTL.bit.TRIGSEL = 3;      //SOC3 will begin conversion on CPU1 Timer 2
```

As configured, when CPU1 Timer 2 generates an event, SOC0, SOC1, SOC2, and SOC3 will eventually be sampled and converted, in that order. The conversion results for ACINA5 (Signal 1) will be in ADCRESULT0. Similarly, The results for ADCINA0 (Signal 2), ADCINA3 (Signal 3), and ADCINA2 (Signal 4) will be in ADCRESULT1, ADCRESULT2, and ADCRESULT3, respectively.

Note

It is possible, but unlikely, that the ADC could begin converting SOC1, SOC2, or SOC3 before SOC0 depending on the position of the round-robin pointer when the CPU Timer trigger is received. See Section 16.5 to understand how the next SOC to be converted is chosen.

16.4.4 Software Triggering of SOCs

At any point, whether or not the SOCs have been configured to accept a specific trigger, a software trigger can set the SOCs to be converted. This is accomplished by writing bits in the ADCSOCFRC1 register.

Software triggering of the previous example without waiting for the CPU1 Timer 2 to generate the trigger could be accomplished by the statement:

```
AdcaRegs.ADCSOCFRC1.all = 0x000F;        //set SOC flags for SOC0 to SOC3
```

16.5 ADC Conversion Priority

When multiple SOC flags are set at the same time, one of two forms of priority determines the order in which they are converted. The default priority method is round robin. In this scheme, no SOC has an inherent higher priority than another. Priority depends on the round robin pointer (RRPOINTER). The RRPOINTER reflected in the ADCSOCPRIORITYCTL register points to the last SOC converted. The highest priority SOC is given to the next value greater than the RRPOINTER value, wrapping around back to SOC0 after SOC15. At reset the value is 16 since 0 indicates a conversion has already occurred. When RRPOINTER equals 16 the highest priority is given to SOC0. The RRPOINTER is reset when the ADC module is reset or when the SOCPRICTL register is written. The ADC module is reset by writing and clearing the SOFTPRES bit corresponding to the ADC instance.

An example of the round robin priority method is given in Figure 16-4.

The SOCPRIORITY field in the ADCSOCPRIORITYCTL register can be used to assign high priority from a single to all of the SOCs. When configured as high priority, an SOC will interrupt the round robin wheel after any current conversion completes and insert itself in as the next conversion. After its conversion completes, the round robin wheel will continue where it was interrupted. If two high priority SOC's are triggered at the same time, the SOC with the lower number will take precedence.

High priority mode is assigned first to SOC0, then in increasing numerical order. The value written in the SOCPRIORITY field defines the first SOC that is not high priority. In other words, if a value of 4 is written into SOCPRIORITY, then SOC0, SOC1, SOC2, and SOC3 are defined as high priority, with SOC0 the highest.

An example using high priority SOC's is given in Figure 16-5.

A After reset, SOC0 is highest priority SOC; SOC7 receives trigger; SOC7 configured channel is converted immediately.

B RRPOINTER changes to point to SOC 7; SOC8 is now highest priority SOC .

C SOC2 & SOC12 triggers rcvd. simultaneously; SOC12 is first on round robin wheel ; SOC12 configured channel is converted while SOC2 stays pending.

D RRPOINTER changes to point to SOC 12; SOC2 configured channel is now converted .

E RRPOINTER changes to point to SOC 2; SOC3 is now highest priority SOC .

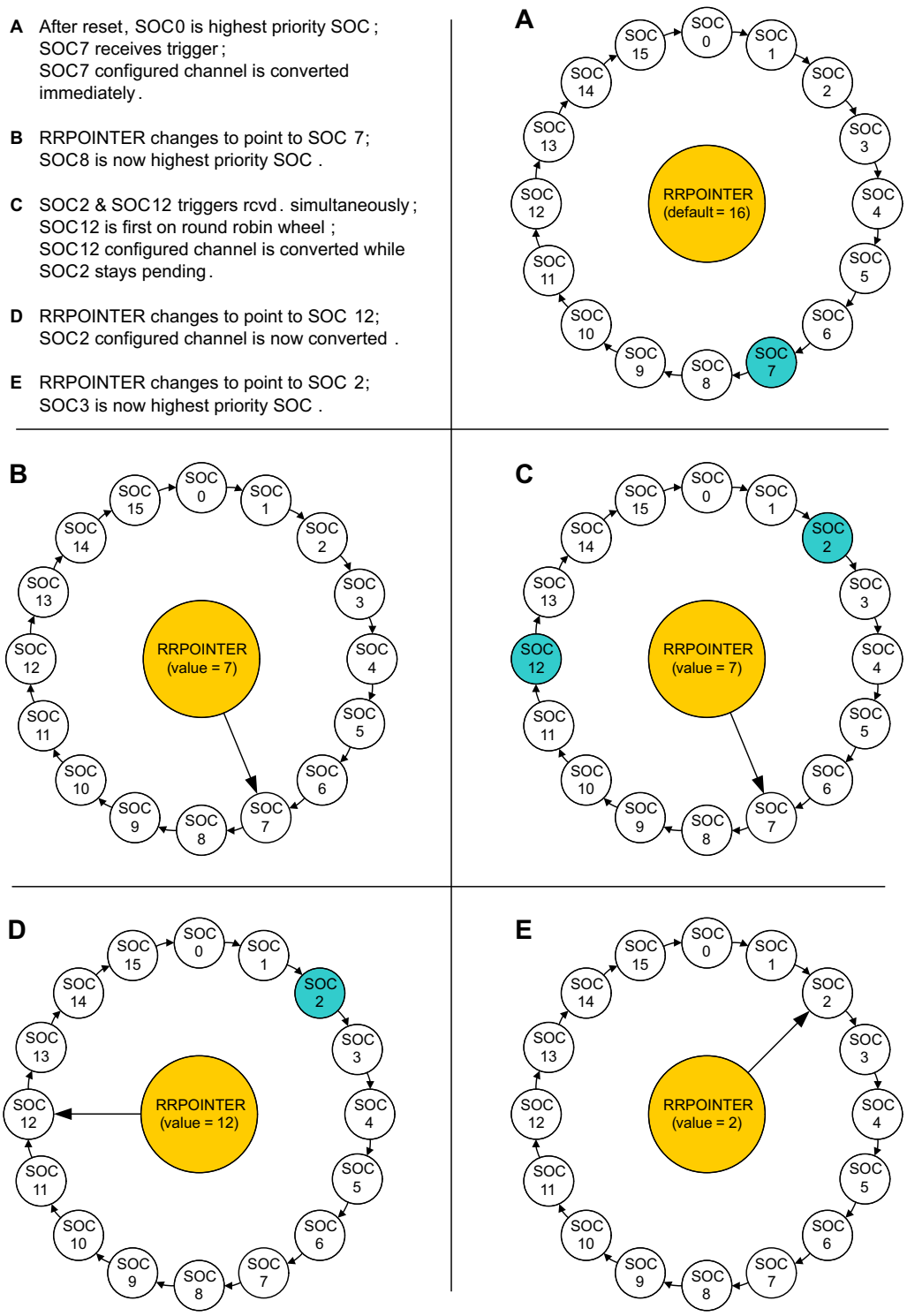


Figure 16-4. Round Robin Priority Example

Example when SOCPRIORITY = 4

A After reset, SOC4 is 1st on round robin wheel ;
SOC7 receives trigger ;
SOC7 configured channel is converted immediately .

B RRPOINTER changes to point to SOC 7;
SOC8 is now 1st on round robin wheel .

C SOC2 & SOC12 triggers rcvd . simultaneously ;
SOC2 interrupts round robin wheel and SOC 2 configured channel is converted while SOC 12 stays pending .

D RRPOINTER stays pointing to 7;
SOC12 configured channel is now converted .

E RRPOINTER changes to point to SOC 12;
SOC13 is now 1st on round robin wheel .

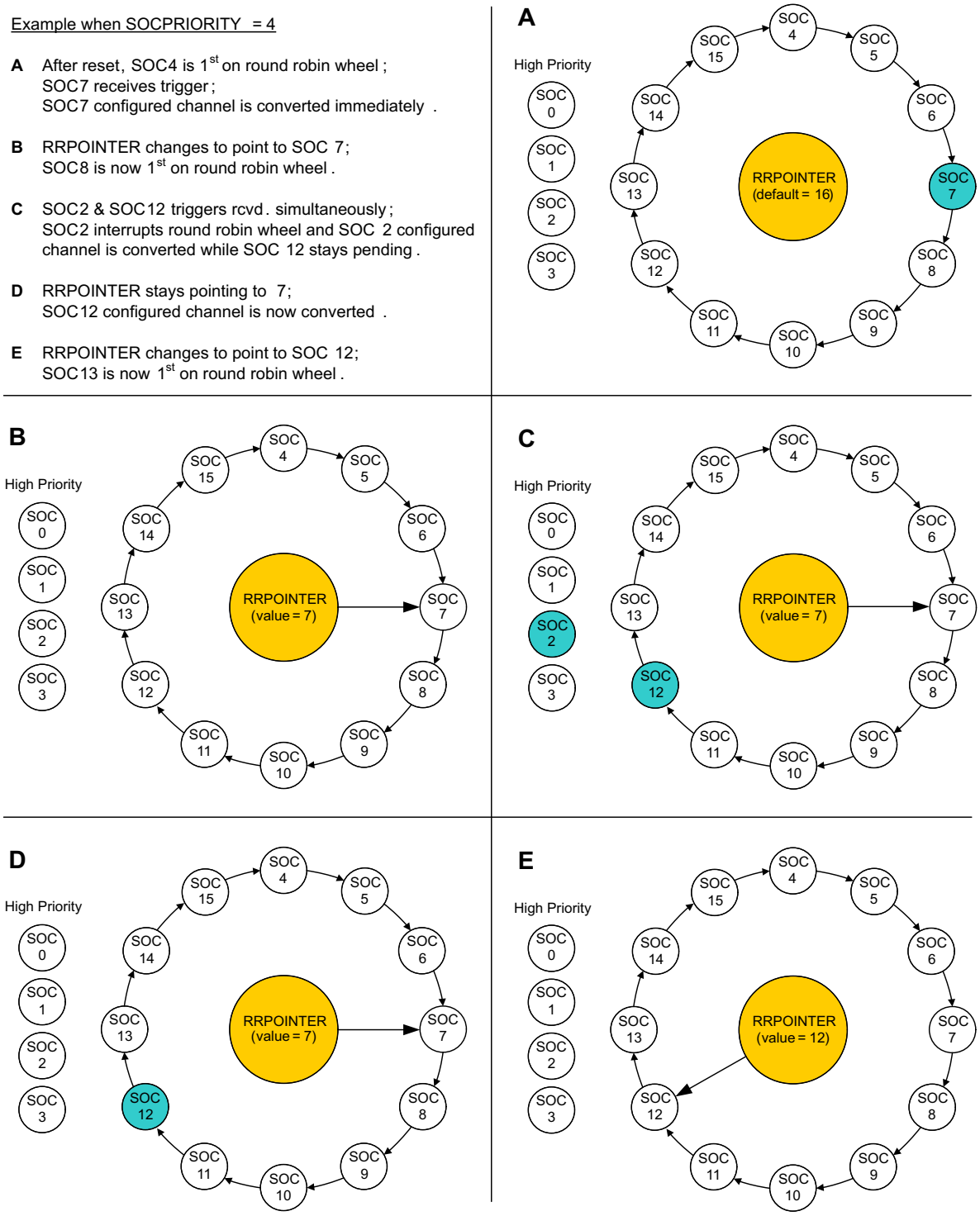


Figure 16-5. High Priority Example

16.6 Burst Mode

Burst mode allows a single trigger to walk through the round-robin SOCs one or more at a time. Setting the bit BURSTEN in the ADCBURSTCTL register configures the ADC wrapper for burst mode. This causes the TRIGSEL field to be ignored, but only for SOCs that are configured for round-robin operation (not high priority). Instead of the TRIGSEL field, all round-robin SOCs are triggered based on the BURSTTRIG field in the ADCBURSTCTL register. Upon reception of the burst trigger, the ADC wrapper will not set all round-robin SOCs to be converted, but only (ADCBURSTCTL.BURSTSIZE + 1) SOCs. The first SOC to be set will be that with the highest priority based on the round-robin pointer, and subsequent SOCs will be set until BURSTSIZE SOCs have been set.

Note

When configuring the ADC for burst mode, the user is responsible for ensuring that each burst of conversions is allowed to complete before the next burst trigger is received.

16.6.1 Burst Mode Example

Burst mode can be used to sample a different set of signals on every other trigger. In the following example, ADCIN7 and ADCIN5 are converted on the first trigger from CPU1 Timer 2 and every other trigger thereafter. ADCIN2 and ACIN3 are converted on the second trigger from CPU1 Timer 2 and every other trigger thereafter. All signals are converted with 20 SYSCLK cycle wide acquisition windows, but different durations could be configured for each SOC as desired.

```
AdcaRegs.BURSTCTL.BURSTEN = 1;              //Enable ADC burst mode
AdcaRegs.BURSTCTL.BURSTTRIG = 3;            //CPU1 Timer 2 will trigger burst of conversions
AdcaRegs.BURSTCTL.BURSTSIZE = 1;            //conversion bursts are 1 + 1 = 2 conversions long
AdcaRegs.SOCPRICTL.bit.SOCPRIORITY = 12;    //SOC0 to SOC11 are high priority
AdcaRegs.ADCSOC12CTL.bit.CHSEL = 7;         //SOC12 will convert ADCINA7
AdcaRegs.ADCSOC12CTL.bit.ACQPS = 19;        //SOC12 will use sample duration of 20 SYSCLK cycles
AdcaRegs.ADCSOC13CTL.bit.CHSEL = 5;         //SOC13 will convert ADCINA5
AdcaRegs.ADCSOC13CTL.bit.ACQPS = 19;        //SOC13 will use sample duration of 20 SYSCLK cycles
AdcaRegs.ADCSOC14CTL.bit.CHSEL = 2;         //SOC14 will convert ADCINA2
AdcaRegs.ADCSOC14CTL.bit.ACQPS = 19;        //SOC14 will use sample duration of 20 SYSCLK cycles
AdcaRegs.ADCSOC15CTL.bit.CHSEL = 3;         //SOC15 will convert ADCINA3
AdcaRegs.ADCSOC15CTL.bit.ACQPS = 19;        //SOC15 will use sample duration of 20 SYSCLK cycles
```

When the first CPU1 Timer 2 trigger is received, SOC12 and SOC13 will be converted immediately if the ADC is idle. If the ADC is busy, SOC12 and SOC13 will be converted once their SOCs gain priority. The results for SOC12 and SOC13 will be in ADCRESULT12 and ADCRESULT13, respectively. After SOC13 completes, the round robin pointer will give highest priority to SOC14. Because of this, when the next CPU1 Timer 2 trigger is received, SOC14 and SOC15 will be set as pending and eventually be converted. The results for SOC14 and SOC15 will be in ADCRESULT14 and ADCRESULT15, respectively. Subsequent triggers will continue to toggle between converting SOC12 and SOC13, and converting SOC14 and SOC15.

While the above example toggles between two sets of conversions, three or more different sets of conversions could be achieved using a similar approach.

16.6.2 Burst Mode Priority Example

An example of priority resolution using burst mode and high-priority SOCs is presented in Figure 16-6.

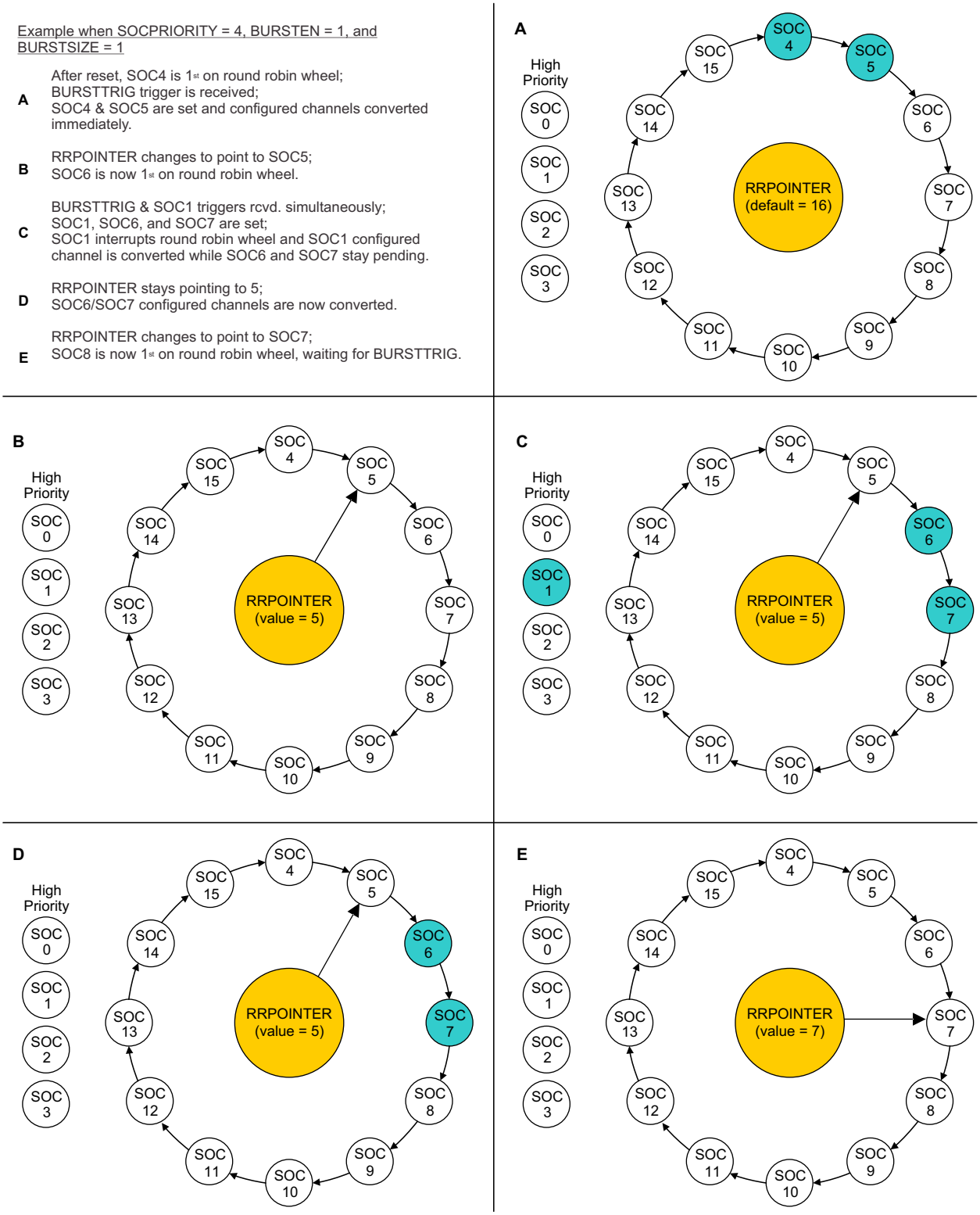


Figure 16-6. Burst Priority Example

16.7 EOC and Interrupt Operation

Each SOC has a corresponding end-of-conversion (EOC) signal. This EOC signal can be used to trigger an ADC interrupt. The ADC can be configured to generate the EOC pulse at either the end of the acquisition window or at the end of the voltage conversion. This is configured using the bit INTPULSEPOS in the ADCCTL1 register. See Section 16.12 for exact EOC pulse location.

Each ADC module has 4 configurable ADC interrupts. These interrupts can be triggered by any of the 16 EOC signals. The flag bit for each ADCINT can be read directly to determine if the associated SOC is complete or the interrupt can be passed on to the PIE.

Note

The ADCCTL1.ADCBSY bit being clear does not indicate that all conversions in a set of SOCs have completed, only that the ADC is ready to process the next conversion. To determine if a sequence of SOCs is complete, link an ADCINT flag to the last SOC in the sequence and monitor that ADCINT flag.

Figure 16-7 shows a block diagram of the ADC interrupt structure.

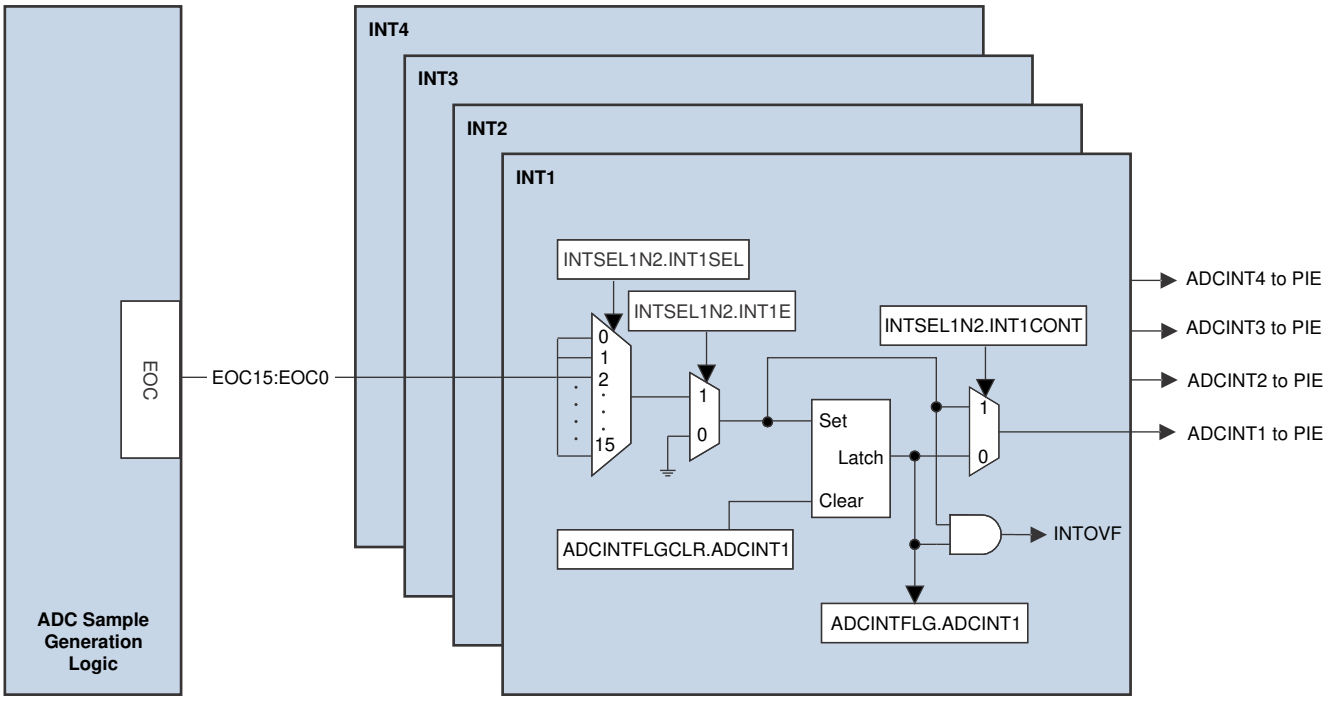


Figure 16-7. ADC EOC Interrupts

16.7.1 Interrupt Overflow

If the EOC signal would set a flag in the ADCINTFLG register, but that flag is already set, an interrupt overflow occurs. By default, overflow interrupts will not be passed on to the PIE module. When an overflow occurs on a given flag in the ADCINTFLG register, the corresponding flag in the ADCINOVF register is set. This overflow flag is only used to detect that an overflow has occurred; it does not block further interrupts from propagating to the PIE module.

When an ADC interrupt overflow could occur, the application should check the appropriate ADCINTOVF flag inside the ISR or in the background loop and take appropriate action when an overflow is detected. The following code snippets demonstrate how to check the ADCINOVF flag inside the ISR after attempting to clear the ADCINT flag.

```
// Clear the interrupt flag
AdcaRegs.ADCINTFLGCLR.bit.ADCINT1 = 1;      //clear INT1 flag for ADC-A

// Check if an overflow has occurred
if(1 == AdcaRegs.ADCINTOVF.bit.ADCINT1)     //ADCINT overflow occurred
{
    AdcaRegs.ADCINTOVFCLR.bit.ADCINT1 = 1   //Clear overflow flag
    AdcaRegs.ADCINTFLGCLR.bit.ADCINT1 = 1   //Re-clear ADCINT flag
}
```

```
//
// Clear the interrupt flag
//
ADC_clearInterruptStatus(ADCA_BASE, ADC_INT_NUMBER1);

//
// Check if an overflow has occurred
//
if(true == ADC_getInterruptOverflowStatus(ADCA_BASE, ADC_INT_NUMBER1))
{
    ADC_clearInterruptOverflowStatus(ADCA_BASE, ADC_INT_NUMBER1);
    ADC_clearInterruptStatus(ADCA_BASE, ADC_INT_NUMBER1);
}
```

16.7.2 Continue to Interrupt Mode

The INTxCONT bits in the ADCINTSEL1N2 and ADCINTSEL3N4 registers configure how interrupts are handled when an ADCINTFLG has not yet been cleared from a prior interrupt. This mode is disabled by default and additional overlapping interrupts will not be issued to the PIE. By activating this mode ADC interrupts will always reach the PIE. The ADCINTOVF register will still be set if an interrupts occur while ADCINTFLG is still set regardless of this configuration.

16.7.3 Early Interrupt Configuration Mode

Enabling early interrupt mode can allow the application to enter the ADC interrupt service routine before the ADC results are ready. This allows the application to do any necessary pre-work so that the application can act on the ADC results immediately when they become available. However, the timing of the early interrupt may be too early and the application may need to waste time until the updated ADC results become available. To prevent this situation, the time the ADC interrupt is entered in early interrupt mode is configurable by way of the DELAY field in the ADCINTCYCLE register.

- To use the configurable interrupt time, the ADC must be in early interrupt mode. To achieve this, clear the bit INTPULSEPOS to 0 in ADCCTL1.
- The DELAY value in the ADCINTCYCLE register sets the number of additional SYSCLK cycles after the falling edge of the SOC pulse before the ADCINT flag is set.
- If the value of DELAY goes beyond EOC, the ADC interrupt will be generated along with EOC.
- Writing values to DELAY when INTPULSEPOS is set to 1 will not have any effect on the interrupt generation.

16.8 Post-Processing Blocks

Each ADC module contains four post-processing blocks (PPB). These blocks can be associated with any of the 16 RESULT registers using the ADCPPBxCONFIG.CONFIG bit field. Each PPB can simultaneously remove an offset associated with the ADCIN channel, subtract out a reference value, flag a zero-crossing point, and flag a high or low compare limit. Furthermore, the zero-crossing and compare flags can trip a PWM and/or generate an interrupt. A PPB is also capable of recording the delay between when the SOC associated with the PPB is triggered and when it actually begins to be sampled. Figure 16-8 presents the structure of each PPB. Subsequent sections explain the use of each submodule.

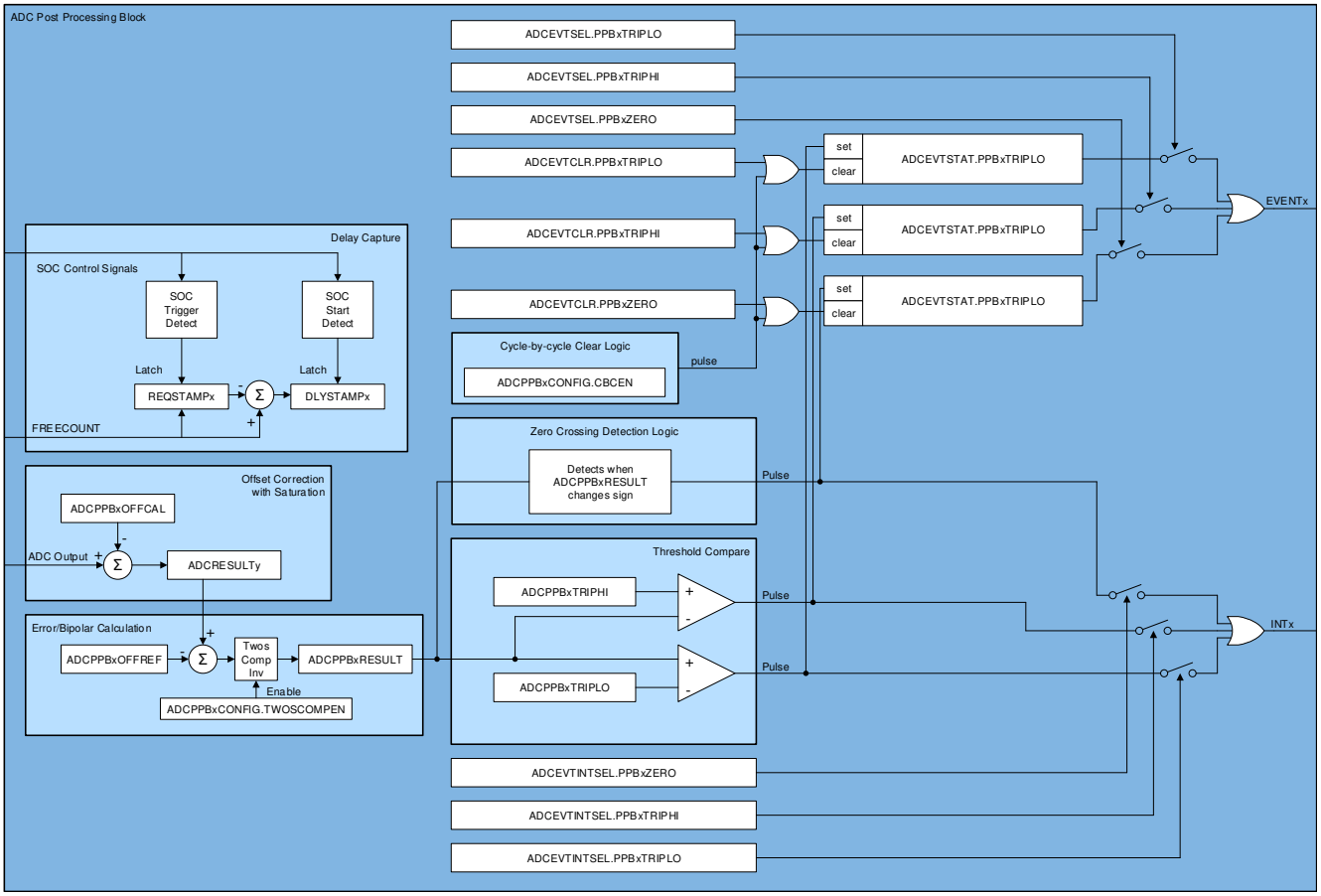


Figure 16-8. ADC PPB Block Diagram

16.8.1 PPB Offset Correction

In many applications, external sensors and signal sources produce an offset. A global trimming of the ADC offset is not enough to compensate for these offsets, which vary from channel to channel. The post-processing block can remove these offsets with zero overhead, saving numerous cycles in tight control loops.

Offset correction is accomplished by first pointing the ADCPPBxCONFIG.CONFIG to the desired SOC, then writing an offset correction value to the ADCPPBxOFFCAL.OFFCAL register. The post-processing block will automatically add or subtract the value in the OFFCAL register from the raw conversion result and store it in the ADCRESULT register.

Note

- Writing a 0 to the OFFCAL register effectively disables the offset correction feature, passing the raw result unchanged to the ADCRESULT register.
- It is possible to point multiple PPBs to the same SOC. In this case, the OFFCAL value that will actually be applied will be that of the PPB with the highest number.
- In particular, care needs to be taken when using the PPB on SOC0, as all the PPB point to this SOC by default. This may cause unintentional overwriting of offset correction of a lower numbered PPB by a higher numbered PPB.

16.8.2 PPB Error Calculation

In many applications, an error from a setpoint or expected value must be computed from the digital output of an ADC conversion. In other cases, a bipolar signal is necessary or convenient for control calculations. The PPB can perform these function automatically, reducing the sample to output latency and reducing software overhead.

Error calculation is accomplished by first pointing the ADCPPBxCONFIG.CONFIG to the desired SOC, then writing a value to the ADCPPBxOFFCAL.OFFREF register. The post-processing block will automatically subtract the value in the OFFREF register from the ADCRESULT value and store it in the ADCPPBxRESULT register. This subtraction will produce a sign-extended 32-bit result. It is also possible to selectively invert the calculated value before storing in the ADCPPBxRESULT register by setting the TWOSCOMPEN bit in the ADCPPBxCONFIG register.

Note

- not write a value larger than 12 bits to the OFFREF register.
- Since the PPBxRESULT register is unique for each PPB, it is possible to point multiple PPBs to the same SOC and get different results for each PPB.
- Writing a 0 to the OFFREF register effectively disables the error calculation feature, passing the ADCRESULT value unchanged to the ADCPPBxRESULT register.

16.8.3 PPB Limit Detection and Zero-Crossing Detection

Many applications perform a limit check against the ADC conversion results. The PPB can automatically perform a check against a high and low limit or whenever ADCPPBxRESULT changes sign. Based on these comparisons, it can generate a trip to the PWM and/or an interrupt automatically, lowering the sample to ePWM latency and reducing software overhead. This functionality also enables safety conscious applications to trip the ePWM based on an out-of-range ADC conversion without any CPU intervention.

To enable this functionality, first point the ADCPPBxCONFIG.CONFIG to the desired SOC, then write a value to one or both of the registers ADCPPBxTRIPHI.LIMITHI and ADCPPBxTRIPLO.LIMITLO (zerocrossing detection does not require further configuration). Whenever these limits are exceeded, the PPBxTRIPHI bit or PPBxTRIPLO bit will be set in the ADCEVTSTAT register. Note that the PPBxZERO bit in the ADCEVTSTAT register is gated by EOC and not by the sign change in the ADCPPBxRESULT register. The ADCEVTCLR

register has corresponding bits to clear these event flags. The ADCEVTSEL register has corresponding bits which allow the events to propagate through to the PWM. The ADCINTSEL register has corresponding bits which allow the events to propagate through to the PIE.

One PIE interrupt is shared between all the PPBs for a given ADC module as shown in Figure 16-9.

Note

- If different actions need to be taken for different PPB events from the same ADC module, then the ADCEVTINT ISR will have to read the PPB event flags in the ADCEVTSTAT register to determine which event caused the interrupt.
- If different ePWM trips need to be generated separately for high compare, low compare, and/or zero-crossing, this can be achieved by pointing multiple PPBs to the same SOC.
- The zero-crossing detect circuit considers a result of zero to be positive.

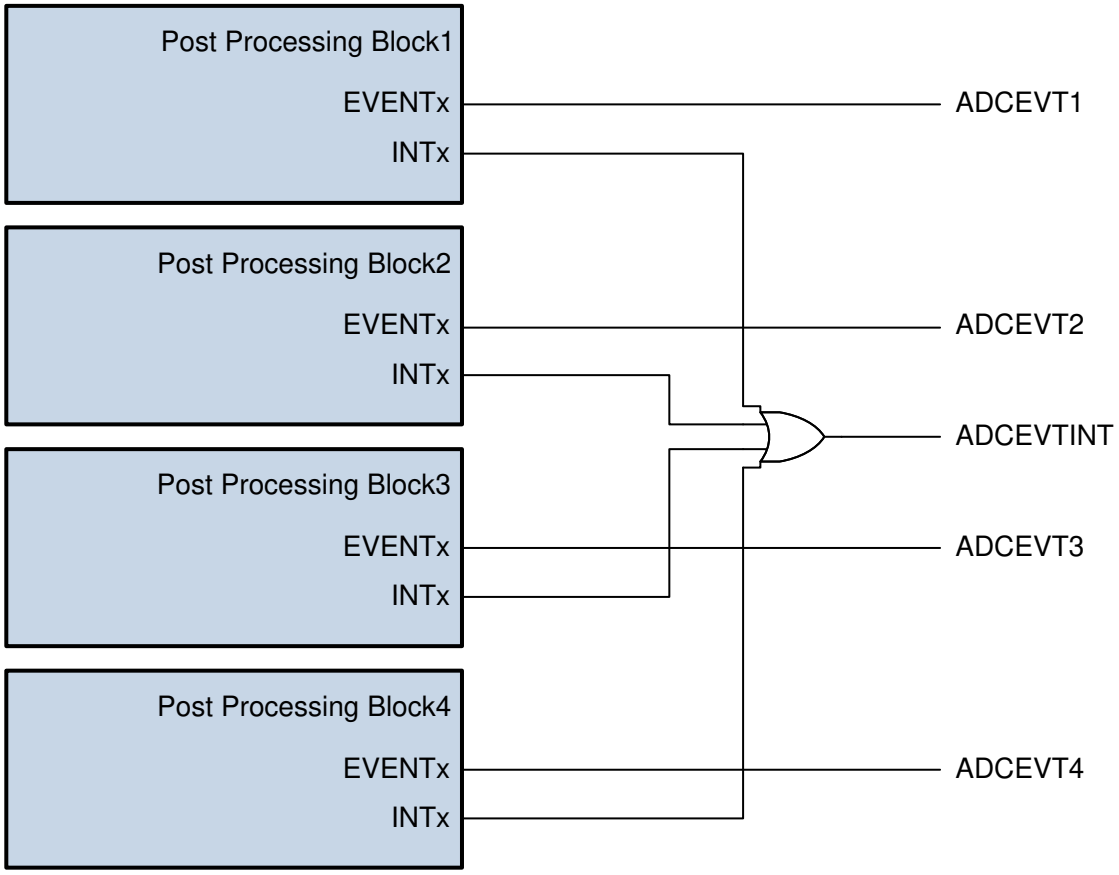


Figure 16-9. ADC PPB Interrupt Event

16.8.4 PPB Sample Delay Capture

When multiple control loops are running asynchronously on the same ADC, there is a chance that an ADC request from two or more loops will collide, causing one of the samples to be delayed. This shows up as a measurement error in the system. By knowing when this delay occurs and the amount of delay that has occurred software can employ extrapolation techniques to reduce the error.

To this effect, each PPB has the field DLYSTAMP in the ADCPPBxSTAMP register. This field will contain the number of SYSCLK cycles between when the associate SOC was triggered and when it began converting.

This is achieved by having a global 12-bit free running counter based off of SYSCLK, which is in the field FREECOUNT in the ADCCOUNTER register. When the trigger for the associated SOC arrives, the value of this counter is loaded into the bit field ADCPPBxTRIPLO.REQSTAMP. When the actual sample window for that SOC begins, the value in REQSTAMP is subtracted from the current FREECOUNT value and stored in DLYSTAMP.

Note

If more than 4096 SYSCLK cycles elapse between the SOC trigger and the actual start of the SOC acquisition, the FREECOUNT register may overflow more than once, leading to incorrect DLYSTAMP value. Be cautious when using very slow conversions to prevent this from happening.

The sample delay capture will not function if the associated SOC is triggered via software. It will, however, correctly record the delay if the software triggering of a different SOC causes the SOC associated with the PPB to be delayed

16.9 Opens/Shorts Detection Circuit (OSDETECT)

The opens/shorts detection circuit (OSDETECT) can be used to detect pin faults in the system. The circuit connects to the ADC input after the channel select multiplexer but before the S+H circuit as shown in Figure 16-10.

Note

- The divider resistance tolerances can vary widely, hence this feature should not be used to check for conversion accuracy.
- See the data sheet for implementation and availability of analog input channels.
- Due to high drive impedance, a S+H duration much longer than the ADC minimum will be needed.

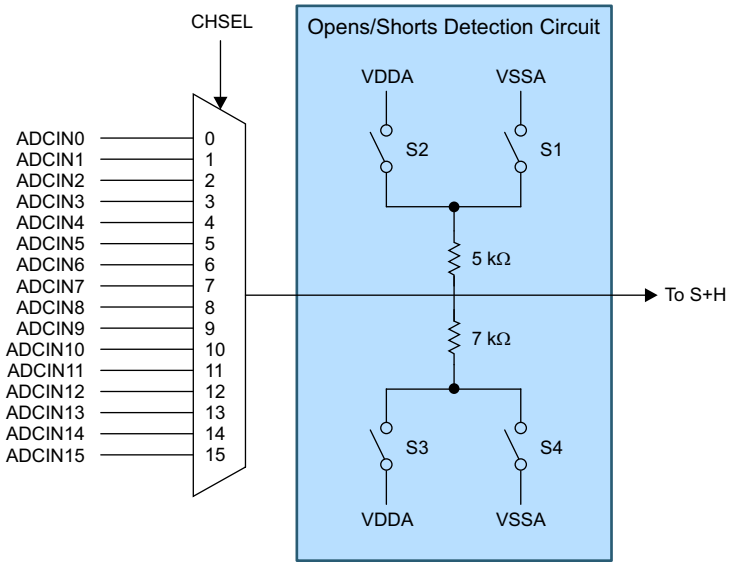


Figure 16-10. Opens/Shorts Detection Circuit

The circuit can be operated by writing a value to the DETECTCFG field in the ADCOSDETECT register. This will cause the circuit to source a voltage onto the input during the S+H phase of any conversion. The voltage and drive strength of the OSDETECT circuit for different DETECTCFG settings is given in Table 16-7.

Table 16-7. DETECTCFG Settings

ADCOSDETECT.DETECTCFG	Source Voltage	S4	S3	S2	S1	Drive Impedance
0	Off	Open	Open	Open	Open	Open
1	Zero Scale	Closed	Open	Open	Closed	5K || 7K
2	Full Scale	Open	Closed	Closed	Open	5K || 7K
3	5/12 VDDA	Open	Closed	Open	Closed	5K || 7K
4	7/12 VDDA	Closed	Open	Closed	Open	5K || 7K
5	Zero Scale	Open	Open	Open	Closed	5K
6	Full Scale	Open	Open	Closed	Open	5K
7	Zero Scale	Closed	Open	Open	Open	7K

16.9.1 Implementation

A representative circuit with the OSDETECT implementation consists of the signal source with series resistance R_S, shunt capacitor C_P, the equivalent OSDETECT resistance $R_{OSDETECT}$ and voltage $V_{OSDETECT}$ is shown in Figure 16-11 and can be used as a basis to calculate the signal level going in to the sampling capacitor. $R_{OSDETECT}$ and $V_{OSDETECT}$ are the equivalent input resistance and voltage source contributed by the OSDETECT circuit with values shown in Table 16-7 for the different configuration settings. Refer to Figure 16-11 when deriving the input signal to S/H if signal source V_S is driving while the OSDETECT feature is enabled.

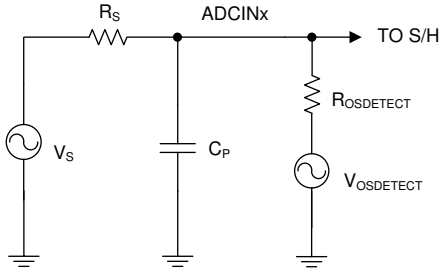


Figure 16-11. Input Circuit Equivalent with OSDETECT Enabled

The input impedance R_S and C_P may be integral parts of the signal source or these could have been implemented in the design to precondition the signal or to control signal settling time to meet S/H requirements. The input path has to be considered when using the OSDETECT feature as this would affect the conversion results. For instance, driving an input signal when this feature is enabled would connect signal V_S to the OSDETECT circuit through R_S and affecting the ADC results. Larger C_P values (in the order greater than hundreds of pF) would require using higher ACQPS to ensure that signal at the input has settled prior to conversion.

To enable the circuit:

1. Configure the ADC for conversion (for example, channel, SOC, ACQPS, prescaler, trigger, and so on).
2. Set up the ADCOSDETECT register for the desired voltage divider connection as shown in Table 16-7.
3. Initiate a conversion and inspect the conversion result.

You must interpret the results based on what is driving on the input side and what are the values of R_S and C_P. If the V_S signal can be disconnected from the input pin, the circuit can be used to detect open and shorted input pins as described in the following sections.

16.9.2 Detecting an Open Input Pin

By cycling through the various OSDETECT settings, the input signal will be pulled towards the sourced voltages. An input with good drive strength (pin not open) will be minimally affected. However, if the pin is open, the sampled voltages will be close to the source voltages specified in Table 16-7.

16.9.3 Detecting a Shorted Input Pin

By cycling through the various OSDETECT settings, the input signal will be pulled towards the sourced voltages. An input with finite drive strength (pin not shorted) will be pulled toward each sourced voltage. However, if the pin is shorted, the signal will remain at the same voltage.

16.10 Power-Up Sequence

Upon device power-up or system level reset, the ADC will be powered down and disabled. When powering up the ADC, use the following sequence:

1. Set the bit to enable the desired ADC clock in the PCLKCR13 register.
2. Set the desired ADC clock divider in the PRESCALE field of ADCCTL2.
3. Power up the ADC by setting the ADCPWDNZ bit in ADCCTL1.
4. Allow a delay before sampling. See the data manual for the necessary time.

If multiple ADCs are powered up simultaneously, steps 1 and step 3 can each be done for all ADCs in one write instruction. Also, only one delay is necessary as long as it occurs after all the ADCs have begun powering up.

16.11 ADC Calibration

During the fabrication and test process, Texas Instruments calibrates the gain, offset, and linearity of the ADCs and the offset of the buffered DACs. These trim settings are embedded into TI reserved OTP memory as part of C-callable functions.

- The Device_cal() function copies the trim values for ADC from OTP memory to their respective trim registers.
- The trim functions in Device_cal() are callable in C2000ware as ADC_setOFFSETTRIM(), ADC_setINLTRIM(). These functions fetch trim values from the TI reserved OTP memory source locations where the values are stored during test process as well as the analog module register destinations where the trim values are copied to.

Until the appropriate factory trim is loaded, the ADC (and other modules) will not be assured to operate within data sheet specifications. Similarly, if trim values other than the factory settings are placed into the trim registers, the ADC (and other modules) will not be assured to operate within data sheet specifications.

The boot ROM will call the calibration functions, so trim values should be initially populated without user intervention. However, if the trims are cleared due to a module reset or modified for some other reason, then the user can call the calibration functions (defined in the headerfiles).

16.11.1 ADC Zero Offset Calibration

Zero offset error is defined as the difference from 0 that occurs when converting a voltage at VREFLO. The zero offset error can be positive or negative. To correct this error, an adjustment of equal magnitude and opposite polarity is written into the ADCOFFTRIM register. The value contained in this register will be applied before the results are available in the ADC result registers. This operation is fully contained within the ADC core, so the timing of the results will not be affected and the full dynamic range of the ADC will be maintained for any trim value.

Using the GetAdcOffsetTrimOTP(Uint16) function, the ADCOFFTRIM register can be loaded with the factory calibrated offset error correction. You can modify the ADCOFFTRIM register to compensate for additional offset error induced by the application environment if desired, but this is not typically necessary to achieve data sheet specified performance.

Note

Regardless of the converter resolution, the size of each ADCOFFTRIM step is (VREFHI-VREFLO)/ 65536.

Use the following procedure to re-calibrate the ADC offset in 12-bit, single-ended mode:

1. Set ADCOFFTRIM to +112 steps (0x70). This adds an artificial offset to account for negative offset that may reside in the ADC core.
2. Perform some multiple of 16 conversions on VREFLO (internal connection), accumulating the results (for example, 32*16 conversions = 512 conversions).
3. Divide the accumulated result by the multiple of 16 (for example, for 512 conversions, divide by 32).
4. Set ADCOFFTRIM to 112 – result from step 3.

16.12 ADC Timings

The process of converting an analog voltage to a digital value is broken down into an S+H phase and a conversion phase. The ADC sample and hold circuits (S+H) are clocked by SYSCLK while the ADC conversion process is clocked by ADCCLK. ADCCLK is generated by dividing down SYSCLK based on the PRESCALE field in the ADCCTL2 register.

The S+H duration is the value of the ACQPS field of the SOC being converted, plus one, times the SYSCLK period. The user must ensure that this duration exceeds both 1 ADCCLK period and the minimum S+H duration specified in the datasheet. The conversion time is approximately 10.5

16.12.1 ADC Timing Diagrams

The following diagrams show the ADC conversion timings for two SOCs given the following assumptions:

- SOC0 and SOC1 are configured to use the same trigger.
- No other SOCs are converting or pending when the trigger occurs.
- The round robin pointer is in a state that causes SOC0 to convert first.
- ADCINTSEL is configured to set an ADCINT flag upon end of conversion for SOC0 (whether this flag propagates through to the CPU to cause an interrupt is determined by the configurations in the PIE module).

Table 16-8 describes the parameters in the following timing diagrams. Table 16-9 lists the ADC timings.

Table 16-8. ADC Timing Parameter Descriptions

Parameter	Description
t_{SH}	The duration of the S+H window. At the end of this window, the value on the S+H capacitor becomes the voltage to be converted into a digital value. The duration is given by (ACQPS + 1) SYSCLK cycles. ACQPS can be configured individually for each SOC, so t_{SH} will not necessarily be the same for different SOCs. **Note:** The value on the S+H capacitor will be captured approximately 5ns before the end of the S+H window regardless of device clock settings.
t_{LAT}	The time from the end of the S+H window until the ADC results latch in the ADCRESULTx register. If the ADCRESULTx register is read before this time, the previous conversion results will be returned.
t_{EOC}	The time from the end of the S+H window until the S+H window for the next ADC conversion can begin. The subsequent sample can start before the conversion results are latched.
t_{INT}	The time from the end of the S+H window until an ADCINT flag is set (if configured). If the INTPULSEPOS bit in the ADCCTL1 register is set, t_{INT} will coincide with the conversion results being latched into the result register. If the INTPULSEPOS bit is 0, t_{INT} will coincide with the end of the S+H window. If t_{INT} triggers a read of the ADC result register (directly through DMA or indirectly by triggering an ISR that reads the result), care must be taken to ensure the read occurs after the results latch (otherwise, the previous results will be read) If the INTPULSEPOS bit is 0, and the OFFSET field in the ADCINTCYCLE register is not 0, then there will be a delay of OFFSET SYSCLK cycles before the ADCINT flag is set. This delay can be used to enter the ISR or trigger the DMA at exactly the time the sample is ready.

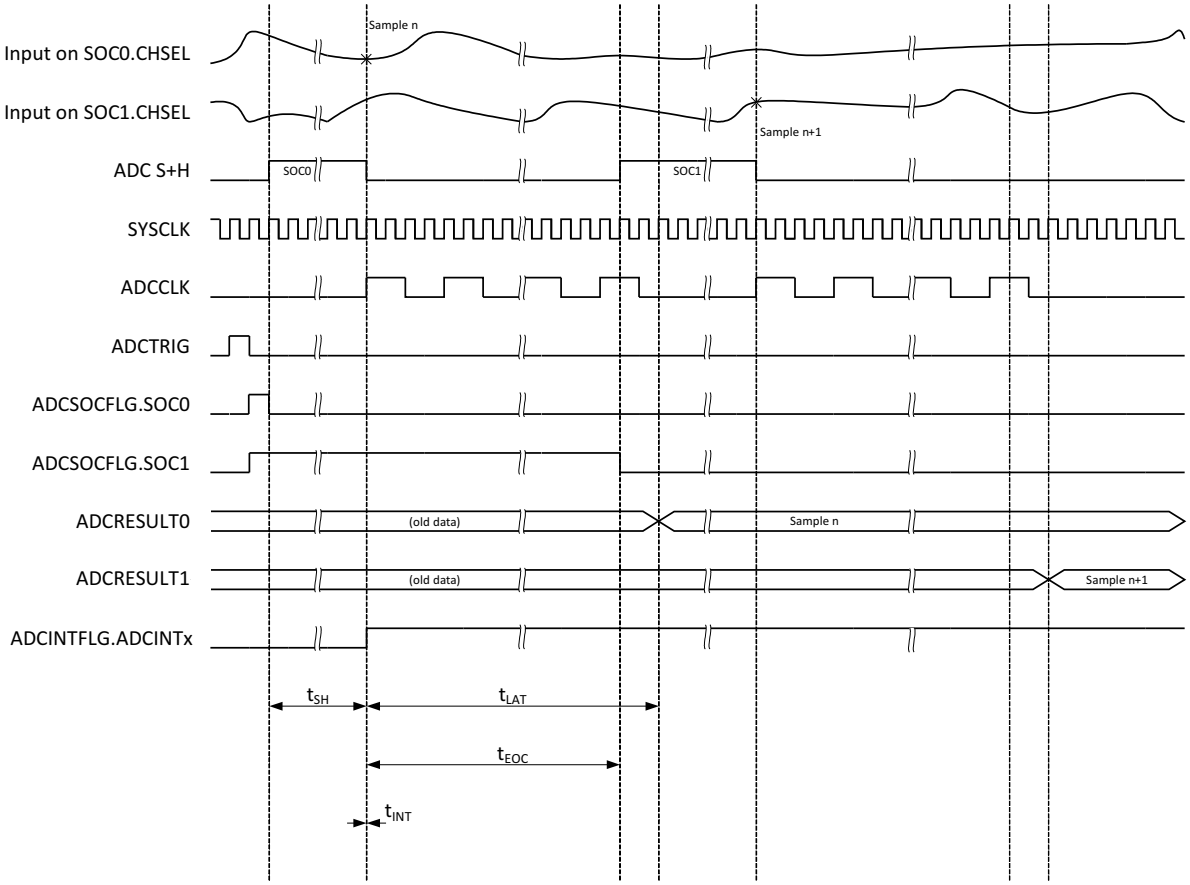


Figure 16-12. ADC Timings for 12-bit Mode in Early Interrupt Mode

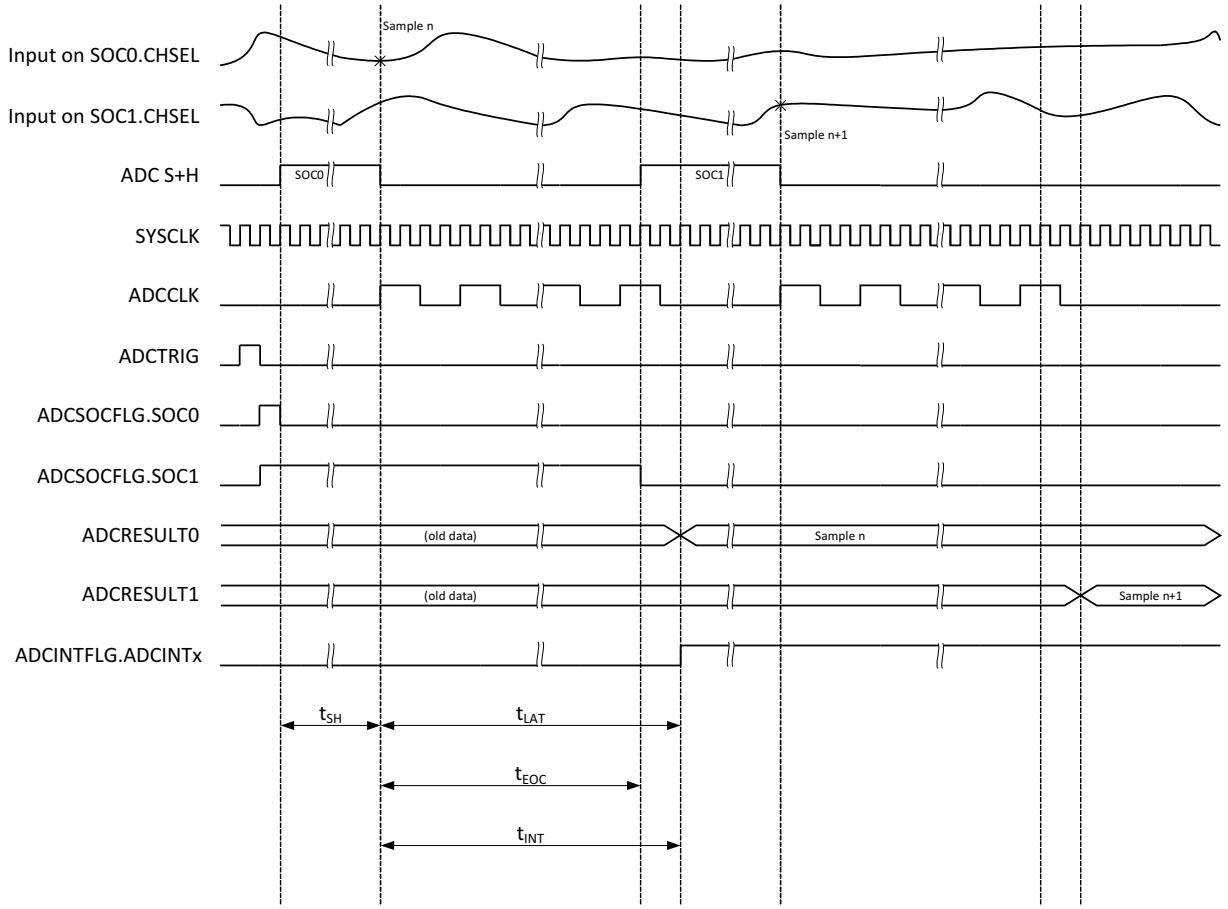


Figure 16-13. ADC Timings for 12-bit Mode in Late Interrupt Mode

Table 16-9. ADC Timings in 12-bit Mode

ADCCLK Prescale		SYSCLK Cycles			
ADCCTL2. PRESCALE	Prescale Ratio	t_{EOC}	t_{LAT}	t_{INT} (Early)	t_{INT} (Late)
0	1	11	13	1	11
2	2	21	23	1	21
4	3	31	34	1	31
6	4	41	44	1	41
8	5	51	55	1	51
10	6	61	65	1	61
12	7	71	76	1	71
14	8	81	86	1	81

16.13 Additional Information

The following sections contain additional practical information.

16.13.1 Ensuring Synchronous Operation

For best performance, all ADCs on the device should be operated synchronously. The device datasheet specifies the performance in both synchronous and asynchronous mode for those parameters which differ between the modes of operation.

To ensure synchronous operation, all ADCs on the device should operate in lockstep. This is accomplished by writing configurations to all ADCs that cause the sampling and conversion phases of all ADCs to be exactly aligned. The easiest way to accomplish this is to write identical values to the SOC configurations for each ADC for trigger select and ACQPS (S+H duration).

16.13.1.1 Basic Synchronous Operation

The following example configures two SOCs each on ADCA and ADCC with identical trigger select and ACQPS values. This will result in synchronous operation between ADCA and ADCC. For devices with more than two ADCs, the same principles can be used to synchronize all the ADCs.

```
Example: Basic Synchronous Operation
AdcaRegs.ADCSOC0CTL.bit.CHSEL = 4;      //SOC0 will convert ADCINA4
AdcaRegs.ADCSOC0CTL.bit.ACQPS = 19;     //SOC0 will use sample duration of 20 SYSCLK cycles
AdcaRegs.ADCSOC0CTL.bit.TRIGSEL = 10;   //SOC0 will begin conversion on ePWM3 SOCB
AdccRegs.ADCSOC0CTL.bit.CHSEL = 0;      //SOC0 will convert ADCINC0
AdccRegs.ADCSOC0CTL.bit.ACQPS = 19;     //SOC0 will use sample duration of 20 SYSCLK cycles
AdccRegs.ADCSOC0CTL.bit.TRIGSEL = 10;   //SOC0 will begin conversion on ePWM3 SOCB

AdcaRegs.ADCSOC1CTL.bit.CHSEL = 4;      //SOC1 will convert ADCINA4
AdcaRegs.ADCSOC1CTL.bit.ACQPS = 30;     //SOC1 will use sample duration of 31 SYSCLK cycles
AdcaRegs.ADCSOC1CTL.bit.TRIGSEL = 10;   //SOC1 will begin conversion on ePWM3 SOCB
AdccRegs.ADCSOC1CTL.bit.CHSEL = 1;      //SOC1 will convert ADCINC1
AdccRegs.ADCSOC1CTL.bit.ACQPS = 30;     //SOC1 will use sample duration of 31 SYSCLK cycles
AdccRegs.ADCSOC1CTL.bit.TRIGSEL = 10;   //SOC1 will begin conversion on ePWM3 SOCB
```

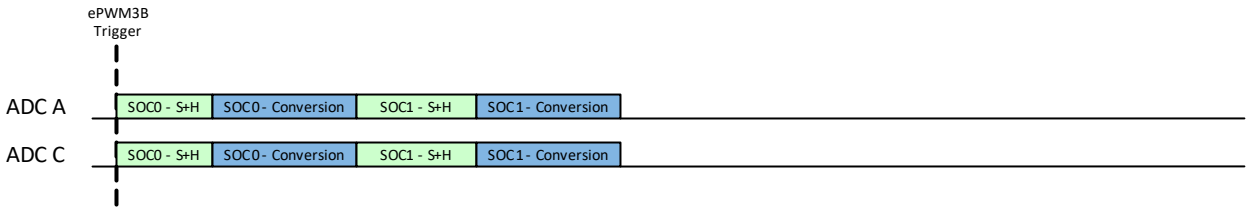


Figure 16-14. Example: Basic Synchronous Operation

Several things should be noted from Figure 16-14. First, while the ACQPS values must be the same for SOCs with the same number, different ACQPS values can be used for SOCs with different numbers. Because of this, synchronous operation does not require a single global S+H time, but instead only channels sampled simultaneously require identical S+H durations. Another important point from this example is that any channel select value can be used for any SOC. Finally, this example assumes round-robin operation. If high priority SOCs are to be used, the priority must be configured the same on all ADCs.

16.13.1.2 Synchronous Operation with Multiple Trigger Sources

As long as each set of SOCs has identical trigger select and ACQPS settings, multiple trigger sources can be used while still achieving synchronous operation.

The following example demonstrates synchronous operation between ADCA and ADCC while using three SOCs and two trigger sources. Figure 16-15 demonstrates that any combination of relative trigger timings still results in synchronous operation.

```
Example: Synchronous Operation With Multiple Trigger Sources
AdcaRegs.ADCSOC0CTL.bit.CHSEL = 4;      //SOC0 will convert ADCINA4
AdcaRegs.ADCSOC0CTL.bit.ACQPS = 19;     //SOC0 will use sample duration of 20 SYSCLK cycles
AdcaRegs.ADCSOC0CTL.bit.TRIGSEL = 10;   //SOC0 will begin conversion on ePWM3 SOCB
AdccRegs.ADCSOC0CTL.bit.CHSEL = 0;      //SOC0 will convert ADCINC0
AdccRegs.ADCSOC0CTL.bit.ACQPS = 19;     //SOC0 will use sample duration of 20 SYSCLK cycles
AdccRegs.ADCSOC0CTL.bit.TRIGSEL = 10;   //SOC0 will begin conversion on ePWM3 SOCB

AdcaRegs.ADCSOC1CTL.bit.CHSEL = 4;      //SOC1 will convert ADCINA4
AdcaRegs.ADCSOC1CTL.bit.ACQPS = 30;     //SOC1 will use sample duration of 31 SYSCLK cycles
AdcaRegs.ADCSOC1CTL.bit.TRIGSEL = 10;   //SOC1 will begin conversion on ePWM3 SOCB
AdccRegs.ADCSOC1CTL.bit.CHSEL = 1;      //SOC1 will convert ADCINC1
AdccRegs.ADCSOC1CTL.bit.ACQPS = 30;     //SOC1 will use sample duration of 31 SYSCLK cycles
AdccRegs.ADCSOC1CTL.bit.TRIGSEL = 10;   //SOC1 will begin conversion on ePWM3 SOCB

AdcaRegs.ADCSOC2CTL.bit.CHSEL = 0;      //SOC2 will convert ADCINA0
AdcaRegs.ADCSOC2CTL.bit.ACQPS = 19;     //SOC2 will use sample duration of 20 SYSCLK cycles
AdcaRegs.ADCSOC2CTL.bit.TRIGSEL = 2;    //SOC2 will begin conversion on CPU Timer1
AdccRegs.ADCSOC2CTL.bit.CHSEL = 2;      //SOC2 will convert ADCINC2
AdccRegs.ADCSOC2CTL.bit.ACQPS = 19;     //SOC2 will use sample duration of 20 SYSCLK cycles
AdccRegs.ADCSOC2CTL.bit.TRIGSEL = 2;    //SOC2 will begin conversion on CPU Timer1
```

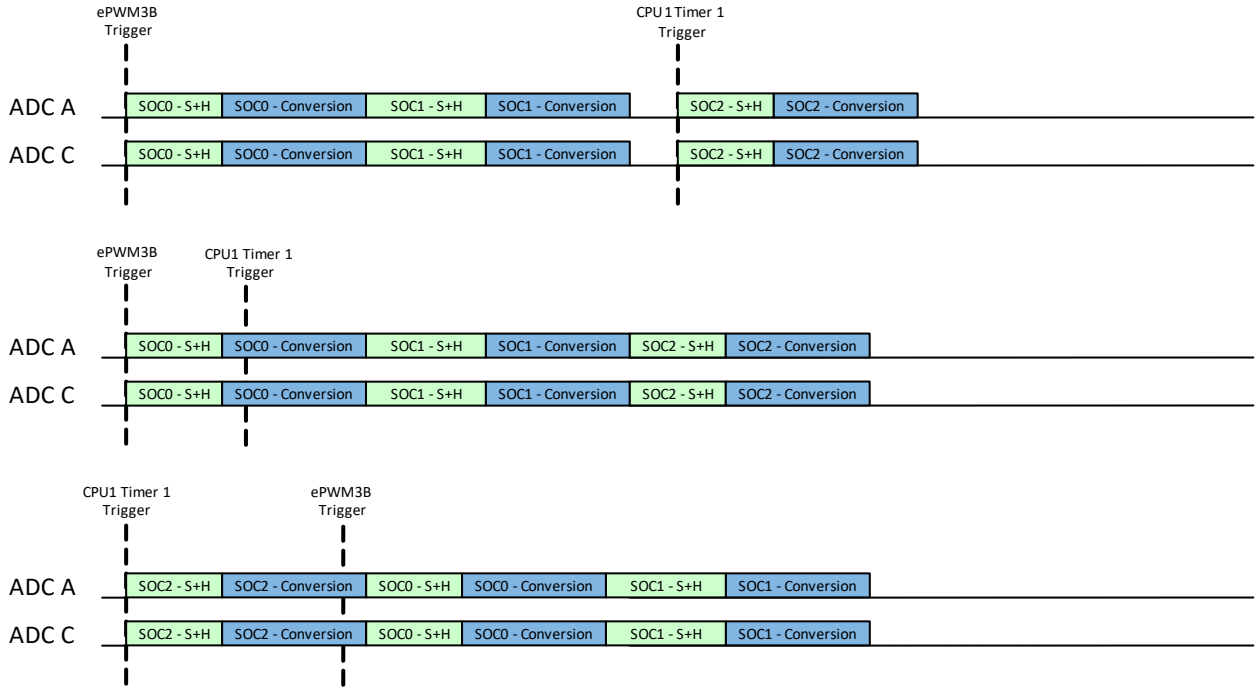


Figure 16-15. Example: Synchronous Operation with Multiple Trigger Sources

Note that any trigger source that can be selected in the TRIGSEL field can be used except for software triggering. There is no way to issue the software triggers for all ADCs simultaneously, so it will likely result in asynchronous operation. ADCINT1 or ADCINT2 can also be used as a trigger as long as the ADCINTSOCSEL1 and ADCINTSOCSEL2 registers are configured identically for all ADCs and software triggering is not used to start the chain of conversions.

16.13.1.3 Synchronous Operation with Uneven SOC Numbers

If only one trigger source is used, one ADC can use more SOCs than the other ADCs while still operating synchronously.

```
Example: Synchronous Operation With Uneven SOC Numbers
AdcaRegs.ADCSOC0CTL.bit.CHSEL = 4;      //SOC0 will convert ADCINA4
AdcaRegs.ADCSOC0CTL.bit.ACQPS = 19;     //SOC0 will use sample duration of 20 SYSCLK cycles
AdcaRegs.ADCSOC0CTL.bit.TRIGSEL = 10;   //SOC0 will begin conversion on ePWM3 SOCB
AdccRegs.ADCSOC0CTL.bit.CHSEL = 0;      //SOC0 will convert ADCINC0
AdccRegs.ADCSOC0CTL.bit.ACQPS = 19;     //SOC0 will use sample duration of 20 SYSCLK cycles
AdccRegs.ADCSOC0CTL.bit.TRIGSEL = 10;   //SOC0 will begin conversion on ePWM3 SOCB

AdcaRegs.ADCSOC1CTL.bit.CHSEL = 4;      //SOC1 will convert ADCINA4
AdcaRegs.ADCSOC1CTL.bit.ACQPS = 30;     //SOC1 will use sample duration of 31 SYSCLK cycles
AdcaRegs.ADCSOC1CTL.bit.TRIGSEL = 10;   //SOC1 will begin conversion on ePWM3 SOCB
AdccRegs.ADCSOC1CTL.bit.CHSEL = 1;      //SOC1 will convert ADCINC1
AdccRegs.ADCSOC1CTL.bit.ACQPS = 30;     //SOC1 will use sample duration of 31 SYSCLK cycles
AdccRegs.ADCSOC1CTL.bit.TRIGSEL = 10;   //SOC1 will begin conversion on ePWM3 SOCB

AdcaRegs.ADCSOC2CTL.bit.CHSEL = 0;      //SOC2 will convert ADCINA0
AdcaRegs.ADCSOC2CTL.bit.ACQPS = 19;     //SOC2 will use sample duration of 20 SYSCLK cycles
AdcaRegs.ADCSOC2CTL.bit.TRIGSEL = 10;   //SOC2 will begin conversion on ePWM3 SOCB
```

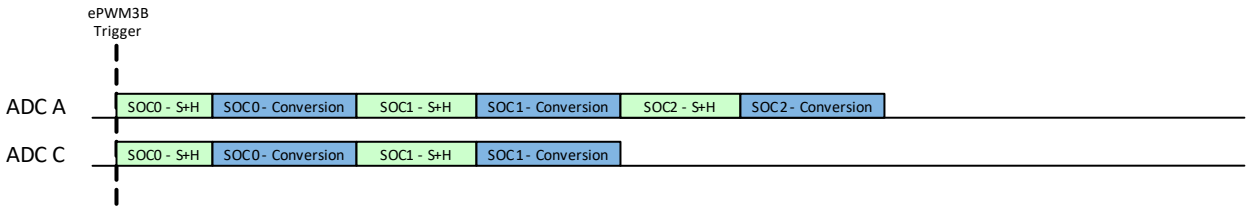


Figure 16-16. Example: Synchronous Operation with Uneven SOC Numbers

Note that if the trigger comes again before all SOCs have completed their conversions, ADCC will begin converting immediately on SOC0 while ADCA will not start converting SOC0 again until SOC2 is complete. This will result in asynchronous operation, so care must be taken to not overflow the trigger.

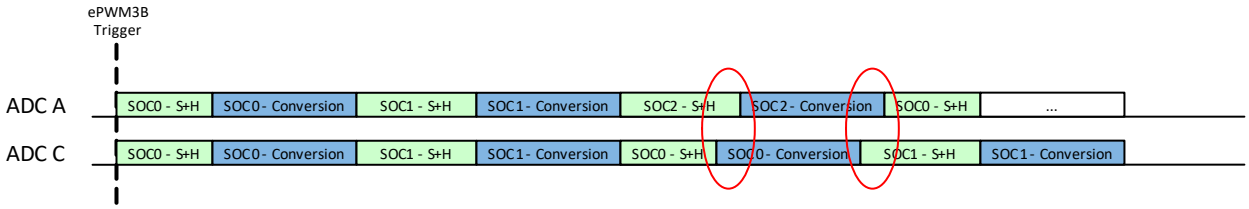


Figure 16-17. Example: Asynchronous Operation with Uneven SOC Numbers – Trigger Overflow

16.13.1.4 Non-overlapping Conversions

If conversion timings can be guaranteed to not overlap by the user, then it is not necessary to configure all SOCs identically on all ADCs to achieve performance equivalent to synchronous operation. For example, if the two ADC triggers in a system come from two ePWM sources which are always 180 degrees out-of-phase, then SOC0 could be used for both ADCA and ADCC with different trigger sources and different ACQPS values.

```
Example: Operation with Non-overlapping Conversions
//ePWM3 SOCA and SOCB are 180 degrees out of phase
AdcaRegs.ADCSOC0CTL.bit.CHSEL = 4;      //SOC0 will convert ADCINA4
AdcaRegs.ADCSOC0CTL.bit.ACQPS = 19;     //SOC0 will use sample duration of 20 SYSCLK cycles
AdcaRegs.ADCSOC0CTL.bit.TRIGSEL = 10;   //SOC0 will begin conversion on ePWM3 SOCB
AdccRegs.ADCSOC0CTL.bit.CHSEL = 0;      //SOC0 will convert ADCINC0
AdccRegs.ADCSOC0CTL.bit.ACQPS = 19;     //SOC0 will use sample duration of 20 SYSCLK cycles
AdccRegs.ADCSOC0CTL.bit.TRIGSEL = 9;    //SOC0 will begin conversion on ePWM3 SOCA
```

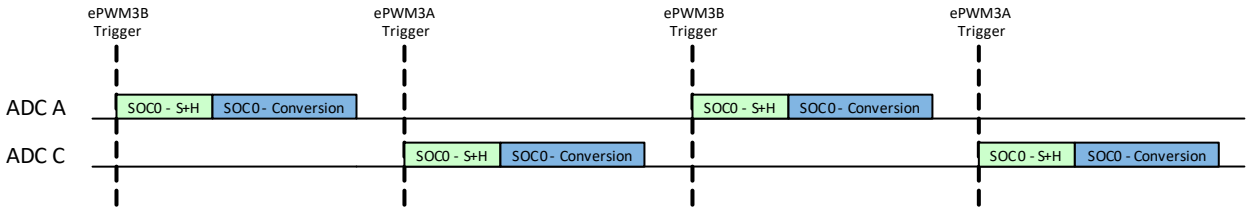


Figure 16-18. Example: Synchronous Equivalent Operation with Non-Overlapping Conversions

16.13.2 Choosing an Acquisition Window Duration

For correct operation, the input signal to the ADC must be allowed adequate time to charge the sample and hold capacitor, Ch. Typically, the S+H duration is chosen such that the sampling capacitor will be charged to within ½ LSB or ¼ LSB of the final value, depending on the tolerable settling error.

The best methodology to determine the required settling time is to simulate the ADC and ADC driving circuits to ensure adequate settling performance. See TI application reports ADC Input Circuit Evaluation for C2000 MCUs and Charge-Sharing Driving Circuits for C2000 ADCs for additional guidance on ADC signal conditioning circuit design and evaluation.

An approximation of the required settling time can also be determined using an RC settling model. The time constant for the model is given by the equation:

$$\tau = (Rs + Ron) \cdot Ch + Rs \cdot (Cs + Cp) \tag{2}$$

And the number of time constants needed is given by the equation:

$$k = ln\left(\frac{2^n}{settling\ error}\right) - ln\left(\frac{C_S + C_P}{CH}\right) \tag{3}$$

So the total S+H time should be set to at least:

$$t = k \cdot \tau \tag{4}$$

Where the following parameters are provided by the ADC input model in the device data manual:

- n = ADC resolution (in bits)
- R_{ON} = ADC sampling switch resistance (in Ohms)
- C_H = ADC sampling capacitor (in pF)

- C_p = ADC channel parasitic input capacitance (in pF)

And the following parameters are dependent on the application design:

- settling error = tolerable settling error (in LSBs)
- R_s = ADC driving circuit source impedance (in Ohms)
- C_S = capacitance on ADC input pin (in pF)

For example, assuming the following parameters:

- n = 12-bits
- R_{ON} = 500Ω
- C_H = 12.5pF
- C_p = 12.7pF
- settling error = ¼ LSB
- R_s = 180Ω
- C_s = 150pF

The time constant would be calculated as:

$$\tau = (180\Omega + 500\Omega) \cdot 12.5pF + 180\Omega \cdot (150pF + 12.7pF) = 8.5ns + \ 29.3ns = 37.8ns \quad (5)$$

And the number of required time constants would be:

$$k = ln\left(\frac{2^{12}}{0.25}\right) - ln\left(\frac{150pF + 12.7pF}{12.5pF}\right) = 9.70 - 2.57 = 7.13 \quad (6)$$

So the S+H time should be set to at least:

37.8ns · 7.13 = 270ns

If SYSCLK = 100 MHz, then each SYSCLK is 10 ns. S+H duration should be 270 ns/10 ns = 27.0 SYSCLKs, so ACQPS for this input should be set to at least CEILING(27.0) – 1 = 26.

While this gives a rough estimate of the required acquisition window, a better method would be to setup a circuit with the ADC input model, a model of the source impedance/capacitance, and any board parasitics in SPICE (or similar software) and simulate to verify that the sampling capacitor settles to the desired accuracy.

Note

The device datasheet will specify a minimum ADC S+H window duration. Do not use an ACQPS value that gives a duration less than this specification.

16.13.3 Achieving Simultaneous Sampling

While each ADC does not have dual S+H circuits, it is easy to achieve simultaneous sampling. This is accomplished by setting the SOC triggers on two or more ADC modules to use the same trigger source. The example below demonstrates simultaneous sampling based on an ePWM3 event. ADCINA3 and ADCINC5 are sampled. An acquisition window of 20 SYSCLK cycles is used, but different durations are possible.

```
AdcaRegs.ADCSOC0CTL.bit.CHSEL = 3;        //SOC0 will convert ADCINA3
AdcaRegs.ADCSOC0CTL.bit.ACQPS = 19;       //SOC0 will use sample duration of 20 SYSCLK cycles
AdcaRegs.ADCSOC0CTL.bit.TRIGSEL = 10;     //SOC0 will begin conversion on ePWM3 SOCB
AdccRegs.ADCSOC0CTL.bit.CHSEL = 5;        //SOC0 will convert ADCINC5
AdccRegs.ADCSOC0CTL.bit.ACQPS = 19;       //SOC0 will use sample duration of 20 SYSCLK cycles
AdccRegs.ADCSOC0CTL.bit.TRIGSEL = 10;     //SOC0 will begin conversion on ePWM3 SOCB
```

When the ePWM3 trigger is received, all ADCs will begin converting in parallel immediately. All results will be in the ADCRESULT0 register for each ADC. Note that this assumes that all ADCs are idle when the trigger is received. If one or more ADCs is busy, the samples will not happen at exactly the same time.

16.13.4 Result Register Mapping

The ADC results and the ADC PPB results are duplicated for each memory bus controller in the system. Bus controllers include all C28x CPUs, C28x DMAs, and CLAs present on the specific part family and part number. For each bus controller, no access configuration is needed to allow read access to the result registers and no contention occurs in cases where multiple bus controllers try to read the ADC results simultaneously.

16.13.5 Internal Temperature Sensor

The internal temperature sensor measures the junction temperature of the device. The output of the sensor can be sampled with the ADC through an internal connection. This can be enabled on channel ADCIN12 on ADCC and on CMPSS2_HP5 input by setting the ENABLE bit in the TSNSCTL register.

To convert the temperature sensor reading into a temperature, pass the temperature sensor reading to the ADC_getTemperatureC() function in F28003x ADC driverlib file.

16.13.6 Designing an External Reference Circuit

Figure 16-19 shows the basic organization of the external voltage reference generation circuitry. The reference voltage should then be buffered by a precision op-amp with good bandwidth and low output impedance before being driven into the reference pin. A capacitor between the high and low reference pins should be placed on the PCB as close to the pins as practical to help absorb high frequency currents. A series resistor (typically <1Ω) in series with this capacitor may be necessary to ensure op-amp stability.

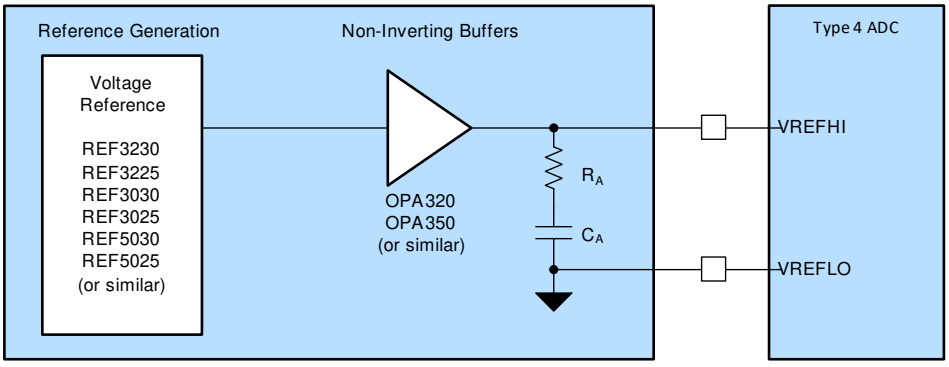


Figure 16-19. ADC Reference System

16.14 Software

16.14.1 ADC Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/adc

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

16.14.1.1 ADC ePWM Triggering Multiple SOC

FILE: adc_ex10_multiple_soc_epwm.c

This example sets up ePWM1 to periodically trigger a set of conversions on ADCA and ADCC. This example demonstrates multiple ADCs working together to process of a batch of conversions using the available parallelism across multiple ADCs.

ADCA Interrupt ISRs are used to read results of both ADCA and ADCC.

External Connections

- A0, A1, A2 and C2, C3, C4 pins should be connected to signals to be converted.

Watch Variables

- *adcAResult0* - Digital representation of the voltage on pin A0
- *adcAResult1* - Digital representation of the voltage on pin A1
- *adcAResult2* - Digital representation of the voltage on pin A2
- *adcCResult0* - Digital representation of the voltage on pin C2
- *adcCResult1* - Digital representation of the voltage on pin C3
- *adcCResult2* - Digital representation of the voltage on pin C4

16.14.1.2 ADC Burst Mode

FILE: adc_ex11_burst_mode_epwm.c

This example sets up ePWM1 to periodically trigger ADCA using burst mode. This allows for different channels to be sampled with each burst.

Each burst triggers 3 conversions. A0 and A1 are part of every burst while the third conversion rotates between A2, A3, and A4. This allows high importance signals to be sampled at high speed while lower priority signals can be sampled at a lower rate.

ADCA Interrupt ISRs are used to read results for ADCA.

External Connections

- A0, A1, A2, A3, A4

Watch Variables

- *adcAResult0* - Digital representation of the voltage on pin A0
- *adcAResult1* - Digital representation of the voltage on pin A1
- *adcAResult2* - Digital representation of the voltage on pin A2
- *adcAResult3* - Digital representation of the voltage on pin A3
- *adcAResult4* - Digital representation of the voltage on pin A4

16.14.1.3 ADC Burst Mode Oversampling

FILE: adc_ex12_burst_mode_oversampling.c

This example sets up ePWM1 to periodically trigger SOC0 and SOC1 on ADCA (to sample A0 and A1). Additionally, the ADC burst mode is also triggered using ePWM1. The burst SOCs are used to accumulate multiple conversions to oversample A2 over multiple ePWM periods.

External Connections

- A0, A1, A2

Watch Variables

- *adcAResult0* - Digital representation of the voltage on pin A0
- *adcAResult1* - Digital representation of the voltage on pin A1
- *adcAResult2* - Digital representation of the voltage on pin A2

16.14.1.4 ADC SOC Oversampling

FILE: adc_ex13_soc_oversampling.c

This example sets up ePWM1 to periodically trigger a set of conversions on ADCA including multiple SOCs that all convert A2 to achieve oversampling on A2.

ADCA Interrupt ISRs are used to read results of ADCA.

External Connections

- A0, A1, A2 should be connected to signals to be converted.

Watch Variables

- *adcAResult0* - Digital representation of the voltage on pin A0
- *adcAResult1* - Digital representation of the voltage on pin A1
- *adcAResult2* - Digital representation of the voltage on pin A2

16.14.1.5 ADC PPB PWM trip (adc_ppb_pwm_trip)

FILE: adc_ex14_ppb_pwm_trip.c

This example demonstrates EPWM tripping through ADC limit detection PPB block. ADCAINT1 is configured to periodically trigger the ADCA channel 2 post initial software forced trigger. The limit detection post-processing block(PPB) is configured and if the ADC results are outside of the defined range, the post-processing block will generate an ADCxEVTy event. This event is configured as EPWM trip source through configuring EPWM XBAR and corresponding EPWM's trip zone and digital compare sub-modules. The example showcases:

- one-shot
- cycle-by-cycle
- and direct tripping of PWMs through ADCAEVT1 source via Digital compare submodule.

The default limits are 0LSBs and 3600LSBs. With VREFHI set to 3.3V, the PPB will generate a trip event if the input voltage goes above about 2.9V.

External Connections

- A2 should be connected to a signal to convert
- Observe the following signals on an oscilloscope
  - ePWM1(GPIO0 - GPIO1)
  - ePWM2(GPIO2 - GPIO3)
  - ePWM3(GPIO4 - GPIO5)

Watch Variables

- *adcA2Results* - Digital representation of the voltage on pin A2

16.14.1.6 ADC Software Triggering

FILE: adc_ex1_soc_software.c

This example converts some voltages on ADCA and ADCC based on a software trigger.

The ADCC will not convert until ADCA is complete, so the ADCs will not run asynchronously. However, this is much less efficient than allowing the ADCs to convert synchronously in parallel (for example, by using an ePWM trigger).

External Connections

- A0, A1, C2, and C3 should be connected to signals to convert

Watch Variables

- *myADC0Result0* - Digital representation of the voltage on pin A0
- *myADC0Result1* - Digital representation of the voltage on pin A1
- *myADC1Result0* - Digital representation of the voltage on pin C2
- *myADC1Result1* - Digital representation of the voltage on pin C3

16.14.1.7 ADC ePWM Triggering

FILE: adc_ex2_soc_epwm.c

This example sets up ePWM1 to periodically trigger a conversion on ADCA.

External Connections

- A0 should be connected to a signal to convert

Watch Variables

- *myADC0Results* - A sequence of analog-to-digital conversion samples from pin A0. The time between samples is determined based on the period of the ePWM timer.

16.14.1.8 ADC Temperature Sensor Conversion

FILE: adc_ex3_temp_sensor.c

This example sets up the ePWM to periodically trigger the ADC. The ADC converts the internal connection to the temperature sensor, which is then interpreted as a temperature by calling the ADC_getTemperatureC() function.

Watch Variables

- *sensorSample* - The raw reading from the temperature sensor
- *sensorTemp* - The interpretation of the sensor sample as a temperature in degrees Celsius.

16.14.1.9 ADC Synchronous SOC Software Force (adc_soc_software_sync)

FILE: adc_ex4_soc_software_sync.c

This example converts some voltages on ADCA and ADCC using input 5 of the input X-BAR as a software force. Input 5 is triggered by toggling GPIO0, but any spare GPIO could be used. This method will ensure that both ADCs start converting at exactly the same time.

External Connections

- A2, A3, C2, C3 pins should be connected to signals to convert

Watch Variables

- *myADC0Result0* : a digital representation of the voltage on pin A2
- *myADC0Result1* : a digital representation of the voltage on pin A3
- *myADC1Result0* : a digital representation of the voltage on pin C2
- *myADC1Result1* : a digital representation of the voltage on pin C3

16.14.1.10 ADC Continuous Triggering (adc_soc_continuous)

FILE: adc_ex5_soc_continuous.c

This example sets up the ADC to convert continuously, achieving maximum sampling rate.
External Connections

- A0 pin should be connected to signal to convert

Watch Variables

- *adcAResults* - A sequence of analog-to-digital conversion samples from pin A0. The time between samples is the minimum possible based on the ADC speed.

16.14.1.11 ADC Continuous Conversions Read by DMA (adc_soc_continuous_dma)

FILE: adc_ex6_soc_continuous_dma.c

This example sets up two ADC channels to convert simultaneously. The results will be transferred by the DMA into a buffer in RAM.

External Connections

- A3, C3 pins should be connected to signals to convert

Watch Variables

- *myADC0DataBuffer*: a digital representation of the voltage on pin A3
- *myADC1DataBuffer*: a digital representation of the voltage on pin C3

16.14.1.12 ADC PPB Offset (adc_ppb_offset)

FILE: adc_ex7_ppb_offset.c

This example software triggers the ADC. Some SOCs have automatic offset adjustment applied by the post-processing block. After the program runs, the memory will contain ADC & post-processing block(PPB) results.

External Connections

- A2, C2 pins should be connected to signals to convert

Watch Variables

- *myADC0Result* : a digital representation of the voltage on pin A2
- *myADC0PPBResult* : a digital representation of the voltage on pin A2, minus 100 LSBs of automatically added offset
- *myADC1Result* : a digital representation of the voltage on pin C2
- *myADC1PPBResult* : a digital representation of the voltage on pin C2 plus 100 LSBs of automatically added offset

16.14.1.13 ADC PPB Limits (adc_ppb_limits)

FILE: adc_ex8_ppb_limits.c

This example sets up the ePWM to periodically trigger the ADC. If the results are outside of the defined range, the post-processing block will generate an interrupt.

The default limits are 1000LSBs and 3000LSBs. With VREFHI set to 3.3V, the PPB will generate an interrupt if the input voltage goes above about 2.4V or below about 0.8V.

External Connections

- A0 should be connected to a signal to convert

Watch Variables

- None

16.14.1.14 ADC PPB Delay Capture (adc_ppb_delay)

FILE: adc_ex9_ppb_delay.c

This example demonstrates delay capture using the post-processing block.

Two asynchronous ADC triggers are setup:

- ePWM1, with period 2048, triggering SOC0 to convert on pin A0
- ePWM2, with period 9999, triggering SOC1 to convert on pin A2
  Each conversion generates an ISR at the end of the conversion. In the ISR for SOC0, a conversion counter is incremented and the PPB is checked to determine if the sample was delayed.
  After the program runs, the memory will contain:

conversion : the sequence of conversions using SOC0 that were delayed

- *delay* : the corresponding delay of each of the delayed conversions

16.15 ADC Registers

This section describes the Analog-to-Digital Converter Registers.

16.15.1 ADC Base Address Table

Table 16-10. ADC Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
AdcaResultRegs	ADC_RESULT_REGS	ADCARESULT_BASE	0x0000_0B00	YES	YES	YES	YES	-
AdcbResultRegs	ADC_RESULT_REGS	ADCBRESULT_BASE	0x0000_0B20	YES	YES	YES	YES	-
AdccResultRegs	ADC_RESULT_REGS	ADCCRESULT_BASE	0x0000_0B40	YES	YES	YES	YES	-
AdcaRegs	ADC_REGS	ADCA_BASE	0x0000_7400	YES	-	-	YES	YES
AdcbRegs	ADC_REGS	ADCB_BASE	0x0000_7480	YES	-	-	YES	YES
AdccRegs	ADC_REGS	ADCC_BASE	0x0000_7500	YES	-	-	YES	YES

16.15.2 ADC_RESULT_REGS Registers

Table 16-11 lists the memory-mapped registers for the ADC_RESULT_REGS registers. All register offset addresses not listed in Table 16-11 should be considered as reserved locations and the register contents should not be modified.

Table 16-11. ADC_RESULT_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	ADCRESULT0	ADC Result 0 Register		Go
1h	ADCRESULT1	ADC Result 1 Register		Go
2h	ADCRESULT2	ADC Result 2 Register		Go
3h	ADCRESULT3	ADC Result 3 Register		Go
4h	ADCRESULT4	ADC Result 4 Register		Go
5h	ADCRESULT5	ADC Result 5 Register		Go
6h	ADCRESULT6	ADC Result 6 Register		Go
7h	ADCRESULT7	ADC Result 7 Register		Go
8h	ADCRESULT8	ADC Result 8 Register		Go
9h	ADCRESULT9	ADC Result 9 Register		Go
Ah	ADCRESULT10	ADC Result 10 Register		Go
Bh	ADCRESULT11	ADC Result 11 Register		Go
Ch	ADCRESULT12	ADC Result 12 Register		Go
Dh	ADCRESULT13	ADC Result 13 Register		Go
Eh	ADCRESULT14	ADC Result 14 Register		Go
Fh	ADCRESULT15	ADC Result 15 Register		Go
10h	ADCPPB1RESULT	ADC Post Processing Block 1 Result Register		Go
12h	ADCPPB2RESULT	ADC Post Processing Block 2 Result Register		Go
14h	ADCPPB3RESULT	ADC Post Processing Block 3 Result Register		Go
16h	ADCPPB4RESULT	ADC Post Processing Block 4 Result Register		Go

Complex bit access types are encoded to fit into small table cells. Table 16-12 shows the codes that are used for access types in this section.

Table 16-12. ADC_RESULT_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.

Table 16-12. ADC_RESULT_REGS Access Type Codes (continued)

Access Type	Code	Description
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

16.15.2.1 ADCRESULT0 Register (Offset = 0h) [Reset = 0h]

ADCRESULT0 is shown in Figure 16-20 and described in Table 16-13.

Return to the Summary Table.

ADC Result 0 Register

Figure 16-20. ADCRESULT0 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 16-13. ADCRESULT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 0 16-bit ADC result. After the ADC completes a conversion of SOC0, the digital result is placed in this bit field. Reset type: SYSRSn

16.15.2.2 ADCRESULT1 Register (Offset = 1h) [Reset = 0h]

ADCRESULT1 is shown in Figure 16-21 and described in Table 16-14.

Return to the Summary Table.

ADC Result 1 Register

Figure 16-21. ADCRESULT1 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 16-14. ADCRESULT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 1 16-bit ADC result. After the ADC completes a conversion of SOC1, the digital result is placed in this bit field. Reset type: SYSRSn

16.15.2.3 ADCRESULT2 Register (Offset = 2h) [Reset = 0h]

ADCRESULT2 is shown in Figure 16-22 and described in Table 16-15.

Return to the Summary Table.

ADC Result 2 Register

Figure 16-22. ADCRESULT2 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 16-15. ADCRESULT2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 2 16-bit ADC result. After the ADC completes a conversion of SOC2, the digital result is placed in this bit field. Reset type: SYSRSn

16.15.2.4 ADCRESULT3 Register (Offset = 3h) [Reset = 0h]

ADCRESULT3 is shown in Figure 16-23 and described in Table 16-16.

Return to the Summary Table.

ADC Result 3 Register

Figure 16-23. ADCRESULT3 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 16-16. ADCRESULT3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 3 16-bit ADC result. After the ADC completes a conversion of SOC3, the digital result is placed in this bit field. Reset type: SYSRSn

16.15.2.5 ADCRESULT4 Register (Offset = 4h) [Reset = 0h]

ADCRESULT4 is shown in Figure 16-24 and described in Table 16-17.

Return to the Summary Table.

ADC Result 4 Register

Figure 16-24. ADCRESULT4 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 16-17. ADCRESULT4 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 4 16-bit ADC result. After the ADC completes a conversion of SOC4, the digital result is placed in this bit field. Reset type: SYSRSn

16.15.2.6 ADCRESULT5 Register (Offset = 5h) [Reset = 0h]

ADCRESULT5 is shown in Figure 16-25 and described in Table 16-18.

Return to the Summary Table.

ADC Result 5 Register

Figure 16-25. ADCRESULT5 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 16-18. ADCRESULT5 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 5 16-bit ADC result. After the ADC completes a conversion of SOC5, the digital result is placed in this bit field. Reset type: SYSRSn

16.15.2.7 ADCRESULT6 Register (Offset = 6h) [Reset = 0h]

ADCRESULT6 is shown in Figure 16-26 and described in Table 16-19.

Return to the Summary Table.

ADC Result 6 Register

Figure 16-26. ADCRESULT6 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 16-19. ADCRESULT6 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 6 16-bit ADC result. After the ADC completes a conversion of SOC6, the digital result is placed in this bit field. Reset type: SYSRSn

16.15.2.8 ADCRESULT7 Register (Offset = 7h) [Reset = 0h]

ADCRESULT7 is shown in Figure 16-27 and described in Table 16-20.

Return to the Summary Table.

ADC Result 7 Register

Figure 16-27. ADCRESULT7 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 16-20. ADCRESULT7 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 7 16-bit ADC result. After the ADC completes a conversion of SOC7, the digital result is placed in this bit field. Reset type: SYSRSn

16.15.2.9 ADCRESULT8 Register (Offset = 8h) [Reset = 0h]

ADCRESULT8 is shown in Figure 16-28 and described in Table 16-21.

Return to the Summary Table.

ADC Result 8 Register

Figure 16-28. ADCRESULT8 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 16-21. ADCRESULT8 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 8 16-bit ADC result. After the ADC completes a conversion of SOC8, the digital result is placed in this bit field. Reset type: SYSRSn

16.15.2.10 ADCRESULT9 Register (Offset = 9h) [Reset = 0h]

ADCRESULT9 is shown in Figure 16-29 and described in Table 16-22.

Return to the Summary Table.

ADC Result 9 Register

Figure 16-29. ADCRESULT9 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 16-22. ADCRESULT9 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 9 16-bit ADC result. After the ADC completes a conversion of SOC9, the digital result is placed in this bit field. Reset type: SYSRSn

16.15.2.11 ADCRESULT10 Register (Offset = Ah) [Reset = 0h]

ADCRESULT10 is shown in Figure 16-30 and described in Table 16-23.

Return to the Summary Table.

ADC Result 10 Register

Figure 16-30. ADCRESULT10 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 16-23. ADCRESULT10 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 10 16-bit ADC result. After the ADC completes a conversion of SOC10, the digital result is placed in this bit field. Reset type: SYSRSn

16.15.2.12 ADCRESULT11 Register (Offset = Bh) [Reset = 0h]

ADCRESULT11 is shown in Figure 16-31 and described in Table 16-24.

Return to the Summary Table.

ADC Result 11 Register

Figure 16-31. ADCRESULT11 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 16-24. ADCRESULT11 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 11 16-bit ADC result. After the ADC completes a conversion of SOC11, the digital result is placed in this bit field. Reset type: SYSRSn

16.15.2.13 ADCRESULT12 Register (Offset = Ch) [Reset = 0h]

ADCRESULT12 is shown in Figure 16-32 and described in Table 16-25.

Return to the Summary Table.

ADC Result 12 Register

Figure 16-32. ADCRESULT12 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 16-25. ADCRESULT12 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 12 16-bit ADC result. After the ADC completes a conversion of SOC12, the digital result is placed in this bit field. Reset type: SYSRSn

16.15.2.14 ADCRESULT13 Register (Offset = Dh) [Reset = 0h]

ADCRESULT13 is shown in Figure 16-33 and described in Table 16-26.

Return to the Summary Table.

ADC Result 13 Register

Figure 16-33. ADCRESULT13 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 16-26. ADCRESULT13 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 13 16-bit ADC result. After the ADC completes a conversion of SOC13, the digital result is placed in this bit field. Reset type: SYSRSn

16.15.2.15 ADCRESULT14 Register (Offset = Eh) [Reset = 0h]

ADCRESULT14 is shown in Figure 16-34 and described in Table 16-27.

Return to the Summary Table.

ADC Result 14 Register

Figure 16-34. ADCRESULT14 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 16-27. ADCRESULT14 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 14 16-bit ADC result. After the ADC completes a conversion of SOC14, the digital result is placed in this bit field. Reset type: SYSRSn

16.15.2.16 ADCRESULT15 Register (Offset = Fh) [Reset = 0h]

ADCRESULT15 is shown in Figure 16-35 and described in Table 16-28.

Return to the Summary Table.

ADC Result 15 Register

Figure 16-35. ADCRESULT15 Register

15	14	13	12	11	10	9	8
RESULT							
R-0h							
7	6	5	4	3	2	1	0
RESULT							
R-0h							

Table 16-28. ADCRESULT15 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RESULT	R	0h	ADC Result 15 16-bit ADC result. After the ADC completes a conversion of SOC15, the digital result is placed in this bit field. Reset type: SYSRSn

16.15.2.17 ADCPPB1RESULT Register (Offset = 10h) [Reset = 0h]

ADCPPB1RESULT is shown in Figure 16-36 and described in Table 16-29.

Return to the Summary Table.

ADC Post Processing Block 1 Result Register

Figure 16-36. ADCPPB1RESULT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIGN																PPBRESULT															
R-0h																R-0h															

Table 16-29. ADCPPB1RESULT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	SIGN	R	0h	Sign Extended Bits. These bits reflect the same value as bit 16. NOTE: If the conversion associated with this Post Processing Block is a 12-bit conversion, the SIGN bits extend down to bit 12, and all reflect the same value as bit 12. Reset type: SYSRSn
15-0	PPBRESULT	R	0h	ADC Post Processing Block Result 1 The result of the offset/reference subtraction post conversion processing is stored in this register. This result is available 1 SYSCLK cycle after the associated ADCRESULT is available. If ADCINTFLG is polled to determine when to read the PPBRESULT, it may be necessary to add a NOP instruction to ensure that the updated post conversion processing result has posted to the register. NOTE: If the conversion associated with this Post Processing Block is a 12-bit conversion, the PPBRESULT bits are limited to bits 12:0. Reset type: SYSRSn

16.15.2.18 ADCPPB2RESULT Register (Offset = 12h) [Reset = 0h]

ADCPPB2RESULT is shown in Figure 16-37 and described in Table 16-30.

Return to the Summary Table.

ADC Post Processing Block 2 Result Register

Figure 16-37. ADCPPB2RESULT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIGN																PPBRESULT															
R-0h																R-0h															

Table 16-30. ADCPPB2RESULT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	SIGN	R	0h	Sign Extended Bits. These bits reflect the same value as bit 16. NOTE: If the conversion associated with this Post Processing Block is a 12-bit conversion, the SIGN bits extend down to bit 12, and all reflect the same value as bit 12. Reset type: SYSRSn
15-0	PPBRESULT	R	0h	ADC Post Processing Block Result 2 The result of the offset/reference subtraction post conversion processing is stored in this register. This result is available 1 SYSCLK cycle after the associated ADCRESULT is available. If ADCINTFLG is polled to determine when to read the PPBRESULT, it may be necessary to add a NOP instruction to ensure that the updated post conversion processing result has posted to the register. NOTE: If the conversion associated with this Post Processing Block is a 12-bit conversion, the PPBRESULT bits are limited to bits 12:0. Reset type: SYSRSn

16.15.2.19 ADCPPB3RESULT Register (Offset = 14h) [Reset = 0h]

ADCPPB3RESULT is shown in Figure 16-38 and described in Table 16-31.

Return to the Summary Table.

ADC Post Processing Block 3 Result Register

Figure 16-38. ADCPPB3RESULT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIGN																PPBRESULT															
R-0h																R-0h															

Table 16-31. ADCPPB3RESULT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	SIGN	R	0h	Sign Extended Bits. These bits reflect the same value as bit 16. NOTE: If the conversion associated with this Post Processing Block is a 12-bit conversion, the SIGN bits extend down to bit 12, and all reflect the same value as bit 12. Reset type: SYSRSn
15-0	PPBRESULT	R	0h	ADC Post Processing Block Result 3 The result of the offset/reference subtraction post conversion processing is stored in this register. This result is available 1 SYSCLK cycle after the associated ADCRESULT is available. If ADCINTFLG is polled to determine when to read the PPBRESULT, it may be necessary to add a NOP instruction to ensure that the updated post conversion processing result has posted to the register. NOTE: If the conversion associated with this Post Processing Block is a 12-bit conversion, the PPBRESULT bits are limited to bits 12:0. Reset type: SYSRSn

16.15.2.20 ADCPPB4RESULT Register (Offset = 16h) [Reset = 0h]

ADCPPB4RESULT is shown in Figure 16-39 and described in Table 16-32.

Return to the Summary Table.

ADC Post Processing Block 4 Result Register

Figure 16-39. ADCPPB4RESULT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIGN																PPBRESULT															
R-0h																R-0h															

Table 16-32. ADCPPB4RESULT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	SIGN	R	0h	Sign Extended Bits. These bits reflect the same value as bit 16. NOTE: If the conversion associated with this Post Processing Block is a 12-bit conversion, the SIGN bits extend down to bit 12, and all reflect the same value as bit 12. Reset type: SYSRSn
15-0	PPBRESULT	R	0h	ADC Post Processing Block Result 4 The result of the offset/reference subtraction post conversion processing is stored in this register. This result is available 1 SYSCLK cycle after the associated ADCRESULT is available. If ADCINTFLG is polled to determine when to read the PPBRESULT, it may be necessary to add a NOP instruction to ensure that the updated post conversion processing result has posted to the register. NOTE: If the conversion associated with this Post Processing Block is a 12-bit conversion, the PPBRESULT bits are limited to bits 12:0. Reset type: SYSRSn

16.15.3 ADC_REGS Registers

Table 16-33 lists the memory-mapped registers for the ADC_REGS registers. All register offset addresses not listed in Table 16-33 should be considered as reserved locations and the register contents should not be modified.

Table 16-33. ADC_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	ADCCTL1	ADC Control 1 Register	EALLOW	Go
1h	ADCCTL2	ADC Control 2 Register	EALLOW	Go
2h	ADCBURSTCTL	ADC Burst Control Register	EALLOW	Go
3h	ADCINTFLG	ADC Interrupt Flag Register		Go
4h	ADCINTFLGCLR	ADC Interrupt Flag Clear Register		Go
5h	ADCINTOVF	ADC Interrupt Overflow Register		Go
6h	ADCINTOVFCLR	ADC Interrupt Overflow Clear Register		Go
7h	ADCINTSEL1N2	ADC Interrupt 1 and 2 Selection Register	EALLOW	Go
8h	ADCINTSEL3N4	ADC Interrupt 3 and 4 Selection Register	EALLOW	Go
9h	ADCSOCPRICTL	ADC SOC Priority Control Register	EALLOW	Go
Ah	ADCINTSOCSEL1	ADC Interrupt SOC Selection 1 Register	EALLOW	Go
Bh	ADCINTSOCSEL2	ADC Interrupt SOC Selection 2 Register	EALLOW	Go
Ch	ADCSOCFLG1	ADC SOC Flag 1 Register		Go
Dh	ADCSOCFRC1	ADC SOC Force 1 Register		Go
Eh	ADCSOCOVF1	ADC SOC Overflow 1 Register		Go
Fh	ADCSOCOVFCLR1	ADC SOC Overflow Clear 1 Register		Go
10h	ADCSOC0CTL	ADC SOC0 Control Register	EALLOW	Go
12h	ADCSOC1CTL	ADC SOC1 Control Register	EALLOW	Go
14h	ADCSOC2CTL	ADC SOC2 Control Register	EALLOW	Go
16h	ADCSOC3CTL	ADC SOC3 Control Register	EALLOW	Go
18h	ADCSOC4CTL	ADC SOC4 Control Register	EALLOW	Go
1Ah	ADCSOC5CTL	ADC SOC5 Control Register	EALLOW	Go
1Ch	ADCSOC6CTL	ADC SOC6 Control Register	EALLOW	Go
1Eh	ADCSOC7CTL	ADC SOC7 Control Register	EALLOW	Go
20h	ADCSOC8CTL	ADC SOC8 Control Register	EALLOW	Go
22h	ADCSOC9CTL	ADC SOC9 Control Register	EALLOW	Go
24h	ADCSOC10CTL	ADC SOC10 Control Register	EALLOW	Go
26h	ADCSOC11CTL	ADC SOC11 Control Register	EALLOW	Go
28h	ADCSOC12CTL	ADC SOC12 Control Register	EALLOW	Go
2Ah	ADCSOC13CTL	ADC SOC13 Control Register	EALLOW	Go
2Ch	ADCSOC14CTL	ADC SOC14 Control Register	EALLOW	Go
2Eh	ADCSOC15CTL	ADC SOC15 Control Register	EALLOW	Go
30h	ADCEVTSTAT	ADC Event Status Register		Go
32h	ADCEVTCLR	ADC Event Clear Register		Go
34h	ADCEVTSEL	ADC Event Selection Register	EALLOW	Go
36h	ADCEVTINTSEL	ADC Event Interrupt Selection Register	EALLOW	Go
38h	ADCOSDETECT	ADC Open and Shorts Detect Register	EALLOW	Go
39h	ADCCOUNTER	ADC Counter Register		Go
3Ah	ADCREV	ADC Revision Register		Go
3Bh	ADCOFFTRIM	ADC Offset Trim Register	EALLOW	Go
40h	ADCPPB1CONFIG	ADC PPB1 Config Register	EALLOW	Go

Table 16-33. ADC_REGS Registers (continued)

Offset	Acronym	Register Name	Write Protection	Section
41h	ADCPPB1STAMP	ADC PPB1 Sample Delay Time Stamp Register		Go
42h	ADCPPB1OFFCAL	ADC PPB1 Offset Calibration Register	EALLOW	Go
43h	ADCPPB1OFFREF	ADC PPB1 Offset Reference Register		Go
44h	ADCPPB1TRIPHI	ADC PPB1 Trip High Register	EALLOW	Go
46h	ADCPPB1TRIPLO	ADC PPB1 Trip Low/Trigger Time Stamp Register	EALLOW	Go
48h	ADCPPB2CONFIG	ADC PPB2 Config Register	EALLOW	Go
49h	ADCPPB2STAMP	ADC PPB2 Sample Delay Time Stamp Register		Go
4Ah	ADCPPB2OFFCAL	ADC PPB2 Offset Calibration Register	EALLOW	Go
4Bh	ADCPPB2OFFREF	ADC PPB2 Offset Reference Register		Go
4Ch	ADCPPB2TRIPHI	ADC PPB2 Trip High Register	EALLOW	Go
4Eh	ADCPPB2TRIPLO	ADC PPB2 Trip Low/Trigger Time Stamp Register	EALLOW	Go
50h	ADCPPB3CONFIG	ADC PPB3 Config Register	EALLOW	Go
51h	ADCPPB3STAMP	ADC PPB3 Sample Delay Time Stamp Register		Go
52h	ADCPPB3OFFCAL	ADC PPB3 Offset Calibration Register	EALLOW	Go
53h	ADCPPB3OFFREF	ADC PPB3 Offset Reference Register		Go
54h	ADCPPB3TRIPHI	ADC PPB3 Trip High Register	EALLOW	Go
56h	ADCPPB3TRIPLO	ADC PPB3 Trip Low/Trigger Time Stamp Register	EALLOW	Go
58h	ADCPPB4CONFIG	ADC PPB4 Config Register	EALLOW	Go
59h	ADCPPB4STAMP	ADC PPB4 Sample Delay Time Stamp Register		Go
5Ah	ADCPPB4OFFCAL	ADC PPB4 Offset Calibration Register	EALLOW	Go
5Bh	ADCPPB4OFFREF	ADC PPB4 Offset Reference Register		Go
5Ch	ADCPPB4TRIPHI	ADC PPB4 Trip High Register	EALLOW	Go
5Eh	ADCPPB4TRIPLO	ADC PPB4 Trip Low/Trigger Time Stamp Register	EALLOW	Go
6Fh	ADCINTCYCLE	ADC Early Interrupt Generation Cycle	EALLOW	Go
70h	ADCINLTRIM1	ADC Linearity Trim 1 Register	EALLOW	Go
72h	ADCINLTRIM2	ADC Linearity Trim 2 Register	EALLOW	Go
74h	ADCINLTRIM3	ADC Linearity Trim 3 Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 16-34 shows the codes that are used for access types in this section.

Table 16-34. ADC_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 16-34. ADC_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

16.15.3.1 ADCCTL1 Register (Offset = 0h) [Reset = 0h]

ADCCTL1 is shown in Figure 16-40 and described in Table 16-35.

Return to the Summary Table.

ADC Control 1 Register

Figure 16-40. ADCCTL1 Register

15	14	13	12	11	10	9	8
RESERVED		ADCBSY	RESERVED	ADCBSYCHN			
R-0h		R-0h	R-0h	R-0h			
7	6	5	4	3	2	1	0
ADCPWDNZ	RESERVED				INTPULSEPOS	RESERVED	
R/W-0h	R-0h				R/W-0h	R-0h	

Table 16-35. ADCCTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	RESERVED	R	0h	Reserved
13	ADCBSY	R	0h	ADC Busy. Set when ADC SOC is generated, cleared by hardware four ADC clocks after negative edge of S/H pulse. Used by the ADC state machine to determine if ADC is available to sample. 0 ADC is available to sample next channel 1 ADC is busy and cannot sample another channel Reset type: SYSRSn
12	RESERVED	R	0h	Reserved
11-8	ADCBSYCHN	R	0h	ADC Busy Channel. Set when an ADC Start of Conversion (SOC) is generated. When ADCBSY=0: holds the value of the last converted SOC When ADCBSY=1: reflects the SOC currently being processed 0h SOC0 is currently processing or was last SOC converted 1h SOC1 is currently processing or was last SOC converted 2h SOC2 is currently processing or was last SOC converted 3h SOC3 is currently processing or was last SOC converted 4h SOC4 is currently processing or was last SOC converted 5h SOC5 is currently processing or was last SOC converted 6h SOC6 is currently processing or was last SOC converted 7h SOC7 is currently processing or was last SOC converted 8h SOC8 is currently processing or was last SOC converted 9h SOC9 is currently processing or was last SOC converted Ah SOC10 is currently processing or was last SOC converted Bh SOC11 is currently processing or was last SOC converted Ch SOC12 is currently processing or was last SOC converted Dh SOC13 is currently processing or was last SOC converted Eh SOC14 is currently processing or was last SOC converted Fh SOC15 is currently processing or was last SOC converted Reset type: SYSRSn
7	ADCPWDNZ	R/W	0h	ADC Power Down (active low). This bit controls the power up and power down of all the analog circuitry inside the analog core. 0 All analog circuitry inside the core is powered down 1 All analog circuitry inside the core is powered up Reset type: SYSRSn
6-3	RESERVED	R	0h	Reserved
2	INTPULSEPOS	R/W	0h	ADC Interrupt Pulse Position. 0 Interrupt pulse generation occurs when ADC begins conversion (at the end of the acquisition window) plus a number of SYSCLK cycles as specified in the ADCINTCYCLE.OFFSET register. 1 Interrupt pulse generation occurs at the end of the conversion, 1 cycle prior to the ADC result latching into its result register Reset type: SYSRSn

Table 16-35. ADCCTL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	RESERVED	R	0h	Reserved

16.15.3.2 ADCCTL2 Register (Offset = 1h) [Reset = 0h]

ADCCTL2 is shown in Figure 16-41 and described in Table 16-36.

Return to the Summary Table.

ADC Control 2 Register

Figure 16-41. ADCCTL2 Register

15	14	13	12	11	10	9	8
RESERVED				RESERVED			
R-0h				R-0h			
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED		PRESCALE			
R/W-0h	R/W-0h	R-0h		R/W-0h			

Table 16-36. ADCCTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12-8	RESERVED	R	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6	RESERVED	R/W	0h	Reserved
5-4	RESERVED	R	0h	Reserved
3-0	PRESCALE	R/W	0h	ADC Clock Prescaler. 0000 ADCCLK = Input Clock / 1.0 0001 Reserved 0010 ADCCLK = Input Clock / 2.0 0011 Reserved 0100 ADCCLK = Input Clock / 3.0 0101 Reserved 0110 ADCCLK = Input Clock / 4.0 0111 Reserved 1000 ADCCLK = Input Clock / 5.0 1001 Reserved 1010 ADCCLK = Input Clock / 6.0 1011 Reserved 1100 ADCCLK = Input Clock / 7.0 1101 Reserved 1110 ADCCLK = Input Clock / 8.0 1111 Reserved Reset type: SYSRSn

16.15.3.3 ADCBURSTCTL Register (Offset = 2h) [Reset = 0h]

ADCBURSTCTL is shown in Figure 16-42 and described in Table 16-37.

Return to the Summary Table.

ADC Burst Control Register

Figure 16-42. ADCBURSTCTL Register

15	14	13	12	11	10	9	8
BURSTEN	RESERVED			BURSTSIZE			
R/W-0h	R-0h			R/W-0h			
7	6	5	4	3	2	1	0
RESERVED		BURSTTRIGSEL					
R-0h		R/W-0h					

Table 16-37. ADCBURSTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	BURSTEN	R/W	0h	SOC Burst Mode Enable. This bit enables the SOC Burst Mode of operation. 0 Burst mode is disabled. 1 Burst mode is enabled. Reset type: SYSRSn
14-12	RESERVED	R	0h	Reserved
11-8	BURSTSIZE	R/W	0h	SOC Burst Size Select. This bit field determines how many SOCs are converted when a burst conversion sequence is started. The first SOC converted is defined by the round robin pointer, which is advanced as each SOC is converted. 0h 1 SOC converted 1h 2 SOCs converted 2h 3 SOCs converted 3h 4 SOCs converted 4h 5 SOCs converted 5h 6 SOCs converted 6h 7 SOCs converted 7h 8 SOCs converted 8h 9 SOCs converted 9h 10 SOCs converted Ah 11 SOCs converted Bh 12 SOCs converted Ch 13 SOCs converted Dh 14 SOCs converted Eh 15 SOCs converted Fh 16 SOCs converted Reset type: SYSRSn
7-6	RESERVED	R	0h	Reserved

Table 16-37. ADCBURSTCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	BURSTTRIGSEL	R/W	0h	SOC Burst Trigger Source Select. Configures which trigger will start a burst conversion sequence. 00h BURSTTRIG0 - Software only 01h BURSTTRIG1 - CPU1 Timer 0, TINT0n 02h BURSTTRIG2 - CPU1 Timer 1, TINT1n 03h BURSTTRIG3 - CPU1 Timer 2, TINT2n 04h BURSTTRIG4 - GPIO, Input X-Bar INPUT5 05h BURSTTRIG5 - ePWM1, ADCSOCA 06h BURSTTRIG6 - ePWM1, ADCSOCB 07h BURSTTRIG7 - ePWM2, ADCSOCA 08h BURSTTRIG8 - ePWM2, ADCSOCB 09h BURSTTRIG9 - ePWM3, ADCSOCA 0Ah BURSTTRIG10 - ePWM3, ADCSOCB 0Bh BURSTTRIG11 - ePWM4, ADCSOCA 0Ch BURSTTRIG12 - ePWM4, ADCSOCB 0Dh BURSTTRIG13 - ePWM5, ADCSOCA 0Eh BURSTTRIG14 - ePWM5, ADCSOCB 0Fh BURSTTRIG15 - ePWM6, ADCSOCA 10h BURSTTRIG16 - ePWM6, ADCSOCB 11h BURSTTRIG17 - ePWM7, ADCSOCA 12h BURSTTRIG18 - ePWM7, ADCSOCB 13h BURSTTRIG19 - ePWM8, ADCSOCA 14h BURSTTRIG20 - ePWM8, ADCSOCB 15h - 3Fh - Reserved Reset type: SYSRSn

16.15.3.4 ADCINTFLG Register (Offset = 3h) [Reset = 0h]

ADCINTFLG is shown in Figure 16-43 and described in Table 16-38.

Return to the Summary Table.

ADC Interrupt Flag Register

Figure 16-43. ADCINTFLG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				ADCINT4	ADCINT3	ADCINT2	ADCINT1
R-0h				R-0h	R-0h	R-0h	R-0h

Table 16-38. ADCINTFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	ADCINT4	R	0h	ADC Interrupt 4 Flag. Reading these flags indicates if the associated ADCINT pulse was generated since the last clear. 0 No ADC interrupt pulse generated 1 ADC interrupt pulse generated If the ADC interrupt is placed in continue to interrupt mode (INTSELxNy register) then further interrupt pulses are generated whenever a selected EOC event occurs even if the flag bit is set. If the continuous mode is not enabled, then no further interrupt pulses are generated until the user clears this flag bit using the ADCINTFLGCLR register. Rather, an ADC interrupt overflow event occurs in the ADCINTOVF register. Reset type: SYSRSn
2	ADCINT3	R	0h	ADC Interrupt 3 Flag. Reading these flags indicates if the associated ADCINT pulse was generated since the last clear. 0 No ADC interrupt pulse generated 1 ADC interrupt pulse generated If the ADC interrupt is placed in continue to interrupt mode (INTSELxNy register) then further interrupt pulses are generated whenever a selected EOC event occurs even if the flag bit is set. If the continuous mode is not enabled, then no further interrupt pulses are generated until the user clears this flag bit using the ADCINTFLGCLR register. Rather, an ADC interrupt overflow event occurs in the ADCINTOVF register. Reset type: SYSRSn
1	ADCINT2	R	0h	ADC Interrupt 2 Flag. Reading these flags indicates if the associated ADCINT pulse was generated since the last clear. 0 No ADC interrupt pulse generated 1 ADC interrupt pulse generated If the ADC interrupt is placed in continue to interrupt mode (INTSELxNy register) then further interrupt pulses are generated whenever a selected EOC event occurs even if the flag bit is set. If the continuous mode is not enabled, then no further interrupt pulses are generated until the user clears this flag bit using the ADCINTFLGCLR register. Rather, an ADC interrupt overflow event occurs in the ADCINTOVF register. Reset type: SYSRSn

Table 16-38. ADCINTFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	ADCINT1	R	0h	ADC Interrupt 1 Flag. Reading these flags indicates if the associated ADCINT pulse was generated since the last clear. 0 No ADC interrupt pulse generated 1 ADC interrupt pulse generated If the ADC interrupt is placed in continue to interrupt mode (INTSELxNy register) then further interrupt pulses are generated whenever a selected EOC event occurs even if the flag bit is set. If the continuous mode is not enabled, then no further interrupt pulses are generated until the user clears this flag bit using the ADCINTFLGCLR register. Rather, an ADC interrupt overflow event occurs in the ADCINTOVF register. Reset type: SYSRSn

16.15.3.5 ADCINTFLGCLR Register (Offset = 4h) [Reset = 0h]

ADCINTFLGCLR is shown in Figure 16-44 and described in Table 16-39.

Return to the Summary Table.

ADC Interrupt Flag Clear Register

Figure 16-44. ADCINTFLGCLR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				ADCINT4	ADCINT3	ADCINT2	ADCINT1
R-0h				R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 16-39. ADCINTFLGCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	ADCINT4	R-0/W1S	0h	ADC Interrupt 4 Flag Clear. Reads return 0. 0 No action 1 Clears ADCINT4 and ADCINT4RESULT flags in the ADCINTFLG register. If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority and the overflow bit will not be set Reset type: SYSRSn
2	ADCINT3	R-0/W1S	0h	ADC Interrupt 3 Flag Clear. Reads return 0. 0 No action 1 Clears ADCINT3 and ADCINT3RESULT flags in the ADCINTFLG register. If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority and the overflow bit will not be set Reset type: SYSRSn
1	ADCINT2	R-0/W1S	0h	ADC Interrupt 2 Flag Clear. Reads return 0. 0 No action 1 Clears ADCINT2 and ADCINT2RESULT flags in the ADCINTFLG register. . If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority and the overflow bit will not be set Reset type: SYSRSn
0	ADCINT1	R-0/W1S	0h	ADC Interrupt 1 Flag Clear. Reads return 0. 0 No action 1 Clears ADCINT1 and ADCINT1RESULT flags in the ADCINTFLG register. If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority and the overflow bit will not be set Reset type: SYSRSn

16.15.3.6 ADCINTOVF Register (Offset = 5h) [Reset = 0h]

ADCINTOVF is shown in Figure 16-45 and described in Table 16-40.

Return to the Summary Table.

ADC Interrupt Overflow Register

Figure 16-45. ADCINTOVF Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				ADCINT4	ADCINT3	ADCINT2	ADCINT1
R-0h				R-0h	R-0h	R-0h	R-0h

Table 16-40. ADCINTOVF Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	ADCINT4	R	0h	ADC Interrupt 4 Overflow Flags Indicates if an overflow occurred when generating ADCINT pulses. If the respective ADCINTFLG bit is set and a selected additional EOC trigger is generated, then an overflow condition occurs. 0 No ADC interrupt overflow event detected. 1 ADC Interrupt overflow event detected. The overflow bit does not care about the continuous mode bit state. An overflow condition is generated irrespective of this mode selection. Reset type: SYSRSn
2	ADCINT3	R	0h	ADC Interrupt 3 Overflow Flags Indicates if an overflow occurred when generating ADCINT pulses. If the respective ADCINTFLG bit is set and a selected additional EOC trigger is generated, then an overflow condition occurs. 0 No ADC interrupt overflow event detected. 1 ADC Interrupt overflow event detected. The overflow bit does not care about the continuous mode bit state. An overflow condition is generated irrespective of this mode selection. Reset type: SYSRSn
1	ADCINT2	R	0h	ADC Interrupt 2 Overflow Flags Indicates if an overflow occurred when generating ADCINT pulses. If the respective ADCINTFLG bit is set and a selected additional EOC trigger is generated, then an overflow condition occurs. 0 No ADC interrupt overflow event detected. 1 ADC Interrupt overflow event detected. The overflow bit does not care about the continuous mode bit state. An overflow condition is generated irrespective of this mode selection. Reset type: SYSRSn
0	ADCINT1	R	0h	ADC Interrupt 1 Overflow Flags Indicates if an overflow occurred when generating ADCINT pulses. If the respective ADCINTFLG bit is set and a selected additional EOC trigger is generated, then an overflow condition occurs. 0 No ADC interrupt overflow event detected. 1 ADC Interrupt overflow event detected. The overflow bit does not care about the continuous mode bit state. An overflow condition is generated irrespective of this mode selection. Reset type: SYSRSn

16.15.3.7 ADCINTOVFCLR Register (Offset = 6h) [Reset = 0h]

ADCINTOVFCLR is shown in Figure 16-46 and described in Table 16-41.

Return to the Summary Table.

ADC Interrupt Overflow Clear Register

Figure 16-46. ADCINTOVFCLR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				ADCINT4	ADCINT3	ADCINT2	ADCINT1
R-0h				R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 16-41. ADCINTOVFCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	ADCINT4	R-0/W1S	0h	ADC Interrupt 4 Overflow Clear Bits 0 No action. 1 Clears the respective overflow bit in the ADCINTOVF register. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCINTOVF register, then hardware has priority and the ADCINTOVF bit will be set. Reset type: SYSRSn
2	ADCINT3	R-0/W1S	0h	ADC Interrupt 3 Overflow Clear Bits 0 No action. 1 Clears the respective overflow bit in the ADCINTOVF register. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCINTOVF register, then hardware has priority and the ADCINTOVF bit will be set. Reset type: SYSRSn
1	ADCINT2	R-0/W1S	0h	ADC Interrupt 2 Overflow Clear Bits 0 No action. 1 Clears the respective overflow bit in the ADCINTOVF register. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCINTOVF register, then hardware has priority and the ADCINTOVF bit will be set. Reset type: SYSRSn
0	ADCINT1	R-0/W1S	0h	ADC Interrupt 1 Overflow Clear Bits 0 No action. 1 Clears the respective overflow bit in the ADCINTOVF register. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCINTOVF register, then hardware has priority and the ADCINTOVF bit will be set. Reset type: SYSRSn

16.15.3.8 ADCINTSEL1N2 Register (Offset = 7h) [Reset = 0h]

ADCINTSEL1N2 is shown in Figure 16-47 and described in Table 16-42.

Return to the Summary Table.

ADC Interrupt 1 and 2 Selection Register

Figure 16-47. ADCINTSEL1N2 Register

15	14	13	12	11	10	9	8
RESERVED	INT2CONT	INT2E	RESERVED	INT2SEL			
R-0h	R/W-0h	R/W-0h	R-0h	R/W-0h			
7	6	5	4	3	2	1	0
RESERVED	INT1CONT	INT1E	RESERVED	INT1SEL			
R-0h	R/W-0h	R/W-0h	R-0h	R/W-0h			

Table 16-42. ADCINTSEL1N2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	INT2CONT	R/W	0h	ADCINT2 Continue to Interrupt Mode 0 No further ADCINT2 pulses are generated until ADCINT2 flag (in ADCINTFLG register) is cleared by user. 1 ADCINT2 pulses are generated whenever an EOC pulse is generated irrespective of whether the flag bit is cleared or not. Reset type: SYSRSn
13	INT2E	R/W	0h	ADCINT2 Interrupt Enable 0 ADCINT2 is disabled 1 ADCINT2 is enabled Reset type: SYSRSn
12	RESERVED	R	0h	Reserved
11-8	INT2SEL	R/W	0h	ADCINT2 EOC Source Select 0h EOC0 is trigger for ADCINT2 1h EOC1 is trigger for ADCINT2 2h EOC2 is trigger for ADCINT2 3h EOC3 is trigger for ADCINT2 4h EOC4 is trigger for ADCINT2 5h EOC5 is trigger for ADCINT2 6h EOC6 is trigger for ADCINT2 7h EOC7 is trigger for ADCINT2 8h EOC8 is trigger for ADCINT2 9h EOC9 is trigger for ADCINT2 Ah EOC10 is trigger for ADCINT2 Bh EOC11 is trigger for ADCINT2 Ch EOC12 is trigger for ADCINT2 Dh EOC13 is trigger for ADCINT2 Eh EOC14 is trigger for ADCINT2 Fh EOC15 is trigger for ADCINT2 Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6	INT1CONT	R/W	0h	ADCINT1 Continue to Interrupt Mode 0 No further ADCINT1 pulses are generated until ADCINT1 flag (in ADCINTFLG register) is cleared by user. 1 ADCINT1 pulses are generated whenever an EOC pulse is generated irrespective of whether the flag bit is cleared or not. Reset type: SYSRSn
5	INT1E	R/W	0h	ADCINT1 Interrupt Enable 0 ADCINT1 is disabled 1 ADCINT1 is enabled Reset type: SYSRSn
4	RESERVED	R	0h	Reserved

Table 16-42. ADCINTSEL1N2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	INT1SEL	R/W	0h	ADCINT1 EOC Source Select 0h EOC0 is trigger for ADCINT1 1h EOC1 is trigger for ADCINT1 2h EOC2 is trigger for ADCINT1 3h EOC3 is trigger for ADCINT1 4h EOC4 is trigger for ADCINT1 5h EOC5 is trigger for ADCINT1 6h EOC6 is trigger for ADCINT1 7h EOC7 is trigger for ADCINT1 8h EOC8 is trigger for ADCINT1 9h EOC9 is trigger for ADCINT1 Ah EOC10 is trigger for ADCINT1 Bh EOC11 is trigger for ADCINT1 Ch EOC12 is trigger for ADCINT1 Dh EOC13 is trigger for ADCINT1 Eh EOC14 is trigger for ADCINT1 Fh EOC15 is trigger for ADCINT1 Reset type: SYSRSn

16.15.3.9 ADCINTSEL3N4 Register (Offset = 8h) [Reset = 0h]

ADCINTSEL3N4 is shown in Figure 16-48 and described in Table 16-43.

Return to the Summary Table.

ADC Interrupt 3 and 4 Selection Register

Figure 16-48. ADCINTSEL3N4 Register

15	14	13	12	11	10	9	8
RESERVED	INT4CONT	INT4E	RESERVED	INT4SEL			
R-0h	R/W-0h	R/W-0h	R-0h	R/W-0h			
7	6	5	4	3	2	1	0
RESERVED	INT3CONT	INT3E	RESERVED	INT3SEL			
R-0h	R/W-0h	R/W-0h	R-0h	R/W-0h			

Table 16-43. ADCINTSEL3N4 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	INT4CONT	R/W	0h	ADCINT4 Continue to Interrupt Mode 0 No further ADCINT4 pulses are generated until ADCINT4 flag (in ADCINTFLG register) is cleared by user. 1 ADCINT4 pulses are generated whenever an EOC pulse is generated irrespective of whether the flag bit is cleared or not. Reset type: SYSRSn
13	INT4E	R/W	0h	ADCINT4 Interrupt Enable 0 ADCINT4 is disabled 1 ADCINT4 is enabled Reset type: SYSRSn
12	RESERVED	R	0h	Reserved
11-8	INT4SEL	R/W	0h	ADCINT4 EOC Source Select 0h EOC0 is trigger for ADCINT4 1h EOC1 is trigger for ADCINT4 2h EOC2 is trigger for ADCINT4 3h EOC3 is trigger for ADCINT4 4h EOC4 is trigger for ADCINT4 5h EOC5 is trigger for ADCINT4 6h EOC6 is trigger for ADCINT4 7h EOC7 is trigger for ADCINT4 8h EOC8 is trigger for ADCINT4 9h EOC9 is trigger for ADCINT4 Ah EOC10 is trigger for ADCINT4 Bh EOC11 is trigger for ADCINT4 Ch EOC12 is trigger for ADCINT4 Dh EOC13 is trigger for ADCINT4 Eh EOC14 is trigger for ADCINT4 Fh EOC15 is trigger for ADCINT4 Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6	INT3CONT	R/W	0h	ADCINT3 Continue to Interrupt Mode 0 No further ADCINT3 pulses are generated until ADCINT3 flag (in ADCINTFLG register) is cleared by user. 1 ADCINT3 pulses are generated whenever an EOC pulse is generated irrespective of whether the flag bit is cleared or not. Reset type: SYSRSn
5	INT3E	R/W	0h	ADCINT3 Interrupt Enable 0 ADCINT3 is disabled 1 ADCINT3 is enabled Reset type: SYSRSn
4	RESERVED	R	0h	Reserved

Table 16-43. ADCINTSEL3N4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	INT3SEL	R/W	0h	ADCINT3 EOC Source Select 0h EOC0 is trigger for ADCINT3 1h EOC1 is trigger for ADCINT3 2h EOC2 is trigger for ADCINT3 3h EOC3 is trigger for ADCINT3 4h EOC4 is trigger for ADCINT3 5h EOC5 is trigger for ADCINT3 6h EOC6 is trigger for ADCINT3 7h EOC7 is trigger for ADCINT3 8h EOC8 is trigger for ADCINT3 9h EOC9 is trigger for ADCINT3 Ah EOC10 is trigger for ADCINT3 Bh EOC11 is trigger for ADCINT3 Ch EOC12 is trigger for ADCINT3 Dh EOC13 is trigger for ADCINT3 Eh EOC14 is trigger for ADCINT3 Fh EOC15 is trigger for ADCINT3 Reset type: SYSRSn

16.15.3.10 ADCSOCPRICTL Register (Offset = 9h) [Reset = 200h]

ADCSOCPRICTL is shown in Figure 16-49 and described in Table 16-44.

Return to the Summary Table.

ADC SOC Priority Control Register

Figure 16-49. ADCSOCPRICTL Register

15	14	13	12	11	10	9	8
RESERVED						RRPOINTER	
R-0h						R-10h	
7	6	5	4	3	2	1	0
RRPOINTER				SOCPRIORITY			
R-10h				R/W-0h			

Table 16-44. ADCSOCPRICTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-5	RRPOINTER	R	10h	Round Robin Pointer. Holds the value of the last converted round robin SOCx to be used by the round robin scheme to determine order of conversions. 00h SOC0 was last round robin SOC to convert, SOC1 is highest round robin priority. 01h SOC1 was last round robin SOC to convert, SOC2 is highest round robin priority. 02h SOC2 was last round robin SOC to convert, SOC3 is highest round robin priority. 03h SOC3 was last round robin SOC to convert, SOC4 is highest round robin priority. 04h SOC4 was last round robin SOC to convert, SOC5 is highest round robin priority. 05h SOC5 was last round robin SOC to convert, SOC6 is highest round robin priority. 06h SOC6 was last round robin SOC to convert, SOC7 is highest round robin priority. 07h SOC7 was last round robin SOC to convert, SOC8 is highest round robin priority. 08h SOC8 was last round robin SOC to convert, SOC9 is highest round robin priority. 09h SOC9 was last round robin SOC to convert, SOC10 is highest round robin priority. 0Ah SOC10 was last round robin SOC to convert, SOC11 is highest round robin priority. 0Bh SOC11 was last round robin SOC to convert, SOC12 is highest round robin priority. 0Ch SOC12 was last round robin SOC to convert, SOC13 is highest round robin priority. 0Dh SOC13 was last round robin SOC to convert, SOC14 is highest round robin priority. 0Eh SOC14 was last round robin SOC to convert, SOC15 is highest round robin priority. 0Fh SOC15 was last round robin SOC to convert, SOC0 is highest round robin priority. 10h Reset value to indicate no SOC has been converted. SOC0 is highest round robin priority. Set to this value when the ADC module is reset by SOFTPRES or when the ADCSOCPRICTL register is written. In the latter case, if a conversion is currently in progress, it will complete and then the new priority will take effect. Others Invalid value. Reset type: SYSRSn

Table 16-44. ADCSOCPRICTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-0	SOCPRIORITY	R/W	0h	SOC Priority Determines the cutoff point for priority mode and round robin arbitration for SOCx 00h SOC priority is handled in round robin mode for all channels. 01h SOC0 is high priority, rest of channels are in round robin mode. 02h SOC0-SOC1 are high priority, SOC2-SOC15 are in round robin mode. 03h SOC0-SOC2 are high priority, SOC3-SOC15 are in round robin mode. 04h SOC0-SOC3 are high priority, SOC4-SOC15 are in round robin mode. 05h SOC0-SOC4 are high priority, SOC5-SOC15 are in round robin mode. 06h SOC0-SOC5 are high priority, SOC6-SOC15 are in round robin mode. 07h SOC0-SOC6 are high priority, SOC7-SOC15 are in round robin mode. 08h SOC0-SOC7 are high priority, SOC8-SOC15 are in round robin mode. 09h SOC0-SOC8 are high priority, SOC9-SOC15 are in round robin mode. 0Ah SOC0-SOC9 are high priority, SOC10-SOC15 are in round robin mode. 0Bh SOC0-SOC10 are high priority, SOC11-SOC15 are in round robin mode. 0Ch SOC0-SOC11 are high priority, SOC12-SOC15 are in round robin mode. 0Dh SOC0-SOC12 are high priority, SOC13-SOC15 are in round robin mode. 0Eh SOC0-SOC13 are high priority, SOC14-SOC15 are in round robin mode. 0Fh SOC0-SOC14 are high priority, SOC15 is in round robin mode. 10h All SOCs are in high priority mode, arbitrated by SOC number. Others Invalid selection. Reset type: SYSRSn

16.15.3.11 ADCINTSOCSEL1 Register (Offset = Ah) [Reset = 0h]

ADCINTSOCSEL1 is shown in Figure 16-50 and described in Table 16-45.

Return to the Summary Table.

ADC Interrupt SOC Selection 1 Register

Figure 16-50. ADCINTSOCSEL1 Register

15	14	13	12	11	10	9	8
SOC7		SOC6		SOC5		SOC4	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
SOC3		SOC2		SOC1		SOC0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 16-45. ADCINTSOCSEL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	SOC7	R/W	0h	SOC7 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC7. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC7. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC7. 10 ADCINT2 will trigger SOC7. 11 Invalid selection. Reset type: SYSRSn
13-12	SOC6	R/W	0h	SOC6 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC6. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC6. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC6. 10 ADCINT2 will trigger SOC6. 11 Invalid selection. Reset type: SYSRSn
11-10	SOC5	R/W	0h	SOC5 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC5. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC5. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC5. 10 ADCINT2 will trigger SOC5. 11 Invalid selection. Reset type: SYSRSn
9-8	SOC4	R/W	0h	SOC4 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC4. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC4. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC4. 10 ADCINT2 will trigger SOC4. 11 Invalid selection. Reset type: SYSRSn

Table 16-45. ADCINTSOCSEL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	SOC3	R/W	0h	SOC3 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC3. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC3. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC3. 10 ADCINT2 will trigger SOC3. 11 Invalid selection. Reset type: SYSRSn
5-4	SOC2	R/W	0h	SOC2 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC2. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC2. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC2. 10 ADCINT2 will trigger SOC2. 11 Invalid selection. Reset type: SYSRSn
3-2	SOC1	R/W	0h	SOC1 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC1. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC1. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC1. 10 ADCINT2 will trigger SOC1. 11 Invalid selection. Reset type: SYSRSn
1-0	SOC0	R/W	0h	SOC0 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC0. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC0. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC0. 10 ADCINT2 will trigger SOC0. 11 Invalid selection. Reset type: SYSRSn

16.15.3.12 ADCINTSOCSEL2 Register (Offset = Bh) [Reset = 0h]

ADCINTSOCSEL2 is shown in Figure 16-51 and described in Table 16-46.

Return to the Summary Table.

ADC Interrupt SOC Selection 2 Register

Figure 16-51. ADCINTSOCSEL2 Register

15	14	13	12	11	10	9	8
SOC15		SOC14		SOC13		SOC12	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
SOC11		SOC10		SOC9		SOC8	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 16-46. ADCINTSOCSEL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	SOC15	R/W	0h	SOC15 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC15. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC15. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC15. 10 ADCINT2 will trigger SOC15. 11 Invalid selection. Reset type: SYSRSn
13-12	SOC14	R/W	0h	SOC14 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC14. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC14. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC14. 10 ADCINT2 will trigger SOC14. 11 Invalid selection. Reset type: SYSRSn
11-10	SOC13	R/W	0h	SOC13 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC13. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC13. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC13. 10 ADCINT2 will trigger SOC13. 11 Invalid selection. Reset type: SYSRSn
9-8	SOC12	R/W	0h	SOC12 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC12. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC12. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC12. 10 ADCINT2 will trigger SOC12. 11 Invalid selection. Reset type: SYSRSn

Table 16-46. ADCINTSOCSEL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-6	SOC11	R/W	0h	SOC11 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC11. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC11. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC11. 10 ADCINT2 will trigger SOC11. 11 Invalid selection. Reset type: SYSRSn
5-4	SOC10	R/W	0h	SOC10 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC10. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC10. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC10. 10 ADCINT2 will trigger SOC10. 11 Invalid selection. Reset type: SYSRSn
3-2	SOC9	R/W	0h	SOC9 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC9. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC9. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC9. 10 ADCINT2 will trigger SOC9. 11 Invalid selection. Reset type: SYSRSn
1-0	SOC8	R/W	0h	SOC8 ADC Interrupt Trigger Select. Selects which, if any, ADCINT triggers SOC8. The trigger selected in this field is in addition to the TRIGSEL field in the ADCSOCxCTL register. 00 No ADCINT will trigger SOC8. TRIGSEL field alone determines SOC0 trigger. 01 ADCINT1 will trigger SOC8. 10 ADCINT2 will trigger SOC8. 11 Invalid selection. Reset type: SYSRSn

16.15.3.13 ADCSOCFLG1 Register (Offset = Ch) [Reset = 0h]

ADCSOCFLG1 is shown in Figure 16-52 and described in Table 16-47.

Return to the Summary Table.

ADC SOC Flag 1 Register

Figure 16-52. ADCSOCFLG1 Register

15	14	13	12	11	10	9	8
SOC15	SOC14	SOC13	SOC12	SOC11	SOC10	SOC9	SOC8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
SOC7	SOC6	SOC5	SOC4	SOC3	SOC2	SOC1	SOC0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 16-47. ADCSOCFLG1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	SOC15	R	0h	SOC15 Start of Conversion Flag. Indicates the state of SOC15 conversions. 0 No sample pending for SOC15. 1 Trigger has been received and sample is pending for SOC15. This bit will be automatically cleared when the SOC15 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn
14	SOC14	R	0h	SOC14 Start of Conversion Flag. Indicates the state of SOC14 conversions. 0 No sample pending for SOC14. 1 Trigger has been received and sample is pending for SOC14. This bit will be automatically cleared when the SOC14 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn
13	SOC13	R	0h	SOC13 Start of Conversion Flag. Indicates the state of SOC13 conversions. 0 No sample pending for SOC13. 1 Trigger has been received and sample is pending for SOC13. This bit will be automatically cleared when the SOC13 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn

Table 16-47. ADCSOCFLG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	SOC12	R	0h	SOC12 Start of Conversion Flag. Indicates the state of SOC12 conversions. 0 No sample pending for SOC12. 1 Trigger has been received and sample is pending for SOC12. This bit will be automatically cleared when the SOC12 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn
11	SOC11	R	0h	SOC11 Start of Conversion Flag. Indicates the state of SOC11 conversions. 0 No sample pending for SOC11. 1 Trigger has been received and sample is pending for SOC11. This bit will be automatically cleared when the SOC11 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn
10	SOC10	R	0h	SOC10 Start of Conversion Flag. Indicates the state of SOC10 conversions. 0 No sample pending for SOC10. 1 Trigger has been received and sample is pending for SOC10. This bit will be automatically cleared when the SOC10 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn
9	SOC9	R	0h	SOC9 Start of Conversion Flag. Indicates the state of SOC9 conversions. 0 No sample pending for SOC9. 1 Trigger has been received and sample is pending for SOC9. This bit will be automatically cleared when the SOC9 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn
8	SOC8	R	0h	SOC8 Start of Conversion Flag. Indicates the state of SOC8 conversions. 0 No sample pending for SOC8. 1 Trigger has been received and sample is pending for SOC8. This bit will be automatically cleared when the SOC8 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn

Table 16-47. ADCSOCFLG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	SOC7	R	0h	SOC7 Start of Conversion Flag. Indicates the state of SOC7 conversions. 0 No sample pending for SOC7. 1 Trigger has been received and sample is pending for SOC7. This bit will be automatically cleared when the SOC7 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn
6	SOC6	R	0h	SOC6 Start of Conversion Flag. Indicates the state of SOC6 conversions. 0 No sample pending for SOC6. 1 Trigger has been received and sample is pending for SOC6. This bit will be automatically cleared when the SOC6 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn
5	SOC5	R	0h	SOC5 Start of Conversion Flag. Indicates the state of SOC5 conversions. 0 No sample pending for SOC5. 1 Trigger has been received and sample is pending for SOC5. This bit will be automatically cleared when the SOC5 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn
4	SOC4	R	0h	SOC4 Start of Conversion Flag. Indicates the state of SOC4 conversions. 0 No sample pending for SOC4. 1 Trigger has been received and sample is pending for SOC4. This bit will be automatically cleared when the SOC4 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn
3	SOC3	R	0h	SOC3 Start of Conversion Flag. Indicates the state of SOC3 conversions. 0 No sample pending for SOC3. 1 Trigger has been received and sample is pending for SOC3. This bit will be automatically cleared when the SOC3 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn

Table 16-47. ADCSOCFLG1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	SOC2	R	0h	SOC2 Start of Conversion Flag. Indicates the state of SOC2 conversions. 0 No sample pending for SOC2. 1 Trigger has been received and sample is pending for SOC2. This bit will be automatically cleared when the SOC2 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn
1	SOC1	R	0h	SOC1 Start of Conversion Flag. Indicates the state of SOC1 conversions. 0 No sample pending for SOC1. 1 Trigger has been received and sample is pending for SOC1. This bit will be automatically cleared when the SOC1 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn
0	SOC0	R	0h	SOC0 Start of Conversion Flag. Indicates the state of SOC0 conversions. 0 No sample pending for SOC0. 1 Trigger has been received and sample is pending for SOC0. This bit will be automatically cleared when the SOC0 conversion is started. If contention exists where this bit receives both a request to set and a request to clear on the same cycle, regardless of the source of either, this bit will be set and the request to clear will be ignored. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether this bit was previously set or not. Reset type: SYSRSn

16.15.3.14 ADCSOCFRC1 Register (Offset = Dh) [Reset = 0h]

ADCSOCFRC1 is shown in Figure 16-53 and described in Table 16-48.

Return to the Summary Table.

ADC SOC Force 1 Register

Figure 16-53. ADCSOCFRC1 Register

15	14	13	12	11	10	9	8
SOC15	SOC14	SOC13	SOC12	SOC11	SOC10	SOC9	SOC8
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
SOC7	SOC6	SOC5	SOC4	SOC3	SOC2	SOC1	SOC0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 16-48. ADCSOCFRC1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	SOC15	R-0/W1S	0h	SOC15 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC15 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC15 flag bit to 1. This will cause a conversion to start once priority is given to SOC15. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC15 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn
14	SOC14	R-0/W1S	0h	SOC14 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC14 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC14 flag bit to 1. This will cause a conversion to start once priority is given to SOC14. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC14 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn
13	SOC13	R-0/W1S	0h	SOC13 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC13 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC13 flag bit to 1. This will cause a conversion to start once priority is given to SOC13. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC13 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn

Table 16-48. ADCSOCFRC1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	SOC12	R-0/W1S	0h	SOC12 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC12 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC12 flag bit to 1. This will cause a conversion to start once priority is given to SOC12. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC12 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn
11	SOC11	R-0/W1S	0h	SOC11 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC11 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC11 flag bit to 1. This will cause a conversion to start once priority is given to SOC11. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC11 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn
10	SOC10	R-0/W1S	0h	SOC10 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC10 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC10 flag bit to 1. This will cause a conversion to start once priority is given to SOC10. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC10 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn
9	SOC9	R-0/W1S	0h	SOC9 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC9 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC9 flag bit to 1. This will cause a conversion to start once priority is given to SOC9. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC9 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn

Table 16-48. ADCSOCFRC1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	SOC8	R-0/W1S	0h	SOC8 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC8 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC8 flag bit to 1. This will cause a conversion to start once priority is given to SOC8. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC8 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn
7	SOC7	R-0/W1S	0h	SOC7 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC7 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC7 flag bit to 1. This will cause a conversion to start once priority is given to SOC7. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC7 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn
6	SOC6	R-0/W1S	0h	SOC6 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC6 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC6 flag bit to 1. This will cause a conversion to start once priority is given to SOC6. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC6 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn
5	SOC5	R-0/W1S	0h	SOC5 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC5 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC5 flag bit to 1. This will cause a conversion to start once priority is given to SOC5. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC5 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn

Table 16-48. ADCSOCFRC1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	SOC4	R-0/W1S	0h	SOC4 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC4 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC4 flag bit to 1. This will cause a conversion to start once priority is given to SOC4. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC4 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn
3	SOC3	R-0/W1S	0h	SOC3 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC3 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC3 flag bit to 1. This will cause a conversion to start once priority is given to SOC3. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC3 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn
2	SOC2	R-0/W1S	0h	SOC2 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC2 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC2 flag bit to 1. This will cause a conversion to start once priority is given to SOC2. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC2 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn
1	SOC1	R-0/W1S	0h	SOC1 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC1 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC1 flag bit to 1. This will cause a conversion to start once priority is given to SOC1. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC1 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn

Table 16-48. ADCSOCFRC1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	SOC0	R-0/W1S	0h	SOC0 Force Start of Conversion Bit. Writing a 1 will force to 1 the SOC0 flag in the ADCSOCFLG1 register. This can be used to initiate a software initiated conversion. Writes of 0 are ignored. This bit will always read as a 0. 0 No action. 1 Force SOC0 flag bit to 1. This will cause a conversion to start once priority is given to SOC0. If software tries to set this bit on the same clock cycle that hardware tries to clear the SOC0 bit in the ADCSOCFLG1 register, then software has priority and the ADCSOCFLG1 bit will be set. In this case the overflow bit in the ADCSOCOVF1 register will not be affected regardless of whether the ADCSOCFLG1 bit was previously set or not. Reset type: SYSRSn

16.15.3.15 ADCSOCOVF1 Register (Offset = Eh) [Reset = 0h]

ADCSOCOVF1 is shown in Figure 16-54 and described in Table 16-49.

Return to the Summary Table.

ADC SOC Overflow 1 Register

Figure 16-54. ADCSOCOVF1 Register

15	14	13	12	11	10	9	8
SOC15	SOC14	SOC13	SOC12	SOC11	SOC10	SOC9	SOC8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
SOC7	SOC6	SOC5	SOC4	SOC3	SOC2	SOC1	SOC0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 16-49. ADCSOCOVF1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	SOC15	R	0h	SOC15 Start of Conversion Overflow Flag. Indicates an SOC15 event was generated in hardware while an existing SOC15 event was already pending. 0 No SOC15 event overflow. 1 SOC15 event overflow. An overflow condition does not stop SOC15 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
14	SOC14	R	0h	SOC14 Start of Conversion Overflow Flag. Indicates an SOC14 event was generated in hardware while an existing SOC14 event was already pending. 0 No SOC14 event overflow. 1 SOC14 event overflow. An overflow condition does not stop SOC14 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
13	SOC13	R	0h	SOC13 Start of Conversion Overflow Flag. Indicates an SOC13 event was generated in hardware while an existing SOC13 event was already pending. 0 No SOC13 event overflow. 1 SOC13 event overflow. An overflow condition does not stop SOC13 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
12	SOC12	R	0h	SOC12 Start of Conversion Overflow Flag. Indicates an SOC12 event was generated in hardware while an existing SOC12 event was already pending. 0 No SOC12 event overflow. 1 SOC12 event overflow. An overflow condition does not stop SOC12 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn

Table 16-49. ADCSOCOVF1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	SOC11	R	0h	SOC11 Start of Conversion Overflow Flag. Indicates an SOC11 event was generated in hardware while an existing SOC11 event was already pending. 0 No SOC11 event overflow. 1 SOC11 event overflow. An overflow condition does not stop SOC11 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
10	SOC10	R	0h	SOC10 Start of Conversion Overflow Flag. Indicates an SOC10 event was generated in hardware while an existing SOC10 event was already pending. 0 No SOC10 event overflow. 1 SOC10 event overflow. An overflow condition does not stop SOC10 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
9	SOC9	R	0h	SOC9 Start of Conversion Overflow Flag. Indicates an SOC9 event was generated in hardware while an existing SOC9 event was already pending. 0 No SOC9 event overflow. 1 SOC9 event overflow. An overflow condition does not stop SOC9 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
8	SOC8	R	0h	SOC8 Start of Conversion Overflow Flag. Indicates an SOC8 event was generated in hardware while an existing SOC8 event was already pending. 0 No SOC8 event overflow. 1 SOC8 event overflow. An overflow condition does not stop SOC8 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
7	SOC7	R	0h	SOC7 Start of Conversion Overflow Flag. Indicates an SOC7 event was generated in hardware while an existing SOC7 event was already pending. 0 No SOC7 event overflow. 1 SOC7 event overflow. An overflow condition does not stop SOC7 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
6	SOC6	R	0h	SOC6 Start of Conversion Overflow Flag. Indicates an SOC6 event was generated in hardware while an existing SOC6 event was already pending. 0 No SOC6 event overflow. 1 SOC6 event overflow. An overflow condition does not stop SOC6 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
5	SOC5	R	0h	SOC5 Start of Conversion Overflow Flag. Indicates an SOC5 event was generated in hardware while an existing SOC5 event was already pending. 0 No SOC5 event overflow. 1 SOC5 event overflow. An overflow condition does not stop SOC5 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn

Table 16-49. ADCSOCOVF1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	SOC4	R	0h	SOC4 Start of Conversion Overflow Flag. Indicates an SOC4 event was generated in hardware while an existing SOC4 event was already pending. 0 No SOC4 event overflow. 1 SOC4 event overflow. An overflow condition does not stop SOC4 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
3	SOC3	R	0h	SOC3 Start of Conversion Overflow Flag. Indicates an SOC3 event was generated in hardware while an existing SOC3 event was already pending. 0 No SOC3 event overflow. 1 SOC3 event overflow. An overflow condition does not stop SOC3 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
2	SOC2	R	0h	SOC2 Start of Conversion Overflow Flag. Indicates an SOC2 event was generated in hardware while an existing SOC2 event was already pending. 0 No SOC2 event overflow. 1 SOC2 event overflow. An overflow condition does not stop SOC2 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
1	SOC1	R	0h	SOC1 Start of Conversion Overflow Flag. Indicates an SOC1 event was generated in hardware while an existing SOC1 event was already pending. 0 No SOC1 event overflow. 1 SOC1 event overflow. An overflow condition does not stop SOC1 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn
0	SOC0	R	0h	SOC0 Start of Conversion Overflow Flag. Indicates an SOC0 event was generated in hardware while an existing SOC0 event was already pending. 0 No SOC0 event overflow. 1 SOC0 event overflow. An overflow condition does not stop SOC0 events from being processed. It simply is an indication that a hardware trigger was missed. A write to the ADCSOCFRC1 register does not affect this bit. Reset type: SYSRSn

16.15.3.16 ADCSOCOVFCLR1 Register (Offset = Fh) [Reset = 0h]

ADCSOCOVFCLR1 is shown in Figure 16-55 and described in Table 16-50.

Return to the Summary Table.

ADC SOC Overflow Clear 1 Register

Figure 16-55. ADCSOCOVFCLR1 Register

15	14	13	12	11	10	9	8
SOC15	SOC14	SOC13	SOC12	SOC11	SOC10	SOC9	SOC8
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
SOC7	SOC6	SOC5	SOC4	SOC3	SOC2	SOC1	SOC0
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 16-50. ADCSOCOVFCLR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	SOC15	R-0/W1S	0h	SOC15 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC15 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC15 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
14	SOC14	R-0/W1S	0h	SOC14 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC14 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC14 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
13	SOC13	R-0/W1S	0h	SOC13 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC13 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC13 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
12	SOC12	R-0/W1S	0h	SOC12 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC12 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC12 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn

Table 16-50. ADCSOCOVFCLR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	SOC11	R-0/W1S	0h	SOC11 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC11 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC11 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
10	SOC10	R-0/W1S	0h	SOC10 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC10 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC10 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
9	SOC9	R-0/W1S	0h	SOC9 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC9 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC9 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
8	SOC8	R-0/W1S	0h	SOC8 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC8 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC8 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
7	SOC7	R-0/W1S	0h	SOC7 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC7 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC7 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
6	SOC6	R-0/W1S	0h	SOC6 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC6 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC6 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
5	SOC5	R-0/W1S	0h	SOC5 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC5 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC5 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn

Table 16-50. ADCSOCOVFCLR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	SOC4	R-0/W1S	0h	SOC4 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC4 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC4 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
3	SOC3	R-0/W1S	0h	SOC3 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC3 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC3 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
2	SOC2	R-0/W1S	0h	SOC2 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC2 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC2 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
1	SOC1	R-0/W1S	0h	SOC1 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC1 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC1 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn
0	SOC0	R-0/W1S	0h	SOC0 Clear Start of Conversion Overflow Bit. Writing a 1 will clear the SOC0 overflow flag in the ADCSOCOVF1 register. Writes of 0 are ignored. Reads will always return a 0. 0 No action. 1 Clear SOC0 overflow flag. If software tries to set this bit on the same clock cycle that hardware tries to set the overflow bit in the ADCSOCOVF1 register, then hardware has priority and the ADCSOCOVF1 bit will be set.. Reset type: SYSRSn

16.15.3.17 ADCSOC0CTL Register (Offset = 10h) [Reset = 0h]

ADCSOC0CTL is shown in Figure 16-56 and described in Table 16-51.

Return to the Summary Table.

ADC SOC0 Control Register

Figure 16-56. ADCSOC0CTL Register

31	30	29	28	27	26	25	24
RESERVED							TRIGSEL
R-0h							R/W-0h
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED	CHSEL		
R/W-0h				R-0h	R/W-0h		
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 16-51. ADCSOC0CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24-20	TRIGSEL	R/W	0h	SOC0 Trigger Source Select. Along with the SOC0 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC0 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h - 1Fh - Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 16-51. ADCSOC0CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC0 Channel Select. Selects the channel to be converted when SOC0 is received by the ADC. 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC0 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

16.15.3.18 ADCSOC1CTL Register (Offset = 12h) [Reset = 0h]

ADCSOC1CTL is shown in Figure 16-57 and described in Table 16-52.

Return to the Summary Table.

ADC SOC1 Control Register

Figure 16-57. ADCSOC1CTL Register

31	30	29	28	27	26	25	24
RESERVED							TRIGSEL
R-0h							R/W-0h
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED	CHSEL		
R/W-0h				R-0h	R/W-0h		
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 16-52. ADCSOC1CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24-20	TRIGSEL	R/W	0h	SOC1 Trigger Source Select. Along with the SOC1 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC1 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h - 1Fh - Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 16-52. ADCSOC1CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC1 Channel Select. Selects the channel to be converted when SOC1 is received by the ADC. 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC1 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

16.15.3.19 ADCSOC2CTL Register (Offset = 14h) [Reset = 0h]

ADCSOC2CTL is shown in Figure 16-58 and described in Table 16-53.

Return to the Summary Table.

ADC SOC2 Control Register

Figure 16-58. ADCSOC2CTL Register

31	30	29	28	27	26	25	24
RESERVED							TRIGSEL
R-0h							R/W-0h
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED	CHSEL		
R/W-0h				R-0h	R/W-0h		
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 16-53. ADCSOC2CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24-20	TRIGSEL	R/W	0h	SOC2 Trigger Source Select. Along with the SOC2 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC2 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h - 1Fh - Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 16-53. ADCSOC2CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC2 Channel Select. Selects the channel to be converted when SOC2 is received by the ADC. 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC2 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

16.15.3.20 ADCSOC3CTL Register (Offset = 16h) [Reset = 0h]

ADCSOC3CTL is shown in Figure 16-59 and described in Table 16-54.

Return to the Summary Table.

ADC SOC3 Control Register

Figure 16-59. ADCSOC3CTL Register

31	30	29	28	27	26	25	24
RESERVED							TRIGSEL
R-0h							R/W-0h
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED	CHSEL		
R/W-0h				R-0h	R/W-0h		
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 16-54. ADCSOC3CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24-20	TRIGSEL	R/W	0h	SOC3 Trigger Source Select. Along with the SOC3 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC3 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h - 1Fh - Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 16-54. ADCSOC3CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC3 Channel Select. Selects the channel to be converted when SOC3 is received by the ADC. 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC3 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

16.15.3.21 ADCSOC4CTL Register (Offset = 18h) [Reset = 0h]

ADCSOC4CTL is shown in Figure 16-60 and described in Table 16-55.

Return to the Summary Table.

ADC SOC4 Control Register

Figure 16-60. ADCSOC4CTL Register

31	30	29	28	27	26	25	24
RESERVED							TRIGSEL
R-0h							R/W-0h
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED	CHSEL		
R/W-0h				R-0h	R/W-0h		
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 16-55. ADCSOC4CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24-20	TRIGSEL	R/W	0h	SOC4 Trigger Source Select. Along with the SOC4 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC4 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h - 1Fh - Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 16-55. ADCSOC4CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC4 Channel Select. Selects the channel to be converted when SOC4 is received by the ADC. 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC4 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

16.15.3.22 ADCSOC5CTL Register (Offset = 1Ah) [Reset = 0h]

ADCSOC5CTL is shown in Figure 16-61 and described in Table 16-56.

Return to the Summary Table.

ADC SOC5 Control Register

Figure 16-61. ADCSOC5CTL Register

31	30	29	28	27	26	25	24
RESERVED							TRIGSEL
R-0h							R/W-0h
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED	CHSEL		
R/W-0h				R-0h	R/W-0h		
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 16-56. ADCSOC5CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24-20	TRIGSEL	R/W	0h	SOC5 Trigger Source Select. Along with the SOC5 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC5 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h - 1Fh - Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 16-56. ADCSOC5CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC5 Channel Select. Selects the channel to be converted when SOC5 is received by the ADC. 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC5 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

16.15.3.23 ADCSOC6CTL Register (Offset = 1Ch) [Reset = 0h]

ADCSOC6CTL is shown in Figure 16-62 and described in Table 16-57.

Return to the Summary Table.

ADC SOC6 Control Register

Figure 16-62. ADCSOC6CTL Register

31	30	29	28	27	26	25	24
RESERVED							TRIGSEL
R-0h							R/W-0h
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED	CHSEL		
R/W-0h				R-0h	R/W-0h		
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 16-57. ADCSOC6CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24-20	TRIGSEL	R/W	0h	SOC6 Trigger Source Select. Along with the SOC6 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC6 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h - 1Fh - Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 16-57. ADCSOC6CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC6 Channel Select. Selects the channel to be converted when SOC6 is received by the ADC. 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC6 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

16.15.3.24 ADCSOC7CTL Register (Offset = 1Eh) [Reset = 0h]

ADCSOC7CTL is shown in Figure 16-63 and described in Table 16-58.

Return to the Summary Table.

ADC SOC7 Control Register

Figure 16-63. ADCSOC7CTL Register

31	30	29	28	27	26	25	24
RESERVED							TRIGSEL
R-0h							R/W-0h
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED	CHSEL		
R/W-0h				R-0h	R/W-0h		
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 16-58. ADCSOC7CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24-20	TRIGSEL	R/W	0h	SOC7 Trigger Source Select. Along with the SOC7 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC7 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h - 1Fh - Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 16-58. ADCSOC7CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC7 Channel Select. Selects the channel to be converted when SOC7 is received by the ADC. 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC7 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

16.15.3.25 ADCSOC8CTL Register (Offset = 20h) [Reset = 0h]

ADCSOC8CTL is shown in Figure 16-64 and described in Table 16-59.

Return to the Summary Table.

ADC SOC8 Control Register

Figure 16-64. ADCSOC8CTL Register

31	30	29	28	27	26	25	24
RESERVED							TRIGSEL
R-0h							R/W-0h
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED	CHSEL		
R/W-0h				R-0h	R/W-0h		
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 16-59. ADCSOC8CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24-20	TRIGSEL	R/W	0h	SOC8 Trigger Source Select. Along with the SOC8 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC8 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h - 1Fh - Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 16-59. ADCSOC8CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC8 Channel Select. Selects the channel to be converted when SOC8 is received by the ADC. 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC8 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

16.15.3.26 ADCSOC9CTL Register (Offset = 22h) [Reset = 0h]

ADCSOC9CTL is shown in Figure 16-65 and described in Table 16-60.

Return to the Summary Table.

ADC SOC9 Control Register

Figure 16-65. ADCSOC9CTL Register

31	30	29	28	27	26	25	24
RESERVED							TRIGSEL
R-0h							R/W-0h
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED	CHSEL		
R/W-0h				R-0h	R/W-0h		
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 16-60. ADCSOC9CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24-20	TRIGSEL	R/W	0h	SOC9 Trigger Source Select. Along with the SOC9 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC9 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h - 1Fh - Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 16-60. ADCSOC9CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC9 Channel Select. Selects the channel to be converted when SOC9 is received by the ADC. 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC9 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

16.15.3.27 ADCSOC10CTL Register (Offset = 24h) [Reset = 0h]

ADCSOC10CTL is shown in Figure 16-66 and described in Table 16-61.

Return to the Summary Table.

ADC SOC10 Control Register

Figure 16-66. ADCSOC10CTL Register

31	30	29	28	27	26	25	24
RESERVED							TRIGSEL
R-0h							R/W-0h
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED	CHSEL		
R/W-0h				R-0h	R/W-0h		
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 16-61. ADCSOC10CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24-20	TRIGSEL	R/W	0h	SOC10 Trigger Source Select. Along with the SOC10 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC10 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h - 1Fh - Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 16-61. ADCSOC10CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC10 Channel Select. Selects the channel to be converted when SOC10 is received by the ADC. 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC10 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

16.15.3.28 ADCSOC11CTL Register (Offset = 26h) [Reset = 0h]

ADCSOC11CTL is shown in Figure 16-67 and described in Table 16-62.

Return to the Summary Table.

ADC SOC11 Control Register

Figure 16-67. ADCSOC11CTL Register

31	30	29	28	27	26	25	24
RESERVED							TRIGSEL
R-0h							R/W-0h
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED	CHSEL		
R/W-0h				R-0h	R/W-0h		
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 16-62. ADCSOC11CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24-20	TRIGSEL	R/W	0h	SOC11 Trigger Source Select. Along with the SOC11 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC11 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h - 1Fh - Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 16-62. ADCSOC11CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC11 Channel Select. Selects the channel to be converted when SOC11 is received by the ADC. 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC11 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

16.15.3.29 ADCSOC12CTL Register (Offset = 28h) [Reset = 0h]

ADCSOC12CTL is shown in Figure 16-68 and described in Table 16-63.

Return to the Summary Table.

ADC SOC12 Control Register

Figure 16-68. ADCSOC12CTL Register

31	30	29	28	27	26	25	24
RESERVED							TRIGSEL
R-0h							R/W-0h
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED	CHSEL		
R/W-0h				R-0h	R/W-0h		
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 16-63. ADCSOC12CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24-20	TRIGSEL	R/W	0h	SOC12 Trigger Source Select. Along with the SOC12 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC12 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h - 1Fh - Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 16-63. ADCSOC12CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC12 Channel Select. Selects the channel to be converted when SOC12 is received by the ADC. 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC12 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

16.15.3.30 ADCSOC13CTL Register (Offset = 2Ah) [Reset = 0h]

ADCSOC13CTL is shown in Figure 16-69 and described in Table 16-64.

Return to the Summary Table.

ADC SOC13 Control Register

Figure 16-69. ADCSOC13CTL Register

31	30	29	28	27	26	25	24
RESERVED							TRIGSEL
R-0h							R/W-0h
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED	CHSEL		
R/W-0h				R-0h	R/W-0h		
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 16-64. ADCSOC13CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24-20	TRIGSEL	R/W	0h	SOC13 Trigger Source Select. Along with the SOC13 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC13 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h - 1Fh - Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 16-64. ADCSOC13CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC13 Channel Select. Selects the channel to be converted when SOC13 is received by the ADC. 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC13 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

16.15.3.31 ADCSOC14CTL Register (Offset = 2Ch) [Reset = 0h]

ADCSOC14CTL is shown in Figure 16-70 and described in Table 16-65.

Return to the Summary Table.

ADC SOC14 Control Register

Figure 16-70. ADCSOC14CTL Register

31	30	29	28	27	26	25	24
RESERVED							TRIGSEL
R-0h							R/W-0h
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED	CHSEL		
R/W-0h				R-0h	R/W-0h		
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 16-65. ADCSOC14CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24-20	TRIGSEL	R/W	0h	SOC14 Trigger Source Select. Along with the SOC14 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC14 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h - 1Fh - Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 16-65. ADCSOC14CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC14 Channel Select. Selects the channel to be converted when SOC14 is received by the ADC. 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC14 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

16.15.3.32 ADCSOC15CTL Register (Offset = 2Eh) [Reset = 0h]

ADCSOC15CTL is shown in Figure 16-71 and described in Table 16-66.

Return to the Summary Table.

ADC SOC15 Control Register

Figure 16-71. ADCSOC15CTL Register

31	30	29	28	27	26	25	24
RESERVED							TRIGSEL
R-0h							R/W-0h
23	22	21	20	19	18	17	16
TRIGSEL				RESERVED	CHSEL		
R/W-0h				R-0h	R/W-0h		
15	14	13	12	11	10	9	8
CHSEL	RESERVED						ACQPS
R/W-0h	R-0h						R/W-0h
7	6	5	4	3	2	1	0
ACQPS							
R/W-0h							

Table 16-66. ADCSOC15CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24-20	TRIGSEL	R/W	0h	SOC15 Trigger Source Select. Along with the SOC15 field in the ADCINTSOCSEL1 register, this bit field configures which trigger will set the SOC15 flag in the ADCSOCFLG1 register to initiate a conversion to start once priority is given to it. 00h ADCTRIG0 - Software only 01h ADCTRIG1 - CPU1 Timer 0, TINT0n 02h ADCTRIG2 - CPU1 Timer 1, TINT1n 03h ADCTRIG3 - CPU1 Timer 2, TINT2n 04h ADCTRIG4 - GPIO, ADCEXTSOC 05h ADCTRIG5 - ePWM1, ADCSOCA 06h ADCTRIG6 - ePWM1, ADCSOCB 07h ADCTRIG7 - ePWM2, ADCSOCA 08h ADCTRIG8 - ePWM2, ADCSOCB 09h ADCTRIG9 - ePWM3, ADCSOCA 0Ah ADCTRIG10 - ePWM3, ADCSOCB 0Bh ADCTRIG11 - ePWM4, ADCSOCA 0Ch ADCTRIG12 - ePWM4, ADCSOCB 0Dh ADCTRIG13 - ePWM5, ADCSOCA 0Eh ADCTRIG14 - ePWM5, ADCSOCB 0Fh ADCTRIG15 - ePWM6, ADCSOCA 10h ADCTRIG16 - ePWM6, ADCSOCB 11h ADCTRIG17 - ePWM7, ADCSOCA 12h ADCTRIG18 - ePWM7, ADCSOCB 13h ADCTRIG19 - ePWM8, ADCSOCA 14h ADCTRIG20 - ePWM8, ADCSOCB 15h - 1Fh - Reserved Reset type: SYSRSn
19	RESERVED	R	0h	Reserved

Table 16-66. ADCSOC15CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-15	CHSEL	R/W	0h	SOC15 Channel Select. Selects the channel to be converted when SOC15 is received by the ADC. 0h ADCIN0 1h ADCIN1 2h ADCIN2 3h ADCIN3 4h ADCIN4 5h ADCIN5 6h ADCIN6 7h ADCIN7 8h ADCIN8 9h ADCIN9 Ah ADCIN10 Bh ADCIN11 Ch ADCIN12 Dh ADCIN13 Eh ADCIN14 Fh ADCIN15 Reset type: SYSRSn
14-9	RESERVED	R	0h	Reserved
8-0	ACQPS	R/W	0h	SOC15 Acquisition Prescale. Controls the sample and hold window for this SOC. The configured acquisition time must be at least as long as one ADCCLK cycle for correct ADC operation. The device datasheet will also specify a minimum sample and hold window duration. 000h Sample window is 1 system clock cycle wide 001h Sample window is 2 system clock cycles wide 002h Sample window is 3 system clock cycles wide ... 1FFh Sample window is 512 system clock cycles wide Reset type: SYSRSn

16.15.3.33 ADCEVTSTAT Register (Offset = 30h) [Reset = 0h]

ADCEVTSTAT is shown in Figure 16-72 and described in Table 16-67.

Return to the Summary Table.

ADC Event Status Register

Figure 16-72. ADCEVTSTAT Register

15	14	13	12	11	10	9	8
RESERVED	PPB4ZERO	PPB4TRIPLO	PPB4TRIPHI	RESERVED	PPB3ZERO	PPB3TRIPLO	PPB3TRIPHI
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	PPB2ZERO	PPB2TRIPLO	PPB2TRIPHI	RESERVED	PPB1ZERO	PPB1TRIPLO	PPB1TRIPHI
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 16-67. ADCEVTSTAT Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	PPB4ZERO	R	0h	Post Processing Block 4 Zero Crossing Flag. When set indicates the ADCPPB4RESULT register has changed sign. This bit is gated by EOC signal. Note: these bits are set even when the corresponding enable in ADCEVTINTSEL is not set. Because of this, an ISR may need to examine both the ADCEVTSTAT and ADCEVTINTSEL registers to determine the source of the interrupt. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
13	PPB4TRIPLO	R	0h	Post Processing Block 4 Trip Low Flag. When set indicates a digital compare trip low event has occurred. Note: these bits are set even when the corresponding enable in ADCEVTINTSEL is not set. Because of this, an ISR may need to examine both the ADCEVTSTAT and ADCEVTINTSEL registers to determine the source of the interrupt. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
12	PPB4TRIPHI	R	0h	Post Processing Block 4 Trip High Flag. When set indicates a digital compare trip high event has occurred. Note: these bits are set even when the corresponding enable in ADCEVTINTSEL is not set. Because of this, an ISR may need to examine both the ADCEVTSTAT and ADCEVTINTSEL registers to determine the source of the interrupt. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
11	RESERVED	R	0h	Reserved
10	PPB3ZERO	R	0h	Post Processing Block 3 Zero Crossing Flag. When set indicates the ADCPPB3RESULT register has changed sign. This bit is gated by EOC signal. Note: these bits are set even when the corresponding enable in ADCEVTINTSEL is not set. Because of this, an ISR may need to examine both the ADCEVTSTAT and ADCEVTINTSEL registers to determine the source of the interrupt. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn

Table 16-67. ADCEVTSTAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	PPB3TRIPLO	R	0h	Post Processing Block 3 Trip Low Flag. When set indicates a digital compare trip low event has occurred. Note: these bits are set even when the corresponding enable in ADCEVTINTSEL is not set. Because of this, an ISR may need to examine both the ADCEVTSTAT and ADCEVTINTSEL registers to determine the source of the interrupt. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
8	PPB3TRIPHI	R	0h	Post Processing Block 3 Trip High Flag. When set indicates a digital compare trip high event has occurred. Note: these bits are set even when the corresponding enable in ADCEVTINTSEL is not set. Because of this, an ISR may need to examine both the ADCEVTSTAT and ADCEVTINTSEL registers to determine the source of the interrupt. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6	PPB2ZERO	R	0h	Post Processing Block 2 Zero Crossing Flag. When set indicates the ADCPPB2RESULT register has changed sign. This bit is gated by EOC signal. Note: these bits are set even when the corresponding enable in ADCEVTINTSEL is not set. Because of this, an ISR may need to examine both the ADCEVTSTAT and ADCEVTINTSEL registers to determine the source of the interrupt. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
5	PPB2TRIPLO	R	0h	Post Processing Block 2 Trip Low Flag. When set indicates a digital compare trip low event has occurred. Note: these bits are set even when the corresponding enable in ADCEVTINTSEL is not set. Because of this, an ISR may need to examine both the ADCEVTSTAT and ADCEVTINTSEL registers to determine the source of the interrupt. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
4	PPB2TRIPHI	R	0h	Post Processing Block 2 Trip High Flag. When set indicates a digital compare trip high event has occurred. Note: these bits are set even when the corresponding enable in ADCEVTINTSEL is not set. Because of this, an ISR may need to examine both the ADCEVTSTAT and ADCEVTINTSEL registers to determine the source of the interrupt. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
3	RESERVED	R	0h	Reserved
2	PPB1ZERO	R	0h	Post Processing Block 1 Zero Crossing Flag. When set indicates the ADCPPB1RESULT register has changed sign. This bit is gated by EOC signal. Note: these bits are set even when the corresponding enable in ADCEVTINTSEL is not set. Because of this, an ISR may need to examine both the ADCEVTSTAT and ADCEVTINTSEL registers to determine the source of the interrupt. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn

Table 16-67. ADCEVTSTAT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	PPB1TRIPLO	R	0h	Post Processing Block 1 Trip Low Flag. When set indicates a digital compare trip low event has occurred. Note: these bits are set even when the corresponding enable in ADCEVTINTSEL is not set. Because of this, an ISR may need to examine both the ADCEVTSTAT and ADCEVTINTSEL registers to determine the source of the interrupt. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
0	PPB1TRIPHI	R	0h	Post Processing Block 1 Trip High Flag. When set indicates a digital compare trip high event has occurred. Note: these bits are set even when the corresponding enable in ADCEVTINTSEL is not set. Because of this, an ISR may need to examine both the ADCEVTSTAT and ADCEVTINTSEL registers to determine the source of the interrupt. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn

16.15.3.34 ADCEVTCLR Register (Offset = 32h) [Reset = 0h]

ADCEVTCLR is shown in Figure 16-73 and described in Table 16-68.

Return to the Summary Table.

ADC Event Clear Register

Figure 16-73. ADCEVTCLR Register

15	14	13	12	11	10	9	8
RESERVED	PPB4ZERO	PPB4TRIPLO	PPB4TRIPHI	RESERVED	PPB3ZERO	PPB3TRIPLO	PPB3TRIPHI
R-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
RESERVED	PPB2ZERO	PPB2TRIPLO	PPB2TRIPHI	RESERVED	PPB1ZERO	PPB1TRIPLO	PPB1TRIPHI
R-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 16-68. ADCEVTCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	PPB4ZERO	R-0/W1S	0h	Post Processing Block 4 Zero Crossing Clear. Clears the corresponding zero crossing flag in the ADCEVTSTAT register. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
13	PPB4TRIPLO	R-0/W1S	0h	Post Processing Block 4 Trip Low Clear. Clears the corresponding trip low flag in the ADCEVTSTAT register. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
12	PPB4TRIPHI	R-0/W1S	0h	Post Processing Block 4 Trip High Clear. Clears the corresponding trip high flag in the ADCEVTSTAT register. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
11	RESERVED	R	0h	Reserved
10	PPB3ZERO	R-0/W1S	0h	Post Processing Block 3 Zero Crossing Clear. Clears the corresponding zero crossing flag in the ADCEVTSTAT register. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
9	PPB3TRIPLO	R-0/W1S	0h	Post Processing Block 3 Trip Low Clear. Clears the corresponding trip low flag in the ADCEVTSTAT register. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
8	PPB3TRIPHI	R-0/W1S	0h	Post Processing Block 3 Trip High Clear. Clears the corresponding trip high flag in the ADCEVTSTAT register. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6	PPB2ZERO	R-0/W1S	0h	Post Processing Block 2 Zero Crossing Clear. Clears the corresponding zero crossing flag in the ADCEVTSTAT register. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn

Table 16-68. ADCEVTCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	PPB2TRIPLO	R-0/W1S	0h	Post Processing Block 2 Trip Low Clear. Clears the corresponding trip low flag in the ADCEVTSTAT register. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
4	PPB2TRIPHI	R-0/W1S	0h	Post Processing Block 2 Trip High Clear. Clears the corresponding trip high flag in the ADCEVTSTAT register. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
3	RESERVED	R	0h	Reserved
2	PPB1ZERO	R-0/W1S	0h	Post Processing Block 1 Zero Crossing Clear. Clears the corresponding zero crossing flag in the ADCEVTSTAT register. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
1	PPB1TRIPLO	R-0/W1S	0h	Post Processing Block 1 Trip Low Clear. Clears the corresponding trip low flag in the ADCEVTSTAT register. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn
0	PPB1TRIPHI	R-0/W1S	0h	Post Processing Block 1 Trip High Clear. Clears the corresponding trip high flag in the ADCEVTSTAT register. Note: If software sets the clear bit on the same cycle that hardware is trying to set the flag bit, then hardware has priority Reset type: SYSRSn

16.15.3.35 ADCEVTSEL Register (Offset = 34h) [Reset = 0h]

ADCEVTSEL is shown in Figure 16-74 and described in Table 16-69.

Return to the Summary Table.

ADC Event Selection Register

Figure 16-74. ADCEVTSEL Register

15	14	13	12	11	10	9	8
RESERVED	PPB4ZERO	PPB4TRIPLO	PPB4TRIPHI	RESERVED	PPB3ZERO	PPB3TRIPLO	PPB3TRIPHI
R-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	PPB2ZERO	PPB2TRIPLO	PPB2TRIPHI	RESERVED	PPB1ZERO	PPB1TRIPLO	PPB1TRIPHI
R-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h

Table 16-69. ADCEVTSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	PPB4ZERO	R/W	0h	Post Processing Block 4 Zero Crossing Event Enable. Setting this bit allows the corresponding rising zero crossing flag to activate the event signal to the PWM blocks. The flag must be cleared before it can produce additional events to the PWM blocks. Reset type: SYSRSn
13	PPB4TRIPLO	R/W	0h	Post Processing Block 4 Trip Low Event Enable. Setting this bit allows the corresponding rising trip low flag to activate the event signal to the PWM blocks. The flag must be cleared before it can produce additional events to the PWM blocks. Reset type: SYSRSn
12	PPB4TRIPHI	R/W	0h	Post Processing Block 4 Trip High Event Enable. Setting this bit allows the corresponding rising trip high flag to activate the event signal to the PWM blocks. The flag must be cleared before it can produce additional events to the PWM blocks. Reset type: SYSRSn
11	RESERVED	R	0h	Reserved
10	PPB3ZERO	R/W	0h	Post Processing Block 3 Zero Crossing Event Enable. Setting this bit allows the corresponding rising zero crossing flag to activate the event signal to the PWM blocks. The flag must be cleared before it can produce additional events to the PWM blocks. Reset type: SYSRSn
9	PPB3TRIPLO	R/W	0h	Post Processing Block 3 Trip Low Event Enable. Setting this bit allows the corresponding rising trip low flag to activate the event signal to the PWM blocks. The flag must be cleared before it can produce additional events to the PWM blocks. Reset type: SYSRSn
8	PPB3TRIPHI	R/W	0h	Post Processing Block 3 Trip High Event Enable. Setting this bit allows the corresponding rising trip high flag to activate the event signal to the PWM blocks. The flag must be cleared before it can produce additional events to the PWM blocks. Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6	PPB2ZERO	R/W	0h	Post Processing Block 2 Zero Crossing Event Enable. Setting this bit allows the corresponding rising zero crossing flag to activate the event signal to the PWM blocks. The flag must be cleared before it can produce additional events to the PWM blocks. Reset type: SYSRSn

Table 16-69. ADCEVTSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	PPB2TRIPLO	R/W	0h	Post Processing Block 2 Trip Low Event Enable. Setting this bit allows the corresponding rising trip low flag to activate the event signal to the PWM blocks. The flag must be cleared before it can produce additional events to the PWM blocks. Reset type: SYSRSn
4	PPB2TRIPHI	R/W	0h	Post Processing Block 2 Trip High Event Enable. Setting this bit allows the corresponding rising trip high flag to activate the event signal to the PWM blocks. The flag must be cleared before it can produce additional events to the PWM blocks. Reset type: SYSRSn
3	RESERVED	R	0h	Reserved
2	PPB1ZERO	R/W	0h	Post Processing Block 1 Zero Crossing Event Enable. Setting this bit allows the corresponding rising zero crossing flag to activate the event signal to the PWM blocks. The flag must be cleared before it can produce additional events to the PWM blocks. Reset type: SYSRSn
1	PPB1TRIPLO	R/W	0h	Post Processing Block 1 Trip Low Event Enable. Setting this bit allows the corresponding rising trip low flag to activate the event signal to the PWM blocks. The flag must be cleared before it can produce additional events to the PWM blocks. Reset type: SYSRSn
0	PPB1TRIPHI	R/W	0h	Post Processing Block 1 Trip High Event Enable. Setting this bit allows the corresponding rising trip high flag to activate the event signal to the PWM blocks. The flag must be cleared before it can produce additional events to the PWM blocks. Reset type: SYSRSn

16.15.3.36 ADCEVTINTSEL Register (Offset = 36h) [Reset = 0h]

ADCEVTINTSEL is shown in Figure 16-75 and described in Table 16-70.

Return to the Summary Table.

ADC Event Interrupt Selection Register

Figure 16-75. ADCEVTINTSEL Register

15	14	13	12	11	10	9	8
RESERVED	PPB4ZERO	PPB4TRIPLO	PPB4TRIPHI	RESERVED	PPB3ZERO	PPB3TRIPLO	PPB3TRIPHI
R-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	PPB2ZERO	PPB2TRIPLO	PPB2TRIPHI	RESERVED	PPB1ZERO	PPB1TRIPLO	PPB1TRIPHI
R-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h

Table 16-70. ADCEVTINTSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	PPB4ZERO	R/W	0h	Post Processing Block 4 Zero Crossing Interrupt Enable. Setting this bit allows the corresponding rising zero crossing flag to activate the event interrupt signal to the PIE. The flag must be cleared before it can produce additional interrupts to the PIE. Reset type: SYSRSn
13	PPB4TRIPLO	R/W	0h	Post Processing Block 4 Trip Low Interrupt Enable. Setting this bit allows the corresponding rising trip low flag to activate the event interrupt signal to the PIE. The flag must be cleared before it can produce additional interrupts to the PIE. Reset type: SYSRSn
12	PPB4TRIPHI	R/W	0h	Post Processing Block 4 Trip High Interrupt Enable. Setting this bit allows the corresponding rising trip high flag to activate the event interrupt signal to the PIE. The flag must be cleared before it can produce additional interrupts to the PIE. Reset type: SYSRSn
11	RESERVED	R	0h	Reserved
10	PPB3ZERO	R/W	0h	Post Processing Block 3 Zero Crossing Interrupt Enable. Setting this bit allows the corresponding rising zero crossing flag to activate the event interrupt signal to the PIE. The flag must be cleared before it can produce additional interrupts to the PIE. Reset type: SYSRSn
9	PPB3TRIPLO	R/W	0h	Post Processing Block 3 Trip Low Interrupt Enable. Setting this bit allows the corresponding rising trip low flag to activate the event interrupt signal to the PIE. The flag must be cleared before it can produce additional interrupts to the PIE. Reset type: SYSRSn
8	PPB3TRIPHI	R/W	0h	Post Processing Block 3 Trip High Interrupt Enable. Setting this bit allows the corresponding rising trip high flag to activate the event interrupt signal to the PIE. The flag must be cleared before it can produce additional interrupts to the PIE. Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6	PPB2ZERO	R/W	0h	Post Processing Block 2 Zero Crossing Interrupt Enable. Setting this bit allows the corresponding rising zero crossing flag to activate the event interrupt signal to the PIE. The flag must be cleared before it can produce additional interrupts to the PIE. Reset type: SYSRSn

Table 16-70. ADCEVTINTSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	PPB2TRIPLO	R/W	0h	Post Processing Block 2 Trip Low Interrupt Enable. Setting this bit allows the corresponding rising trip low flag to activate the event interrupt signal to the PIE. The flag must be cleared before it can produce additional interrupts to the PIE. Reset type: SYSRSn
4	PPB2TRIPHI	R/W	0h	Post Processing Block 2 Trip High Interrupt Enable. Setting this bit allows the corresponding rising trip high flag to activate the event interrupt signal to the PIE. The flag must be cleared before it can produce additional interrupts to the PIE. Reset type: SYSRSn
3	RESERVED	R	0h	Reserved
2	PPB1ZERO	R/W	0h	Post Processing Block 1 Zero Crossing Interrupt Enable. Setting this bit allows the corresponding rising zero crossing flag to activate the event interrupt signal to the PIE. The flag must be cleared before it can produce additional interrupts to the PIE. Reset type: SYSRSn
1	PPB1TRIPLO	R/W	0h	Post Processing Block 1 Trip Low Interrupt Enable. Setting this bit allows the corresponding rising trip low flag to activate the event interrupt signal to the PIE. The flag must be cleared before it can produce additional interrupts to the PIE. Reset type: SYSRSn
0	PPB1TRIPHI	R/W	0h	Post Processing Block 1 Trip High Interrupt Enable. Setting this bit allows the corresponding rising trip high flag to activate the event interrupt signal to the PIE. The flag must be cleared before it can produce additional interrupts to the PIE. Reset type: SYSRSn

16.15.3.37 ADCOSDETECT Register (Offset = 38h) [Reset = 0h]

ADCOSDETECT is shown in Figure 16-76 and described in Table 16-71.

Return to the Summary Table.

ADC Open and Shorts Detect Register

Figure 16-76. ADCOSDETECT Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					DETECTCFG		
R-0h					R/W-0h		

Table 16-71. ADCOSDETECT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-3	RESERVED	R	0h	Reserved
2-0	DETECTCFG	R/W	0h	ADC Opens and Shorts Detect Configuration. This bit field defines the open/shorts detection circuit state. 0h Open/Shorts detection circuit is disabled. 1h Open/Shorts detection circuit is enabled at zero scale. 2h Open/Shorts detection circuit is enabled at full scale. 3h Open/Shorts detection circuit is enabled at (nominal) 5/12 scale. 4h Open/Shorts detection circuit is enabled at (nominal) 7/12 scale. 5h Open/Shorts detection circuit is enabled with a (nominal) 5K pulldown to VSSA. 6h Open/Shorts detection circuit is enabled with a (nominal) 5K pullup to VDDA. 7h Open/Shorts detection circuit is enabled with a (nominal) 7K pulldown to VSSA. Reset type: SYSRSn

16.15.3.38 ADCCOUNTER Register (Offset = 39h) [Reset = 0h]

ADCCOUNTER is shown in Figure 16-77 and described in Table 16-72.

Return to the Summary Table.

ADC Counter Register

Figure 16-77. ADCCOUNTER Register

15	14	13	12	11	10	9	8
RESERVED				FREECOUNT			
R-0h				R-0h			
7	6	5	4	3	2	1	0
FREECOUNT							
R-0h							

Table 16-72. ADCCOUNTER Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-0	FREECOUNT	R	0h	ADC Free Running Counter Value. This bit field reflects the status of the free running ADC counter. Reset type: SYSRSn

16.15.3.39 ADCREV Register (Offset = 3Ah) [Reset = 5h]

ADCREV is shown in Figure 16-78 and described in Table 16-73.

Return to the Summary Table.

ADC Revision Register

Figure 16-78. ADCREV Register

15	14	13	12	11	10	9	8
REV							
R-0h							
7	6	5	4	3	2	1	0
TYPE							
R-5h							

Table 16-73. ADCREV Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	REV	R	0h	ADC Revision. To allow documentation of differences between revisions. First version is labeled as 00h. Reset type: SYSRSn
7-0	TYPE	R	5h	ADC Type. Always set to 5 for this ADC. Reset type: SYSRSn

16.15.3.40 ADCOFFTRIM Register (Offset = 3Bh) [Reset = 0h]

ADCOFFTRIM is shown in Figure 16-79 and described in Table 16-74.

Return to the Summary Table.

ADC Offset Trim Register

Figure 16-79. ADCOFFTRIM Register

15	14	13	12	11	10	9	8
RESERVED				RESERVED			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
OFFTRIM							
R/W-0h							

Table 16-74. ADCOFFTRIM Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-8	RESERVED	R/W	0h	Reserved
7-0	OFFTRIM	R/W	0h	ADC Offset Trim Adjusts the conversion results of the converter up or down to account for offset error in the ADC. A factory trim setting will be loaded during device boot. Offset can be corrected in the range of +7 to -8 LSBs. Value is 16*Offset in 8-bit 2's complement: 7 LSB (16*7) = 112 6 LSB (16*6) = 96 5 LSB (16*5) = 80 4 LSB (16*4) = 64 3 LSB (16*3) = 48 2 LSB (16*2) = 32 1 LSB (16*1) = 16 0 LSB (16*0) = 0 -1 LSB (16*(-1)) = 240 : : -7LSB(16*(-7)) = 144 Reset type: SYSRSn

16.15.3.41 ADCPPB1CONFIG Register (Offset = 40h) [Reset = 0h]

ADCPPB1CONFIG is shown in Figure 16-80 and described in Table 16-75.

Return to the Summary Table.

ADC PPB1 Config Register

Figure 16-80. ADCPPB1CONFIG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		CBCEN	TWOSCOMPEN	CONFIG			
R-0h		R/W-0h	R/W-0h	R/W-0h			

Table 16-75. ADCPPB1CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-6	RESERVED	R	0h	Reserved
5	CBCEN	R/W	0h	ADC Post Processing Block Cycle By Cycle Enable. When set, this bit enables the post conversion hardware processing circuit to automatically clear the ADCEVTSTAT on a conversion if the event condition is no longer present. Reset type: SYSRSn
4	TWOSCOMPEN	R/W	0h	ADC Post Processing Block 1 Two's Complement Enable. When set this bit enables the post conversion hardware processing circuit that performs a two's complement on the output of the offset/reference subtraction unit before storing the result in the ADCPPB1RESULT register. 0 ADCPPB1RESULT = ADCRESULTx - ADCPPB1OFFREF 1 ADCPPB1RESULT = ADCPPB1OFFREF - ADCRESULTx Reset type: SYSRSn

Table 16-75. ADCPPB1CONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	CONFIG	R/W	0h	ADC Post Processing Block 1 Configuration. This bit field defines which SOC/EOC/RESULT is assocatied with this post processing block. 0000 SOC0/EOC0/RESULT0 is associated with post processing block 1 0001 SOC1/EOC1/RESULT1 is associated with post processing block 1 0010 SOC2/EOC2/RESULT2 is associated with post processing block 1 0011 SOC3/EOC3/RESULT3 is associated with post processing block 1 0100 SOC4/EOC4/RESULT4 is associated with post processing block 1 0101 SOC5/EOC5/RESULT5 is associated with post processing block 1 0110 SOC6/EOC6/RESULT6 is associated with post processing block 1 0111 SOC7/EOC7/RESULT7 is associated with post processing block 1 1000 SOC8/EOC8/RESULT8 is associated with post processing block 1 1001 SOC9/EOC9/RESULT9 is associated with post processing block 1 1010 SOC10/EOC10/RESULT10 is associated with post processing block 1 1011 SOC11/EOC11/RESULT11 is associated with post processing block 1 1100 SOC12/EOC12/RESULT12 is associated with post processing block 1 1101 SOC13/EOC13/RESULT13 is associated with post processing block 1 1110 SOC14/EOC14/RESULT14 is associated with post processing block 1 1111 SOC15/EOC15/RESULT15 is associated with post processing block 1 Reset type: SYSRSn

16.15.3.42 ADCPPB1STAMP Register (Offset = 41h) [Reset = 0h]

ADCPPB1STAMP is shown in Figure 16-81 and described in Table 16-76.

Return to the Summary Table.

ADC PPB1 Sample Delay Time Stamp Register

Figure 16-81. ADCPPB1STAMP Register

15	14	13	12	11	10	9	8
RESERVED				DLYSTAMP			
R-0h				R-0h			
7	6	5	4	3	2	1	0
DLYSTAMP							
R-0h							

Table 16-76. ADCPPB1STAMP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-0	DLYSTAMP	R	0h	ADC Post Processing Block 1 Delay Time Stamp. When an SOC starts sampling the value contained in REQSTAMP is subtracted from the value in ADCCOUNTER.FREECOUNT and loaded into this bit field, thereby giving the number of system clock cycles delay between the SOC trigger and the actual start of the sample. Reset type: SYSRSn

16.15.3.43 ADCPPB1OFFCAL Register (Offset = 42h) [Reset = 0h]

ADCPPB1OFFCAL is shown in Figure 16-82 and described in Table 16-77.

Return to the Summary Table.

ADC PPB1 Offset Calibration Register

Figure 16-82. ADCPPB1OFFCAL Register

15	14	13	12	11	10	9	8
RESERVED						OFFCAL	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
OFFCAL							
R/W-0h							

Table 16-77. ADCPPB1OFFCAL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	OFFCAL	R/W	0h	ADC Post Processing Block 1 Offset Correction. This bit field can be used to digitally remove any system level offset inherent in the ADCIN circuit. This 10-bit signed value is subtracted from the ADC output before being stored in the ADCRESULT register. 000h No change. The ADC output is stored directly into ADCRESULT. 001h ADC output - 1 is stored into ADCRESULT. 002h ADC output - 2 is stored into ADCRESULT. ... 200h ADC output + 512 is stored into ADCRESULT. ... 3FFh ADC output + 1 is stored into ADCRESULT. NOTE: In 16-bit mode, the subtraction will saturate at 0000h and FFFFh before being stored into the ADCRESULT register. In 12-bit mode, the subtraction will saturate at 0000h and 0FFFh before being stored into the ADCRESULT register. Note: in the case that multiple PPBs point to the same SOC, only the OFFCAL of the highest numbered PPB will be applied. Reset type: SYSRSn

16.15.3.44 ADCPPB1OFFREF Register (Offset = 43h) [Reset = 0h]

ADCPPB1OFFREF is shown in Figure 16-83 and described in Table 16-78.

Return to the Summary Table.

ADC PPB1 Offset Reference Register

Figure 16-83. ADCPPB1OFFREF Register

15	14	13	12	11	10	9	8
OFFREF							
R/W-0h							
7	6	5	4	3	2	1	0
OFFREF							
R/W-0h							

Table 16-78. ADCPPB1OFFREF Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	OFFREF	R/W	0h	ADC Post Processing Block 1 Offset Correction. This bit field can be used to either calculate the feedback error or convert a unipolar signal to bipolar by subtracting a reference value. This 16-bit unsigned value is subtracted from the ADCRESULT register before being passed through an optional two's complement function and stored in the ADCPPB1RESULT register. This subtraction is not saturated. 0000h No change. The ADCRESULT value is passed on. 0001h ADCRESULT - 1 is passed on. 0002h ADCRESULT - 2 is passed on. ... 8000h ADCRESULT - 32,768 is passed on. ... FFFFh ADCRESULT - 65,535 is passed on. NOTE: In 12-bit mode the size of this register does not change from 16-bits. It is the user's responsibility to ensure that only a 12-bit value is written to this register when in 12-bit mode. Reset type: SYSRSn

16.15.3.45 ADCPPB1TRIPHI Register (Offset = 44h) [Reset = 0h]

ADCPPB1TRIPHI is shown in Figure 16-84 and described in Table 16-79.

Return to the Summary Table.

ADC PPB1 Trip High Register

Figure 16-84. ADCPPB1TRIPHI Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							HSIGN
R-0h							R/W-0h
15	14	13	12	11	10	9	8
LIMITHI							
R/W-0h							
7	6	5	4	3	2	1	0
LIMITHI							
R/W-0h							

Table 16-79. ADCPPB1TRIPHI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16	HSIGN	R/W	0h	High Limit Sign Bit. This is the sign bit (17th bit) to the LIMITHI bit field when in 16-bit ADC mode. Reset type: SYSRSn
15-0	LIMITHI	R/W	0h	ADC Post Processing Block 1 Trip High Limit. This value sets the digital comparator trip high limit. In 16-bit mode all 17 bits will be compared against the 17 bits of the PPBRESULT bit field of the ADCPPB1RESULT register. In 12-bit mode bits 12:0 will be compared against bits 12:0 of the PPBRESULT bit field of the ADCPPB1RESULT register. Reset type: SYSRSn

16.15.3.46 ADCPPB1TRIPLO Register (Offset = 46h) [Reset = 0h]

ADCPPB1TRIPLO is shown in Figure 16-85 and described in Table 16-80.

Return to the Summary Table.

ADC PPB1 Trip Low/Trigger Time Stamp Register

Figure 16-85. ADCPPB1TRIPLO Register

31	30	29	28	27	26	25	24
REQSTAMP							
R-0h							
23	22	21	20	19	18	17	16
REQSTAMP				RESERVED			LSIGN
R-0h				R-0h			R/W-0h
15	14	13	12	11	10	9	8
LIMITLO							
R/W-0h							
7	6	5	4	3	2	1	0
LIMITLO							
R/W-0h							

Table 16-80. ADCPPB1TRIPLO Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	REQSTAMP	R	0h	ADC Post Processing Block 1 Request Time Stamp. When a trigger sets the associated SOC flag in the ADCSOCFLG1 register the value of ADCCOUNTER.FREECOUNT is loaded into this bit field. Reset type: SYSRSn
19-17	RESERVED	R	0h	Reserved
16	LSIGN	R/W	0h	Low Limit Sign Bit. This is the sign bit (17th bit) to the LIMITLO bit field when in 16-bit ADC mode. Reset type: SYSRSn
15-0	LIMITLO	R/W	0h	ADC Post Processing Block 1 Trip Low Limit. This value sets the digital comparator trip low limit. In 16-bit mode all 17 bits will be compared against the 17 bits of the PPBRESULT bit field of the ADCPPB1RESULT register. In 12-bit mode bits 12:0 will be compared against bits 12:0 of the PPBRSULT bit field of the ADCPPB1RESULT register. Reset type: SYSRSn

16.15.3.47 ADCPPB2CONFIG Register (Offset = 48h) [Reset = 0h]

ADCPPB2CONFIG is shown in Figure 16-86 and described in Table 16-81.

Return to the Summary Table.

ADC PPB2 Config Register

Figure 16-86. ADCPPB2CONFIG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		CBCEN	TWOSCOMPEN	CONFIG			
R-0h		R/W-0h	R/W-0h	R/W-0h			

Table 16-81. ADCPPB2CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-6	RESERVED	R	0h	Reserved
5	CBCEN	R/W	0h	ADC Post Processing Block Cycle By Cycle Enable. When set, this bit enables the post conversion hardware processing circuit to automatically clear the ADCEVTSTAT on a conversion if the event condition is no longer present. Reset type: SYSRSn
4	TWOSCOMPEN	R/W	0h	ADC Post Processing Block 2 Two's Complement Enable. When set this bit enables the post conversion hardware processing circuit that performs a two's complement on the output of the offset/reference subtraction unit before storing the result in the ADCPPB2RESULT register. 0 ADCPPB2RESULT = ADCRESULTx - ADCPPB2OFFREF 1 ADCPPB2RESULT = ADCPPB2OFFREF - ADCRESULTx Reset type: SYSRSn

Table 16-81. ADCPPB2CONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	CONFIG	R/W	0h	ADC Post Processing Block 2 Configuration. This bit field defines which SOC/EOC/RESULT is assocatied with this post processing block. 0000 SOC0/EOC0/RESULT0 is associated with post processing block 2 0001 SOC1/EOC1/RESULT1 is associated with post processing block 2 0010 SOC2/EOC2/RESULT2 is associated with post processing block 2 0011 SOC3/EOC3/RESULT3 is associated with post processing block 2 0100 SOC4/EOC4/RESULT4 is associated with post processing block 2 0101 SOC5/EOC5/RESULT5 is associated with post processing block 2 0110 SOC6/EOC6/RESULT6 is associated with post processing block 2 0111 SOC7/EOC7/RESULT7 is associated with post processing block 2 1000 SOC8/EOC8/RESULT8 is associated with post processing block 2 1001 SOC9/EOC9/RESULT9 is associated with post processing block 2 1010 SOC10/EOC10/RESULT10 is associated with post processing block 2 1011 SOC11/EOC11/RESULT11 is associated with post processing block 2 1100 SOC12/EOC12/RESULT12 is associated with post processing block 2 1101 SOC13/EOC13/RESULT13 is associated with post processing block 2 1110 SOC14/EOC14/RESULT14 is associated with post processing block 2 1111 SOC15/EOC15/RESULT15 is associated with post processing block 2 Reset type: SYSRSn

16.15.3.48 ADCPPB2STAMP Register (Offset = 49h) [Reset = 0h]

ADCPPB2STAMP is shown in Figure 16-87 and described in Table 16-82.

Return to the Summary Table.

ADC PPB2 Sample Delay Time Stamp Register

Figure 16-87. ADCPPB2STAMP Register

15	14	13	12	11	10	9	8
RESERVED				DLYSTAMP			
R-0h				R-0h			
7	6	5	4	3	2	1	0
DLYSTAMP							
R-0h							

Table 16-82. ADCPPB2STAMP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-0	DLYSTAMP	R	0h	ADC Post Processing Block 2 Delay Time Stamp. When an SOC starts sampling the value contained in REQSTAMP is subtracted from the value in ADCCOUNTER.FREECOUNT and loaded into this bit field, thereby giving the number of system clock cycles delay between the SOC trigger and the actual start of the sample. Reset type: SYSRSn

16.15.3.49 ADCPPB2OFFCAL Register (Offset = 4Ah) [Reset = 0h]

ADCPPB2OFFCAL is shown in Figure 16-88 and described in Table 16-83.

Return to the Summary Table.

ADC PPB2 Offset Calibration Register

Figure 16-88. ADCPPB2OFFCAL Register

15	14	13	12	11	10	9	8
RESERVED						OFFCAL	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
OFFCAL							
R/W-0h							

Table 16-83. ADCPPB2OFFCAL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	OFFCAL	R/W	0h	ADC Post Processing Block 2 Offset Correction. This bit field can be used to digitally remove any system level offset inherent in the ADCIN circuit. This 10-bit signed value is subtracted from the ADC output before being stored in the ADCRESULT register. 000h No change. The ADC output is stored directly into ADCRESULT. 001h ADC output - 1 is stored into ADCRESULT. 002h ADC output - 2 is stored into ADCRESULT. ... 200h ADC output + 512 is stored into ADCRESULT. ... 3FFh ADC output + 1 is stored into ADCRESULT. NOTE: In 16-bit mode, the subtraction will saturate at 0000h and FFFFh before being stored into the ADCRESULT register. In 12-bit mode, the subtraction will saturate at 0000h and 0FFFh before being stored into the ADCRESULT register. Note: in the case that multiple PPBs point to the same SOC, only the OFFCAL of the highest numbered PPB will be applied. Reset type: SYSRSn

16.15.3.50 ADCPPB2OFFREF Register (Offset = 4Bh) [Reset = 0h]

ADCPPB2OFFREF is shown in Figure 16-89 and described in Table 16-84.

Return to the Summary Table.

ADC PPB2 Offset Reference Register

Figure 16-89. ADCPPB2OFFREF Register

15	14	13	12	11	10	9	8
OFFREF							
R/W-0h							
7	6	5	4	3	2	1	0
OFFREF							
R/W-0h							

Table 16-84. ADCPPB2OFFREF Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	OFFREF	R/W	0h	ADC Post Processing Block 2 Offset Correction. This bit field can be used to either calculate the feedback error or convert a unipolar signal to bipolar by subtracting a reference value. This 16-bit unsigned value is subtracted from the ADCRESULT register before being passed through an optional two's complement function and stored in the ADCPPB2RESULT register. This subtraction is not saturated. 0000h No change. The ADCRESULT value is passed on. 0001h ADCRESULT - 1 is passed on. 0002h ADCRESULT - 2 is passed on. ... 8000h ADCRESULT - 32,768 is passed on. ... FFFFh ADCRESULT - 65,535 is passed on. NOTE: In 12-bit mode the size of this register does not change from 16-bits. It is the user's responsibility to ensure that only a 12-bit value is written to this register when in 12-bit mode. Reset type: SYSRSn

16.15.3.51 ADCPPB2TRIPHI Register (Offset = 4Ch) [Reset = 0h]

ADCPPB2TRIPHI is shown in Figure 16-90 and described in Table 16-85.

Return to the Summary Table.

ADC PPB2 Trip High Register

Figure 16-90. ADCPPB2TRIPHI Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							HSIGN
R-0h							R/W-0h
15	14	13	12	11	10	9	8
LIMITHI							
R/W-0h							
7	6	5	4	3	2	1	0
LIMITHI							
R/W-0h							

Table 16-85. ADCPPB2TRIPHI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16	HSIGN	R/W	0h	High Limit Sign Bit. This is the sign bit (17th bit) to the LIMITHI bit field when in 16-bit ADC mode. Reset type: SYSRSn
15-0	LIMITHI	R/W	0h	ADC Post Processing Block 2 Trip High Limit. This value sets the digital comparator trip high limit. In 16-bit mode all 17 bits will be compared against the 17 bits of the PPBRESULT bit field of the ADCPPB2RESULT register. In 12-bit mode bits 12:0 will be compared against bits 12:0 of the PPBRESULT bit field of the ADCPPB2RESULT register. Reset type: SYSRSn

16.15.3.52 ADCPPB2TRIPLO Register (Offset = 4Eh) [Reset = 0h]

ADCPPB2TRIPLO is shown in Figure 16-91 and described in Table 16-86.

Return to the Summary Table.

ADC PPB2 Trip Low/Trigger Time Stamp Register

Figure 16-91. ADCPPB2TRIPLO Register

31	30	29	28	27	26	25	24
REQSTAMP							
R-0h							
23	22	21	20	19	18	17	16
REQSTAMP				RESERVED			LSIGN
R-0h				R-0h			R/W-0h
15	14	13	12	11	10	9	8
LIMITLO							
R/W-0h							
7	6	5	4	3	2	1	0
LIMITLO							
R/W-0h							

Table 16-86. ADCPPB2TRIPLO Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	REQSTAMP	R	0h	ADC Post Processing Block 2 Request Time Stamp. When a trigger sets the associated SOC flag in the ADCSOCFLG1 register the value of ADCCOUNTER.FREECOUNT is loaded into this bit field. Reset type: SYSRSn
19-17	RESERVED	R	0h	Reserved
16	LSIGN	R/W	0h	Low Limit Sign Bit. This is the sign bit (17th bit) to the LIMITLO bit field when in 16-bit ADC mode. Reset type: SYSRSn
15-0	LIMITLO	R/W	0h	ADC Post Processing Block 2 Trip Low Limit. This value sets the digital comparator trip low limit. In 16-bit mode all 17 bits will be compared against the 17 bits of the PPBRESULT bit field of the ADCPPB2RESULT register. In 12-bit mode bits 12:0 will be compared against bits 12:0 of the PPBRSULT bit field of the ADCPPB2RESULT register. Reset type: SYSRSn

16.15.3.53 ADCPPB3CONFIG Register (Offset = 50h) [Reset = 0h]

ADCPPB3CONFIG is shown in Figure 16-92 and described in Table 16-87.

Return to the Summary Table.

ADC PPB3 Config Register

Figure 16-92. ADCPPB3CONFIG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		CBCEN	TWOSCOMPEN	CONFIG			
R-0h		R/W-0h	R/W-0h	R/W-0h			

Table 16-87. ADCPPB3CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-6	RESERVED	R	0h	Reserved
5	CBCEN	R/W	0h	ADC Post Processing Block Cycle By Cycle Enable. When set, this bit enables the post conversion hardware processing circuit to automatically clear the ADCEVTSTAT on a conversion if the event condition is no longer present. Reset type: SYSRSn
4	TWOSCOMPEN	R/W	0h	ADC Post Processing Block 3 Two's Complement Enable. When set this bit enables the post conversion hardware processing circuit that performs a two's complement on the output of the offset/reference subtraction unit before storing the result in the ADCPPB3RESULT register. 0 ADCPPB3RESULT = ADCRESULTx - ADCPPB3OFFREF 1 ADCPPB3RESULT = ADCPPB3OFFREF - ADCRESULTx Reset type: SYSRSn

Table 16-87. ADCPPB3CONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	CONFIG	R/W	0h	ADC Post Processing Block 3 Configuration. This bit field defines which SOC/EOC/RESULT is assocatied with this post processing block. 0000 SOC0/EOC0/RESULT0 is associated with post processing block 3 0001 SOC1/EOC1/RESULT1 is associated with post processing block 3 0010 SOC2/EOC2/RESULT2 is associated with post processing block 3 0011 SOC3/EOC3/RESULT3 is associated with post processing block 3 0100 SOC4/EOC4/RESULT4 is associated with post processing block 3 0101 SOC5/EOC5/RESULT5 is associated with post processing block 3 0110 SOC6/EOC6/RESULT6 is associated with post processing block 3 0111 SOC7/EOC7/RESULT7 is associated with post processing block 3 1000 SOC8/EOC8/RESULT8 is associated with post processing block 3 1001 SOC9/EOC9/RESULT9 is associated with post processing block 3 1010 SOC10/EOC10/RESULT10 is associated with post processing block 3 1011 SOC11/EOC11/RESULT11 is associated with post processing block 3 1100 SOC12/EOC12/RESULT12 is associated with post processing block 3 1101 SOC13/EOC13/RESULT13 is associated with post processing block 3 1110 SOC14/EOC14/RESULT14 is associated with post processing block 3 1111 SOC15/EOC15/RESULT15 is associated with post processing block 3 Reset type: SYSRSn

16.15.3.54 ADCPPB3STAMP Register (Offset = 51h) [Reset = 0h]

ADCPPB3STAMP is shown in Figure 16-93 and described in Table 16-88.

Return to the Summary Table.

ADC PPB3 Sample Delay Time Stamp Register

Figure 16-93. ADCPPB3STAMP Register

15	14	13	12	11	10	9	8
RESERVED				DLYSTAMP			
R-0h				R-0h			
7	6	5	4	3	2	1	0
DLYSTAMP							
R-0h							

Table 16-88. ADCPPB3STAMP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-0	DLYSTAMP	R	0h	ADC Post Processing Block 3 Delay Time Stamp. When an SOC starts sampling the value contained in REQSTAMP is subtracted from the value in ADCCOUNTER.FREECOUNT and loaded into this bit field, thereby giving the number of system clock cycles delay between the SOC trigger and the actual start of the sample. Reset type: SYSRSn

16.15.3.55 ADCPPB3OFFCAL Register (Offset = 52h) [Reset = 0h]

ADCPPB3OFFCAL is shown in Figure 16-94 and described in Table 16-89.

Return to the Summary Table.

ADC PPB3 Offset Calibration Register

Figure 16-94. ADCPPB3OFFCAL Register

15	14	13	12	11	10	9	8
RESERVED						OFFCAL	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
OFFCAL							
R/W-0h							

Table 16-89. ADCPPB3OFFCAL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	OFFCAL	R/W	0h	ADC Post Processing Block 3 Offset Correction. This bit field can be used to digitally remove any system level offset inherent in the ADCIN circuit. This 10-bit signed value is subtracted from the ADC output before being stored in the ADCRESULT register. 000h No change. The ADC output is stored directly into ADCRESULT. 001h ADC output - 1 is stored into ADCRESULT. 002h ADC output - 2 is stored into ADCRESULT. ... 200h ADC output + 512 is stored into ADCRESULT. ... 3FFh ADC output + 1 is stored into ADCRESULT. NOTE: In 16-bit mode, the subtraction will saturate at 0000h and FFFFh before being stored into the ADCRESULT register. In 12-bit mode, the subtraction will saturate at 0000h and 0FFFh before being stored into the ADCRESULT register. Note: in the case that multiple PPBs point to the same SOC, only the OFFCAL of the highest numbered PPB will be applied. Reset type: SYSRSn

16.15.3.56 ADCPPB3OFFREF Register (Offset = 53h) [Reset = 0h]

ADCPPB3OFFREF is shown in Figure 16-95 and described in Table 16-90.

Return to the Summary Table.

ADC PPB3 Offset Reference Register

Figure 16-95. ADCPPB3OFFREF Register

15	14	13	12	11	10	9	8
OFFREF							
R/W-0h							
7	6	5	4	3	2	1	0
OFFREF							
R/W-0h							

Table 16-90. ADCPPB3OFFREF Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	OFFREF	R/W	0h	ADC Post Processing Block 3 Offset Correction. This bit field can be used to either calculate the feedback error or convert a unipolar signal to bipolar by subtracting a reference value. This 16-bit unsigned value is subtracted from the ADCRESULT register before being passed through an optional two's complement function and stored in the ADCPPB3RESULT register. This subtraction is not saturated. 0000h No change. The ADCRESULT value is passed on. 0001h ADCRESULT - 1 is passed on. 0002h ADCRESULT - 2 is passed on. ... 8000h ADCRESULT - 32,768 is passed on. ... FFFFh ADCRESULT - 65,535 is passed on. NOTE: In 12-bit mode the size of this register does not change from 16-bits. It is the user's responsibility to ensure that only a 12-bit value is written to this register when in 12-bit mode. Reset type: SYSRSn

16.15.3.57 ADCPPB3TRIPHI Register (Offset = 54h) [Reset = 0h]

ADCPPB3TRIPHI is shown in Figure 16-96 and described in Table 16-91.

Return to the Summary Table.

ADC PPB3 Trip High Register

Figure 16-96. ADCPPB3TRIPHI Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							HSIGN
R-0h							R/W-0h
15	14	13	12	11	10	9	8
LIMITHI							
R/W-0h							
7	6	5	4	3	2	1	0
LIMITHI							
R/W-0h							

Table 16-91. ADCPPB3TRIPHI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16	HSIGN	R/W	0h	High Limit Sign Bit. This is the sign bit (17th bit) to the LIMITHI bit field when in 16-bit ADC mode. Reset type: SYSRSn
15-0	LIMITHI	R/W	0h	ADC Post Processing Block 3 Trip High Limit. This value sets the digital comparator trip high limit. In 16-bit mode all 17 bits will be compared against the 17 bits of the PPBRESULT bit field of the ADCPPB3RESULT register. In 12-bit mode bits 12:0 will be compared against bits 12:0 of the PPBRESULT bit field of the ADCPPB3RESULT register. Reset type: SYSRSn

16.15.3.58 ADCPPB3TRIPLO Register (Offset = 56h) [Reset = 0h]

ADCPPB3TRIPLO is shown in Figure 16-97 and described in Table 16-92.

Return to the Summary Table.

ADC PPB3 Trip Low/Trigger Time Stamp Register

Figure 16-97. ADCPPB3TRIPLO Register

31	30	29	28	27	26	25	24
REQSTAMP							
R-0h							
23	22	21	20	19	18	17	16
REQSTAMP				RESERVED			LSIGN
R-0h				R-0h			R/W-0h
15	14	13	12	11	10	9	8
LIMITLO							
R/W-0h							
7	6	5	4	3	2	1	0
LIMITLO							
R/W-0h							

Table 16-92. ADCPPB3TRIPLO Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	REQSTAMP	R	0h	ADC Post Processing Block 3 Request Time Stamp. When a trigger sets the associated SOC flag in the ADCSOCFLG1 register the value of ADCCOUNTER.FREECOUNT is loaded into this bit field. Reset type: SYSRSn
19-17	RESERVED	R	0h	Reserved
16	LSIGN	R/W	0h	Low Limit Sign Bit. This is the sign bit (17th bit) to the LIMITLO bit field when in 16-bit ADC mode. Reset type: SYSRSn
15-0	LIMITLO	R/W	0h	ADC Post Processing Block 3 Trip Low Limit. This value sets the digital comparator trip low limit. In 16-bit mode all 17 bits will be compared against the 17 bits of the PPBRESULT bit field of the ADCPPB3RESULT register. In 12-bit mode bits 12:0 will be compared against bits 12:0 of the PPBRSULT bit field of the ADCPPB3RESULT register. Reset type: SYSRSn

16.15.3.59 ADCPPB4CONFIG Register (Offset = 58h) [Reset = 0h]

ADCPPB4CONFIG is shown in Figure 16-98 and described in Table 16-93.

Return to the Summary Table.

ADC PPB4 Config Register

Figure 16-98. ADCPPB4CONFIG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		CBCEN	TWOSCOMPEN	CONFIG			
R-0h		R/W-0h	R/W-0h	R/W-0h			

Table 16-93. ADCPPB4CONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-6	RESERVED	R	0h	Reserved
5	CBCEN	R/W	0h	ADC Post Processing Block Cycle By Cycle Enable. When set, this bit enables the post conversion hardware processing circuit to automatically clear the ADCEVTSTAT on a conversion if the event condition is no longer present. Reset type: SYSRSn
4	TWOSCOMPEN	R/W	0h	ADC Post Processing Block 4 Two's Complement Enable. When set this bit enables the post conversion hardware processing circuit that performs a two's complement on the output of the offset/reference subtraction unit before storing the result in the ADCPPB4RESULT register. 0 ADCPPB4RESULT = ADCRESULTx - ADCPPB4OFFREF 1 ADCPPB4RESULT = ADCPPB4OFFREF - ADCRESULTx Reset type: SYSRSn

Table 16-93. ADCPPB4CONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	CONFIG	R/W	0h	ADC Post Processing Block 4 Configuration. This bit field defines which SOC/EOC/RESULT is assocatied with this post processing block. 0000 SOC0/EOC0/RESULT0 is associated with post processing block 4 0001 SOC1/EOC1/RESULT1 is associated with post processing block 4 0010 SOC2/EOC2/RESULT2 is associated with post processing block 4 0011 SOC3/EOC3/RESULT3 is associated with post processing block 4 0100 SOC4/EOC4/RESULT4 is associated with post processing block 4 0101 SOC5/EOC5/RESULT5 is associated with post processing block 4 0110 SOC6/EOC6/RESULT6 is associated with post processing block 4 0111 SOC7/EOC7/RESULT7 is associated with post processing block 4 1000 SOC8/EOC8/RESULT8 is associated with post processing block 4 1001 SOC9/EOC9/RESULT9 is associated with post processing block 4 1010 SOC10/EOC10/RESULT10 is associated with post processing block 4 1011 SOC11/EOC11/RESULT11 is associated with post processing block 4 1100 SOC12/EOC12/RESULT12 is associated with post processing block 4 1101 SOC13/EOC13/RESULT13 is associated with post processing block 4 1110 SOC14/EOC14/RESULT14 is associated with post processing block 4 1111 SOC15/EOC15/RESULT15 is associated with post processing block 4 Reset type: SYSRSn

16.15.3.60 ADCPPB4STAMP Register (Offset = 59h) [Reset = 0h]

ADCPPB4STAMP is shown in Figure 16-99 and described in Table 16-94.

Return to the Summary Table.

ADC PPB4 Sample Delay Time Stamp Register

Figure 16-99. ADCPPB4STAMP Register

15	14	13	12	11	10	9	8
RESERVED				DLYSTAMP			
R-0h				R-0h			
7	6	5	4	3	2	1	0
DLYSTAMP							
R-0h							

Table 16-94. ADCPPB4STAMP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-0	DLYSTAMP	R	0h	ADC Post Processing Block 4 Delay Time Stamp. When an SOC starts sampling the value contained in REQSTAMP is subtracted from the value in ADCCOUNTER.FREECOUNT and loaded into this bit field, thereby giving the number of system clock cycles delay between the SOC trigger and the actual start of the sample. Reset type: SYSRSn

16.15.3.61 ADCPPB4OFFCAL Register (Offset = 5Ah) [Reset = 0h]

ADCPPB4OFFCAL is shown in Figure 16-100 and described in Table 16-95.

Return to the Summary Table.

ADC PPB4 Offset Calibration Register

Figure 16-100. ADCPPB4OFFCAL Register

15	14	13	12	11	10	9	8
RESERVED						OFFCAL	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
OFFCAL							
R/W-0h							

Table 16-95. ADCPPB4OFFCAL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	OFFCAL	R/W	0h	ADC Post Processing Block 4 Offset Correction. This bit field can be used to digitally remove any system level offset inherent in the ADCIN circuit. This 10-bit signed value is subtracted from the ADC output before being stored in the ADCRESULT register. 000h No change. The ADC output is stored directly into ADCRESULT. 001h ADC output - 1 is stored into ADCRESULT. 002h ADC output - 2 is stored into ADCRESULT. ... 200h ADC output + 512 is stored into ADCRESULT. ... 3FFh ADC output + 1 is stored into ADCRESULT. NOTE: In 16-bit mode, the subtraction will saturate at 0000h and FFFFh before being stored into the ADCRESULT register. In 12-bit mode, the subtraction will saturate at 0000h and 0FFFh before being stored into the ADCRESULT register. Note: in the case that multiple PPBs point to the same SOC, only the OFFCAL of the highest numbered PPB will be applied. Reset type: SYSRSn

16.15.3.62 ADCPPB4OFFREF Register (Offset = 5Bh) [Reset = 0h]

ADCPPB4OFFREF is shown in Figure 16-101 and described in Table 16-96.

Return to the Summary Table.

ADC PPB4 Offset Reference Register

Figure 16-101. ADCPPB4OFFREF Register

15	14	13	12	11	10	9	8
OFFREF							
R/W-0h							
7	6	5	4	3	2	1	0
OFFREF							
R/W-0h							

Table 16-96. ADCPPB4OFFREF Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	OFFREF	R/W	0h	ADC Post Processing Block 4 Offset Correction. This bit field can be used to either calculate the feedback error or convert a unipolar signal to bipolar by subtracting a reference value. This 16-bit unsigned value is subtracted from the ADCRESULT register before being passed through an optional two's complement function and stored in the ADCPPB4RESULT register. This subtraction is not saturated. 0000h No change. The ADCRESULT value is passed on. 0001h ADCRESULT - 1 is passed on. 0002h ADCRESULT - 2 is passed on. ... 8000h ADCRESULT - 32,768 is passed on. ... FFFFh ADCRESULT - 65,535 is passed on. NOTE: In 12-bit mode the size of this register does not change from 16-bits. It is the user's responsibility to ensure that only a 12-bit value is written to this register when in 12-bit mode. Reset type: SYSRSn

16.15.3.63 ADCPPB4TRIPHI Register (Offset = 5Ch) [Reset = 0h]

ADCPPB4TRIPHI is shown in Figure 16-102 and described in Table 16-97.

Return to the Summary Table.

ADC PPB4 Trip High Register

Figure 16-102. ADCPPB4TRIPHI Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							HSIGN
R-0h							R/W-0h
15	14	13	12	11	10	9	8
LIMITHI							
R/W-0h							
7	6	5	4	3	2	1	0
LIMITHI							
R/W-0h							

Table 16-97. ADCPPB4TRIPHI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-17	RESERVED	R	0h	Reserved
16	HSIGN	R/W	0h	High Limit Sign Bit. This is the sign bit (17th bit) to the LIMITHI bit field when in 16-bit ADC mode. Reset type: SYSRSn
15-0	LIMITHI	R/W	0h	ADC Post Processing Block 4 Trip High Limit. This value sets the digital comparator trip high limit. In 16-bit mode all 17 bits will be compared against the 17 bits of the PPBRESULT bit field of the ADCPPB4RESULT register. In 12-bit mode bits 12:0 will be compared against bits 12:0 of the PPBRESULT bit field of the ADCPPB4RESULT register. Reset type: SYSRSn

16.15.3.64 ADCPPB4TRIPLO Register (Offset = 5Eh) [Reset = 0h]

ADCPPB4TRIPLO is shown in Figure 16-103 and described in Table 16-98.

Return to the Summary Table.

ADC PPB4 Trip Low/Trigger Time Stamp Register

Figure 16-103. ADCPPB4TRIPLO Register

31	30	29	28	27	26	25	24
REQSTAMP							
R-0h							
23	22	21	20	19	18	17	16
REQSTAMP				RESERVED			LSIGN
R-0h				R-0h			R/W-0h
15	14	13	12	11	10	9	8
LIMITLO							
R/W-0h							
7	6	5	4	3	2	1	0
LIMITLO							
R/W-0h							

Table 16-98. ADCPPB4TRIPLO Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	REQSTAMP	R	0h	ADC Post Processing Block 4 Request Time Stamp. When a trigger sets the associated SOC flag in the ADCSOCFLG1 register the value of ADCCOUNTER.FREECOUNT is loaded into this bit field. Reset type: SYSRSn
19-17	RESERVED	R	0h	Reserved
16	LSIGN	R/W	0h	Low Limit Sign Bit. This is the sign bit (17th bit) to the LIMITLO bit field when in 16-bit ADC mode. Reset type: SYSRSn
15-0	LIMITLO	R/W	0h	ADC Post Processing Block 4 Trip Low Limit. This value sets the digital comparator trip low limit. In 16-bit mode all 17 bits will be compared against the 17 bits of the PPBRESULT bit field of the ADCPPB4RESULT register. In 12-bit mode bits 12:0 will be compared against bits 12:0 of the PPBRSULT bit field of the ADCPPB4RESULT register. Reset type: SYSRSn

16.15.3.65 ADCINTCYCLE Register (Offset = 6Fh) [Reset = 0h]

ADCINTCYCLE is shown in Figure 16-104 and described in Table 16-99.

Return to the Summary Table.

ADC Early Interrupt Generation Cycle

Figure 16-104. ADCINTCYCLE Register

15	14	13	12	11	10	9	8
DELAY							
R/W-0h							
7	6	5	4	3	2	1	0
DELAY							
R/W-0h							

Table 16-99. ADCINTCYCLE Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	DELAY	R/W	0h	ADC Early Interrupt Generation Cycle Delay: Defines the delay from the fall edge of ADCSOC in terms of system clock cycles, for the interrupt to be generated. Reset type: SYSRSn

16.15.3.66 ADCINLTRIM1 Register (Offset = 70h) [Reset = X]

ADCINLTRIM1 is shown in Figure 16-105 and described in Table 16-100.

Return to the Summary Table.

ADC Linearity Trim 1 Register

Figure 16-105. ADCINLTRIM1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INLTRIM31TO0																															
R/W-X																															

Table 16-100. ADCINLTRIM1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	INLTRIM31TO0	R/W	X	ADC Linearity Trim Bits 31-0. This register should not be modified unless specifically indicated by TI Errata or other documentation. Modifying the contents of this register could cause this module to operate outside of datasheet specifications. Reset type: SYSRSn

16.15.3.67 ADCINLTRIM2 Register (Offset = 72h) [Reset = X]

ADCINLTRIM2 is shown in Figure 16-106 and described in Table 16-101.

Return to the Summary Table.

ADC Linearity Trim 2 Register

Figure 16-106. ADCINLTRIM2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INLTRIM63TO32																															
R/W-X																															

Table 16-101. ADCINLTRIM2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	INLTRIM63TO32	R/W	X	ADC Linearity Trim Bits 63-32. This register should not be modified unless specifically indicated by TI Errata or other documentation. Modifying the contents of this register could cause this module to operate outside of datasheet specifications. Reset type: SYSRSn

16.15.3.68 ADCINLTRIM3 Register (Offset = 74h) [Reset = X]

ADCINLTRIM3 is shown in Figure 16-107 and described in Table 16-102.

Return to the Summary Table.

ADC Linearity Trim 3 Register

Figure 16-107. ADCINLTRIM3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INLTRIM95TO64																															
R/W-X																															

Table 16-102. ADCINLTRIM3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	INLTRIM95TO64	R/W	X	ADC Linearity Trim Bits 95-64. This register should not be modified unless specifically indicated by TI Errata or other documentation. Modifying the contents of this register could cause this module to operate outside of datasheet specifications. Reset type: SYSRSn

16.15.4 ADC Registers to Driverlib Functions

Table 16-103. ADC Registers to Driverlib Functions

File	Driverlib Function
ADCCTL1	
adc.h	ADC_setInterruptPulseMode
adc.h	ADC_enableConverter
adc.h	ADC_disableConverter
adc.h	ADC_isBusy
ADCCTL2	
adc.h	ADC_setPrescaler
ADCBURSTCTL	
adc.h	ADC_setBurstModeConfig
adc.h	ADC_enableBurstMode
adc.h	ADC_disableBurstMode
ADCINTFLG	
adc.h	ADC_getInterruptStatus
adc.h	ADC_clearInterruptStatus
ADCINTFLGCLR	
adc.h	ADC_clearInterruptStatus
ADCINTOVF	
adc.h	ADC_getInterruptOverflowStatus
adc.h	ADC_clearInterruptOverflowStatus
ADCINTOVFCLR	
adc.h	ADC_clearInterruptOverflowStatus
ADCINTSEL1N2	
adc.h	ADC_enableInterrupt
adc.h	ADC_disableInterrupt
adc.h	ADC_setInterruptSource
adc.h	ADC_enableContinuousMode
adc.h	ADC_disableContinuousMode

Table 16-103. ADC Registers to Driverlib Functions (continued)

File	Driverlib Function
ADCINTSEL3N4	
-	See INTSEL1N2
ADCSOCPRICTL	
adc.h	ADC_setSOCPriority
ADCINTSOCSEL1	
adc.h	ADC_setInterruptSOCTrigger
ADCINTSOCSEL2	
-	See INTSOCSEL1
ADCSOCFLG1	
-	
ADCSOCFRC1	
adc.h	ADC_forceSOC
adc.h	ADC_forceMultipleSOC
ADCSOCOVF1	
-	
ADCSOCOVFCLR1	
-	
ADCSOC0CTL	
adc.h	ADC_setupSOC
ADCSOC1CTL	
-	See SOC0CTL
ADCSOC2CTL	
-	See SOC0CTL
ADCSOC3CTL	
-	See SOC0CTL
ADCSOC4CTL	
-	See SOC0CTL
ADCSOC5CTL	
-	See SOC0CTL
ADCSOC6CTL	
-	See SOC0CTL
ADCSOC7CTL	
-	See SOC0CTL
ADCSOC8CTL	
-	See SOC0CTL
ADCSOC9CTL	
-	See SOC0CTL
ADCSOC10CTL	
-	See SOC0CTL
ADCSOC11CTL	
-	See SOC0CTL
ADCSOC12CTL	
-	See SOC0CTL
ADCSOC13CTL	
-	See SOC0CTL

Table 16-103. ADC Registers to Driverlib Functions (continued)

File	Driverlib Function
ADCSOC14CTL	
-	See SOC0CTL
ADCSOC15CTL	
-	See SOC0CTL
ADCEVTSTAT	
adc.h	ADC_getPPBEventStatus
ADCEVTCLR	
adc.h	ADC_clearPPBEventStatus
ADCEVTSEL	
adc.h	ADC_enablePPBEvent
adc.h	ADC_disablePPBEvent
ADCEVTINTSEL	
adc.h	ADC_enablePPBEventInterrupt
adc.h	ADC_disablePPBEventInterrupt
ADCOSDETECT	
adc.h	ADC_configOSDetectMode
ADCCOUNTER	
-	
ADCREV	
-	
ADCOFFTRIM	
adc.c	ADC_setOffsetTrim
adc.c	ADC_setOffsetTrimAll
ADCPPB1CONFIG	
adc.h	ADC_setupPPB
adc.h	ADC_enablePPBEventCBCClear
adc.h	ADC_disablePPBEventCBCClear
adc.h	ADC_enablePPBTwosComplement
adc.h	ADC_disablePPBTwosComplement
ADCPPB1STAMP	
adc.h	ADC_getPPBDelayTimeStamp
ADCPPB1OFFCAL	
adc.h	ADC_setPPBCalibrationOffset
ADCPPB1OFFREF	
adc.h	ADC_setPPBReferenceOffset
ADCPPB1TRIPHI	
adc.c	ADC_setPPBTripLimits
ADCPPB1TRIPLO	
adc.c	ADC_setPPBTripLimits
ADCPPB2CONFIG	
-	See PPB1CONFIG
ADCPPB2STAMP	
-	See PPB1STAMP
ADCPPB2OFFCAL	
-	See PPB1OFFCAL

Table 16-103. ADC Registers to Driverlib Functions (continued)

File	Driverlib Function
ADCPPB2OFFREF	
-	See PPB1OFFREF
ADCPPB2TRIPHI	
-	See PPB1TRIPHI
ADCPPB2TRIPLO	
-	See PPB1TRIPLO
ADCPPB3CONFIG	
-	See PPB1CONFIG
ADCPPB3STAMP	
-	See PPB1STAMP
ADCPPB3OFFCAL	
-	See PPB1OFFCAL
ADCPPB3OFFREF	
-	See PPB1OFFREF
ADCPPB3TRIPHI	
-	See PPB1TRIPHI
ADCPPB3TRIPLO	
-	See PPB1TRIPLO
ADCPPB4CONFIG	
-	See PPB1CONFIG
ADCPPB4STAMP	
-	See PPB1STAMP
ADCPPB4OFFCAL	
-	See PPB1OFFCAL
ADCPPB4OFFREF	
-	See PPB1OFFREF
ADCPPB4TRIPHI	
-	See PPB1TRIPHI
ADCPPB4TRIPLO	
-	See PPB1TRIPLO
ADCINTCYCLE	
adc.h	ADC_setInterruptCycleOffset
ADCINLTRIM1	
-	
ADCINLTRIM2	
adc.c	ADC_setINLTrim
ADCINLTRIM3	
-	
ADCRESULT0	
adc.h	ADC_readResult
ADCRESULT1	
-	See RESULT0
ADCRESULT2	
-	See RESULT0
ADCRESULT3	

Table 16-103. ADC Registers to Driverlib Functions (continued)

File	Driverlib Function
-	See RESULT0
ADCRESULT4	
-	See RESULT0
ADCRESULT5	
-	See RESULT0
ADCRESULT6	
-	See RESULT0
ADCRESULT7	
-	See RESULT0
ADCRESULT8	
-	See RESULT0
ADCRESULT9	
-	See RESULT0
ADCRESULT10	
-	See RESULT0
ADCRESULT11	
-	See RESULT0
ADCRESULT12	
-	See RESULT0
ADCRESULT13	
-	See RESULT0
ADCRESULT14	
-	See RESULT0
ADCRESULT15	
-	See RESULT0
ADCPPB1RESULT	
adc.h	ADC_readPPBResult
ADCPPB2RESULT	
-	See PPB1RESULT
ADCPPB3RESULT	
-	See PPB1RESULT
ADCPPB4RESULT	
-	See PPB1RESULT

This page intentionally left blank.

Chapter 17

Buffered Digital-to-Analog Converter (DAC)

The buffered digital-to-analog converter (DAC) is an analog module that can output a programmable, arbitrary reference voltage.

17.1 Introduction

The buffered DAC module consists of an internal 12-bit DAC and an analog output buffer that is capable of driving an external load. For driving even higher loads than typical, a trade-off can be made between load size and output voltage swing. For the load conditions of the buffered DAC, see the device-specific data manual. The buffered DAC is a general-purpose DAC that can be used to generate a DC voltage in addition to AC waveforms such as sine waves, square waves, triangle waves and so forth. Software writes to the DAC value register can take effect immediately or can be synchronized with EPWMSYNCPER events.

17.1.1 DAC Related Collateral

Foundational Materials

- C2000 Academy - Analog Subsystem
- *High Speed, Digital to Analog Converters Basics Application Report*
- *Real-Time Control Reference Guide*
  - Refer to the DAC section
- *Understanding Data Converters Application Report*

17.1.2 Features

Each buffered DAC has the following features:

- 12-bit programmable internal DAC
- Selectable reference voltage source
- x1 and x2 gain modes when using internal VREFHI
- Ability to synchronize with EPWMSYNCPER

17.1.3 Block Diagram

The block diagram for the buffered DAC is shown in Figure 17-1.

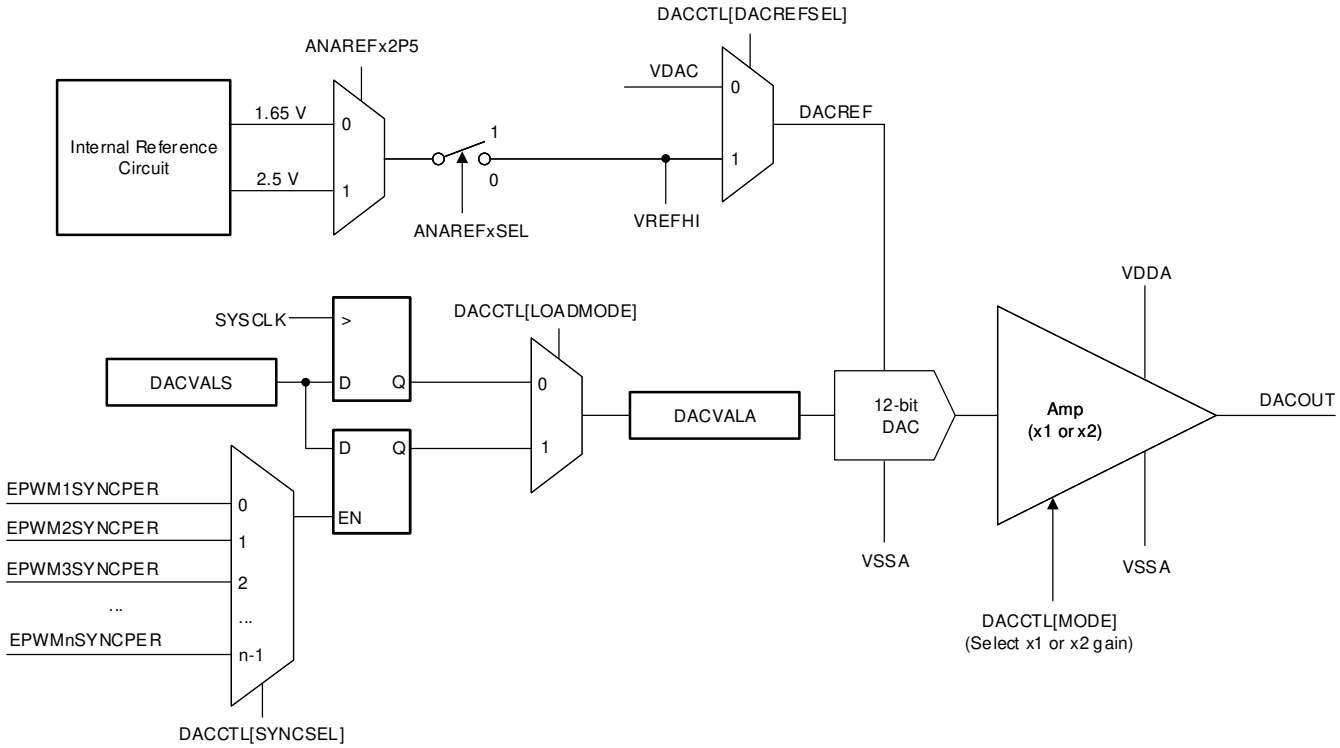


Figure 17-1. DAC Module Block Diagram

17.2 Using the DAC

The internal DAC's reference voltage source, DACREF, is selectable between VDAC and VREFHI. The x2 gain mode is only available when VREFHI is set as DACREF and internal reference mode is used (see the *Analog Subsystem* chapter on how to switch to internal reference mode). Even though the buffered DAC has an x2 gain mode, the maximum output voltage from the buffered DAC will not be greater than VDDA. Table 17-1 lists the gain mode combinations supported by the buffered DAC. In this table, x = A or B, X = Don't Care, VDAC/VREFHI = 2.5v, VDDA = 3.3v, and DACVAL = 4095.

Table 17-1. DAC Supported Gain Mode Combinations

DACREFSEL	ANAREFSEL	ANAREF2P5	Reference Source	Reference Voltage (V)	Mode	Maximum DAC Output (V)	Support Status
0	X	X	External	VDAC	0	2.5	Supported
0	X	X	External	VDAC	1	2.5	Not Supported
1	0	0	Internal	1.65	0	1.65	Not Supported
1	0	0	Internal	1.65	1	3.3	Supported
1	0	1	Internal	2.5	0	2.5	Supported
1	0	1	Internal	2.5	1	3.3	Not Supported
1	1	X	External	VREFHI	0	2.5	Supported
1	1	X	External	VREFHI	1	2.5	Not Supported

Two sets of DACVAL registers, DACVALA and DACVALS, are present in the buffered DAC module. DACVALA is a read-only register that actively controls the buffered DAC value. DACVALS is a writable shadow register that loads into DACVALA either immediately or synchronized with the next EPWMSYNCPER event. If the clock to the buffered DAC is disabled while the buffered DAC is outputting a voltage, the output voltage remains unaffected, but DACVALA and DACVALS will no longer be updated with register writes. Enabling the clock to the buffered DAC restores it to the state before the clock was disabled.

The ideal output of the internal DAC is calculated with the following equation:

$$DACOUT = \frac{DACVALA * DACREF}{4096} \quad (7)$$

The output buffer of the buffered DAC may exhibit non-linear behavior near the supply rails (VDDA/VSSA). To determine the linear range of the buffered DAC, see the device-specific data manual.

17.2.1 Initialization Sequence

1. Enable the buffered DAC clock.
2. Set DACREF with DACREFSEL.
3. Power up the buffered DAC with DACOUTEN.
4. Wait for the power-up time to elapse before outputting a voltage. To determine the power-up time of the buffered DAC, see the device-specific data manual.
5. For predictable behavior of the buffered DAC, consecutive writes to DACVALS should be spaced apart according to the settling time of the buffered DAC. To determine the settling time of the buffered DAC, see the device-specific data manual.

17.2.2 DAC Offset Adjustment

Zero offset error is defined as the difference between the voltage at midcode (2048) and 1.25v (for 2.5v reference voltage). DAC offset error is calibrated at 2.5v reference voltage and loaded into the DAC offset trim register as part of the Device_cal() function. If the DAC is used at any reference voltage other than 2.5v, the offset trim must be adjusted to ensure the offset error performance stays within the device-specific data manual limits. The DAC offset register is a 16-bit register that contains the 8-bit signed offset trim in the lower half of the register. Use the function call DAC_tuneOffsetTrim() found in C2000Ware to adjust the offset.

17.2.3 EPWMSYNCPER Signal

EPWMSYNCPER comes from the Time-Base submodule of the EPWM. For a detailed description of how this signal is generated, refer to the Time-Base Submodule section of the *Enhanced Pulse Width Modulator (ePWM)* chapter.

The EPWMSYNCPER signal that loads DACVALA when DACCTL [LOADMODE] = 1 is a level trigger load. If TBCTR and TBPRD of the EPWM are both 0, EPWMSYNCPER will be held at level high and DACVALA will be loaded immediately from DACVALS irrespective of the value of DACCTL [LOADMODE]. Due to this, it is recommended to configure the EPWM first before setting DACCTL [LOADMODE] to 1.

Note

The name of the sync signal that the GPDAC receives from the EPWM has been updated from PWMSYNC to EPWMSYNCPER (SYNCPER/PWMSYNCPER/EPWMxSYNCPER) to avoid confusion with the other EPWM sync signals EPWMSYNCI and EPWMSYNCO. For a description of what are these signals, see the *Enhanced Pulse Width Modulator (ePWM)* chapter.

17.3 Lock Registers

A DACLOCK register is provided to prevent spurious writes from modifying the DACCTL, DACVALS, and DACOUTEN registers. Once a register is protected through DACLOCK, write access will be locked out until the device is reset.

17.4 Software

17.4.1 DAC Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/dac

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

17.4.1.1 Buffered DAC Enable

FILE: buffdac_ex1_enable.c

This example generates a voltage on the buffered DAC output, DACOUTA/ADCINA0 and uses the default DAC reference setting of VDAC.

External Connections

- When the DAC reference is set to VDAC, an external reference voltage must be applied to the VDAC pin. This can be accomplished by connecting a jumper wire from 3.3V to ADCINB3.

Watch Variables

- None.

17.4.1.2 Buffered DAC Random

FILE: buffdac_ex2_random.c

This example generates random voltages on the buffered DAC output, DACOUTA/ADCINA0 and uses the default DAC reference setting of VDAC.

External Connections

- When the DAC reference is set to VDAC, an external reference voltage must be applied to the VDAC pin. This can accomplished by connecting a jumper wire from 3.3V to ADCINB3.

Watch Variables

- None.

17.4.1.3 Buffered DAC Sine (buffdac_sine)

FILE: buffdac_ex3_waveform.c

This example generates a sine wave on the buffered DAC output, DACOUTA/ADCINA0 (HSEC Pin 9) and uses the default DAC reference setting of VDAC.

When the DAC reference is set to VDAC, an external reference voltage must be applied to the VDAC pin. This can accomplished by connecting a jumper wire from 3.3V to ADCINB3.

Run the included .js file to add the watch variables. This example uses the SGEN module. Documentation for the SGEN module can be found in the SGEN library directory.

The generated waveform can be adjusted with the following variables while running:

- *waveformGain* : Adjust the magnitude of the waveform. Range is from 0.0 to 1.0. The default value of 0.8003 centers the waveform within the linear range of the DAC
- *waveformOffset* : Adjust the offset of the waveform. Range is from -1.0 to 1.0. The default value of 0 centers the waveform
- *outputFreq_hz* : Adjust the output frequency of the waveform. Range is from 0 to maxOutputFreq_hz
- *maxOutputFreq_hz* : Adjust the max output frequency of the waveform. Range - See SGEN module documentation for how this affects other parameters

The generated waveform can be adjusted with the following variables/macros but require recompile:

- *samplingFreq_hz* : Adjust the rate at which the DAC is updated. Range - See SGEN module documentation for how this affects other parameters
- *SINEWAVE_TYPE* : The type of sine generated. Range - LOW_THD_SINE, HIGH_PRECISION_SINE

The following variables give additional information about the generated waveform: See SGEN module documentation for details

- *freqResolution_hz*
- *maxOutput_lsb* : Maximum value written to the DAC.
- *minOutput_lsb* : Minimum value written to the DAC.
- *pk_to_pk_lsb* : Magnitude of generated waveform.
- *cpuPeriod_us* : Period of cpu.
- *samplingPeriod_us* : The rate at which the DAC is updated. Note that samplingPeriod_us has to be greater than the DAC settling time.
- *interruptCycles* : Interrupt duration in cycles.
- *interruptDuration_us* : Interrupt duration in uS.
- *sgen* : The SGEN module instance.
- *DataLog* : Circular log of writes to the DAC.

17.5 DAC Registers

This section describes the Buffered Digital to Analog Converter registers.

17.5.1 DAC Base Address Table

Table 17-2. DAC Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
DacaRegs	DAC_REGS	DACA_BASE	0x0000_5C00	YES	YES	YES	YES	YES
DacbRegs	DAC_REGS	DACB_BASE	0x0000_5C10	YES	YES	YES	YES	YES

17.5.2 DAC_REGS Registers

Table 17-3 lists the memory-mapped registers for the DAC_REGS registers. All register offset addresses not listed in Table 17-3 should be considered as reserved locations and the register contents should not be modified.

Table 17-3. DAC_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	DACREV	DAC Revision Register		Go
1h	DACCTL	DAC Control Register	EALLOW	Go
2h	DACVALA	DAC Value Register - Active		Go
3h	DACVALS	DAC Value Register - Shadow		Go
4h	DACOUTEN	DAC Output Enable Register	EALLOW	Go
5h	DACLOCK	DAC Lock Register	EALLOW	Go
6h	DACTRIM	DAC Trim Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 17-4 shows the codes that are used for access types in this section.

Table 17-4. DAC_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

17.5.2.1 DACREV Register (Offset = 0h) [Reset = 0h]

DACREV is shown in Figure 17-2 and described in Table 17-5.

Return to the Summary Table.

DAC Revision Register

Figure 17-2. DACREV Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
REV							
R-0h							

Table 17-5. DACREV Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-0	REV	R	0h	DAC Revision Reset type: SYSRSn

17.5.2.2 DACCTL Register (Offset = 1h) [Reset = 0h]

DACCTL is shown in Figure 17-3 and described in Table 17-6.

Return to the Summary Table.

DAC Control Register

Figure 17-3. DACCTL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
SYNCSEL				RESERVED	LOADMODE	MODE	DACREFSEL
R/W-0h				R-0h	R/W-0h	R/W-0h	R/W-0h

Table 17-6. DACCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-4	SYNCSEL	R/W	0h	DAC EPWMSYNCPER select. Determines which EPWMSYNCPER signal will update the DACVALA register. Where n represents the maximum number of EPWMSYNCPER signals available on the device: 0 EPWM1SYNCPER 1 EPWM2SYNCPER 2 EPWM3SYNCPER ... n-1 EPWMnSYNCPER Reset type: SYSRSn
3	RESERVED	R	0h	Reserved
2	LOADMODE	R/W	0h	DACVALA load mode. Determines when the DACVALA register is updated with the value from DACVALS. 0 Load on next SYSCLK 1 Load on next EPWMSYNCPER specified by SYNCSEL Reset type: SYSRSn
1	MODE	R/W	0h	DAC gain mode select. Selects the gain mode for the buffered output. The MODE value is only used when DACREFSEL=1 and internal ADC reference mode is selected. 0 Gain is 1 1 Gain is 2 Reset type: SYSRSn
0	DACREFSEL	R/W	0h	DAC reference select. Selects which voltage references are used by the DAC. 0 VDAC/VSSA are the reference voltages 1 ADC VREFHI/VSSA are the reference voltages Reset type: SYSRSn

17.5.2.3 DACVALA Register (Offset = 2h) [Reset = 0h]

DACVALA is shown in Figure 17-4 and described in Table 17-7.

Return to the Summary Table.

DAC Value Register - Active

Figure 17-4. DACVALA Register

15	14	13	12	11	10	9	8
RESERVED				DACVALA			
R-0h				R-0h			
7	6	5	4	3	2	1	0
DACVALA							
R-0h							

Table 17-7. DACVALA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-0	DACVALA	R	0h	Active output code currently driven by the DAC Reset type: SYSRSn

17.5.2.4 DACVALS Register (Offset = 3h) [Reset = 0h]

DACVALS is shown in Figure 17-5 and described in Table 17-8.

Return to the Summary Table.

DAC Value Register - Shadow

Figure 17-5. DACVALS Register

15	14	13	12	11	10	9	8
RESERVED				DACVALS			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
DACVALS							
R/W-0h							

Table 17-8. DACVALS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-0	DACVALS	R/W	0h	Shadow output code to be loaded into DACVALA Reset type: SYSRSn

17.5.2.5 DACOUTEN Register (Offset = 4h) [Reset = 0h]

DACOUTEN is shown in Figure 17-6 and described in Table 17-9.

Return to the Summary Table.

DAC Output Enable Register

Figure 17-6. DACOUTEN Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							DACOUTEN
R-0h							R/W-0h

Table 17-9. DACOUTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R	0h	Reserved
0	DACOUTEN	R/W	0h	DAC output enable 0 DAC output is disabled 1 DAC output is enabled Reset type: SYSRSn

17.5.2.6 DACLOCK Register (Offset = 5h) [Reset = 0h]

DACLOCK is shown in Figure 17-7 and described in Table 17-10.

Return to the Summary Table.

DAC Lock Register

Figure 17-7. DACLOCK Register

15	14	13	12	11	10	9	8
KEY				RESERVED			
R-0/W-0h				R-0h			
7	6	5	4	3	2	1	0
RESERVED					DACOUTEN	DACVAL	DACCTL
R-0h					R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 17-10. DACLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	KEY	R-0/W	0h	Writes to this register succeed only if this field is written with a value of 0xA. Only 16-bit writes will succeed (provided the KEY matches). Read-modify-writes to individual bits in this register will be ignored. Reset type: SYSRSn
11-3	RESERVED	R	0h	Reserved
2	DACOUTEN	R/WSonce	0h	Lock write-access to the DACOUTEN register. 0 DACOUTEN register is not locked. Write 0 to this bit has no effect. 1 DACOUTEN register is locked. Only a system reset can clear this bit. Reset type: SYSRSn
1	DACVAL	R/WSonce	0h	Lock write-access to the DACVALS register. 0 DACVALS register is not locked. Write 0 to this bit has no effect. 1 DACVALS register is locked. Only a system reset can clear this bit. Reset type: SYSRSn
0	DACCTL	R/WSonce	0h	Lock write-access to the DACCTL register. 0 DACCTL register is not locked. Write 0 to this bit has no effect. 1 DACCTL register is locked. Only a system reset can clear this bit. Reset type: SYSRSn

17.5.2.7 DACTRIM Register (Offset = 6h) [Reset = 0h]

DACTRIM is shown in Figure 17-8 and described in Table 17-11.

Return to the Summary Table.

DAC Trim Register

Figure 17-8. DACTRIM Register

15	14	13	12	11	10	9	8
RESERVED				RESERVED			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
OFFSET_TRIM							
R/W-0h							

Table 17-11. DACTRIM Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-8	RESERVED	R/W	0h	Reserved
7-0	OFFSET_TRIM	R/W	0h	DAC Offset Trim. This register should not be modified unless specifically indicated by TI Errata or other documentation. Modifying the contents of this register could cause this module to operate outside of datasheet specifications. Reset type: SYSRSn

17.5.3 DAC Registers to Driverlib Functions

Table 17-12. DAC Registers to Driverlib Functions

File	Driverlib Function
DACREV	
dac.h	DAC_getRevision
DACCTL	
dac.h	DAC_setReferenceVoltage
dac.h	DAC_setGainMode
dac.h	DAC_setLoadMode
dac.h	DAC_setPWMSyncSignal
DACVALA	
dac.h	DAC_getActiveValue
DACVALS	
dac.h	DAC_setShadowValue
dac.h	DAC_getShadowValue
DACOUTEN	
dac.h	DAC_enableOutput
dac.h	DAC_disableOutput
DACLOCK	
dac.h	DAC_lockRegister
dac.h	DAC_isRegisterLocked
DACTRIM	
dac.c	DAC_tuneOffsetTrim
dac.h	DAC_setOffsetTrim

Table 17-12. DAC Registers to Driverlib Functions (continued)

File	Driverlib Function
dac.h	DAC_getOffsetTrim

This page intentionally left blank.

Chapter 18

Comparator Subsystem (CMPSS)

The Comparator Subsystem (CMPSS) consists of analog comparators and supporting circuits that are useful for power applications such as peak current mode control, switched-mode power, power factor correction, voltage trip monitoring, and so forth.

See the *C2000 Real-Time Control Peripheral Reference Guide* for a list of all devices with modules of the same type, to determine the differences between the types, and for a list of device-specific differences within a type.

18.1 Introduction

The comparator subsystem is built around a number of modules. Each subsystem contains two comparators, two reference 12-bit DACs, and two digital filters. The subsystem also includes one ramp generator. The ramp generator ramps down only. Comparators are denoted "H" or "L" within each module where “H” and “L” represent high and low, respectively. Each comparator generates a digital output which indicates whether the voltage on the positive input is greater than the voltage on the negative input. The positive input of the comparator is driven from an external pin (see the *Analog Subsystem* chapter for mux options available to the CMPSS). The negative input can be driven by an external pin or by the programmable reference 12-bit DAC . Each comparator output passes through a programmable digital filter that can remove spurious trip signals. An unfiltered output is also available if filtering is not required. A ramp generator circuit is optionally available to control the reference 12-bit DAC value for the high comparator in the subsystem.

18.1.1 CMPSS Related Collateral

Foundational Materials

- C2000 Academy - Analog Subsystem
- *Real-Time Control Reference Guide*
  - Refer to the Comparator section

Getting Started Materials

- *Comparator with Hysteresis Reference Design*

Expert Materials

- *Peak Current Control Realization for Boost Circuit Based On C2000 MCU Application Report*
- *Peak Current Mode Controlled PSFB Converter Reference Design Using C2000 Real-time MCU*
- *Understanding and Applying Current-Mode Control Theory Application Report*

18.1.2 Features

Each CMPSS includes:

- Two analog comparators
- Two programmable reference 12-bit DACs
- One decrementing ramp generator
- Two digital filters, max filter clock prescale = 2^{16}
- Ability to synchronize submodules with EPWMSYNCPER
- Ability to extend clear signal with EPWMBLANK
- Ability to synchronize output with SYSCLK
- Ability to latch output
- Ability to invert output
- Option to use hysteresis on the input
- Option for negative input of comparator to be driven by an external signal or by the reference DAC
- The DAC reference voltage can be set to VDDA or VDAC

18.1.3 Block Diagram

The block diagram for the CMPSS is shown in Figure 18-1.

- CTRIPx(x= "H" or "L") signals are connected to the ePWM X-BAR for ePWM trip response. See the *Enhanced Pulse Width Modulator (ePWM)* chapter for more details on the ePWM X-BAR mux configuration.
- CTRIPxOUTx(x= "H" or "L") signals are connected to the Output X-BAR for external signaling. See the *General-Purpose Input/Output (GPIO)* chapter for more details on the Output X-BAR mux configuration.

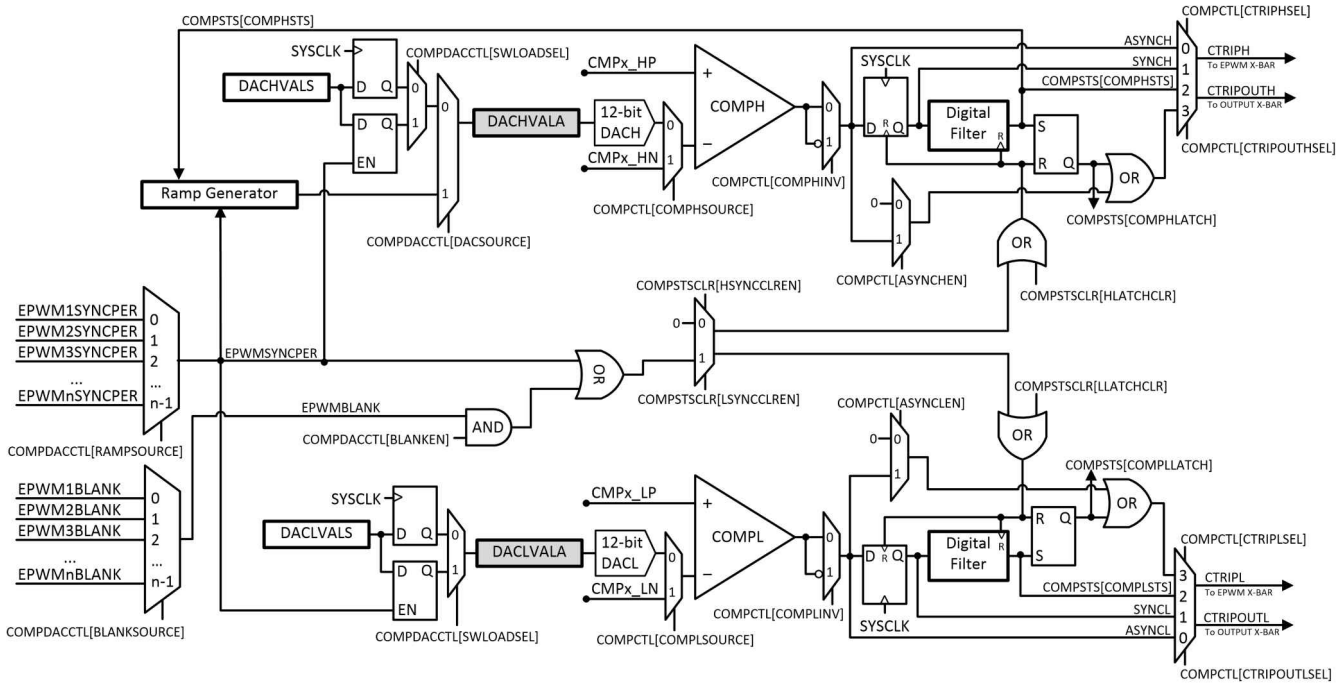


Figure 18-1. CMPSS Module Block Diagram

18.2 Comparator

The comparator generates a high digital output when the voltage on the positive input is greater than the voltage on the negative input, and a low digital output when the voltage on the positive input is less than the voltage on the negative input. The comparator is illustrated in Figure 18-2.

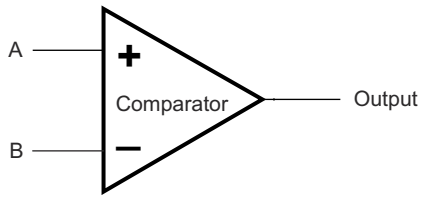


Figure 18-2. Comparator Block Diagram

Voltages	Output
Voltage A > Voltage B	1
Voltage A < Voltage B	0

18.3 Reference DAC

Each reference 12-bit DAC can be configured to drive a reference voltage into the negative input of its respective comparator. The reference 12-bit DAC output is internal only and cannot be observed externally.

Two sets of DACxVAL registers, DACxVALA and DACxVALS, are present for each reference 12-bit DAC. DACxVALA is a read-only register that actively controls the reference 12-bit DAC value. DACxVALS is a writable shadow register that loads into DACxVALA either immediately or synchronized with the next EPWMSYNCPER event. The high reference 12-bit DAC (DACH) can optionally source its DACHVALA value from the ramp generator instead of DACHVALS.

The operating range of the reference 12-bit DAC is bounded by DACREF and VSSA. The high voltage reference is VDDA by default, but it can be configured to be VDAC. The reference 12-bit DAC is illustrated in Figure 18-3.

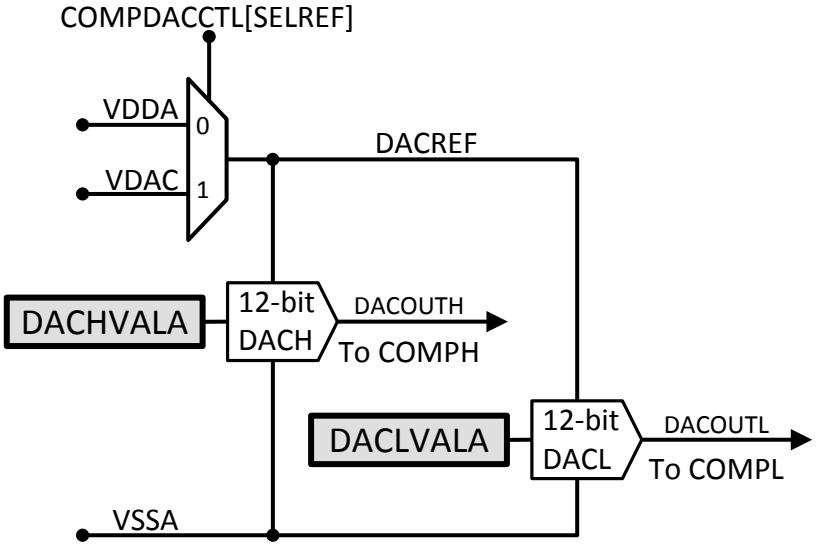


Figure 18-3. Reference DAC Block Diagram

The ideal output of the reference 12-bit DAC can be calculated as:

$$DACOUT = \frac{DACVALA * DACREF}{4096} \quad (8)$$

Note

The CMPSS instance should be enabled before programming the reference DAC values.

18.4 Ramp Generator

This section discusses the characteristics of the ramp generator and its behavior.

18.4.1 Ramp Generator Overview

The ramp generator produces a falling ramp input for the high reference 12-bit DAC when selected. In this mode, the reference 12-bit DAC uses the most significant 12 bits of the RAMPSTS countdown register as its input. The low 4 bits of the RAMPSTS countdown register effectively act as a prescale for the falling ramp rate configurable with RAMPDECVALA..

The ramp generator is enabled by setting DACSOURCE = 1. On setting DACSOURCE = 1, the value of RAMPSTS is loaded from RAMPMAXREFS and the register remains static until the selected EPWMSYNCPER signal is received. After receiving the selected EPWMSYNCPER signal, the value of RAMPDECVALA. is subtracted from RAMPSTS on every subsequent SYSCLK cycle.

To prevent the subtraction from commencing a SYSCLK cycle after a EPWMSYNCPER event, the RAMPDLYA register which serves as a delay counter can be used to hold off the RAMPSTS subtraction. On receiving a EPWMSYNCPER event, the value of RAMPDLYA is decremented by one on every SYSCLK cycle until the register reaches zero. The RAMPSTS subtraction will only begin when RAMPDLYA is zero.

18.4.2 Ramp Generator Behavior

The ramp generator makes state changes on every rising edge of DACSOURCE, EPWMSYNCPER and COMPHSTS.

On the rising edge of DACSOURCE, RAMPMAXREFA, RAMPDECVALA and RAMPDLYA are loaded with their shadow registers. RAMPSTS is loaded with RAMPMAXREFS.

On the rising edge of the selected EPWMSYNCPER, RAMPMAXREFA, RAMPDECVALA and RAMPDLYA are loaded with their shadow registers. RAMPSTS is loaded with RAMPMAXREFS and starts decrementing when RAMPDLYA counter reaches zero.

On the rising edge of COMPHSTS with RAMPLOADSEL = 1, RAMPMAXREFA, RAMPDECVALA and RAMPDLYA are loaded with their shadow registers. RAMPSTS is loaded with RAMPMAXREFS and stops decrementing.

On the rising edge of COMPHSTS with RAMPLOADSEL = 0, RAMPSTS is loaded with RAMPMAXREFA and stops decrementing.

Additionally, if the value of RAMPSTS reaches zero, the RAMPSTS register will remain static at zero until the next EPWMSYNCPER is received. These state changes are illustrated in the ramp generator block diagram in Figure 18-4.

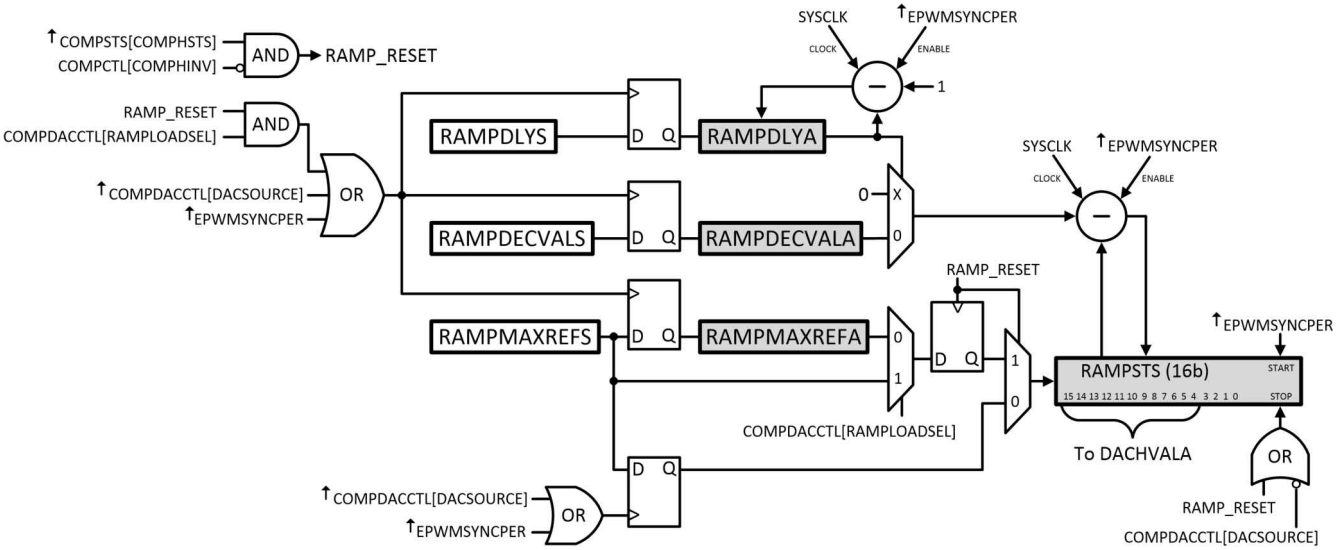


Figure 18-4. Ramp Generator Block Diagram

18.4.3 Ramp Generator Behavior at Corner Cases

Since the ramp generator makes state changes on every rising edge of EPWMSYNCPER and COMPHSTS, the following behavior can be expected on instances when these two events occur simultaneously or very close together.

Case 1: COMPHSTS rising edge occurs one or more cycles before EPWMSYNCPER rising edge.
RAMPSTS stops decrementing on COMPHSTS rising edge event. RAMPSTS starts decrementing on EPWMSYNCPER rising edge event when RAMPDLYA reaches 0.

Case 2: COMPHSTS rising edge occurs simultaneously as EPWMSYNCPER rising edge.
EPWMSYNCPER rising edge event takes precedence and RAMPSTS starts decrementing when RAMPDLYA reaches 0. COMPHSTS rising edge event is ignored and does not halt RAMPSTS.

Case 3: COMPHSTS rising edge occurs one or more cycles after EPWMSYNCPER rising edge but before RAMPDLYA reaches 0.
RAMPSTS does not decrement when RAMPDLYA reaches 0.

Case 4: COMPHSTS rising edge occurs simultaneously as RAMPDLYA reaches 0 from EPWMSYNCPER rising edge.
RAMPSTS does not decrement.

This behavior is also illustrated in Figure 18-5.

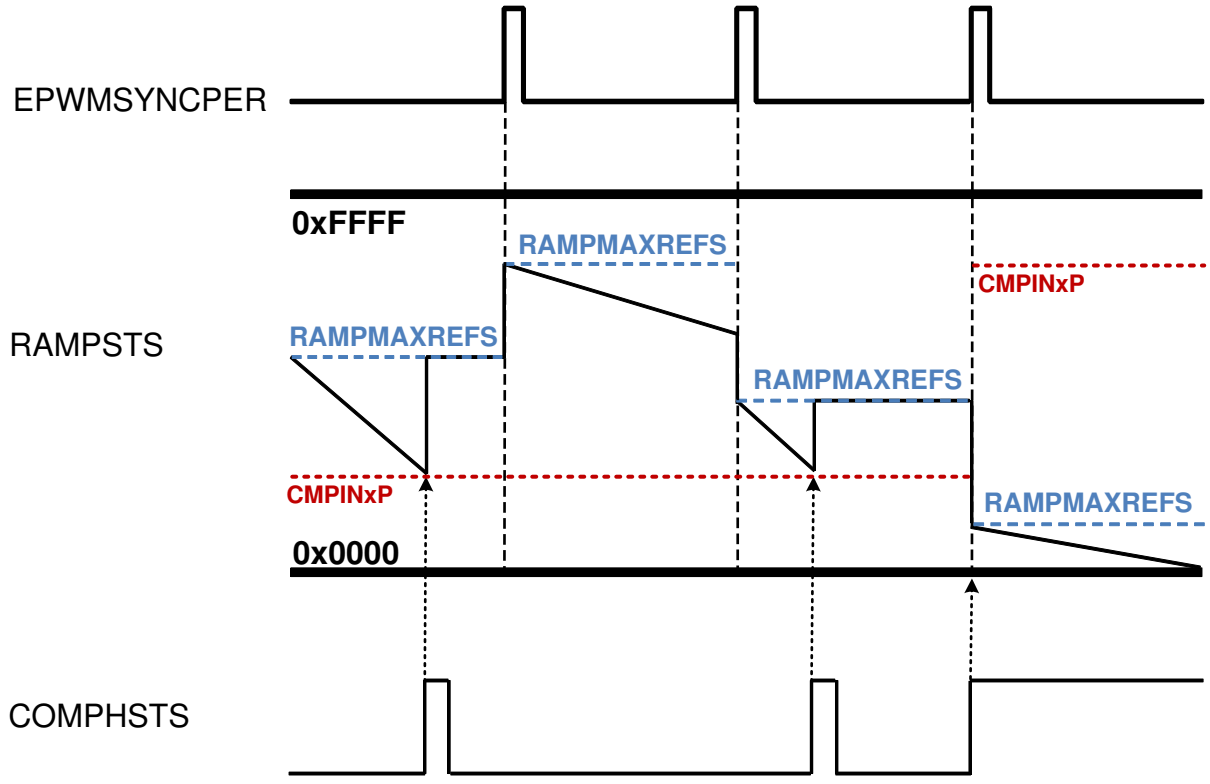


Figure 18-5. Ramp Generator Behavior

18.5 Digital Filter

The digital filter works on a window of FIFO samples (SAMPWIN) taken from the input. The filter output resolves to the majority value of the sample window, where majority is defined by the threshold (THRESH) value. If the majority threshold is not satisfied, the filter output remains unchanged.

For proper operation, the value of THRESH must be greater than SAMPWIN / 2 and less than or equal to SAMPWIN.

A prescale function (CLKPRESCALE) determines the filter sampling rate, where the filter FIFO captures one sample every prescale system clocks. Old data from the FIFO is discarded.

Note that for SAMPWIN, THRESH and CLKPRESCALE, the internal number used by the digital filter is + 1 in all cases. In essence, samples = SAMPWIN + 1, threshold = THRESH + 1 and prescale = CLKPRESCALE + 1.

A conceptual model of the digital filter is shown in Figure 18-6.

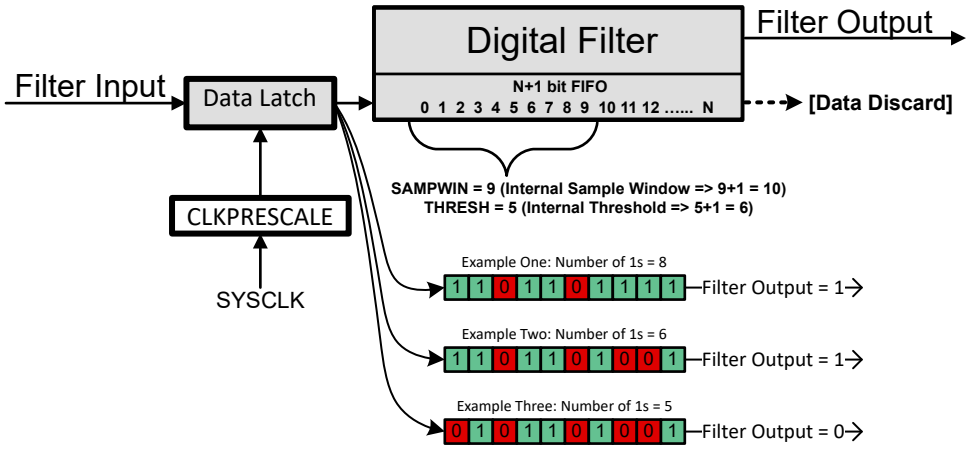


Figure 18-6. Digital Filter Behavior

Equivalent C code of the filter implementation is:

```
if (FILTER_OUTPUT == 0) {
    if (Num_1s_in_SAMPWIN >= THRESH) {
        FILTER_OUTPUT = 1;
    }
}
else {
    if (Num_0s_in_SAMPWIN >= THRESH) {
        FILTER_OUTPUT = 0;
    }
}
```

18.5.1 Filter Initialization Sequence

To ensure proper operation of the digital filter, the following initialization sequence is recommended:

1. Configure and enable the comparator for operation.
2. Configure the digital filter parameters for operation:
   - Set SAMPWIN for the number of samples to monitor in the FIFO window.
   - Set THRESH for the threshold required for majority qualification.
   - Set CLKPRESCALE for the digital filter clock prescale value.
3. Initialize the sample values in the digital FIFO window by setting FILINIT.
4. Clear COMPSTS latch via COMPSTSCLR if the latched path is desired.
5. Configure the CTRIP and CTRIPOUT signal paths.
6. If desired, configure the destination module, for example, ePWM, GPIO, and so on to accept the filtered signals.

18.6 Using the CMPSS

18.6.1 LATCHCLR, EPWMSYNCPER and EPWMBLANK Signals

The LATCHCLR signal holds the digital filter, synchronization block and the latch output in reset (0) after the required delays. It is activated in software using xLATCHCLR(x= “H” or “L”). It can also be activated by EPWMSYNCPER when xSYNCCLREN(x= "H" or "L") is set. If a longer LATCHCLR signal is required, the EPWMBLANK signal can be used to extend it by setting BLANKEN.

EPWMSYNCPER and EPWMBLANK (BLANKWDW) come from the Time-Base and Digital Compare submodules of the EPWM, respectively. For a detailed description of how these two signals are generated, refer to the respective submodule section in the *Enhanced Pulse Width Modulator (ePWM)* chapter.

The EPWMSYNCPER signal that loads DACxVALA when COMPDACCTL [SWLOADSEL] = 1 is a level trigger load. If TBCTR and TBPRD of the EPWM are both 0, EPWMSYNCPER will be held at level high and DACxVALA will be loaded immediately from DACxVALS irrespective of the value of COMPDACCTL [SWLOADSEL]. Due to this, it is recommended to configure the EPWM first before setting COMPDACCTL [SWLOADSEL] to 1.

Note

The name of the sync signal that the CMPSS receives from the EPWM has been updated from PWMSYNC to EPWMSYNCPER (SYNCPER/PWMSYNCPER/EPWMxSYNCPER) to avoid confusion with the other EPWM sync signals EPWMSYNCI and EPWMSYNCO. For a description of what are these signals, see the *Enhanced Pulse Width Modulator (ePWM)* chapter.

18.6.2 Synchronizer, Digital Filter and Latch Delays

The synchronization block adds a delay of 1-2 sysclks. If the digital filter is bypassed (all filter settings are 0), the digital filter will add a delay of 2 sysclks. The latch adds 1 sysclk delay.

18.6.3 Calibrating the CMPSS

The CMPSS has two sources of offset errors: comparator offset error and compdac offset error. In the data manual, the comparator offset error is referred to as ***Input referred offset error*** and compdac offset error is referred to as ***Static offset error***. See the device specific data manual for their values.

If both inputs of the comparator are driven from a pin, only the comparator offset error applies. However if the inverting input of the comparator is driven from the compdac, then only the compdac offset error applies. This is because the compdac offset error includes comparator offset error.

Due to the offset errors, it is recommended that the CMPSS be calibrated to ensure trips happen at the expected levels. The flow below outlines how the calibration can be performed if the inverting input of the comparator is driven from the compdac.

Notes before calibration:

1. A static DC signal is required on the non-inverting input of the comparator.
2. Hysteresis should be disabled for calibration. It can be re-enabled after calibration is complete.
3. A noisy input can make calibration difficult so it is recommended to use the latch with non-zero filter settings depending on how noisy the signal on the non-inverting input is.

This approach sweeps down the compdac:

1. Set the starting compdac value to max, 0xFFF.
   - Optional: Instead of setting the starting compdac value to max, it can be set to ***Vtarget*** + ***Static offset error*** + ***Margin***. Where ***Vtarget*** is the approximate DC voltage on the non-inverting input, ***Static offset error*** is the compdac offset error specification and ***Margin*** is some amount of guard band. This can lead to a faster calibration but will only work if ***Vtarget*** is known. Alternatively if ***Vtarget*** is unknown, the ADC can be used to convert it.
2. Decrement compdac value by 1.
3. Wait for compdac to settle.
4. Clear latch.
5. Wait for possible latch set.
6. If latch is set, trip code is found exit.
   - Optional: The trip code can be double checked by:
     a. Increasing compdac value by 1.
     b. Clear latch.
     c. Wait for possible latch set.
     d. Latch should be unset.
7. If latch is unset, go back to step 2 and repeat.

It is also possible to calibrate the CMPSS if both inputs of the comparator are driven from a pin. For this case, the flow stays the same but the voltage on the inverting pin of the comparator is swept externally.

18.6.4 Enabling and Disabling the CMPSS Clock

If the clock to the CMPSS module is disabled while the comparator is active, the following behavior can be expected:

- The comparator remains unaffected and will continue to trip from voltages on its inputs.
- If the reference 12-bit DAC is driving the negative input of the comparator, the voltage on the negative input remains static and unaffected but DACVALA would no longer be updated from the ramp generator or DACVALS.
- The ramp generator, synchronize block and digital filter freeze on their current states.

Enabling the clock to the CMPSS restores it to the state before the clock was disabled.

18.7 Software

18.7.1 CMPSS Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/cmpss

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

18.7.1.1 CMPSS Asynchronous Trip

FILE: cmpss_ex1_asynch.c

This example enables the CMPSS1 COMPH comparator and feeds the asynchronous CTRIPOUTH signal to the GPIO14/OUTPUTXBAR3 pin and CTRIPH to GPIO13/EPWM7B.

CMPSS is configured to generate trip signals to trip the EPWM signals. CMPIN1P is used to give positive input and internal DAC is configured to provide the negative input. Internal DAC is configured to provide a signal at VDD/2. An EPWM signal is generated at GPIO13 and is configured to be tripped by CTRIPOUTH.

When a low input(VSS) is provided to CMPIN1P,
- Trip signal(GPIO14) output is low
- PWM7B(GPIO13) gives a PWM signal

When a high input(higher than VDD/2) is provided to CMPIN1P,
- Trip signal(GPIO14) output turns high
- PWM7B(GPIO13) gets tripped and outputs as high

External Connections
- Give input on CMPIN1P (The pin is shared with ADCINA2)
- Outputs can be observed on GPIO14 and GPIO13 using an oscilloscope

Watch Variables
- None

18.7.1.2 CMPSS Digital Filter Configuration

FILE: cmpss_ex2_digital_filter.c

This example enables the CMPSS1 COMPH comparator and feeds the output through the digital filter to the GPIO14/OUTPUTXBAR3 pin.

CMPIN1P is used to give positive input and internal DAC is configured to provide the negative input. Internal DAC is configured to provide a signal at VDD/2.

When a low input(VSS) is provided to CMPIN1P,
- GPIO14 output is low

When a high input(higher than VDD/2) is provided to CMPIN1P,
- GPIO14 output turns high

18.8 CMPSS Registers

This section describes the CMPSS Registers.

18.8.1 CMPSS Base Address Table

Table 18-1. CMPSS Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
Cmpss1Regs	CMPSS_REGS	CMPSS1_BASE	0x0000_5C80	YES	YES	YES	YES	YES
Cmpss2Regs	CMPSS_REGS	CMPSS2_BASE	0x0000_5CA0	YES	YES	YES	YES	YES
Cmpss3Regs	CMPSS_REGS	CMPSS3_BASE	0x0000_5CC0	YES	YES	YES	YES	YES
Cmpss4Regs	CMPSS_REGS	CMPSS4_BASE	0x0000_5CE0	YES	YES	YES	YES	YES

18.8.2 CMPSS_REGS Registers

Table 18-2 lists the memory-mapped registers for the CMPSS_REGS registers. All register offset addresses not listed in Table 18-2 should be considered as reserved locations and the register contents should not be modified.

Table 18-2. CMPSS_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	COMPCTL	CMPSS Comparator Control Register	EALLOW	Go
1h	COMPHYSCTL	CMPSS Comparator Hysteresis Control Register	EALLOW	Go
2h	COMPSTS	CMPSS Comparator Status Register		Go
3h	COMPSTSCLR	CMPSS Comparator Status Clear Register	EALLOW	Go
4h	COMPDACCTL	CMPSS DAC Control Register	EALLOW	Go
6h	DACHVALS	CMPSS High DAC Value Shadow Register		Go
7h	DACHVALA	CMPSS High DAC Value Active Register		Go
8h	RAMPMAXREFA	CMPSS Ramp Max Reference Active Register		Go
Ah	RAMPMAXREFS	CMPSS Ramp Max Reference Shadow Register		Go
Ch	RAMPDECVALA	CMPSS Ramp Decrement Value Active Register		Go
Eh	RAMPDECVALS	CMPSS Ramp Decrement Value Shadow Register		Go
10h	RAMPSTS	CMPSS Ramp Status Register		Go
12h	DACLVALS	CMPSS Low DAC Value Shadow Register		Go
13h	DACLVALA	CMPSS Low DAC Value Active Register		Go
14h	RAMPDLYA	CMPSS Ramp Delay Active Register		Go
15h	RAMPDLYS	CMPSS Ramp Delay Shadow Register		Go
16h	CTRIPLFILCTL	CTRIPL Filter Control Register	EALLOW	Go
17h	CTRIPLFILCLKCTL	CTRIPL Filter Clock Control Register	EALLOW	Go
18h	CTRIPHFILCTL	CTRIPH Filter Control Register	EALLOW	Go
19h	CTRIPHFILCLKCTL	CTRIPH Filter Clock Control Register	EALLOW	Go
1Ah	COMPLOCK	CMPSS Lock Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 18-3 shows the codes that are used for access types in this section.

Table 18-3. CMPSS_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 18-3. CMPSS_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

18.8.2.1 COMPCTL Register (Offset = 0h) [Reset = 0h]

COMPCTL is shown in Figure 18-7 and described in Table 18-4.

Return to the Summary Table.

CMPSS Comparator Control Register

Figure 18-7. COMPCTL Register

15	14	13	12	11	10	9	8
COMPDACE	ASYNCLEN	CTRIPOUTLSEL		CTRIPLSEL		COMPLINV	COMPLSOURCE
R/W-0h	R/W-0h	R/W-0h		R/W-0h		R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED	ASYNCHEN	CTRIPOUTHSEL		CTRIPHSEL		COMPHINV	COMPHSOURCE
R-0h	R/W-0h	R/W-0h		R/W-0h		R/W-0h	R/W-0h

Table 18-4. COMPCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	COMPDACE	R/W	0h	Comparator/DAC enable. 0 Comparator/DAC disabled 1 Comparator/DAC enabled Reset type: SYSRSn
14	ASYNCLEN	R/W	0h	Low comparator asynchronous path enable. Allows asynchronous comparator output to feed into OR gate with latched digital filter signal when CTRIPLSEL=3 or CTRIPOUTLSEL=3. 0 Asynchronous comparator output does not feed into OR gate with latched digital filter output 1 Asynchronous comparator output feeds into OR gate with latched digital filter output Reset type: SYSRSn
13-12	CTRIPOUTLSEL	R/W	0h	Low comparator CTRIPOUTL source select. 0 Asynchronous comparator output drives CTRIPOUTL 1 Synchronous comparator output drives CTRIPOUTL 2 Output of digital filter drives CTRIPOUTL 3 Latched output of digital filter drives CTRIPOUTL Reset type: SYSRSn
11-10	CTRIPLSEL	R/W	0h	Low comparator CTRIPL source select. 0 Asynchronous comparator output drives CTRIPL 1 Synchronous comparator output drives CTRIPL 2 Output of digital filter drives CTRIPL 3 Latched output of digital filter drives CTRIPL Reset type: SYSRSn
9	COMPLINV	R/W	0h	Low comparator output invert. 0 Output of comparator is not inverted 1 Output of comparator is inverted Reset type: SYSRSn
8	COMPLSOURCE	R/W	0h	Low comparator input source. 0 Inverting input of comparator driven by internal DAC 1 Inverting input of comparator driven through external pin Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6	ASYNCHEN	R/W	0h	High comparator asynchronous path enable. Allows asynchronous comparator output to feed into OR gate with latched digital filter signal when CTRIPHSEL=3 or CTRIPOUTHSEL=3. 0 Asynchronous comparator output does not feed into OR gate with latched digital filter output 1 Asynchronous comparator output feeds into OR gate with latched digital filter output Reset type: SYSRSn

Table 18-4. COMPCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-4	CTRIPOUTHSEL	R/W	0h	High comparator CTRIPOUTH source select. 0 Asynchronous comparator output drives CTRIPOUTH 1 Synchronous comparator output drives CTRIPOUTH 2 Output of digital filter drives CTRIPOUTH 3 Latched output of digital filter drives CTRIPOUTH Reset type: SYSRSn
3-2	CTRIPHSEL	R/W	0h	High comparator CTRIPH source select. 0 Asynchronous comparator output drives CTRIPH 1 Synchronous comparator output drives CTRIPH 2 Output of digital filter drives CTRIPH 3 Latched output of digital filter drives CTRIPH Reset type: SYSRSn
1	COMPHINV	R/W	0h	High comparator output invert. 0 Output of comparator is not inverted 1 Output of comparator is inverted Reset type: SYSRSn
0	COMPHSOURCE	R/W	0h	High comparator input source. 0 Inverting input of comparator driven by internal DAC 1 Inverting input of comparator driven through external pin Reset type: SYSRSn

18.8.2.2 COMPHYSCTL Register (Offset = 1h) [Reset = 0h]

COMPHYSCTL is shown in Figure 18-8 and described in Table 18-5.

Return to the Summary Table.

CMPSS Comparator Hysteresis Control Register

Figure 18-8. COMPHYSCTL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				COMPHYS			
R-0h				R/W-0h			

Table 18-5. COMPHYSCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3-0	COMPHYS	R/W	0h	Comparator hysteresis. Sets the amount of hysteresis on the comparator inputs. 0 None 1 Set to typical hysteresis 2 Set to 2x of typical hysteresis 3 Set to 3x of typical hysteresis 4 Set to 4x of typical hysteresis others : undefined Reset type: SYSRSn

18.8.2.3 COMPSTS Register (Offset = 2h) [Reset = 0h]

COMPSTS is shown in Figure 18-9 and described in Table 18-6.

Return to the Summary Table.

CMPSS Comparator Status Register

Figure 18-9. COMPSTS Register

15	14	13	12	11	10	9	8
RESERVED						COMPLLATCH	COMPLSTS
R-0h						R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED						COMPHLATCH	COMPHSTS
R-0h						R-0h	R-0h

Table 18-6. COMPSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9	COMPLLATCH	R	0h	Latched value of low comparator digital filter output Reset type: SYSRSn
8	COMPLSTS	R	0h	Low comparator digital filter output Reset type: SYSRSn
7-2	RESERVED	R	0h	Reserved
1	COMPHLATCH	R	0h	Latched value of high comparator digital filter output Reset type: SYSRSn
0	COMPHSTS	R	0h	High comparator digital filter output Reset type: SYSRSn

18.8.2.4 COMPSTSCLR Register (Offset = 3h) [Reset = 0h]

COMPSTSCLR is shown in Figure 18-10 and described in Table 18-7.

Return to the Summary Table.

CMPSS Comparator Status Clear Register

Figure 18-10. COMPSTSCLR Register

15	14	13	12	11	10	9	8
RESERVED					LSYNCCLREN	LLATCHCLR	RESERVED
R-0h					R/W-0h	R-0/W1S-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED					HSYNCCLREN	HLATCHCLR	RESERVED
R-0h					R/W-0h	R-0/W1S-0h	R-0h

Table 18-7. COMPSTSCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R	0h	Reserved
10	LSYNCCLREN	R/W	0h	Low comparator latch EPWMSYNCPER clear. Enable EPWMSYNCPER reset of low comparator digital filter output latch COMPSTS[COMPLLATCH]. 0 EPWMSYNCPER will not reset latch 1 EPWMSYNCPER will reset latch Reset type: SYSRSn
9	LLATCHCLR	R-0/W1S	0h	Low comparator latch software clear. Perform software reset of low comparator digital filter output latch COMPSTS[COMPLLATCH]. Reads always return 0. 0 No effect 1 Generate a single pulse of latch reset signal for COMPSTS[COMPLLATCH] Reset type: SYSRSn
8-3	RESERVED	R	0h	Reserved
2	HSYNCCLREN	R/W	0h	High comparator latch EPWMSYNCPER clear. Enable EPWMSYNCPER reset of high comparator digital filter output latch COMPSTS[COMPHLATCH]. 0 EPWMSYNCPER will not reset latch 1 EPWMSYNCPER will reset latch Reset type: SYSRSn
1	HLATCHCLR	R-0/W1S	0h	High comparator latch software clear. Perform software reset of high comparator digital filter output latch COMPSTS[COMPHLATCH]. Reads always return 0. 0 No effect 1 Generate a single pulse of latch reset signal for COMPSTS[COMPHLATCH] Reset type: SYSRSn
0	RESERVED	R	0h	Reserved

18.8.2.5 COMPDACCTL Register (Offset = 4h) [Reset = 0h]

COMPDACCTL is shown in Figure 18-11 and described in Table 18-8.

Return to the Summary Table.

CMPSS DAC Control Register

Figure 18-11. COMPDACCTL Register

15	14	13	12	11	10	9	8
FREESOFT		RESERVED	BLANKEN	BLANKSOURCE			
R/W-0h		R-0h	R/W-0h	R/W-0h			
7	6	5	4	3	2	1	0
SWLOADSEL	RAMPLOADSEL	SELREF	RAMPSOURCE				DACSOURCE
R/W-0h	R/W-0h	R/W-0h	R/W-0h				R/W-0h

Table 18-8. COMPDACCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	FREESOFT	R/W	0h	Free-run or software-run emulation behavior. Behavior of the ramp generator during emulation suspend. 00b Ramp generator stops immediately during emulation suspend 01b Ramp generator completes current ramp and stops at next EPWMSYNCPER during emulation suspend 1Xb Ramp generator runs freely Reset type: SYSRSn
13	RESERVED	R	0h	Reserved
12	BLANKEN	R/W	0h	EPWMBLANK enable. This bit enables the EPWMBLANK signal. 0 EPWMBLANK signal is disabled. 1 EPWMBLANK signal is enabled. Reset type: SYSRSn
11-8	BLANKSOURCE	R/W	0h	EPWMBLANK source select. This bit field determines which EPWMnBLANK is passed on as the EPWMBLANK signal. Where n represents the maximum number of EPWMBLANK signals available on the device: 0 EPWM1BLANK 1 EPWM2BLANK 2 EPWM3BLANK ... n-1 EPWMnBLANK Reset type: SYSRSn
7	SWLOADSEL	R/W	0h	Software load select. Determines whether DACxVALA is updated from DACxVALS on SYSCLK or EPWMSYNCPER. 0 DACxVALA is updated from DACxVALS on SYSCLK 1 DACxVALA is updated from DACxVALS on EPWMSYNCPER Reset type: SYSRSn
6	RAMPLOADSEL	R/W	0h	Ramp load select. Determines whether RAMPSTS is updated from RAMPMAXREFA or RAMPMAXREFS when COMPSTS[COMPHSTS] is triggered. 0 RAMPSTS is loaded from RAMPMAXREFA 1 RAMPSTS is loaded from RAMPMAXREFS Reset type: SYSRSn
5	SELREF	R/W	0h	DAC reference select. Determines which voltage supply is used as the reference for the internal comparator DACs. 0 VDDA is the voltage reference for the DAC 1 VDAC is the voltage reference for the DAC Reset type: SYSRSn

Table 18-8. COMPDACCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-1	RAMPSOURCE	R/W	0h	EPWMSYNCPER source select. Determines which EPWMnSYNCPER signal is used within the CMPSS module. Where n represents the maximum number of EPWMSYNCPER signals available on the device: 0 EPWM1SYNCPER 1 EPWM2SYNCPER 2 EPWM3SYNCPER ... n-1 EPWMnSYNCPER Reset type: SYSRSn
0	DACSOURCE	R/W	0h	DAC source select. Determines whether DACHVALA is updated from DACHVALS or from the ramp generator. 0 DACHVALA is updated from DACHVALS 1 DACHVALA is updated from the ramp generator Reset type: SYSRSn

18.8.2.6 DACHVALS Register (Offset = 6h) [Reset = 0h]

DACHVALS is shown in Figure 18-12 and described in Table 18-9.

Return to the Summary Table.

CMPSS High DAC Value Shadow Register

Figure 18-12. DACHVALS Register

15	14	13	12	11	10	9	8
RESERVED				DACVAL			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
DACVAL							
R/W-0h							

Table 18-9. DACHVALS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-0	DACVAL	R/W	0h	High DAC shadow value. When COMPDACCTL[DACSOURCE]=0, the value of DACHVALS is loaded into DACHVALA on the trigger signal selected by COMPDACCTL[SWLOADSEL]. Reset type: SYSRSn

18.8.2.7 DACHVALA Register (Offset = 7h) [Reset = 0h]

DACHVALA is shown in Figure 18-13 and described in Table 18-10.

Return to the Summary Table.

CMPSS High DAC Value Active Register

Figure 18-13. DACHVALA Register

15	14	13	12	11	10	9	8
RESERVED				DACVAL			
R-0h				R-0h			
7	6	5	4	3	2	1	0
DACVAL							
R-0h							

Table 18-10. DACHVALA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-0	DACVAL	R	0h	High DAC active value. Value that is actively driven by the high DAC. Reset type: SYSRSn

18.8.2.8 RAMPMAXREFA Register (Offset = 8h) [Reset = 0h]

RAMPMAXREFA is shown in Figure 18-14 and described in Table 18-11.

Return to the Summary Table.

CMPSS Ramp Max Reference Active Register

Figure 18-14. RAMPMAXREFA Register

15	14	13	12	11	10	9	8
RAMPMAXREF							
R-0h							
7	6	5	4	3	2	1	0
RAMPMAXREF							
R-0h							

Table 18-11. RAMPMAXREFA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RAMPMAXREF	R	0h	Ramp maximum reference active value. Latched value to be loaded into ramp generator RAMPSTS. Reset type: SYSRSn

18.8.2.9 RAMPMAXREFS Register (Offset = Ah) [Reset = 0h]

RAMPMAXREFS is shown in Figure 18-15 and described in Table 18-12.

Return to the Summary Table.

CMPSS Ramp Max Reference Shadow Register

Figure 18-15. RAMPMAXREFS Register

15	14	13	12	11	10	9	8
RAMPMAXREF							
R/W-0h							
7	6	5	4	3	2	1	0
RAMPMAXREF							
R/W-0h							

Table 18-12. RAMPMAXREFS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RAMPMAXREF	R/W	0h	Ramp maximum reference shadow. Unlatched value to be loaded into ramp generator RAMPSTS. Reset type: SYSRSn

18.8.2.10 RAMPDECVALA Register (Offset = Ch) [Reset = 0h]

RAMPDECVALA is shown in Figure 18-16 and described in Table 18-13.

Return to the Summary Table.

CMPSS Ramp Decrement Value Active Register

Figure 18-16. RAMPDECVALA Register

15	14	13	12	11	10	9	8
RAMPDECVAL							
R-0h							
7	6	5	4	3	2	1	0
RAMPDECVAL							
R-0h							

Table 18-13. RAMPDECVALA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RAMPDECVAL	R	0h	Ramp decrement value active. Latched value that will be subtracted from RAMPSTS. Reset type: SYSRSn

18.8.2.11 RAMPDECVALS Register (Offset = Eh) [Reset = 0h]

RAMPDECVALS is shown in Figure 18-17 and described in Table 18-14.

Return to the Summary Table.

CMPSS Ramp Decrement Value Shadow Register

Figure 18-17. RAMPDECVALS Register

15	14	13	12	11	10	9	8
RAMPDECVAL							
R/W-0h							
7	6	5	4	3	2	1	0
RAMPDECVAL							
R/W-0h							

Table 18-14. RAMPDECVALS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RAMPDECVAL	R/W	0h	Ramp decrement value shadow. Unlatched value to be loaded into RAMPDECVALA. Reset type: SYSRSn

18.8.2.12 RAMPSTS Register (Offset = 10h) [Reset = 0h]

RAMPSTS is shown in Figure 18-18 and described in Table 18-15.

Return to the Summary Table.

CMPSS Ramp Status Register

Figure 18-18. RAMPSTS Register

15	14	13	12	11	10	9	8
RAMPVALUE							
R-0h							
7	6	5	4	3	2	1	0
RAMPVALUE							
R-0h							

Table 18-15. RAMPSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RAMPVALUE	R	0h	Ramp value. Present value of ramp generator. Reset type: SYSRSn

18.8.2.13 DACLVALS Register (Offset = 12h) [Reset = 0h]

DACLVALS is shown in Figure 18-19 and described in Table 18-16.

Return to the Summary Table.

CMPSS Low DAC Value Shadow Register

Figure 18-19. DACLVALS Register

15	14	13	12	11	10	9	8
RESERVED				DACVAL			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
DACVAL							
R/W-0h							

Table 18-16. DACLVALS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-0	DACVAL	R/W	0h	Low DAC shadow value. value to be loaded into DACLVALA on the trigger signal selected by COMPDACCTL[SWLOADSEL]. Reset type: SYSRSn

18.8.2.14 DACLVALA Register (Offset = 13h) [Reset = 0h]

DACLVALA is shown in Figure 18-20 and described in Table 18-17.

Return to the Summary Table.

CMPSS Low DAC Value Active Register

Figure 18-20. DACLVALA Register

15	14	13	12	11	10	9	8
RESERVED				DACVAL			
R-0h				R-0h			
7	6	5	4	3	2	1	0
DACVAL							
R-0h							

Table 18-17. DACLVALA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-0	DACVAL	R	0h	Low DAC active value. Value that is actively driven by the low DAC. Reset type: SYSRSn

18.8.2.15 RAMPDLYA Register (Offset = 14h) [Reset = 0h]

RAMPDLYA is shown in Figure 18-21 and described in Table 18-18.

Return to the Summary Table.

CMPSS Ramp Delay Active Register

Figure 18-21. RAMPDLYA Register

15	14	13	12	11	10	9	8
RESERVED			DELAY				
R-0h			R-0h				
7	6	5	4	3	2	1	0
DELAY							
R-0h							

Table 18-18. RAMPDLYA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12-0	DELAY	R	0h	Ramp delay active value. Latched value of the number of cycles to delay the start of the ramp generator decrementer after a EPWMSYNCPER is received. Reset type: SYSRSn

18.8.2.16 RAMPDLYS Register (Offset = 15h) [Reset = 0h]

RAMPDLYS is shown in Figure 18-22 and described in Table 18-19.

Return to the Summary Table.

CMPSS Ramp Delay Shadow Register

Figure 18-22. RAMPDLYS Register

15	14	13	12	11	10	9	8
RESERVED			DELAY				
R-0h			R/W-0h				
7	6	5	4	3	2	1	0
DELAY							
R/W-0h							

Table 18-19. RAMPDLYS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12-0	DELAY	R/W	0h	Ramp delay shadow value. Unlatched value to be loaded into RAMPDLYA. Reset type: SYSRSn

18.8.2.17 CTRIPLFILCTL Register (Offset = 16h) [Reset = 0h]

CTRIPLFILCTL is shown in Figure 18-23 and described in Table 18-20.

Return to the Summary Table.

CTRIPL Filter Control Register

Figure 18-23. CTRIPLFILCTL Register

15	14	13	12	11	10	9	8
FILINIT	RESERVED	THRESH					SAMPWIN
R-0/W1S-0h	R-0h	R/W-0h					R/W-0h
7	6	5	4	3	2	1	0
SAMPWIN				RESERVED			
R/W-0h				R-0h			

Table 18-20. CTRIPLFILCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	FILINIT	R-0/W1S	0h	Low filter initialization. 0 No effect 1 Initialize all samples to the filter input value Reset type: SYSRSn
14	RESERVED	R	0h	Reserved
13-9	THRESH	R/W	0h	Low filter majority voting threshold. At least THRESH samples of the opposite state must appear within the sample window in order for the output to change state. Threshold used is THRESH+1. Reset type: SYSRSn
8-4	SAMPWIN	R/W	0h	Low filter sample window size. Number of samples to monitor is SAMPWIN+1. Reset type: SYSRSn
3-0	RESERVED	R	0h	Reserved

18.8.2.18 CTRIPLFILCLKCTL Register (Offset = 17h) [Reset = 0h]

CTRIPLFILCLKCTL is shown in Figure 18-24 and described in Table 18-21.

Return to the Summary Table.

CTRIPL Filter Clock Control Register

Figure 18-24. CTRIPLFILCLKCTL Register

15	14	13	12	11	10	9	8
CLKPRESCALE							
R/W-0h							
7	6	5	4	3	2	1	0
CLKPRESCALE							
R/W-0h							

Table 18-21. CTRIPLFILCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	CLKPRESCALE	R/W	0h	Low filter sample clock prescale. Number of system clocks between samples is CLKPRESCALE+1. Reset type: SYSRSn

18.8.2.19 CTRIPHFILCTL Register (Offset = 18h) [Reset = 0h]

CTRIPHFILCTL is shown in Figure 18-25 and described in Table 18-22.

Return to the Summary Table.

CTRIPH Filter Control Register

Figure 18-25. CTRIPHFILCTL Register

15	14	13	12	11	10	9	8
FILINIT	RESERVED	THRESH					SAMPWIN
R-0/W1S-0h	R-0h	R/W-0h					R/W-0h
7	6	5	4	3	2	1	0
SAMPWIN				RESERVED			
R/W-0h				R-0h			

Table 18-22. CTRIPHFILCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	FILINIT	R-0/W1S	0h	High filter initialization. 0 No effect 1 Initialize all samples to the filter input value Reset type: SYSRSn
14	RESERVED	R	0h	Reserved
13-9	THRESH	R/W	0h	High filter majority voting threshold. At least THRESH samples of the opposite state must appear within the sample window in order for the output to change state. Threshold used is THRESH+1. Reset type: SYSRSn
8-4	SAMPWIN	R/W	0h	High filter sample window size. Number of samples to monitor is SAMPWIN+1. Reset type: SYSRSn
3-0	RESERVED	R	0h	Reserved

18.8.2.20 CTRIPHFILCLKCTL Register (Offset = 19h) [Reset = 0h]

CTRIPHFILCLKCTL is shown in Figure 18-26 and described in Table 18-23.

Return to the Summary Table.

CTRIPH Filter Clock Control Register

Figure 18-26. CTRIPHFILCLKCTL Register

15	14	13	12	11	10	9	8
CLKPRESCALE							
R/W-0h							
7	6	5	4	3	2	1	0
CLKPRESCALE							
R/W-0h							

Table 18-23. CTRIPHFILCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	CLKPRESCALE	R/W	0h	High filter sample clock prescale. Number of system clocks between samples is CLKPRESCALE+1. Reset type: SYSRSn

18.8.2.21 COMPLOCK Register (Offset = 1Ah) [Reset = 0h]

COMPLOCK is shown in Figure 18-27 and described in Table 18-24.

Return to the Summary Table.

CMPSS Lock Register

Figure 18-27. COMPLOCK Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			RESERVED	CTRIP	DACCTL	COMPHYSCTL	COMPCTL
R-0h			R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 18-24. COMPLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R	0h	Reserved
4	RESERVED	R/WSonce	0h	Reserved
3	CTRIP	R/WSonce	0h	Lock write-access to the CTRIPxFILTCTL and CTRIPxFILCLKCTL* registers. 0 CTRIPxFILCTL and CTRIPxFILCLKCTL* registers are not locked. Write 0 to this bit has no effect. 1 CTRIPxFILCTL and CTRIPxFILCLKCTL* registers are locked. Only a system reset can clear this bit. Reset type: SYSRSn
2	DACCTL	R/WSonce	0h	Lock write-access to the COMPDAC*CTL* register(s). 0 COMPDAC*CTL* register(s) not locked. Write 0 to this bit has no effect. 1 COMPDAC*CTL* register(s) locked. Only a system reset can clear this bit. Reset type: SYSRSn
1	COMPHYSCTL	R/WSonce	0h	Lock write-access to the COMPHYSCTL register. 0 COMPHYSCTL register is not locked. Write 0 to this bit has no effect. 1 COMPHYSCTL register is locked. Only a system reset can clear this bit. Reset type: SYSRSn
0	COMPCTL	R/WSonce	0h	Lock write-access to the COMPCTL register. 0 COMPCTL register is not locked. Write 0 to this bit has no effect. 1 COMPCTL register is locked. Only a system reset can clear this bit. Reset type: SYSRSn

18.8.3 CMPSS Registers to Driverlib Functions

Table 18-25. CMPSS Registers to Driverlib Functions

File	Driverlib Function
COMPCTL	
cmpss.h	CMPSS_enableModule
cmpss.h	CMPSS_disableModule
cmpss.h	CMPSS_configHighComparator
cmpss.h	CMPSS_configLowComparator
cmpss.h	CMPSS_configOutputsHigh
cmpss.h	CMPSS_configOutputsLow
COMPHYSCTL	

Table 18-25. CMPSS Registers to Driverlib Functions (continued)

File	Driverlib Function
cmpss.h	CMPSS_setHysteresis
COMPSTS	
cmpss.c	CMPSS_configLatchOnPWMSYNC
cmpss.h	CMPSS_getStatus
cmpss.h	CMPSS_clearFilterLatchHigh
cmpss.h	CMPSS_clearFilterLatchLow
cmpss.h	CMPSS_enableLatchResetOnPWMSYNCHigh
cmpss.h	CMPSS_disableLatchResetOnPWMSYNCHigh
cmpss.h	CMPSS_enableLatchResetOnPWMSYNCLow
cmpss.h	CMPSS_disableLatchResetOnPWMSYNCLow
COMPSTSCLR	
cmpss.c	CMPSS_configLatchOnPWMSYNC
cmpss.h	CMPSS_clearFilterLatchHigh
cmpss.h	CMPSS_clearFilterLatchLow
cmpss.h	CMPSS_enableLatchResetOnPWMSYNCHigh
cmpss.h	CMPSS_disableLatchResetOnPWMSYNCHigh
cmpss.h	CMPSS_enableLatchResetOnPWMSYNCLow
cmpss.h	CMPSS_disableLatchResetOnPWMSYNCLow
COMPDACCTL	
cmpss.c	CMPSS_configRamp
cmpss.h	CMPSS_configDAC
cmpss.h	CMPSS_configBlanking
cmpss.h	CMPSS_enableBlanking
cmpss.h	CMPSS_disableBlanking
DACHVALS	
cmpss.h	CMPSS_setDACValueHigh
DACHVALA	
cmpss.h	CMPSS_getDACValueHigh
RAMPMAXREFA	
cmpss.h	CMPSS_getMaxRampValue
RAMPMAXREFS	
cmpss.c	CMPSS_configRamp
cmpss.h	CMPSS_setMaxRampValue
RAMPDECVALA	
cmpss.h	CMPSS_getRampDecValue
RAMPDECVALS	
cmpss.c	CMPSS_configRamp
cmpss.h	CMPSS_setRampDecValue
RAMPSTS	
-	
DACLVALS	
cmpss.h	CMPSS_setDACValueLow
DACLVALA	
cmpss.h	CMPSS_getDACValueLow
RAMPDLYA	

Table 18-25. CMPSS Registers to Driverlib Functions (continued)

File	Driverlib Function
cmpss.h	CMPSS_getRampDelayValue
RAMPDLYS	
cmpss.c	CMPSS_configRamp
cmpss.h	CMPSS_setRampDelayValue
CTRIPLFILCTL	
cmpss.c	CMPSS_configFilterLow
cmpss.h	CMPSS_initFilterLow
CTRIPLFILCLKCTL	
cmpss.c	CMPSS_configFilterLow
CTRIPHFILCTL	
cmpss.c	CMPSS_configFilterHigh
cmpss.h	CMPSS_initFilterHigh
CTRIPHFILCLKCTL	
cmpss.c	CMPSS_configFilterHigh
COMPLOCK	
-	

This page intentionally left blank.

Chapter 19

Sigma Delta Filter Module (SDFM)

The sigma delta filter module (SDFM) is a four-channel digital filter designed specifically for current measurement and resolver position decoding in motor control applications. Each input channel can receive an independent delta-sigma (ΔΣ) modulator bit stream. The bit streams are processed by four individually-programmable digital decimation filters. The filter set includes a fast comparator (secondary filter) for immediate digital threshold comparisons for over-current and under-current monitoring, and zeros crossing detection.

19.1 Introduction

Figure 19-1 shows the SDFM CPU Interface.

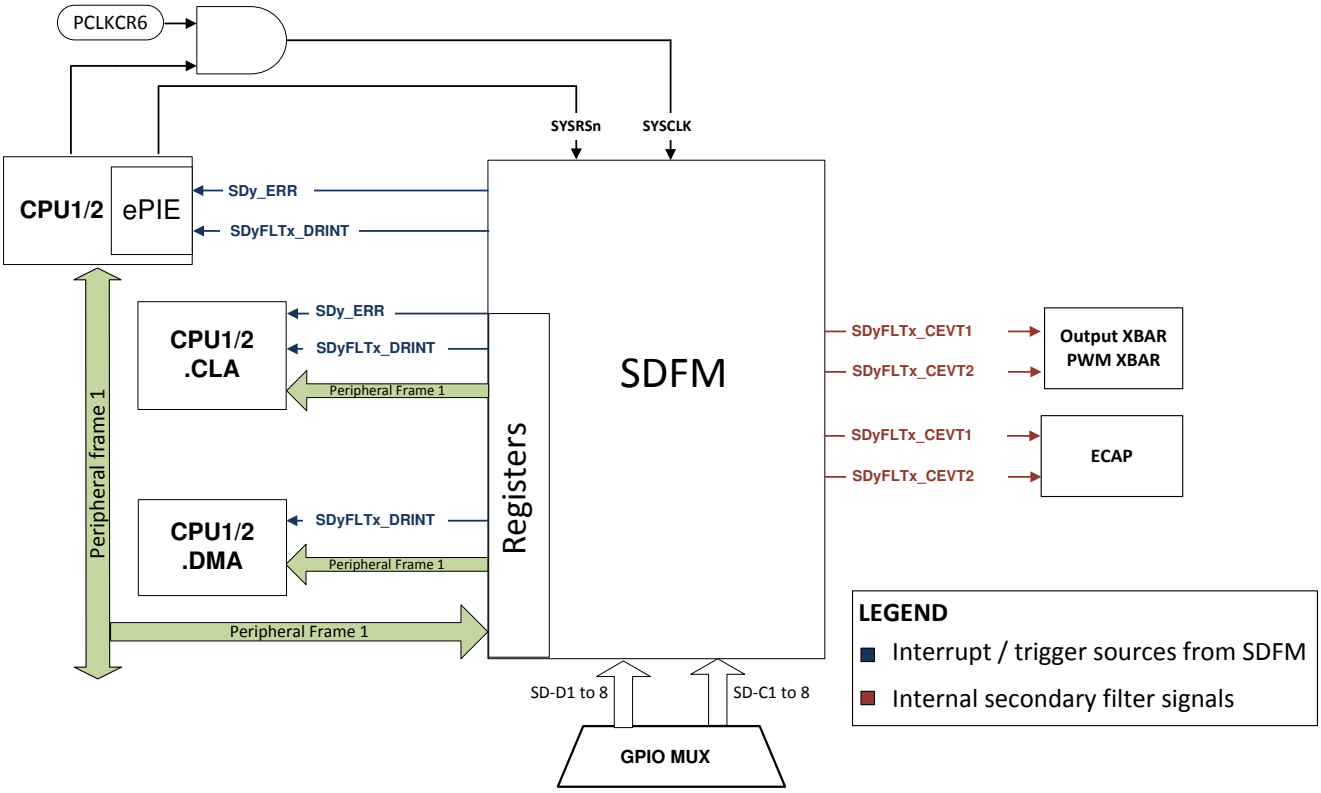


Figure 19-1. Sigma Delta Filter Module (SDFM) CPU Interface

19.1.1 SDFM Related Collateral

Foundational Materials

- *How delta-sigma ADCs work, Part 1 Application Report*
- *How delta-sigma ADCs work, Part 2 Application Report*
- Nuts and Bolts of the Delta-Sigma Converter (Video)

Getting Started Materials

- *Achieving Better Signal Integrity With Isolated Delta-Sigma Modulators in Motor Drives Application Report*
  - Critical information for a hardware designer
- Using Sigma Delta Filter Module (SDFM) to Measure the Analog Input Signal
  - NOTE: This is a non-TI (third party) site.

Expert Materials

- C2000 DesignDRIVE Development Kit for Industrial Motor Control
- *Isolated Current Shunt and Voltage Measurement Kit Application Report*
- Isolated, Shunt-Based Current Sensing Reference Design
- *Quick Response Control of PMSM Using Fast Current Loop Application Report*
- Single-Phase Inverter Reference Design With Voltage Source and Grid Connected Modes
- The case for isolated delta-sigma modulators: Is my system fast enough for short-circuit detection?
- Vienna Rectifier-Based Three Phase Power Factor Correction Reference Design Using C2000 MCU

19.1.2 Features

SDFM features include:

- Eight external pins per SDFM module
  - Four sigma-delta data input pins per SDFM module (SD-Dx, where x = 1 to 4)
  - Four sigma-delta clock input pins per SDFM module (SD-Cx, where x = 1 to 4)
- Different configurable modulator clock modes supported:
  - Mode 0: Modulator clock rate equals the modulator data rate.
- Four independent, configurable secondary filter (comparator) units per SDFM module:
  - Four different filter type selection (Sinc1/Sinc2/SincFast/Sinc3) options available
  - Ability to detect over-value condition, under-value condition, and Threshold-crossing conditions
    1. Two independent Higher Threshold comparators (used to detect over-value condition)
    2. Two independent Lower Threshold comparators (used to detect under-value condition)
    3. One independent Threshold-Crossing comparator (used to measure duty cycle/frequency with eCAP)
  - OSR value for comparator filter unit (COSR) programmable from 1 to 32

Four independent configurable primary filter (data filter) units per SDFM module:

- Four different filter type selection (Sinc1/Sinc2/SincFast/Sinc3) options available
- OSR value for data filter unit (DOSR) programmable from 1 to 256
- Ability to enable or disable (or both) individual filter module
- Ability to synchronize all four independent filters of an SDFM module by using the Main Filter Enable (MFE) bit or by using PWM signals
- Data filter output can be represented in either 16 bits or 32 bits.
- Data filter unit has a programmable mode FIFO to reduce interrupt overhead. The FIFO has the following features:
  - The primary filter (data filter) has a 16-deep x 32-bit FIFO.
  - The FIFO can interrupt the CPU after programmable number of data-ready events.
  - FIFO Wait-for-Sync feature: Ability to ignore data-ready events until the PWM synchronization signal (SDSYNC) is received. Once the SDSYNC event is received, the FIFO is populated on every data-ready event.
  - Data filter output can be represented in either 16 bits or 32 bits.
- PWMx.SOCA/SOCB can be configured to serve as SDSYNC source on a per-data-filter-channel basis.
- PWMs can be used to generate a modulator clock for sigma-delta modulators.
- Configurable Input Qualification available for both SD-Cx and SD-Dx
- Ability to use one filter channel clock (SD-C1) to provide clock to other filter clock channels.
- Configurable digital filter available on comparator filter events to blankout comparator events caused by spurious noise

19.1.3 Block Diagram

Each SDFM module has four independent filter modules. These filter modules are identical and can be configured independently. Each individual filter module has the following units:

- Input control unit
- Primary filter (data filter) unit
- Secondary filter (comparator filter) unit with 4 independent comparators

Figure 19-2 shows the SDFM module block diagram. The SDFM port operation is configured and controlled by the registers listed in Section 19.12.

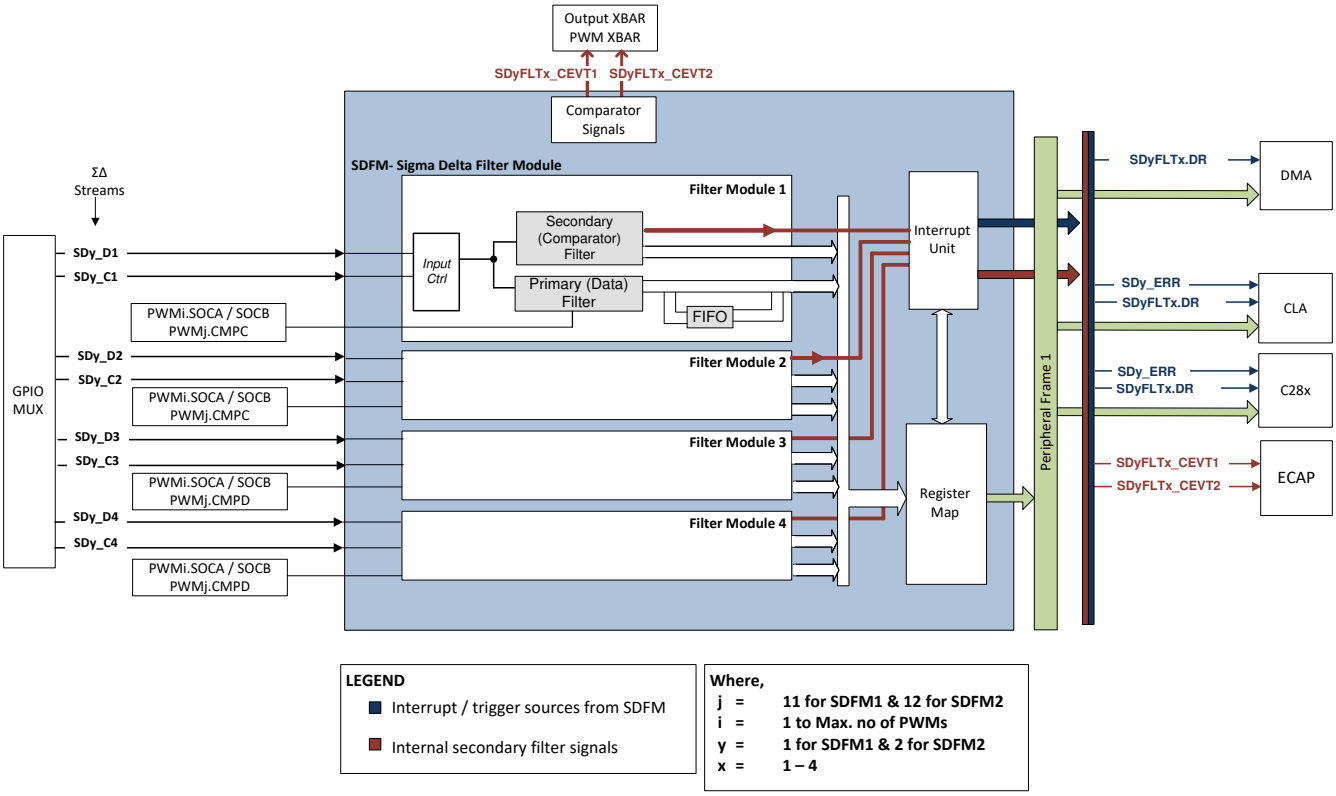


Figure 19-2. Sigma Delta Filter Module (SDFM) Block Diagram

Each filter module shown in Figure 19-3 has a primary (data) filter and a secondary (comparator) filter pair that receives the same bit stream. Except for the input bit stream, both the primary and secondary filter are completely independent of each other. Each of these filter modules can be independently configured. So, in a SDFM module, there is a total of four primary filters and four secondary filters.

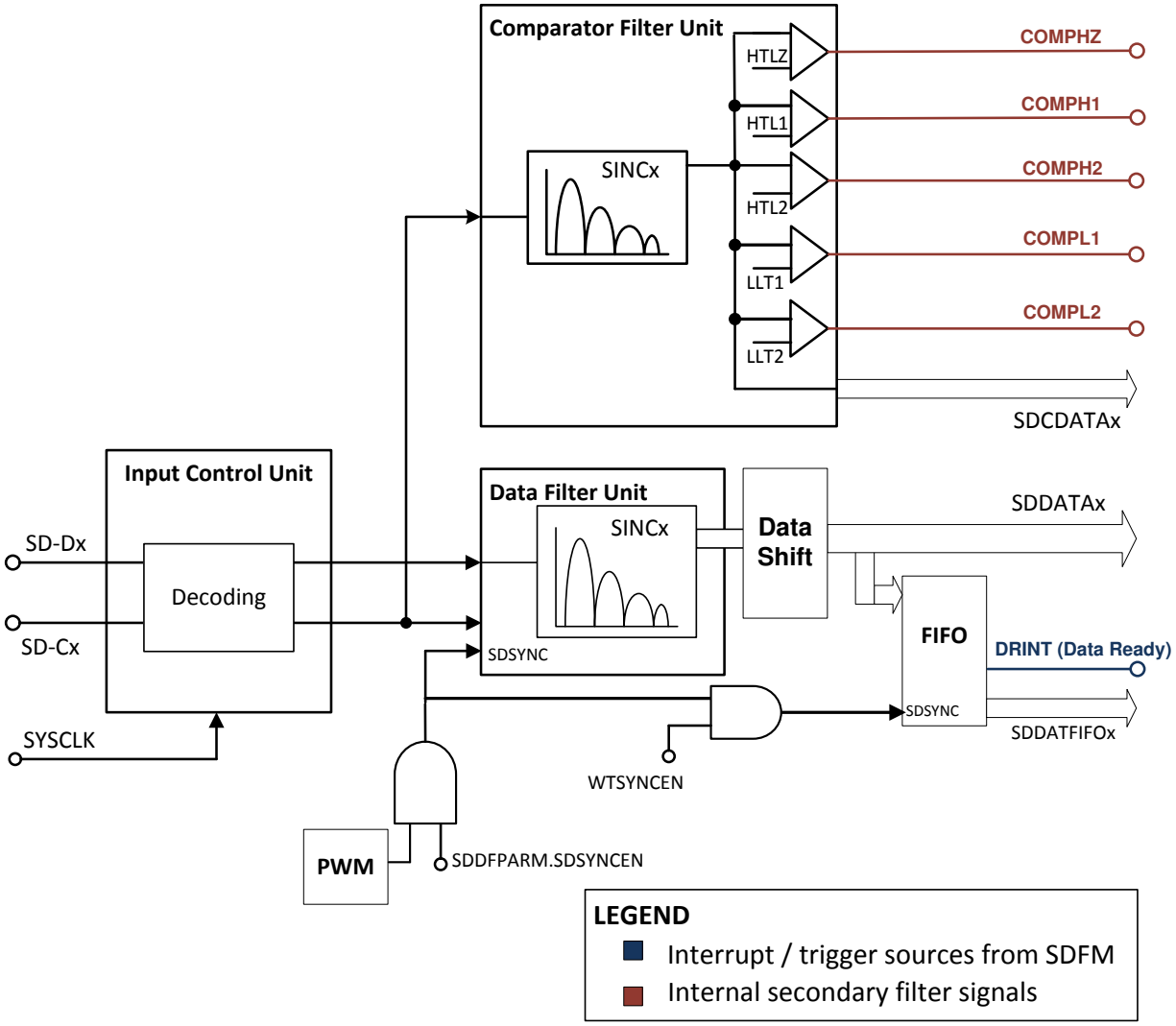


Figure 19-3. Block Diagram of One Filter Module

19.2 Configuring Device Pins

The GPIO mux registers must be configured to connect this peripheral to the device pins. To avoid glitches on the pins, the GPyGMUX bits must be configured first (while keeping the corresponding GPyMUX bits at the default of zero), followed by writing the GPyMUX register to the desired value.

For proper SDFM operation, the following GPIO input qualification can be used. Other GPIO qualifications are not supported.

- If GPIO Input qualification is ASYNC, make sure to check the SDFM Electrical Data and Timing (Using ASYNC) requirement is met and be aware of the following caution message.

CAUTION

The SDFM clock inputs (SDx_Cy pins) directly clock the SDFM module. Any glitches or ringing noise on these inputs can corrupt the SDFM module operation. Special precautions should be taken on these signals to ensure a clean and noise-free signal that meets SDFM timing requirements. Precautions such as series termination for ringing due to any impedance mismatch of the clock driver and spacing of traces from other noisy signals are recommended.

Note

The SDFM module expects SD-Dx to change on the falling edge of SD-Cx and strobes for SD-Dx on the rising edge. But some SD-modulators in the market change SD-Dx on the rising edge and expect SDFM to strobe for data on the falling edge. In such cases, the GPIO inversion feature (GPxINV) is used on SD-Cx pin to change polarity and make it compatible with the SDFM.

See the *General-Purpose Input/Output (GPIO)* chapter for more details on GPIO mux and settings.

19.3 Input Qualification

Impulse noise sources such as EMI, crosstalk, and so on, has the possibility of corrupting SDCLK and SDDATA bit streams, resulting in corrupted SDFM filtered data. This impulse noise effects could be mitigated when using the input qualification feature which synchronizes SDCLK / SDDATA signals with PLLCLK.
By default, both SDCLK and SDDATA bit stream are not synchronized. SDCLK can be synchronized to PLLCLK by setting SDCTLPARMx.SDCLKSYNC = 1 and SDDATA can be synchronized to PLLCLK by setting SDCTLPARMx.SDDATASYNC = 1. Figure 19-4 shows optional Input Qualification option on SDCLK and SDDATA lines.

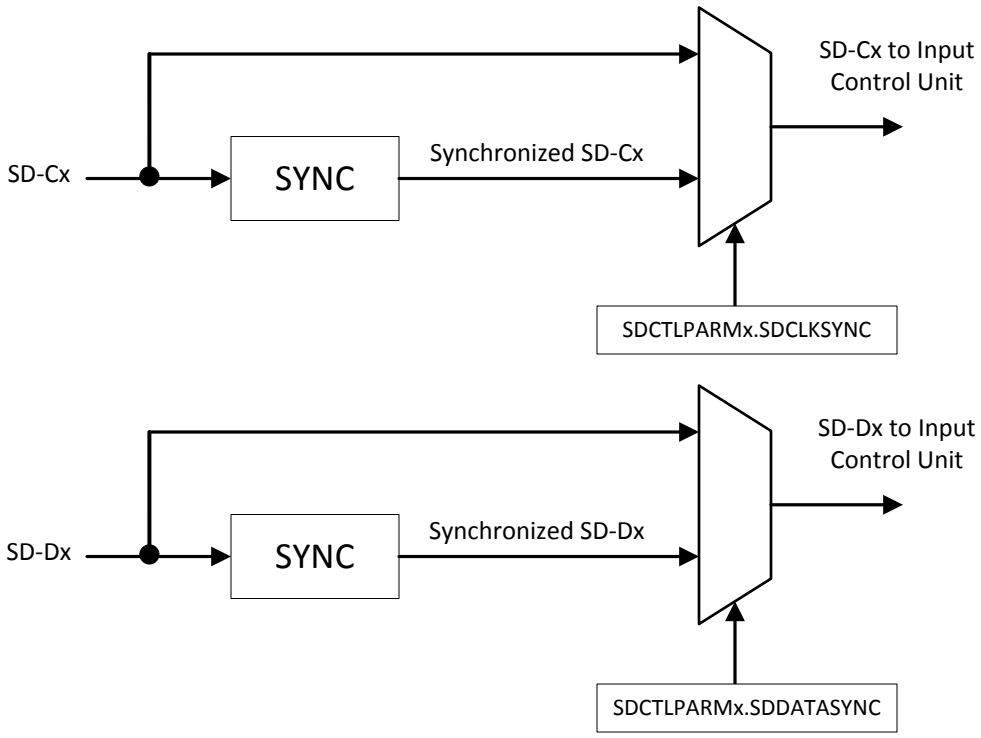


Figure 19-4. Input Qualification on SD-Cx and SD-Dx

19.4 Input Control Unit

The input control unit receives sigma delta modulated data and a sigma delta modulated clock. The modulated data received is captured and passed on to the data filter unit and comparator unit. This unit can be configured to receive the modulated data in Mode 0. Table 19-1 and Figure 19-5 show how SDCTLPARMx.MOD bits can be configured in Mode 0.

Table 19-1. Modulator Clock Modes

Modulator Mode [MOD]	Description
0	The modulator clock is running with the modulator data rate. The modulator data is strobed at every rising edge of the modulator clock.
1	Reserved
2	Reserved
3	Reserved

Note

In order to achieve the maximum value, the sigma-delta modulator has to be operated at absolute maximum positive or negative full scale, which is outside of the recommended full scale range of 80% of most sigma-delta modulators.

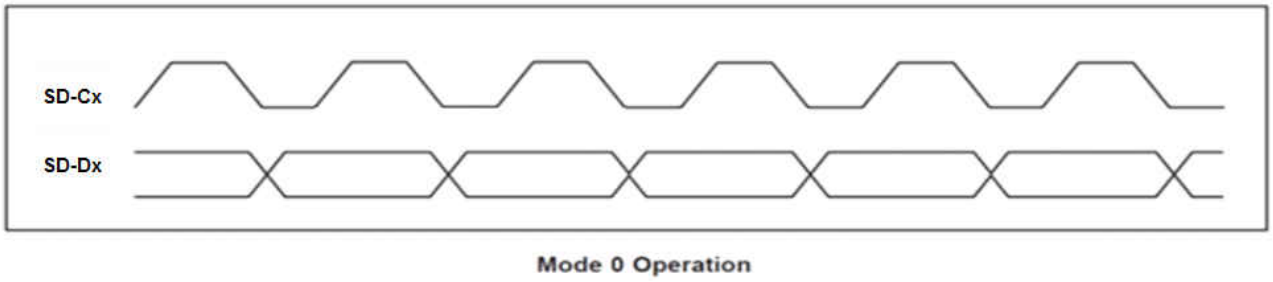


Figure 19-5. Different Modulator Modes Supported

19.5 SDFM Clock Control

In systems, the modulator clock could be generated using PWMs. Assuming all the SD-CLKs see the same delay on board traces, you could potentially use just one clock to clock multiple filters, thereby saving on the number of pins used for SDFM. In order to enable this Filter1 SDCLK (SD-C1) can possibly fed to other filter channels if required. SDCTLPARAMx.SDCLKSEL register bit field can be configured to select filter channel SDCLK. See Figure 19-6 to view this feature.

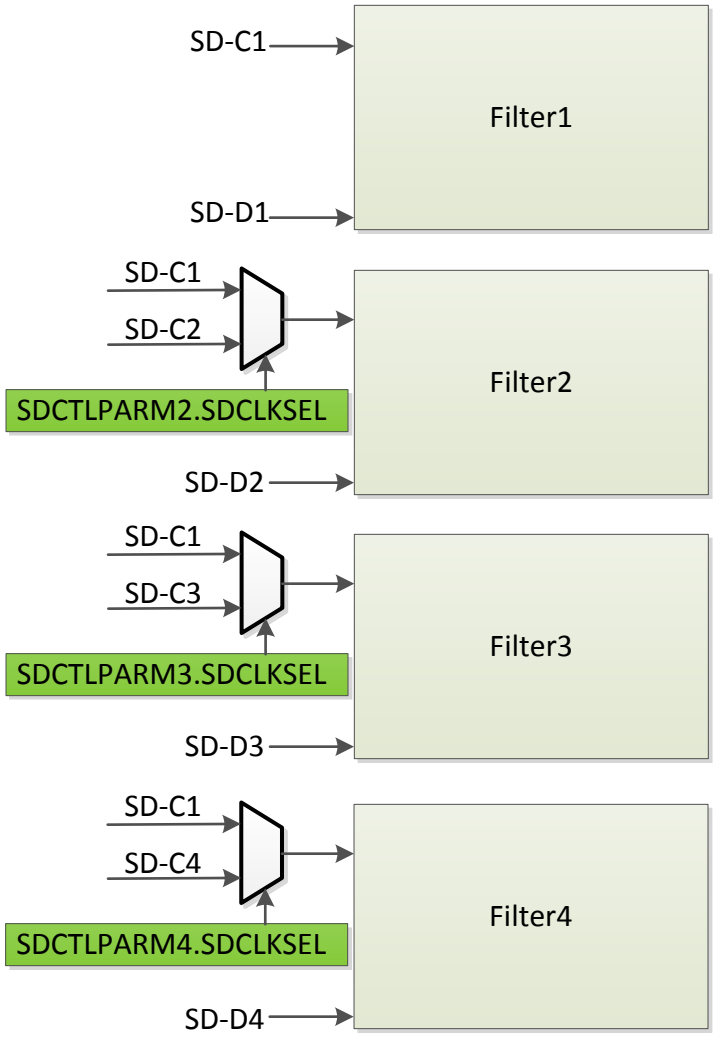


Figure 19-6. SDFM Clock Control

19.6 Sinc Filter

Both comparator filter and data filter available in SDFM have the $Sinc^N$ filter as its core. The $Sinc^N$ filter is essentially a low-pass filter that converts the input bit stream into digital data by digital filtering and decimation. This filtered digital data represents analog input given to the sigma delta modulator. Simplified $Sinc^N$ architecture consists of cascaded integrators and differentiators separated by a down-sampler as shown in Figure 19-7. The Z-transfer function of the Sinc filter of order N is shown in Figure 19-8.

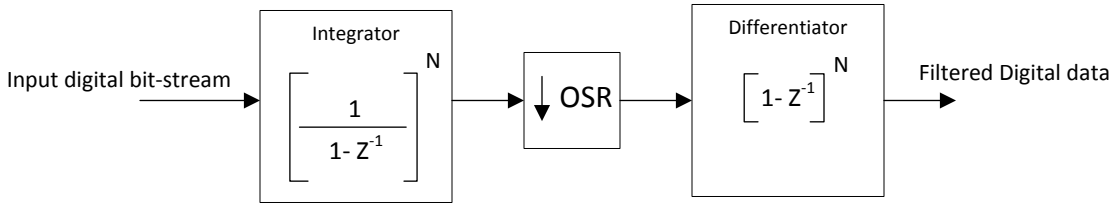


Figure 19-7. Simplified Sinc Filter Architecture

$$H(Z) = \left[\frac{1 - Z^{-OSR}}{1 - Z^{-1}} \right]^{N}$$

N = Order of Sinc filter
OSR = Over Sampling Ratio

Figure 19-8. Z-Transform of Sinc Filter of Order N

Effective resolution of the Sinc filter (ENOB) depends upon filter type, OSR and sigma-delta modulator frequency. Typically, higher resolution (or) ENOB can be achieved by higher OSR for a given filter type; however, the tradeoff is increased filter delay. It is important to choose the right sigma delta modulator by studying the optimal speed versus resolution tradeoff. Refer to the corresponding sigma delta modulator datasheet to determine the effective resolution for a given Sinc filter configuration. Figure 19-9 shows the frequency response of different filter structures when OSR = 32 and when the sigma delta modulator frequency is 10 MHz.

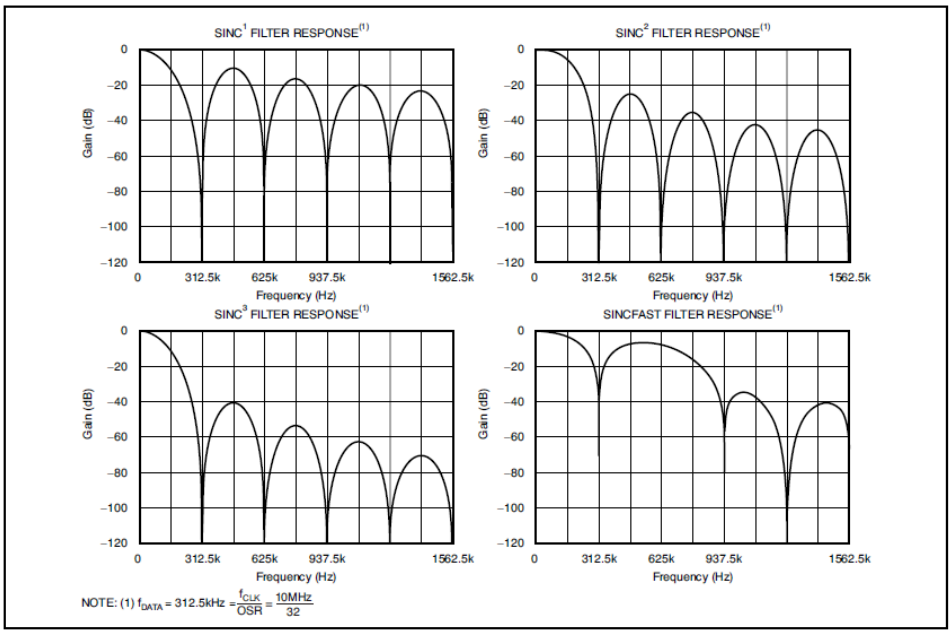


Figure 19-9. Frequency Response of Different Sinc Filters

The order of different sinc filter is shown in Table 19-2.

Table 19-2. Order of Sinc Filter

Filter Type	Order of Sinc Filter
Sinc1	1
Sinc2	2
Sinc3	3
SincFast	3

19.6.1 Data Rate and Latency of the Sinc Filter

The data rate of the sinc filter (filter throughput) represented in samples/sec is calculated by the following formula:

$$Data\ rate\ of\ Sinc\ filter = \frac{Modulator\ data\ rate}{OSR} \quad (9)$$

The latency of the sinc filter represented in secs is defined as the amount of time taken by a sinc filter type to deliver the correct filtered output upon initiation. For a given filter type, latency is calculated by the following formula:

$$Latency\ of\ Sinc\ filter = \frac{Order\ of\ Sinc\ filter}{Data\ rate\ of\ Sinc\ filter} \quad (10)$$

Example configuration:

Sinc filter type	= sinc3
Modulator data rate	= 10 MHz
OSR	= 256
Data rate of Sinc Filter	= 10 MHz / 256 = 39.1 K samples / sec
Sinc filter latency	= 76.8 µs
Sinc filter type	= sinc2
Modulator data rate	= 10 MHz
OSR	= 256
Data rate of Sinc Filter	= 10 MHz / 256 = 39.1 K samples / sec
Sinc filter latency	=51.2 µs

19.7 Data (Primary) Filter Unit

The data filter is a configurable Sinc filter which supports the following filter types: Sinc1, Sinc2, Sinc3, and SincFast. The data filter OSR (DOSR) settings can be configured from 1 to 256 and is independent of the comparator filter. Effective resolution of the data filter (ENOB) depends upon Data filter type, DOSR, and sigma-delta modulator frequency. By default, the data filter is disabled and setting of SDDFPARMx.FEN = 1 enables the data filter. The data filter output is represented in 26-bit signed integer in two's complement format. This filter unit translates a low input signal as '-1' and a high input signal as '1'. The resulting calculation gives both positive and negative values for the output of the data filter. Table 19-3 shows the different full scale values that the data filter can store using different OSRs.

See Section 19.6.1 to understand how to calculate data rate and latency of data filter.

Table 19-3. Peak Data Values for Different DOSR/Filter Combinations

DOSR	Sinc1	Sinc2	Sinc3	SincFast
x	x	x^2	x^3	$2x^2$
4	–4 to 4	–16 to 16	–64 to 64	–32 to 32
8	–8 to 8	–64 to 64	–512 to 512	–128 to 128
16	–16 to 16	–256 to 256	–4096 to 4096	–512 to 512
32	–32 to 32	–1024 to 1024	–32,768 to 32,768	–2048 to 2048
64	–64 to 64	–4096 to 4096	–262,144 to 262,144	–8192 to 8192
128	–128 to 128	–16,384 to 16,384	–2,097,152 to 2,097,152	–32,768 to 32,768
256	–256 to 256	–65,536 to 65,536	–16,777,216 to 16,777,216	–131,072 to 131,072

19.7.1 32-bit or 16-bit Data Filter Output Representation

The data filter output can be represented in either 32-bit or 16-bit format.

32-bit data filter representation:

- When SDDPARMx.DR = 1, data filter output is represented in 32-bit format. Writes to shift control bits do not have any bearing on the output of the data filter in this configuration.

16-bit data filter representation:

- By default, data filter output is represented in 16-bit format
- When SDDPARMx.DR = 0, data filter output is represented in 16-bit format. But it is the responsibility of the user to configure the corresponding shift control bits in the SDDPARMx register to control which 16-bit part of the 32-bit word is sent to the register map.

  For example, for the data filter configuration below:
  - Filter type = Sinc3
  - OSR = 128
  - SDDPARMx.DR = 0

  The data filter with a 26-bit signed output value can be in the range of –16,777,216 to 16,777,216. But, 16-bit signed output supports values only from –32,768 to 32,767. Therefore, it is required to configure shift control bits (SDDPARMx.SH) to 7 to represent the data filter output correctly in 16-bit format. Table 19-4 shows the configuration settings of shift control bits for different OSR and filter types.

Table 19-4. Shift Control Bit Configuration Settings

OSR	Sinc1	Sinc2	SincFast	Sinc3
1 to 31	0	0	0	0
32 to 40	0	0	0	1
41 to 50	0	0	0	2
51 to 63	0	0	0	3
64 to 80	0	0	0	4
81 to 101	0	0	0	5
102 to 127	0	0	0	6
128 to 161	0	0	1	7
162 to 181	0	0	1	8
182 to 203	0	1	2	8
204 to 255	0	1	2	9
256	0	2	3	10

CAUTION

Configuring shift control bits incorrectly will result in getting incorrect 16-bit data filter output.

19.7.2 Data FIFO

Each primary (data) filter channel has a 16-level deep, 32-bit FIFO.

FIFOs can be configured to collect a programmable number of data filter samples before issuing data-ready interrupt. This reduces the number of data-ready interrupts generated and resulting interrupt overhead for managed data flow.

By default, FIFO operation is disabled. FIFOs can be enabled by setting SDFIFOCTLx.FFEN = 1. When FIFO is enabled, each data-ready event from the data filter will populate the FIFO, and the status of the FIFO at any given time is updated in the SDFIFOCTLx.SDFFST bit field.

Setting up FIFO to interrupt after receiving programmable number of data ready events:

- Enable SDFM FIFO (Set SDFIFOCTLx.FFEN = 1)
- Enable SDFM FIFO interrupt (Set SDFIFOCTLx.FFIEN = 1)
- Configure SDFIFOCTLx.SDFFIL bit field to any value between 0 to 16
- Configure SDFM data ready event to interrupt on FIFO interrupt (SDFFINT) (Set SDFFINTx = 1)
- Select data-ready interrupt source is SDFFINTx (DRINTx = SDFFINTx) (SDFIFOCTLx.DRINTSEL = 1)

When the SDFIFOCTLx.SDFFST >= SDFIFOCTLx.SDFFIL condition is met, the SDIFLG.SDFFINTx bit is set and an interrupt is generated on the DRINTx. SDIFLG.SDFFINTx flag can be cleared by setting the SDIFLGCLR. SDFFINTx bit field.

Wait for Sync feature:

The FIFO wait for sync feature can be used to ignore data-ready events from the data filter until the SDSYNC (from PWM) event is triggered.

By default, the Wait for Sync feature is disabled. This feature can be enabled by setting SDSYNCx.WTSYNCEN = 1

When the wait for sync feature is disabled:

FIFOs get populated on every data ready event until the FIFO gets full (or) when SDFIFOCTLx.SDFFST >= SDFIFOCTLx.SDFFIL.

When the wait for sync feature enabled:

FIFOs will not be get populated on every data ready event until it receives a SDSYNC event. On a SYSYNC event, it sets SDSYNCx.WTSYNFLG = 1 and data ready events from primary filter will start populating FIFO until either the FIFOs get full (or) when SDFIFOCTLx.SDFFST >= SDFIFOCTLx.SDFFIL. WTSYNFLG can be cleared either automatically (or) manually.

When WTSYNFLG = 0, FIFOs contents are frozen and subsequent data ready events don't get populate FIFO until next SDSYNC event.

WTSYNFLG automatic clear mode:

By default, this mode is enabled. When SDSYNCx.WTSCLREN = 1, WTSYNFLG is automatically cleared on SDFFINT event.

WTSYNFLG manual clear mode:

Setting SDSYNCx.WTSYNCLR = 1 can be used to clear WTSYNFLG manually.

Clearing FIFO contents:

FIFO contents can cleared by any of the following methods:-

- Disabling FIFO clear FIFO contents. This can be done by clearing SDFIFOCTLx.FFEN = 0.
- Disabling Primary filter clear FIFO contents. This can done by either clearing SDDFPARMx.FEN = 0 (or) by clearing SDMFILEN.MFE = 0.
- FIFO contents can also be automatically cleared upon receiving the SDSYNC event. By default, this feature is disabled and this feature can be enabled by setting FIFO Clear-on-SDSYNC enable (SDSYNCx.FFSYNCCLREN = 1).

  Note: The above feature is only enabled when wait for sync feature is enabled (SDSYNCx. WTSYNCEN = 1).

FIFO debug access behavior:

Debug access of SDDATFIFOx registers does not affect FIFO pointers. On a CPU / CLA / DMA access to SDDATFIFOx register, FIFO read pointers would be advanced to next available entry in FIFO.

19.7.3 SDSYNC Event

Primary (data) filters can be synchronized with respect to the PWM event (called SDSYNC event). The SDSYNC signal from the PWM module is used to reset the DOSR counter. This feature is by default disabled and can be enabled by setting SDDFPARMx.SDSYNCEN = 1. Each primary filter can be synchronized from any of the available PWMx. SOCA / SOCB signals (see Table 19-5). The SDSYNCx.SDSYNCSEL bits allow the user to configure which PWM signal provides the SDSYNC pulse to the primary filter. Figure 19-10 shows how the PWM signals are connected to SDFM.

Table 19-5. SDSYNCx.SYNCSEL

SYNCSEL[5:2]	SDSYNCx.SYNCSEL[1:0]			
	0	1	2	3
0	PWM1.SOCA	PWM1.SOCB	Rsvd	Rsvd
1	PWM2.SOCA	PWM2.SOCB	Rsvd	Rsvd
2	PWM3.SOCA	PWM3.SOCB	Rsvd	Rsvd
3	PWM4.SOCA	PWM4.SOCB	Rsvd	Rsvd
4	PWM5.SOCA	PWM5.SOCB	Rsvd	Rsvd
5	PWM6.SOCA	PWM6.SOCB	Rsvd	Rsvd
6	PWM7.SOCA	PWM7.SOCB	Rsvd	Rsvd
7	PWM8.SOCA	PWM8.SOCB	Rsvd	Rsvd
8	Rsvd	Rsvd	Rsvd	Rsvd
9	Rsvd	Rsvd	Rsvd	Rsvd
10	Rsvd	Rsvd	Rsvd	Rsvd
11	Rsvd	Rsvd	Rsvd	Rsvd
12	Rsvd	Rsvd	Rsvd	Rsvd
13	Rsvd	Rsvd	Rsvd	Rsvd
14	Rsvd	Rsvd	Rsvd	Rsvd
15	Rsvd	Rsvd	Rsvd	Rsvd

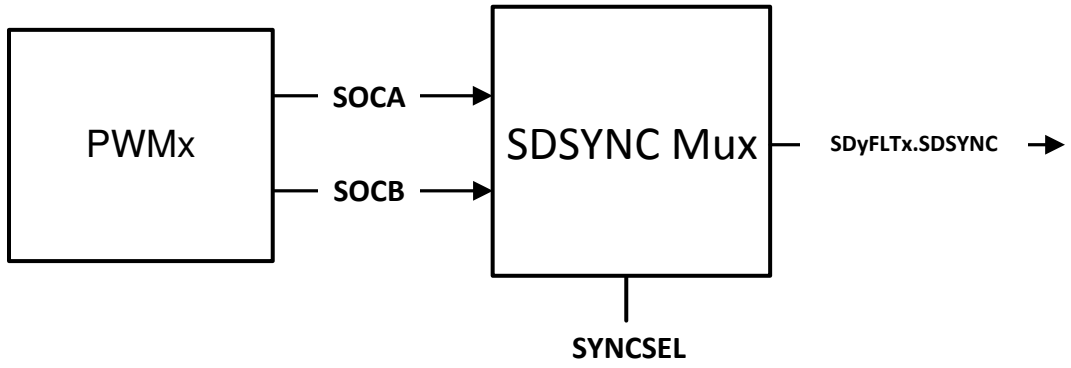


Figure 19-10. SDSYNC Event

Note

Ensure that ONLY ONE SDSYNC event will be generated per PWM timer period. Using PWM in up-count or down-count mode would automatically ensure that you get ONLY SDSYNC event. But, if up-down count mode is used, then make sure that only one SDSYNC event per PWM cycle is generated; otherwise, the filter synchronizer will corrupt SDFM timing by providing two pulses per PWM cycle.

Because of the inherent architecture of the Sinc filter (Sinc1, Sinc2, Sinc3, SincFast), the first few samples, depending upon filter type will be incorrect. Table 19-6 shows the number of incorrect samples on the following conditions:

- When Sinc filter is enabled / configured for first time.
- When Sinc filter is disabled / re-enabled or reconfigured in the middle of operation.
- When data filter receives SDSYNC event from PWM.

Table 19-6. Number of Incorrect Samples Tabulated

Filter Type	Number of Incorrect Samples After the Filter is Enabled and Configured
Sinc1	No incorrect sample
Sinc2	The first sample of the Sinc2 filter is incorrect
SincFast	The first two samples of the SincFast filter are incorrect
Sinc3	The first two samples of the Sinc3 filter are incorrect

CAUTION

SDFM comparator interrupts should be enabled only after providing sufficient settling time to make sure the comparator filter does not trip on these incorrect samples. Therefore, SDFM comparator interrupts (CEVT1 and CEVT2) should be enabled only after a sufficient delay is provided after the comparator filter is configured. This sufficient delay is calculated by adding the latency of the comparator filter and five SD-Cx clock cycles.

19.8 Comparator (Secondary) Filter Unit

Most control systems require protection of the system by tripping the PWM in case the current or voltage goes out of bounds. The primary purpose of the secondary (comparator) filter is to allow the user to monitor input conditions with a fast settling time. This allows the user to trip PWMs to protect the system from potential damage.

Note

The secondary (comparator) filter cannot be synchronized with respect to the PWM event (SDSYNC event).

The comparator filter is a configurable Sinc filter which supports the following filter types: Sinc1, Sinc2, Sinc3, and SincFast. The comparator OSR (COSR) settings can be configured from 1 to 32 and is independent of the data filter. Effective resolution of the comparator filter (ENOB) depends upon the comparator filter type, COSR, and sigma-delta modulator frequency. By default, the comparator filter is disabled and setting SDCPARMx.CEN = 1 enables the comparator filter. The comparator filter output is represented in 16-bit unsigned format. This filter unit translates a low input signal as '0' and a high input signal as '1'. The resulting calculations give only positive values for the output of the comparator filter. Table 19-7 shows the different full-scale values that the comparator filter can store using different OSRs.

Table 19-7. Peak Data Values for Different OSR/Filter Combinations

OSR	Sinc1	Sinc2	Sinc3	SincFast
x	0 to x	0 to x2	0 to x3	0 to 2x2
4	0 to 4	0 to 16	0 to 64	0 to 32
8	0 to 8	0 to 64	0 to 512	0 to 128
16	0 to 16	0 to 256	0 to 4096	0 to 512
32	0 to 32	0 to 1024	0 to 32,768	0 to 2048

See Section 19.6.1 to understand how to calculate data rate and latency of comparator filter.

The output of the comparator filter is memory-mapped and can be read in the SDCDATAx register. This register, SDCDATAx, is updated every COSR number of SD-Cx cycles. The comparator filter digital output is connected to digital comparators explained below.

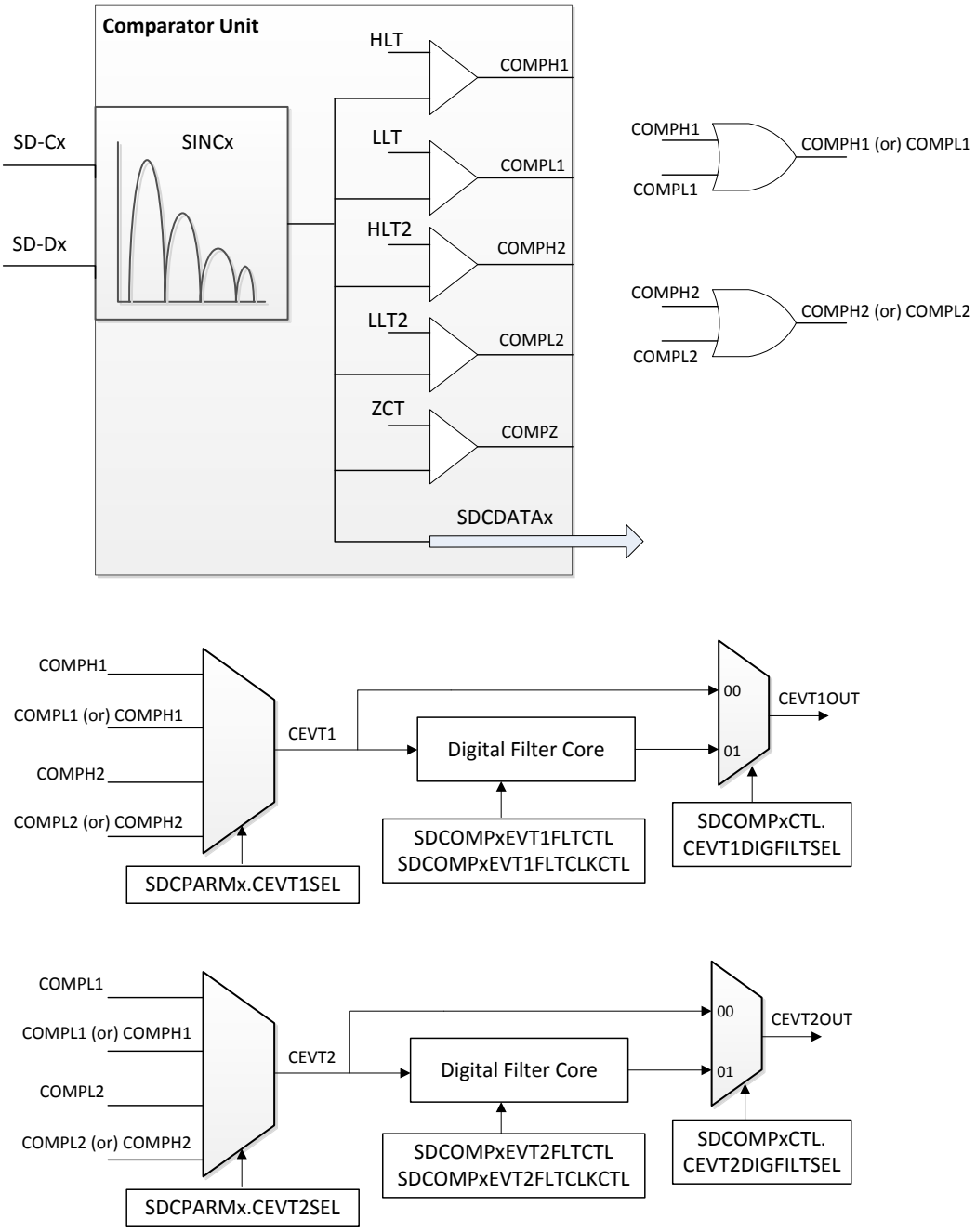


Figure 19-11. Comparator Unit Structure

19.8.1 Higher Threshold (HLT) Comparators

- High threshold comparator can be used to detect over-value condition.
- When comparator data > = higher threshold register, a high threshold event is generated.
- Higher threshold comparator events except for COMPHZx can be configured to trigger following events: CPU interrupt, CLA task, PWM trip.
- This device has three High Threshold comparators:
  - **Higher Threshold 1 (HLT1) Comparator** :
    - When comparator data > = (SDFLTxCMPH1.HLT), HLT1 comparator generates COMPH1 event.
    - The COMPH1 event is connected to both CEVT1 and CEVT2.
  - **Higher Threshold 2 (HLT2) Comparator**
    - When comparator data > = (SDFLTxCMPH2.HLT), HLT2 comparator generates COMPH2 event.
    - The COMPH2 event is connected to both CEVT1 and CEVT2.
  - **Higher Threshold (HTLZ) Comparator**
    - When comparator data > = (SDFLT1CMPHZ.CMPHZ), it can generate a Higher Threshold (B) event (COMPHZx) and sets the corresponding SDSTATUS.HZx flag. But, this event cannot be configured to generate SDFM interrupt (SDx_ERR). The COMPHZ signals from HTLZ comparator are connected to CLB XBAR.

19.8.2 Lower Threshold (LLT) Comparators

- The low threshold comparator can be used to detect under-value condition.
- When comparator data < = Lower Threshold register, a low threshold event is generated.
- Lower threshold comparator events can be configured to trigger following events : CPU interrupt, CLA task, PWM trip.
- Lower threshold comparator events can be used in conjunction with ECAP to measure the frequency / duty cycle of Threshold crossing
- This device has two low threshold comparators.
  - **Lower Threshold 1 (LLT1) Comparator**
    - When comparator data < = (SDFLTxCMPL1.LLT), the LLT1 comparator generates COMPL1 event.
    - The COMPL1 event is connected to both CEVT1 and CEVT2.
  - **Lower Threshold 2 (LLT2) Comparator**
    - When comparator data < = (SDFLTxCMPL2.LLT), LLT2 comparator generates COMPL2 event.
    - The COMPL1 event is connected to both CEVT1 and CEVT2.

19.8.3 Digital Filter

The digital filter works on a window of FIFO samples (SAMPWIN + 1) taken from the input. The filter output resolves to the majority value of the sample window, where majority is defined by the threshold (THRESH) value. If the majority threshold is not satisfied, the filter output remains unchanged.

For proper operation, the value of THRESH must be greater than SAMPWIN / 2.

A prescale function (CLKPRESCALE) determines the filter sampling rate, where the filter FIFO captures one sample every CLKPRESCALE system clocks. Old data from the FIFO is discarded.

A conceptual model of the digital filter is shown in Figure 19-12.

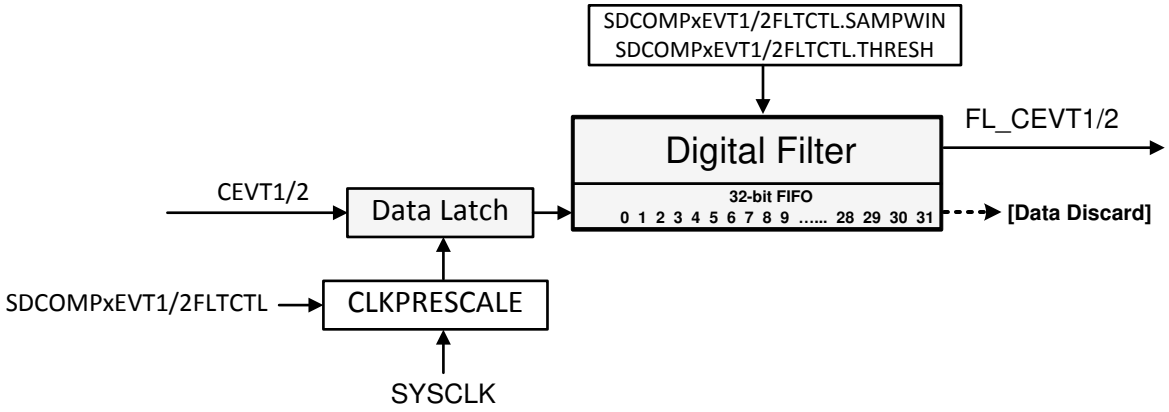


Figure 19-12. Digital Filter

Equivalent C code of the filter implementation is:

```
if (FILTER_OUTPUT == 0) {
    if (Num_1s_in_SAMPWIN >= THRESH) {
        FILTER_OUTPUT = 1;
    }
}
else {
    if (Num_0s_in_SAMPWIN >= THRESH) {
        FILTER_OUTPUT = 0;
    }
}
```

Filter Initialization Sequence

To ensure proper operation of the digital filter, the following initialization sequence is recommended:

1. Configure the digital filter parameters for operation:
   - Set SAMPWIN for the number of samples to monitor in the FIFO window.
   - Set THRESH for the threshold required for majority qualification.
   - Set CLKPRESCALE for the digital filter clock prescale value.
2. Initialize the sample values in the digital FIFO window by setting FILINIT = 1.

19.9 Theoretical SDFM Filter Output

The following equations can be used to derive a theoretical filter output of an SDFM filter output for both a comparator filter and a data filter.

$$\text{Density of ones in bitstream} = \frac{\text{Input Voltage} + \text{Vclipping}}{2 \times \text{Vclipping}} \tag{11}$$

Where:

- Vclipping = maximum differential voltage input range of modulator
- Input voltage = Differential input voltage applied to the modulator

$$\text{Comparator Filter Output (Theoretical)} = \text{Density of ones in bitstream} \times \text{Maximum Filter Output (FilterType, COSR)} \tag{12}$$

$$\text{FilterOutput} = \left\{\frac{\text{absolute(Input voltage)}}{\text{Vclipping}}\right\} \times \text{Maximum Filter Output (FilterType, DOSR)} \tag{13}$$

$$\text{Data Filter Output\_32bit(Theoretical)} = \begin{cases} \text{FilterOutput} & \text{if Input Voltage is +ve voltage} \\ \text{2's complement of FilterOutput} & \text{if input voltage is -ve voltage} \end{cases} \tag{14}$$

$$\text{Data Filter Output\_16bit(Theoretical)} = \text{Data Filter Output\_32bit(Theoretical)} >> \text{Shift value(FilterType, OSR)} \tag{15}$$

For example, when using the AMC1306x25 modulator:

AMC1306x25	Vclipping = Input voltage (AINP - AINN) =	320 mV 100 mV
SDFM filter settings	Filter type = Comparator OSR (COSR) = Data filter OSR (DOSR) =	3 32 100

Density of ones in bitstream	Using Equation 11	0.65625
Comparator filter output Filter type = Sinc3 COSR = 32	Using Equation 12	21504
Data filter output (32-bit) Filter type = Sinc3 DOSR = 100	Using Equation 13 and Equation 14	312500
Data filter output (32-bit) Filter type = Sinc3 DOSR = 100 (Right shift by 5)	Using Equation 15	9765

19.10 Interrupt Unit

Each SDFM can generate five CPU interrupts such as SDFM Error (SDy_ERR) and SDFM data ready (SDy_DRINT1 / SDy_DRINT2, SDy_DRINT3, SDy_DRINT4) interrupts for each filter module.

19.10.1 SDFM (SDyERR) Interrupt Sources

Figure 19-13 shows the structure of SDy_ERR interrupt. SDy_ERR interrupt can be triggered by any of these 16 events.

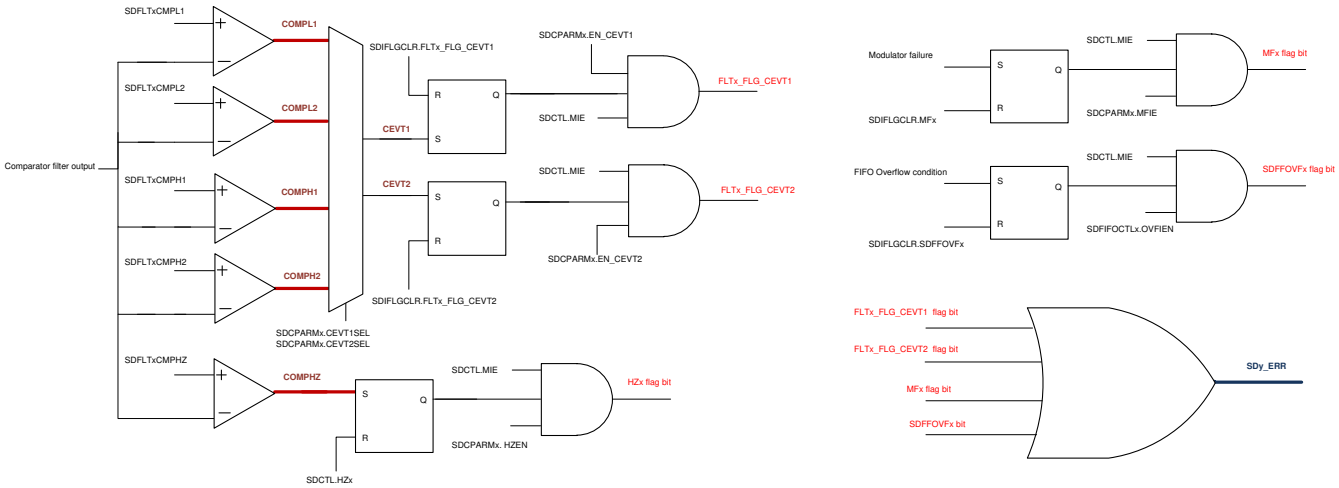


Figure 19-13. SDFM Error (SD_ERR) Interrupt Sources

1. Comparator Event1 (CEVT1)

CEVT1 events from any of the four comparator filter module can trigger CPU interrupt. This event can be configured to trigger SDy_ERR interrupt only if below configurations are made:

- Enable Main interrupt enable (SDCTL.MIE = 1)
- Enable comparator Event1 interrupt (SDCPARMx.EN_CEVT1 = 1)

On a CEVT1 event, SDIFLG. FLTx_FLG_CEVT1 flag bit is set. This flag bit can only be reset if the corresponding bit in SDIFLGCLR register is set and if the interrupt source is no longer active.

2. Comparator Event2 (CEVT2)

CEVT2 events from any of the four comparator filter module can trigger CPU interrupt. This event can be configured to trigger SDy_ERR interrupt only if below configurations are made:

- Enable Main interrupt enable (SDCTL.MIE = 1)
- Enable comparator event1 interrupt (SDCPARMx.EN_CEVT2 = 1)

On a CEVT2 event, SDIFLG.FLTx_FLG_CEVT2 flag bit is set. This flag bit can only be reset if the corresponding bit in SDIFLGCLR register is set and if the interrupt source is no longer active.

3. Modulator Failure (MFx) event

Modulator failures (MFx) are generated when SD-Cx goes missing. The modulator clock is considered missing if SD-Cx does not toggle for 64-SYSCLKs. MFx events from any of the four filter modules can trigger CPU interrupt. This event can be configured to trigger SDy_ERR interrupt only if below configurations are made:

- Enable Main Interrupt Enable (SDCTL.MIE = 1)
- Enable modulator clock failure interrupt source (SDCPARMx.MFIE = 1)

On a MFx event, SDIFLG.MFx flag bit is set. This flag bit can only be reset if the corresponding bit in SDIFLGCLR register is set and if the interrupt source is no longer active.

4. FIFO overflow (SDFFOVFx) event

The number of filter data available in FIFO at any given point can be tracked in SDFIFOCTLx.SDFFST. If the number of words received in FIFO is greater than Max FIFO depth (16), SDFFOVFx event is generated. SDFFOVFx events from any of the four filter modules can trigger CPU interrupt. This event can be configured to trigger SDy_ERR interrupt, only if below configurations are made:

- Enable SDFM FIFO (Set SDFIFOCTLx.FFEN = 1)
- Enable SDFM FIFO overflow interrupt (Set SDFIFOCTLx.OVFIEN = 1) and
- Enable Main interrupt enable (Set SDCTL.MIE = 1)

On a SDFFOVFx event, all subsequent data (primary) filter data is lost and is not stored in FIFO. SDIFLG.SDFFOVFx flag bit is set on a FIFO overflow event and this bit can be cleared if the corresponding bit in SDIFLGCLR register is set and if the interrupt source is no longer active.

19.10.2 Data Ready (DRINT) Interrupt Sources

Figure 19-14 shows the structure of interrupt SDy_DRINTx interrupt. Each SDy_DRINTx interrupt is triggered by corresponding Data Filter channel.

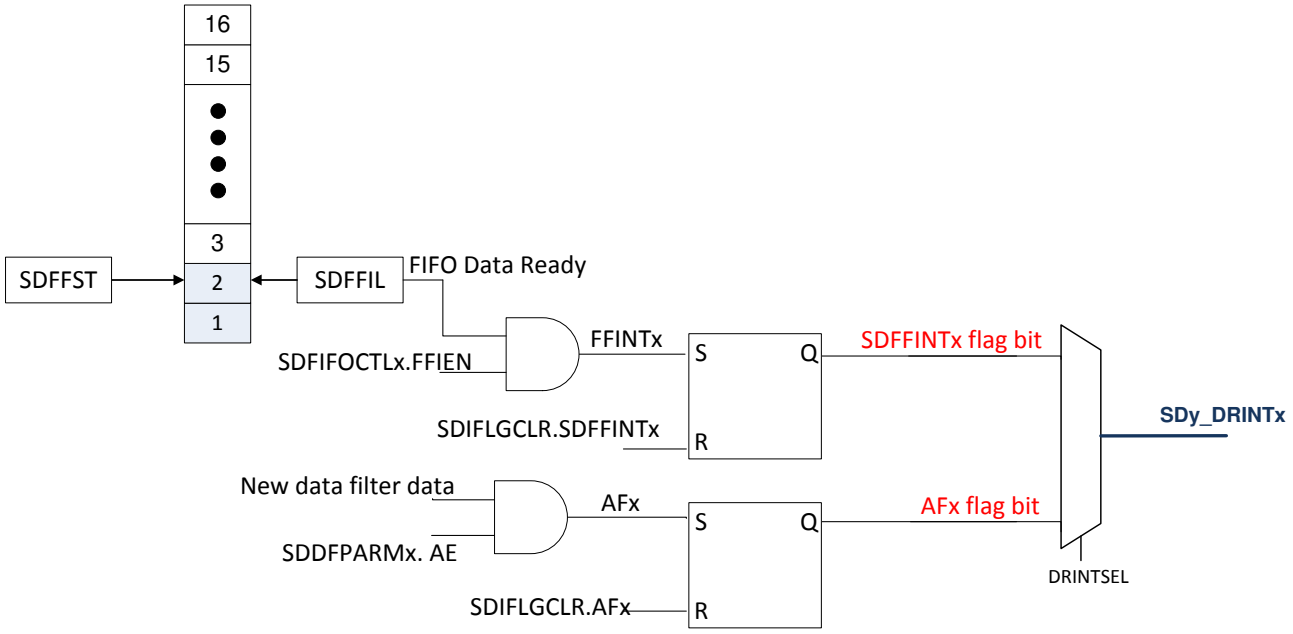


Figure 19-14. SDFM Data Ready (SDy_DRINTx) Interrupt

1. Data Acknowledge (AFx)

When the primary filter is ready with a new filter data, AFx event is generated. AFx events from each filter can generate its own SDy_DRINTx interrupt. This event can be configured to trigger SDy_DRINTx interrupt only if below configurations are made:

- Enable individual filter interrupts (SDDFPARMx. AE = 1)
- Select data-ready interrupt source AFx (DRINTx = AFx) (SDFIFOCTLx.DRINTSEL = 0)

On an AFx event, the SDIFLG.AFx flag bit is set. This flag bit can only be reset if the corresponding bit in SDIFLGCLR register is set and if the interrupt source is no longer active.

2. Four FIFO Data ready interrupt (SDFFINTx)

FIFO Data Ready event is generated whenever SDFIFOCTLx.SDFFST >= SDFIFOCTLx.SDFFIL condition is met. FIFO data ready events from each filter can generate its own SDy_DRINTx interrupt. This event can be configured to trigger SDy_DRINTx interrupt only if below configurations are made:

Table 19-8 shows how the DRINTx output is selected.

- Enable SDFM FIFO (Set SDFIFOCTLx.FFEN = 1) and
- Enable SDFM FIFO interrupt (Set SDFIFOCTLx.FFIEN = 1)
- Select data-Ready interrupt source is SDFFINTx (DRINTx = SDFFINTx) (SDFIFOCTLx.DRINTSEL = 1)

Table 19-8. SDFM Data-Ready Interrupt (SDy_DRINTx) Output Selection

DRINTSEL	AE	FFIEN	FFEN	DRINTx
0	0	x	X	0
0	1	x	X	AFx
1	x	0	X	0
1	x	x	0	0
1	x	1	1	SDFFINTx

19.11 Software

19.11.1 SDFM Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/sdfm

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

19.11.1.1 SDFM Filter Sync CPU

FILE: sdfm_ex1_filter_sync_cpuread.c

In this example, SDFM filter data is read by CPU in SDFM ISR routine. The SDFM configuration is shown below:

- SDFM used in this example - SDFM1
- Input control mode selected - MODE0
- Comparator settings
  - Sinc3 filter selected
  - OSR = 32
  - HLT = 0x7FFF (Higher threshold setting)
  - LLT = 0x0000(Lower threshold setting)
- Data filter settings
  - All the 4 filter modules enabled
  - Sinc3 filter selected
  - OSR = 128
  - All the 4 filters are synchronized by using MFE (Master Filter enable bit)
  - Filter output represented in 16 bit format
  - In order to convert 25 bit Data filter into 16 bit format user needs to right shift by 7 bits for Sinc3 filter with OSR = 128
- Interrupt module settings for SDFM filter
  - All the 4 higher threshold comparator interrupts disabled
  - All the 4 lower threshold comparator interrupts disabled
  - All the 4 modulator failure interrupts disabled
  - All the 4 filter will generate interrupt when a new filter data is available.

External Connections

- SDFM_PIN_MUX_OPTION1 Connect Sigma-Delta streams to (SDx-D1, SDx-C1 to SDx-D4,SDx-C4) on GPIO16-GPIO31
- SDFM_PIN_MUX_OPTION2 Connect Sigma-Delta streams to (SDx-D1, SDx-C1 to SDx-D4,SDx-C4) on GPIO46-GPIO61

Watch Variables

- *filter1Result* - Output of filter 1
- *filter2Result* - Output of filter 2
- *filter3Result* - Output of filter 3
- *filter4Result* - Output of filter 4

19.11.1.2 SDFM Filter Sync CLA

FILE: sdfm_ex2_filter_sync_claread.c

In this example, SDFM filter data is read by CLA in Cla1Task1. The SDFM configuration is shown below:

- SDFM1 used in this example.For using SDFM2, few modifications would be needed in the example.
- MODE0 Input control mode selected
- Comparator settings
  - Sinc3 filter selected
  - OSR = 32
  - hlt = 0x7FFF (Higher threshold setting)
  - llt = 0x0000(Lower threshold setting)

- Data filter settings
  - All the 4 filter modules enabled
  - Sinc3 filter selected
  - OSR = 256
  - All the 4 filters are synchronized by using MFE (Master Filter enable bit)
  - Filter output represented in 16 bit format
  - In order to convert 25 bit Data filter into 16 bit format user needs to right shift by 10 bits for Sinc3 filter with OSR = 256
- Interrupt module settings for SDFM filter
  - All the 4 higher threshold comparator interrupts disabled
  - All the 4 lower threshold comparator interrupts disabled
  - All the 4 modulator failure interrupts disabled
  - All the 4 filter will generate interrupt when a new filter data is available

External Connections

- SDFM_PIN_MUX_OPTION1 Connect Sigma-Delta streams to (SDx-D1, SDx-C1 to SDx-D4,SDx-C4) on GPIO16-GPIO31
- SDFM_PIN_MUX_OPTION2 Connect Sigma-Delta streams to (SDx-D1, SDx-C1 to SDx-D4,SDx-C4) on GPIO46-GPIO61

Watch Variables

- *filter1Result* - Output of filter 1
- *filter2Result* - Output of filter 2
- *filter3Result* - Output of filter 3
- *filter4Result* - Output of filter 4

19.11.1.3 SDFM Filter Sync DMA

FILE: sdfm_ex3_filter_sync_dmaread.c

In this example, SDFM filter data is read by DMA. The SDFM configuration is shown below:

- SDFM1 used in this example. For using SDFM2, few modifications would be needed in the example.
- MODE0 Input control mode selected
- Comparator settings
  - Sinc3 filter selected
  - OSR = 32
  - hlt = 0x7FFF (Higher threshold setting)
  - llt = 0x0000(Lower threshold setting)
- Data filter settings
  - All the 4 filter modules enabled
  - Sinc3 filter selected
  - OSR = 256
  - All the 4 filters are synchronized by using MFE (Master Filter enable bit)
  - Filter output represented in 16 bit format
  - In order to convert 25 bit Data filter into 16 bit format user needs to right shift by 10 bits for Sinc3 filter with OSR = 256
- Interrupt module settings for SDFM filter
  - All the 4 higher threshold comparator interrupts disabled
  - All the 4 lower threshold comparator interrupts disabled
  - All the 4 modulator failure interrupts disabled
  - All the 4 filter will generate interrupt when a new filter data is available

External Connections

- SDFM_PIN_MUX_OPTION1 Connect Sigma-Delta streams to (SDx-D1, SDx-C1 to SDx-D4,SDx-C4) on GPIO16-GPIO31
- SDFM_PIN_MUX_OPTION2 Connect Sigma-Delta streams to (SDx-D1, SDx-C1 to SDx-D4,SDx-C4) on GPIO46-GPIO61

Watch Variables

- *filter1Result* - Output of filter 1
- *filter2Result* - Output of filter 2
- *filter3Result* - Output of filter 3
- *filter4Result* - Output of filter 4

19.11.1.4 SDFM PWM Sync

FILE: sdfm_ex4_pwm_sync_cpuread.c

In this example, SDFM filter data is read by CPU in SDFM ISR routine. The SDFM configuration is shown below:

- SDFM1 is used in this example. For using SDFM2, few modifications would be needed in the example.
- MODE0 Input control mode selected
- Comparator settings
  - Sinc3 filter selected
  - OSR = 32
  - hlt = 0x7FFF (Higher threshold setting)
  - llt = 0x0000(Lower threshold setting)

Data filter settings

- All the 4 filter modules enabled
- Sinc3 filter selected
- OSR = 256
- All the 4 filters are synchronized by using PWM (Master Filter enable bit)
- Filter output represented in 16 bit format
- In order to convert 25 bit Data filter into 16 bit format user needs to right shift by 10 bits for Sinc3 filter with OSR = 256

Interrupt module settings for SDFM filter

- All the 4 higher threshold comparator interrupts disabled
- All the 4 lower threshold comparator interrupts disabled
- All the 4 modulator failure interrupts disabled
- All the 4 filter will generate interrupt when a new filter data is available

External Connections

- SDFM_PIN_MUX_OPTION1 Connect Sigma-Delta streams to (SDx-D1, SDx-C1 to SDx-D4,SDx-C4) on GPIO16-GPIO31
- SDFM_PIN_MUX_OPTION2 Connect Sigma-Delta streams to (SDx-D1, SDx-C1 to SDx-D4,SDx-C4) on GPIO46-GPIO61

Watch Variables

- *filter1Result* - Output of filter 1
- *filter2Result* - Output of filter 2
- *filter3Result* - Output of filter 3
- *filter4Result* - Output of filter 4

19.11.1.5 SDFM Type 1 Filter FIFO

FILE: sdfm_ex5_type1_filter_fifo_cpuread.c

This example configures SDFM1 filter in type 1 to demonstrate data read through CPU in FIFO & non-FIFO mode. Data filter is configured in mode 0 to select SINC3 filter with OSR of 256. Filter output is configured for 16-bit format and data shift of 10 is used.

This example demonstrates the FIFO usage if enabled. FIFO length is set at 16 and data ready interrupt is configured to be triggered when FIFO is full. In this example, SDFM filter data is read by CPU in SDFM Data Ready ISR routine.

External Connections

- SDFM_PIN_MUX_OPTION1 Connect Sigma-Delta streams(SD1-D1, SD1-C1) to (GPIO16, GPIO17)

- SDFM_PIN_MUX_OPTION2 Connect Sigma-Delta streams(SD1-D1, SD1-C1) to (GPIO48, GPIO49)

Watch Variables

- *filter1Result* - Output of filter 1

19.11.1.6 SDFM Filter Sync CLA

FILE: sdfm_ex6_FIFO_freeze_claread.c

In this example, SDFM FIFO will not be filled until a SDSYNC event. On a SDSYNC event, SDFM data filter output will start filling FIFO and stop filling after programmable number 'N' of FIFO is filled.

SDy-C1 (Filter1 channel clock) is internally configured to connected SDy-C2 / SDy-C3 / SDy-C4 SDFM configuration is shown below:

- SDFM1 used in this example.For using SDFM2, few modifications would be needed in the example.
- MODE0 Input control mode selected
- Comparator settings
  - Sinc3 filter selected
  - OSR = 32
  - hlt = 0x7FFF (Higher threshold setting)
  - llt = 0x0000(Lower threshold setting)
- Data filter settings
  - All the 4 filter modules enabled
  - Sinc3 filter selected
  - OSR = 256
  - All the 4 filters are synchronized by using MFE (Master Filter enable bit)
  - Filter output represented in 16 bit format
  - In order to convert 25 bit Data filter into 16 bit format user needs to right shift by 10 bits for Sinc3 filter with OSR = 256
- Interrupt module settings for SDFM filter
  - All the 4 higher threshold comparator interrupts disabled
  - All the 4 lower threshold comparator interrupts disabled
  - All the 4 modulator failure interrupts disabled
  - All the 4 filter will generate interrupt when a new filter data is available

External Connections

- SDFM_PIN_MUX_OPTION1 Connect Sigma-Delta streams to (SDx-D1, SDx-C1 to SDx-D4,SDx-C4) on GPIO16-GPIO31
- SDFM_PIN_MUX_OPTION2 Connect Sigma-Delta streams to (SDx-D1, SDx-C1 to SDx-D4,SDx-C4) on GPIO46-GPIO61

Watch Variables

- *filter1Result* - Output of filter 1
- *filter2Result* - Output of filter 2
- *filter3Result* - Output of filter 3
- *filter4Result* - Output of filter 4

19.12 SDFM Registers

This section describes the Sigma Delta Filter Module registers.

19.12.1 SDFM Base Address Table

Table 19-9. SDFM Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
Sdfm1Regs	SDFM_REGS	SDFM1_BASE	0x0000_5E00	YES	YES	YES	YES	YES
Sdfm2Regs	SDFM_REGS	SDFM2_BASE	0x0000_5E80	YES	YES	YES	YES	YES

19.12.2 SDFM_REGS Registers

Table 19-10 lists the memory-mapped registers for the SDFM_REGS registers. All register offset addresses not listed in Table 19-10 should be considered as reserved locations and the register contents should not be modified.

Table 19-10. SDFM_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	SDIFLG	SD Interrupt Flag Register		Go
2h	SDIFLGCLR	SD Interrupt Flag Clear Register		Go
4h	SDCTL	SD Control Register	EALLOW	Go
6h	SDMFILEN	SD Master Filter Enable	EALLOW	Go
7h	SDSTATUS	SD Status Register		Go
10h	SDCTLPARM1	Control Parameter Register for Ch1	EALLOW	Go
11h	SDDFPARM1	Data Filter Parameter Register for Ch1	EALLOW	Go
12h	SDDPARM1	Data Parameter Register for Ch1	EALLOW	Go
13h	SDFLT1CMPH1	High-level Threshold Register for Ch1	EALLOW	Go
14h	SDFLT1CMPL1	Low-level Threshold Register for Ch1	EALLOW	Go
15h	SDCPARM1	Comparator Filter Parameter Register for Ch1	EALLOW	Go
16h	SDDATA1	Data Filter Data Register (16 or 32bit) for Ch1		Go
18h	SDDATFIFO1	Filter Data FIFO Output(32b) for Ch1		Go
1Ah	SDCDATA1	Comparator Filter Data Register (16b) for Ch1		Go
1Bh	SDFLT1CMPH2	Second high level threhold for CH1	EALLOW	Go
1Ch	SDFLT1CMPHZ	High-level (Z) Threshold Register for Ch1	EALLOW	Go
1Dh	SDFIFOCTL1	FIFO Control Register for Ch1	EALLOW	Go
1Eh	SDSYNC1	SD Filter Sync control for Ch1	EALLOW	Go
1Fh	SDFLT1CMPL2	Second low level threhold for CH1	EALLOW	Go
20h	SDCTLPARM2	Control Parameter Register for Ch2	EALLOW	Go
21h	SDDFPARM2	Data Filter Parameter Register for Ch2	EALLOW	Go
22h	SDDPARM2	Data Parameter Register for Ch2	EALLOW	Go
23h	SDFLT2CMPH1	High-level Threshold Register for Ch2	EALLOW	Go
24h	SDFLT2CMPL1	Low-level Threshold Register for Ch2	EALLOW	Go
25h	SDCPARM2	Comparator Filter Parameter Register for Ch2	EALLOW	Go
26h	SDDATA2	Data Filter Data Register (16 or 32bit) for Ch2		Go
28h	SDDATFIFO2	Filter Data FIFO Output(32b) for Ch2		Go
2Ah	SDCDATA2	Comparator Filter Data Register (16b) for Ch2		Go
2Bh	SDFLT2CMPH2	Second high level threhold for CH2	EALLOW	Go
2Ch	SDFLT2CMPHZ	High-level (Z) Threshold Register for Ch2	EALLOW	Go
2Dh	SDFIFOCTL2	FIFO Control Register for Ch2	EALLOW	Go
2Eh	SDSYNC2	SD Filter Sync control for Ch2	EALLOW	Go
2Fh	SDFLT2CMPL2	Second low level threhold for CH2	EALLOW	Go
30h	SDCTLPARM3	Control Parameter Register for Ch3	EALLOW	Go
31h	SDDFPARM3	Data Filter Parameter Register for Ch3	EALLOW	Go
32h	SDDPARM3	Data Parameter Register for Ch3	EALLOW	Go
33h	SDFLT3CMPH1	High-level Threshold Register for Ch3	EALLOW	Go
34h	SDFLT3CMPL1	Low-level Threshold Register for Ch3	EALLOW	Go
35h	SDCPARM3	Comparator Filter Parameter Register for Ch3	EALLOW	Go
36h	SDDATA3	Data Filter Data Register (16 or 32bit) for Ch3		Go
38h	SDDATFIFO3	Filter Data FIFO Output(32b) for Ch3		Go

Table 19-10. SDFM_REGS Registers (continued)

Offset	Acronym	Register Name	Write Protection	Section
3Ah	SDCDATA3	Comparator Filter Data Register (16b) for Ch3		Go
3Bh	SDFLT3CMPH2	Second high level threhold for CH3	EALLOW	Go
3Ch	SDFLT3CMPHZ	High-level (Z) Threshold Register for Ch3	EALLOW	Go
3Dh	SDFIFOCTL3	FIFO Control Register for Ch3	EALLOW	Go
3Eh	SDSYNC3	SD Filter Sync control for Ch3	EALLOW	Go
3Fh	SDFLT3CMPL2	Second low level threhold for CH3	EALLOW	Go
40h	SDCTLPARM4	Control Parameter Register for Ch4	EALLOW	Go
41h	SDDFPARM4	Data Filter Parameter Register for Ch4	EALLOW	Go
42h	SDDPARM4	Data Parameter Register for Ch4	EALLOW	Go
43h	SDFLT4CMPH1	High-level Threshold Register for Ch4	EALLOW	Go
44h	SDFLT4CMPL1	Low-level Threshold Register for Ch4	EALLOW	Go
45h	SDCPARM4	Comparator Filter Parameter Register for Ch4	EALLOW	Go
46h	SDDATA4	Data Filter Data Register (16 or 32bit) for Ch4		Go
48h	SDDATFIFO4	Filter Data FIFO Output(32b) for Ch4		Go
4Ah	SDCDATA4	Comparator Filter Data Register (16b) for Ch4		Go
4Bh	SDFLT4CMPH2	Second high level threhold for CH4	EALLOW	Go
4Ch	SDFLT4CMPHZ	High-level (Z) Threshold Register for Ch4	EALLOW	Go
4Dh	SDFIFOCTL4	FIFO Control Register for Ch4	EALLOW	Go
4Eh	SDSYNC4	SD Filter Sync control for Ch4	EALLOW	Go
4Fh	SDFLT4CMPL2	Second low level threhold for CH4	EALLOW	Go
60h	SDCOMP1CTL	SD Comparator event filter1 Control Register	EALLOW	Go
61h	SDCOMP1EVT2FLTCTL	COMPL/CEVT2 Digital filter1 Control Register	EALLOW	Go
62h	SDCOMP1EVT2FLTCLKCTL	COMPL/CEVT2 Digital filter1 Clock Control Register	EALLOW	Go
63h	SDCOMP1EVT1FLTCTL	COMPH/CEVT1 Digital filter1 Control Register	EALLOW	Go
64h	SDCOMP1EVT1FLTCLKCTL	COMPH/CEVT1 Digital filter1 Clock Control Register	EALLOW	Go
67h	SDCOMP1LOCK	SD compartor event filter1 Lock Register	EALLOW	Go
68h	SDCOMP2CTL	SD Comparator event filter2 Control Register	EALLOW	Go
69h	SDCOMP2EVT2FLTCTL	COMPL/CEVT2 Digital filter2 Control Register	EALLOW	Go
6Ah	SDCOMP2EVT2FLTCLKCTL	COMPL/CEVT2 Digital filter2 Clock Control Register	EALLOW	Go
6Bh	SDCOMP2EVT1FLTCTL	COMPH/CEVT1 Digital filter2 Control Register	EALLOW	Go
6Ch	SDCOMP2EVT1FLTCLKCTL	COMPH/CEVT1 Digital filter2 Clock Control Register	EALLOW	Go
6Fh	SDCOMP2LOCK	SD compartor event filter2 Lock Register	EALLOW	Go
70h	SDCOMP3CTL	SD Comparator event filter3 Control Register	EALLOW	Go
71h	SDCOMP3EVT2FLTCTL	COMPL/CEVT2 Digital filter3 Control Register	EALLOW	Go
72h	SDCOMP3EVT2FLTCLKCTL	COMPL/CEVT2 Digital filter3 Clock Control Register	EALLOW	Go
73h	SDCOMP3EVT1FLTCTL	COMPH/CEVT1 Digital filter3 Control Register	EALLOW	Go
74h	SDCOMP3EVT1FLTCLKCTL	COMPH/CEVT1 Digital filter3 Clock Control Register	EALLOW	Go
77h	SDCOMP3LOCK	SD compartor event filter3 Lock Register	EALLOW	Go
78h	SDCOMP4CTL	SD Comparator event filter4 Control Register	EALLOW	Go
79h	SDCOMP4EVT2FLTCTL	COMPL/CEVT2 Digital filter4 Control Register	EALLOW	Go

Table 19-10. SDFM_REGS Registers (continued)

Offset	Acronym	Register Name	Write Protection	Section
7Ah	SDCOMP4EVT2FLTCLKCTL	COMPL/CEVT2 Digital filter4 Clock Control Register	EALLOW	Go
7Bh	SDCOMP4EVT1FLTCTL	COMPH/CEVT1 Digital filter4 Control Register	EALLOW	Go
7Ch	SDCOMP4EVT1FLTCLKCTL	COMPH/CEVT1 Digital filter4 Clock Control Register	EALLOW	Go
7Fh	SDCOMP4LOCK	SD compartor event filter4 Lock Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 19-11 shows the codes that are used for access types in this section.

Table 19-11. SDFM_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

19.12.2.1 SDIFLG Register (Offset = 0h) [Reset = 0h]

SDIFLG is shown in Figure 19-15 and described in Table 19-12.

Return to the Summary Table.

SD Interrupt Flag Register

Figure 19-15. SDIFLG Register

31	30	29	28	27	26	25	24
MIF	RESERVED						
R-0h	R-0h						
23	22	21	20	19	18	17	16
SDFFINT4	SDFFINT3	SDFFINT2	SDFFINT1	SDFFOVF4	SDFFOVF3	SDFFOVF2	SDFFOVF1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
AF4	AF3	AF2	AF1	MF4	MF3	MF2	MF1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
FLT4_FLG_CEVT2	FLT4_FLG_CEVT1	FLT3_FLG_CEVT2	FLT3_FLG_CEVT1	FLT2_FLG_CEVT2	FLT2_FLG_CEVT1	FLT1_FLG_CEVT2	FLT1_FLG_CEVT1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 19-12. SDIFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MIF	R	0h	Set whenever any "error" interrupt (MF1-4,IFL1-4,IFH1-4,SDFFOVF1-4) is active Reset type: SYSRSn
30-24	RESERVED	R-0	0h	Reserved
23	SDFFINT4	R	0h	SDFIFO data ready interrupt for Ch4 Reset type: SYSRSn
22	SDFFINT3	R	0h	SDFIFO data ready interrupt for Ch3 Reset type: SYSRSn
21	SDFFINT2	R	0h	SDFIFO data ready interrupt for Ch2 Reset type: SYSRSn
20	SDFFINT1	R	0h	SDFIFO data ready interrupt for Ch1 0: SDFIFO data ready interrupt has NOT occurred 1: SDFIFO data ready interrupt has occurred Reset type: SYSRSn
19	SDFFOVF4	R	0h	FIFO Overflow Flag for Ch4 Reset type: SYSRSn
18	SDFFOVF3	R	0h	FIFO Overflow Flag for Ch3 Reset type: SYSRSn
17	SDFFOVF2	R	0h	FIFO Overflow Flag for Ch2 Reset type: SYSRSn
16	SDFFOVF1	R	0h	FIFO Overflow Flag for Ch1 0 - FIFO has not overflowed 1 - FIFO overflowed. # words received in FIFO > FIFO depth (16), NEW word is lost Reset type: SYSRSn
15	AF4	R	0h	Acknowledge flag for Filter 4 0: No new data available for Filter (in non-FIFO mode) 1: New data available for Filter (in non-FIFO mode) Reset type: SYSRSn

Table 19-12. SDIFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
14	AF3	R	0h	Acknowledge flag for Filter 3 0: No new data available for Filter (in non-FIFO mode) 1: New data available for Filter (in non-FIFO mode) Reset type: SYSRSn
13	AF2	R	0h	Acknowledge flag for Filter 2 0: No new data available for Filter (in non-FIFO mode) 1: New data available for Filter (in non-FIFO mode) Reset type: SYSRSn
12	AF1	R	0h	Acknowledge flag for Filter 1 0: No new data available for Filter (in non-FIFO mode) 1: New data available for Filter (in non-FIFO mode) Reset type: SYSRSn
11	MF4	R	0h	Modulator Failure for Filter 4 0: Modulator is operating normally for Filter 1: Modulator failure for Filter Reset type: SYSRSn
10	MF3	R	0h	Modulator Failure for Filter 3 0: Modulator is operating normally for Filter 1: Modulator failure for Filter Reset type: SYSRSn
9	MF2	R	0h	Modulator Failure for Filter 2 0: Modulator is operating normally for Filter 1: Modulator failure for Filter Reset type: SYSRSn
8	MF1	R	0h	Modulator Failure for Filter 1 0: Modulator is operating normally for Filter 1: Modulator failure for Filter Reset type: SYSRSn
7	FLT4_FLG_CEVT2	R	0h	CEVT2 Interrupt flag for filter4 0: CEVT2 event has not occured 1: CEVT2 event has occurred Reset type: SYSRSn
6	FLT4_FLG_CEVT1	R	0h	CEVT1 Interrupt flag for filter4 0: CEVT1 event has not occured 1: CEVT1 event has occurred Reset type: SYSRSn
5	FLT3_FLG_CEVT2	R	0h	CEVT2 Interrupt flag for filter3 0: CEVT2 event has not occured 1: CEVT2 event has occurred Reset type: SYSRSn
4	FLT3_FLG_CEVT1	R	0h	CEVT1 Interrupt flag for filter3 0: CEVT1 event has not occured 1: CEVT1 event has occurred Reset type: SYSRSn
3	FLT2_FLG_CEVT2	R	0h	CEVT2 Interrupt flag for filter2 0: CEVT2 event has not occured 1: CEVT2 event has occurred Reset type: SYSRSn
2	FLT2_FLG_CEVT1	R	0h	CEVT1 Interrupt flag for filter2 0: CEVT1 event has not occured 1: CEVT1 event has occurred Reset type: SYSRSn
1	FLT1_FLG_CEVT2	R	0h	CEVT2 Interrupt flag for filter1 0: CEVT2 event has not occured 1: CEVT2 event has occurred Reset type: SYSRSn

Table 19-12. SDIFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	FLT1_FLG_CEVT1	R	0h	CEVT1 Interrupt flag for filter1 0: CEVT1 event has not occured 1: CEVT1 event has occurred Reset type: SYSRSn

19.12.2.2 SDIFLGCLR Register (Offset = 2h) [Reset = 0h]

SDIFLGCLR is shown in Figure 19-16 and described in Table 19-13.

Return to the Summary Table.

SD Module Interrupt Flag Clear Bits:
Writing a "1" will clear the respective flag bit in the SDIFLG register.
Writes of "0" are ignored.
Note: If user writes a "1" to clear a bit on the same cycle that the hardware is trying to set the bit to "1", then hardware has priority and the bit will not be cleared.

Figure 19-16. SDIFLGCLR Register

31	30	29	28	27	26	25	24
MIF	RESERVED						
R-0/W1S-0h	R-0h						
23	22	21	20	19	18	17	16
SDFFINT4	SDFFINT3	SDFFINT2	SDFFINT1	SDFFOVF4	SDFFOVF3	SDFFOVF2	SDFFOVF1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
15	14	13	12	11	10	9	8
AF4	AF3	AF2	AF1	MF4	MF3	MF2	MF1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
FLT4_FLG_CE VT2	FLT4_FLG_CE VT1	FLT3_FLG_CE VT2	FLT3_FLG_CE VT1	FLT2_FLG_CE VT2	FLT2_FLG_CE VT1	FLT1_FLG_CE VT2	FLT1_FLG_CE VT1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 19-13. SDIFLGCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MIF	R-0/W1S	0h	Flag-clear bit for SDFM Master Interrupt flag. Writing a 1 to clear MIF flag in SDIFLG register Writes of "0" are ignored. Note: If the MIF flag is cleared and other Interrupts are still pending, MIF will again be set to 1 on the following SysClk cycle, and the INT output will be reasserted (pulsed low) Reset type: SYSRSn
30-24	RESERVED	R-0	0h	Reserved
23	SDFFINT4	R-0/W1S	0h	SDFIFO data ready Interrupt flag-clear bit for Ch4 Reset type: SYSRSn
22	SDFFINT3	R-0/W1S	0h	SDFIFO data ready Interrupt flag-clear bit for Ch3 Reset type: SYSRSn
21	SDFFINT2	R-0/W1S	0h	SDFIFO data ready Interrupt flag-clear bit for Ch2 Reset type: SYSRSn
20	SDFFINT1	R-0/W1S	0h	SDFIFO data ready Interrupt flag-clear bit for Ch1 Reset type: SYSRSn
19	SDFFOVF4	R-0/W1S	0h	SDFIFO overflow clear Ch4 Reset type: SYSRSn
18	SDFFOVF3	R-0/W1S	0h	SDFIFO overflow clear Ch3 Reset type: SYSRSn
17	SDFFOVF2	R-0/W1S	0h	SDFIFO overflow clear Ch2 Reset type: SYSRSn
16	SDFFOVF1	R-0/W1S	0h	SDFIFO overflow clear Ch1 Reset type: SYSRSn
15	AF4	R-0/W1S	0h	Flag-clear bit for Acknowledge flag for Filter 4 Reset type: SYSRSn

Table 19-13. SDIFLGCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
14	AF3	R-0/W1S	0h	Flag Clear bit for AF3 Reset type: SYSRSn
13	AF2	R-0/W1S	0h	Flag Clear bit for AF2 Reset type: SYSRSn
12	AF1	R-0/W1S	0h	Flag Clear bit for AF1 Reset type: SYSRSn
11	MF4	R-0/W1S	0h	Flag Clear bit for MF4 Reset type: SYSRSn
10	MF3	R-0/W1S	0h	Flag Clear bit for MF3 Reset type: SYSRSn
9	MF2	R-0/W1S	0h	Flag Clear bit for MF2 Reset type: SYSRSn
8	MF1	R-0/W1S	0h	Flag Clear bit for MF1 Reset type: SYSRSn
7	FLT4_FLG_CEVT2	R-0/W1S	0h	Flag Clear bit for FLT4_FLG_CEVT2 Reset type: SYSRSn
6	FLT4_FLG_CEVT1	R-0/W1S	0h	Flag Clear bit for FLT4_FLG_CEVT1 Reset type: SYSRSn
5	FLT3_FLG_CEVT2	R-0/W1S	0h	Flag Clear bit for FLT3_FLG_CEVT2 Reset type: SYSRSn
4	FLT3_FLG_CEVT1	R-0/W1S	0h	Flag Clear bit for FLT3_FLG_CEVT1 Reset type: SYSRSn
3	FLT2_FLG_CEVT2	R-0/W1S	0h	Flag Clear bit for FLT2_FLG_CEVT2 Reset type: SYSRSn
2	FLT2_FLG_CEVT1	R-0/W1S	0h	Flag Clear bit for FLT2_FLG_CEVT1 Reset type: SYSRSn
1	FLT1_FLG_CEVT2	R-0/W1S	0h	Flag Clear bit for FLT1_FLG_CEVT2 Reset type: SYSRSn
0	FLT1_FLG_CEVT1	R-0/W1S	0h	Flag Clear bit for FLT1_FLG_CEVT1 Reset type: SYSRSn

19.12.2.3 SDCTL Register (Offset = 4h) [Reset = 0h]

SDCTL is shown in Figure 19-17 and described in Table 19-14.

Return to the Summary Table.

SD Control Register

Figure 19-17. SDCTL Register

15	14	13	12	11	10	9	8
RESERVED	RESERVED	MIE	RESERVED				
R-0h	R-0h	R/W-0h	R-0h				
7	6	5	4	3	2	1	0
RESERVED				HZ4	HZ3	HZ2	HZ1
R-0h				R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 19-14. SDCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14	RESERVED	R-0	0h	Reserved
13	MIE	R/W	0h	Master SDy_ERR interrupt enable 0: SDy_ERR Interrupt and interrupt flags are disabled 1: SDy_ERR Interrupt and interrupt flags are enabled Reset type: SYSRSn
12-4	RESERVED	R-0	0h	Reserved
3	HZ4	R-0/W1S	0h	Flag Clear bit for HZ4 Reset type: SYSRSn
2	HZ3	R-0/W1S	0h	Flag Clear bit for HZ3 Reset type: SYSRSn
1	HZ2	R-0/W1S	0h	Flag Clear bit for HZ2 Reset type: SYSRSn
0	HZ1	R-0/W1S	0h	Flag Clear bit for HZ1 Reset type: SYSRSn

19.12.2.4 SDMFILEN Register (Offset = 6h) [Reset = 0h]

SDMFILEN is shown in Figure 19-18 and described in Table 19-15.

Return to the Summary Table.

SD Master Filter Enable

Figure 19-18. SDMFILEN Register

15	14	13	12	11	10	9	8
RESERVED			RESERVED	MFE	RESERVED	RESERVED	RESERVED
R-0h			R-0h	R/W-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED			RESERVED			
R-0h	R-0h			R-0h			

Table 19-15. SDMFILEN Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R-0	0h	Reserved
12	RESERVED	R-0	0h	Reserved
11	MFE	R/W	0h	Master Filter Enable 0: All the four data filter units of SDFM module are disabled. All FIFOs are cleared 1: Data filter units can be enabled if bit FEN is '1'. Reset type: SYSRSn
10	RESERVED	R-0	0h	Reserved
9	RESERVED	R-0	0h	Reserved
8-7	RESERVED	R-0	0h	Reserved
6-4	RESERVED	R-0	0h	Reserved
3-0	RESERVED	R-0	0h	Reserved

19.12.2.5 SDSTATUS Register (Offset = 7h) [Reset = 0h]

SDSTATUS is shown in Figure 19-19 and described in Table 19-16.

Return to the Summary Table.

SD Status Register

Figure 19-19. SDSTATUS Register

15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED				HZ4	HZ3	HZ2	HZ1
R-0h				R-0h	R-0h	R-0h	R-0h

Table 19-16. SDSTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	RESERVED	R	0h	Reserved
13	RESERVED	R	0h	Reserved
12	RESERVED	R	0h	Reserved
11	RESERVED	R	0h	Reserved
10	RESERVED	R	0h	Reserved
9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7-4	RESERVED	R-0	0h	Reserved
3	HZ4	R	0h	High-level Threshold crossing (Z) flag Ch4 Primarily intended for detecting "zero"-crossing events. Unlike the primary comparator IFHx flag, it does not have the ability to generate an interrupt. 0: Comparator filter output < SDCMPHZ4.HLTZ 1: Comparator filter output >= SDCMPHZ4.HLTZ Reset type: SYSRSn
2	HZ3	R	0h	High-level Threshold crossing (Z) flag Ch3 Primarily intended for detecting "zero"-crossing events. Unlike the primary comparator IFHx flag, it does not have the ability to generate an interrupt. 0: Comparator filter output < SDCMPHZ3.HLTZ 1: Comparator filter output >= SDCMPHZ3.HLTZ Reset type: SYSRSn
1	HZ2	R	0h	High-level Threshold crossing (Z) flag Ch2 Primarily intended for detecting "zero"-crossing events. Unlike the primary comparator IFHx flag, it does not have the ability to generate an interrupt. 0: Comparator filter output < SDCMPHZ2.HLTZ 1: Comparator filter output >= SDCMPHZ2.HLTZ Reset type: SYSRSn
0	HZ1	R	0h	High-level Threshold crossing (Z) flag Ch1 Primarily intended for detecting "zero"-crossing events. Unlike the primary comparator IFHx flag, it does not have the ability to generate an interrupt. 0: Comparator filter output < SDCMPHZ1.HLTZ 1: Comparator filter output >= SDCMPHZ1.HLTZ Reset type: SYSRSn

19.12.2.6 SDCTLPARM1 Register (Offset = 10h) [Reset = 0h]

SDCTLPARM1 is shown in Figure 19-20 and described in Table 19-17.

Return to the Summary Table.

Control Parameter Register for Ch1

Figure 19-20. SDCTLPARM1 Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	SDDATASYNC	RESERVED	SDCLKSYNC	SDCLKSEL	RESERVED	MOD	
R-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	

Table 19-17. SDCTLPARM1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R-0	0h	Reserved
7	RESERVED	R-0	0h	Reserved
6	SDDATASYNC	R/W	0h	0: SD Data is not passed through a synchronizer. 1: SD Data is passed through a synchronizer. Reset type: SYSRSn
5	RESERVED	R-0	0h	Reserved
4	SDCLKSYNC	R/W	0h	0: SD Clock is not passed through a synchronizer. 1: SD Clock is passed through a synchronizer. Reset type: SYSRSn
3	SDCLKSEL	R/W	0h	SD1 Clock source select. 0: Clock source to SDFM filter is its channel clock. 1: Clock source to SDFM filter is SD1 filter clock. Reset type: SYSRSn
2	RESERVED	R	0h	Reserved
1-0	MOD	R/W	0h	Modulator clock modes 0: Mode 0: Modulator clock running at 1x data rate 1: Reserved 2: Reserved 3: Reserved Reset type: SYSRSn

19.12.2.7 SDDFPARM1 Register (Offset = 11h) [Reset = 0h]

SDDFPARM1 is shown in Figure 19-21 and described in Table 19-18.

Return to the Summary Table.

Data Filter Parameter Register for Ch1

Figure 19-21. SDDFPARM1 Register

15	14	13	12	11	10	9	8
RESERVED			SDSYNCEN	SST		AE	FEN
R-0h			R/W-0h	R/W-0h		R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DOSR							
R/W-0h							

Table 19-18. SDDFPARM1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12	SDSYNCEN	R/W	0h	PWM synchronization (SDSYNC) of data filter 0: PWM synchronization of data filter is disabled 1: PWM synchronization of data filter is enabled Note: SDSYNCx.SYNCSEL bits define which PWM signal is used to synchronize PWMs Reset type: SYSRSn
11-10	SST	R/W	0h	Data filter structure 00: Data filter runs with a Sincfast structure 01: Data filter runs with a Sinc1 structure 10: Data filter runs with a Sinc2 structure 11: Data filter runs with a Sinc3 structure Reset type: SYSRSn
9	AE	R/W	0h	Data filter Acknowledge Enable 0: Acknowledge flag is disabled for the particular filter 1: Acknowledge flag is enabled for the particular filter Reset type: SYSRSn
8	FEN	R/W	0h	Filter Enable 0: The data filter is disabled and no data is produced 1: The data filter is enabled and data are produced in the data filter Note: When filter is disabled, DOSR counter held in reset, filter data erased. Also resets FIFO pointers and clears the FIFO Reset type: SYSRSn
7-0	DOSR	R/W	0h	Data filter Oversampling ratio The actual oversampling ratio of data filter is DOSR + 1 These bits set the oversampling ratio of the data filter. 0x0FF represents an oversampling ratio of 256. Reset type: SYSRSn

19.12.2.8 SDDPARM1 Register (Offset = 12h) [Reset = 0h]

SDDPARM1 is shown in Figure 19-22 and described in Table 19-19.

Return to the Summary Table.

Data Parameter Register for Ch1

Figure 19-22. SDDPARM1 Register

15	14	13	12	11	10	9	8
SH					DR	RESERVED	
R/W-0h					R/W-0h	R-0h	
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

Table 19-19. SDDPARM1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	SH	R/W	0h	Shift Control These bits indicate by how many bits the 16-bit window is shifted up when 16-bit data representation is chosen. Reset type: SYSRSn
10	DR	R/W	0h	Data filter Data representation 0: Data stored in 16b 2's complement 1: Data stored in 32b 2's complement Reset type: SYSRSn
9-0	RESERVED	R-0	0h	Reserved

19.12.2.9 SDFLT1CMPH1 Register (Offset = 13h) [Reset = 7FFFh]

SDFLT1CMPH1 is shown in Figure 19-23 and described in Table 19-20.

Return to the Summary Table.

High-level Threshold Register for Ch1

Figure 19-23. SDFLT1CMPH1 Register

15	14	13	12	11	10	9	8
RESERVED	HLT						
R-0h	R/W-7FFFh						
7	6	5	4	3	2	1	0
HLT							
R/W-7FFFh							

Table 19-20. SDFLT1CMPH1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	HLT	R/W	7FFFh	Unsigned high-level threshold for the comparator filter output. Reset type: SYSRSn

19.12.2.10 SDFLT1CMPL1 Register (Offset = 14h) [Reset = 0h]

SDFLT1CMPL1 is shown in Figure 19-24 and described in Table 19-21.

Return to the Summary Table.

Low-level Threshold Register for Ch1

Figure 19-24. SDFLT1CMPL1 Register

15	14	13	12	11	10	9	8
RESERVED	LLT						
R-0h	R/W-0h						
7	6	5	4	3	2	1	0
LLT							
R/W-0h							

Table 19-21. SDFLT1CMPL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	LLT	R/W	0h	Unsigned low-level threshold for the comparator filter output. Reset type: SYSRSn

19.12.2.11 SDCPARM1 Register (Offset = 15h) [Reset = 2000h]

SDCPARM1 is shown in Figure 19-25 and described in Table 19-22.

Return to the Summary Table.

Comparator Filter Parameter Register for Ch1

Figure 19-25. SDCPARM1 Register

15	14	13	12	11	10	9	8
CEVT2SEL		CEN	CEVT1SEL		HZEN	MFIE	CS1_CS0
R/W-0h		R/W-1h	R/W-0h		R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CS1_CS0	EN_CEVT2	EN_CEVT1	COSR				
R/W-0h	R/W-0h	R/W-0h	R/W-0h				

Table 19-22. SDCPARM1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	CEVT2SEL	R/W	0h	Comparator Event2 Select 00: COMPL1 01: COMPL1 OR COMPH1 10: COMPL2 11: COMPL2 OR COMPH2 Reset type: SYSRSn
13	CEN	R/W	1h	Comparator Filter enable 0: Disable comparator filter 1: Enable comparator filter Reset type: SYSRSn
12-11	CEVT1SEL	R/W	0h	Comparator Event1 Select 00: COMPH1 01: COMPL1 OR COMPH1 10: COMPH2 11: COMPL2 OR COMPH2 Reset type: SYSRSn
10	HZEN	R/W	0h	High level (Z) Threshold crossing output enable 0: Disable Higher level Threshold (Z) crossing 1: Enable Higher level Threhold (Z) crossing Reset type: SYSRSn
9	MFIE	R/W	0h	Modulator Failure Interrupt Enable 0: Disable modulator failure interrupt and its flag 1: Enable modulator failure interrupt and its flag Reset type: SYSRSn
8-7	CS1_CS0	R/W	0h	Comparator filter structure 00: Comparator filter runs with a sincfast structure 01: Comparator filter runs with a Sinc1 structure 10: Comparator filter runs with a Sinc2 structure 11: Comparator filter runs with a Sinc3 structure Reset type: SYSRSn
6	EN_CEVT2	R/W	0h	CEVT2 interrupt enable 0: Disable CEVT2 interrupt 1: Enable CEVT2 interrupt Reset type: SYSRSn
5	EN_CEVT1	R/W	0h	CEVT1 interrupt enable 0: Disable CEVT1 interrupt 1: Enable CEVT1 interrupt Reset type: SYSRSn

Table 19-22. SDCPARM1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-0	COSR	R/W	0h	Comparator Oversampling ratio. The actual rate is COSR + 1. These bits set the oversampling ratio of the filter. 0x1F represents an oversampling ratio of 32 Reset type: SYSRSn

19.12.2.12 SDDATA1 Register (Offset = 16h) [Reset = 0h]

SDDATA1 is shown in Figure 19-26 and described in Table 19-23.

Return to the Summary Table.

Data Filter Data Register (16 or 32bit) for Ch1

Figure 19-26. SDDATA1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA32HI																DATA16															
R-0h																R-0h															

Table 19-23. SDDATA1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DATA32HI	R	0h	Hi-order 16b in 32b mode, 16-bit Data in 16b mode Reset type: SYSRSn
15-0	DATA16	R	0h	Lo-order 16b in 32b mode Reset type: SYSRSn

19.12.2.13 SDDATFIFO1 Register (Offset = 18h) [Reset = 0h]

SDDATFIFO1 is shown in Figure 19-27 and described in Table 19-24.

Return to the Summary Table.

Filter Data FIFO Output(32b) for Ch1

Figure 19-27. SDDATFIFO1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA32HI																DATA16															
R-0h																R-0h															

Table 19-24. SDDATFIFO1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DATA32HI	R	0h	Hi-order 16b in 32b mode, 16-bit Data in 16b mode Reset type: SYSRSn
15-0	DATA16	R	0h	Lo-order 16b in 32b mode Reset type: SYSRSn

19.12.2.14 SDCDATA1 Register (Offset = 1Ah) [Reset = 0h]

SDCDATA1 is shown in Figure 19-28 and described in Table 19-25.

Return to the Summary Table.

Comparator Filter Data Register (16b) for Ch1

Figure 19-28. SDCDATA1 Register

15	14	13	12	11	10	9	8
DATA16							
R-0h							
7	6	5	4	3	2	1	0
DATA16							
R-0h							

Table 19-25. SDCDATA1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	DATA16	R	0h	Comparator Data output - 16b only Reset type: SYSRSn

19.12.2.15 SDFLT1CMPH2 Register (Offset = 1Bh) [Reset = 7FFFh]

SDFLT1CMPH2 is shown in Figure 19-29 and described in Table 19-26.

Return to the Summary Table.

Second high level threhold for CH1

Figure 19-29. SDFLT1CMPH2 Register

15	14	13	12	11	10	9	8
RESERVED	HLT2						
R-0h	R/W-7FFFh						
7	6	5	4	3	2	1	0
HLT2							
R/W-7FFFh							

Table 19-26. SDFLT1CMPH2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	HLT2	R/W	7FFFh	Second Unsigned high-level threshold for the comparator filter output. Reset type: SYSRSn

19.12.2.16 SDFLT1CMPHZ Register (Offset = 1Ch) [Reset = 0h]

SDFLT1CMPHZ is shown in Figure 19-30 and described in Table 19-27.

Return to the Summary Table.

High-level (Z) Threshold Register for Ch1

Figure 19-30. SDFLT1CMPHZ Register

15	14	13	12	11	10	9	8
RESERVED	HLTZ						
R-0h	R/W-0h						
7	6	5	4	3	2	1	0
HLTZ							
R/W-0h							

Table 19-27. SDFLT1CMPHZ Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	HLTZ	R/W	0h	Unsigned High-level threshold (Z) for the comparator filter output Primarily intended for detecting "zero"-crossing events. Unlike the primary comparator SDCMPHx, it does not have the ability to generate an interrupt. Reset type: SYSRSn

19.12.2.17 SDFIFOCTL1 Register (Offset = 1Dh) [Reset = 0h]

SDFIFOCTL1 is shown in Figure 19-31 and described in Table 19-28.

Return to the Summary Table.

FIFO Control Register for Ch1

Figure 19-31. SDFIFOCTL1 Register

15	14	13	12	11	10	9	8
OVFIEN	DRINTSEL	FFEN	FFIEN	RESERVED	SDFFST		
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R-0h		
7	6	5	4	3	2	1	0
SDFFST		RESERVED	SDFFIL				
R-0h		R-0h	R/W-0h				

Table 19-28. SDFIFOCTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	OVFIEN	R/W	0h	SDFIFO Overflow interrupt enable 0: SDFIFO Overflow condition will not generate an interrupt 1: SDFIFO overflow condition generates an interrupt on SDy_ERR Reset type: SYSRSn
14	DRINTSEL	R/W	0h	Data-Ready Interrupt (DRINT) source select 0 = AF1 (Select non-FIFO data-ready interrupt) 1 = SDFFINT1 (Select FIFO data-ready interrupt) Reset type: SYSRSn
13	FFEN	R/W	0h	SDFIFO Enable 0: Disable FIFO operation 1: Enable FIFO operation Note: When FIFO is disabled, FIFO contents are cleared Reset type: SYSRSn
12	FFIEN	R/W	0h	SDFIFO data ready Interrupt Enable Reset type: SYSRSn
11	RESERVED	R-0	0h	Reserved
10-6	SDFFST	R	0h	SDFIFO Status 00000 FIFO empty 00001 FIFO has 1 word . . . . 10000 FIFO has 16 words Reset type: SYSRSn
5	RESERVED	R-0	0h	Reserved
4-0	SDFFIL	R/W	0h	SDFIFO interrupt level bits The FIFO will generate an interrupt when the FIFO status (SDFFST) >= FIFO level (SDFFIL) Reset type: SYSRSn

19.12.2.18 SDSYNC1 Register (Offset = 1Eh) [Reset = 43Fh]

SDSYNC1 is shown in Figure 19-32 and described in Table 19-29.

Return to the Summary Table.

SD Filter Sync control for Ch1

Figure 19-32. SDSYNC1 Register

15	14	13	12	11	10	9	8
RESERVED					WTSCLREN	FFSYNCCLREN	WTSYNCLR
R-0h					R/W-1h	R/W-0h	R-0/W-0h
7	6	5	4	3	2	1	0
WTSYNFLG	WTSYNCEN	SYNCSEL					
R-0h	R/W-0h	R/W-3Fh					

Table 19-29. SDSYNC1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R-0	0h	Reserved
10	WTSCLREN	R/W	1h	WTSYNFLG Clear-on-FIFOINT Enable 0: WTSYNFLG can only be cleared manually (using WTSYNCLR bit) 1: WTSYNFLG is cleared automatically on SDFFINT Reset type: SYSRSn
9	FFSYNCCLREN	R/W	0h	FIFO Clear-on-SDSYNC Enable 0: SDFIFO is not automaticaly cleared upon receiving SDSYNC 1: SDFIFO is automaticaly cleared upon receiving SDSYNC Reset type: SYSRSn
8	WTSYNCLR	R-0/W	0h	Wait-for-Sync Flag Clear (always reads 0) 0: Write of 0 has no affect 1: Write of 1 clears WTSYNFLG Reset type: SYSRSn
7	WTSYNFLG	R	0h	Wait-for-Sync Flag 0: SDSYNC event has not occurred 1: SDSYNC event occurred. Reset type: SYSRSn
6	WTSYNCEN	R/W	0h	Wait-for-Sync Enable 0: Incoming Data written to SDFIFO on every Data-Ready (DR) Event 1: Incoming Data written to SDFIFO on DR event only after SDSYNC event occurs Reset type: SYSRSn
5-0	SYNCSEL	R/W	3Fh	Defines source for the SDSYNC Input on this channel Refer SDSYNCx.SYNCSEL table Reset type: SYSRSn

19.12.2.19 SDFLT1CMPL2 Register (Offset = 1Fh) [Reset = 0h]

SDFLT1CMPL2 is shown in Figure 19-33 and described in Table 19-30.

Return to the Summary Table.

Second low level threhold for CH1

Figure 19-33. SDFLT1CMPL2 Register

15	14	13	12	11	10	9	8
RESERVED	LLT2						
R-0h	R/W-0h						
7	6	5	4	3	2	1	0
LLT2							
R/W-0h							

Table 19-30. SDFLT1CMPL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	LLT2	R/W	0h	Second Unsigned low-level threshold for the comparator filter output. Reset type: SYSRSn

19.12.2.20 SDCTLPARM2 Register (Offset = 20h) [Reset = 0h]

SDCTLPARM2 is shown in Figure 19-34 and described in Table 19-31.

Return to the Summary Table.

Control Parameter Register for Ch2

Figure 19-34. SDCTLPARM2 Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	SDDATASYNC	RESERVED	SDCLKSYNC	SDCLKSEL	RESERVED	MOD	
R-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	

Table 19-31. SDCTLPARM2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	RESERVED	R-0	0h	Reserved
6	SDDATASYNC	R/W	0h	0: SD Data is not passed through a synchronizer. 1: SD Data is passed through a synchronizer. Reset type: SYSRSn
5	RESERVED	R-0	0h	Reserved
4	SDCLKSYNC	R/W	0h	0: SD Clock is not passed through a synchronizer. 1: SD Clock is passed through a synchronizer. Reset type: SYSRSn
3	SDCLKSEL	R/W	0h	SD2 Clock source select. 0: Clock source to SDFM filter is its channel clock. 1: Clock source to SDFM filter is SD1 filter clock. Reset type: SYSRSn
2	RESERVED	R	0h	Reserved
1-0	MOD	R/W	0h	Modulator clock modes 0: Mode 0: Modulator clock running at 1x data rate 1: Reserved 2: Reserved 3: Reserved Reset type: SYSRSn

19.12.2.21 SDDFPARM2 Register (Offset = 21h) [Reset = 0h]

SDDFPARM2 is shown in Figure 19-35 and described in Table 19-32.

Return to the Summary Table.

Data Filter Parameter Register for Ch2

Figure 19-35. SDDFPARM2 Register

15	14	13	12	11	10	9	8
RESERVED			SDSYNCEN	SST		AE	FEN
R-0h			R/W-0h	R/W-0h		R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DOSR							
R/W-0h							

Table 19-32. SDDFPARM2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12	SDSYNCEN	R/W	0h	PWM synchronization (SDSYNC) of data filter 0: PWM synchronization of data filter is disabled 1: PWM synchronization of data filter is enabled Note: SDSYNCx.SYNCSEL bits define which PWM signal is used to synchronize PWMs Reset type: SYSRSn
11-10	SST	R/W	0h	Data filter structure 00: Data filter runs with a Sincfast structure 01: Data filter runs with a Sinc1 structure 10: Data filter runs with a Sinc2 structure 11: Data filter runs with a Sinc3 structure Reset type: SYSRSn
9	AE	R/W	0h	Data filter Acknowledge Enable 0: Acknowledge flag is disabled for the particular filter 1: Acknowledge flag is enabled for the particular filter Reset type: SYSRSn
8	FEN	R/W	0h	Filter Enable 0: The data filter is disabled and no data is produced 1: The data filter is enabled and data are produced in the data filter Note: When filter is disabled, DOSR counter held in reset, filter data erased. Also resets FIFO pointers and clears the FIFO Reset type: SYSRSn
7-0	DOSR	R/W	0h	Data filter Oversampling ratio The actual oversampling ratio of data filter is DOSR + 1 These bits set the oversampling ratio of the data filter. 0x0FF represents an oversampling ratio of 256. Reset type: SYSRSn

19.12.2.22 SDDPARM2 Register (Offset = 22h) [Reset = 0h]

SDDPARM2 is shown in Figure 19-36 and described in Table 19-33.

Return to the Summary Table.

Data Parameter Register for Ch2

Figure 19-36. SDDPARM2 Register

15	14	13	12	11	10	9	8
SH					DR	RESERVED	
R/W-0h					R/W-0h	R-0h	
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

Table 19-33. SDDPARM2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	SH	R/W	0h	Shift Control These bits indicate by how many bits the 16-bit window is shifted up when 16-bit data representation is chosen. Reset type: SYSRSn
10	DR	R/W	0h	Data filter Data representation 0: Data stored in 16b 2's complement 1: Data stored in 32b 2's complement Reset type: SYSRSn
9-0	RESERVED	R-0	0h	Reserved

19.12.2.23 SDFLT2CMPH1 Register (Offset = 23h) [Reset = 7FFFh]

SDFLT2CMPH1 is shown in Figure 19-37 and described in Table 19-34.

Return to the Summary Table.

High-level Threshold Register for Ch2

Figure 19-37. SDFLT2CMPH1 Register

15	14	13	12	11	10	9	8
RESERVED	HLT						
R-0h	R/W-7FFFh						
7	6	5	4	3	2	1	0
HLT							
R/W-7FFFh							

Table 19-34. SDFLT2CMPH1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	HLT	R/W	7FFFh	Unsigned high-level threshold for the comparator filter output. Reset type: SYSRSn

19.12.2.24 SDFLT2CMPL1 Register (Offset = 24h) [Reset = 0h]

SDFLT2CMPL1 is shown in Figure 19-38 and described in Table 19-35.

Return to the Summary Table.

Low-level Threshold Register for Ch2

Figure 19-38. SDFLT2CMPL1 Register

15	14	13	12	11	10	9	8
RESERVED	LLT						
R-0h	R/W-0h						
7	6	5	4	3	2	1	0
LLT							
R/W-0h							

Table 19-35. SDFLT2CMPL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	LLT	R/W	0h	Unsigned low-level threshold for the comparator filter output. Reset type: SYSRSn

19.12.2.25 SDCPARM2 Register (Offset = 25h) [Reset = 2000h]

SDCPARM2 is shown in Figure 19-39 and described in Table 19-36.

Return to the Summary Table.

Comparator Filter Parameter Register for Ch2

Figure 19-39. SDCPARM2 Register

15	14	13	12	11	10	9	8
CEVT2SEL		CEN	CEVT1SEL		HZEN	MFIE	CS1_CS0
R/W-0h		R/W-1h	R/W-0h		R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CS1_CS0	EN_CEVT2	EN_CEVT1	COSR				
R/W-0h	R/W-0h	R/W-0h	R/W-0h				

Table 19-36. SDCPARM2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	CEVT2SEL	R/W	0h	Comparator Event2 Select 00: COMPL1 01: COMPL1 OR COMPH1 10: COMPL2 11: COMPL2 OR COMPH2 Reset type: SYSRSn
13	CEN	R/W	1h	Comparator Filter enable 0: Disable comparator filter 1: Enable comparator filter Reset type: SYSRSn
12-11	CEVT1SEL	R/W	0h	Comparator Event1 Select 00: COMPH1 01: COMPL1 OR COMPH1 10: COMPH2 11: COMPL2 OR COMPH2 Reset type: SYSRSn
10	HZEN	R/W	0h	High level (Z) Threshold crossing output enable 0: Disable Higher level Threshold (Z) crossing 1: Enable Higher level Threhold (Z) crossing Reset type: SYSRSn
9	MFIE	R/W	0h	Modulator Failure Interrupt Enable 0: Disable modulator failure interrupt and its flag 1: Enable modulator failure interrupt and its flag Reset type: SYSRSn
8-7	CS1_CS0	R/W	0h	Comparator filter structure 00: Comparator filter runs with a sincfast structure 01: Comparator filter runs with a Sinc1 structure 10: Comparator filter runs with a Sinc2 structure 11: Comparator filter runs with a Sinc3 structure Reset type: SYSRSn
6	EN_CEVT2	R/W	0h	CEVT2 interrupt enable 0: Disable CEVT2 interrupt 1: Enable CEVT2 interrupt Reset type: SYSRSn
5	EN_CEVT1	R/W	0h	CEVT1 interrupt enable 0: Disable CEVT1 interrupt 1: Enable CEVT1 interrupt Reset type: SYSRSn

Table 19-36. SDCPARM2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-0	COSR	R/W	0h	Comparator Oversampling ratio. The actual rate is COSR + 1. These bits set the oversampling ratio of the filter. 0x1F represents an oversampling ratio of 32 Reset type: SYSRSn

19.12.2.26 SDDATA2 Register (Offset = 26h) [Reset = 0h]

SDDATA2 is shown in Figure 19-40 and described in Table 19-37.

Return to the Summary Table.

Data Filter Data Register (16 or 32bit) for Ch2

Figure 19-40. SDDATA2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA32HI																DATA16															
R-0h																R-0h															

Table 19-37. SDDATA2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DATA32HI	R	0h	Hi-order 16b in 32b mode, 16-bit Data in 16b mode Reset type: SYSRSn
15-0	DATA16	R	0h	Lo-order 16b in 32b mode Reset type: SYSRSn

19.12.2.27 SDDATFIFO2 Register (Offset = 28h) [Reset = 0h]

SDDATFIFO2 is shown in Figure 19-41 and described in Table 19-38.

Return to the Summary Table.

Filter Data FIFO Output(32b) for Ch2

Figure 19-41. SDDATFIFO2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA32HI																DATA16															
R-0h																R-0h															

Table 19-38. SDDATFIFO2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DATA32HI	R	0h	Hi-order 16b in 32b mode, 16-bit Data in 16b mode Reset type: SYSRSn
15-0	DATA16	R	0h	Lo-order 16b in 32b mode Reset type: SYSRSn

19.12.2.28 SDCDATA2 Register (Offset = 2Ah) [Reset = 0h]

SDCDATA2 is shown in Figure 19-42 and described in Table 19-39.

Return to the Summary Table.

Comparator Filter Data Register (16b) for Ch2

Figure 19-42. SDCDATA2 Register

15	14	13	12	11	10	9	8
DATA16							
R-0h							
7	6	5	4	3	2	1	0
DATA16							
R-0h							

Table 19-39. SDCDATA2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	DATA16	R	0h	Comparator Data output - 16b only Reset type: SYSRSn

19.12.2.29 SDFLT2CMPH2 Register (Offset = 2Bh) [Reset = 7FFFh]

SDFLT2CMPH2 is shown in Figure 19-43 and described in Table 19-40.

Return to the Summary Table.

Second high level threhold for CH2

Figure 19-43. SDFLT2CMPH2 Register

15	14	13	12	11	10	9	8
RESERVED	HLT2						
R-0h	R/W-7FFFh						
7	6	5	4	3	2	1	0
HLT2							
R/W-7FFFh							

Table 19-40. SDFLT2CMPH2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	HLT2	R/W	7FFFh	Second Unsigned high-level threshold for the comparator filter output. Reset type: SYSRSn

19.12.2.30 SDFLT2CMPHZ Register (Offset = 2Ch) [Reset = 0h]

SDFLT2CMPHZ is shown in Figure 19-44 and described in Table 19-41.

Return to the Summary Table.

High-level (Z) Threshold Register for Ch2

Figure 19-44. SDFLT2CMPHZ Register

15	14	13	12	11	10	9	8
RESERVED	HLTZ						
R-0h	R/W-0h						
7	6	5	4	3	2	1	0
HLTZ							
R/W-0h							

Table 19-41. SDFLT2CMPHZ Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	HLTZ	R/W	0h	Unsigned High-level threshold (Z) for the comparator filter output Primarily intended for detecting "zero"-crossing events. Unlike the primary comparator SDCMPHx, it does not have the ability to generate an interrupt. Reset type: SYSRSn

19.12.2.31 SDFIFOCTL2 Register (Offset = 2Dh) [Reset = 0h]

SDFIFOCTL2 is shown in Figure 19-45 and described in Table 19-42.

Return to the Summary Table.

FIFO Control Register for Ch2

Figure 19-45. SDFIFOCTL2 Register

15	14	13	12	11	10	9	8
OVFIEN	DRINTSEL	FFEN	FFIEN	RESERVED	SDFFST		
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R-0h		
7	6	5	4	3	2	1	0
SDFFST		RESERVED	SDFFIL				
R-0h		R-0h	R/W-0h				

Table 19-42. SDFIFOCTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	OVFIEN	R/W	0h	SDFIFO Overflow interrupt enable 0: SDFIFO Overflow condition will not generate an interrupt 1: SDFIFO overflow condition generates an interrupt on SDy_ERR Reset type: SYSRSn
14	DRINTSEL	R/W	0h	Data-Ready Interrupt (DRINT) source select 0 = AF1 (Select non-FIFO data-ready interrupt) 1 = SDFFINT1 (Select FIFO data-ready interrupt) Reset type: SYSRSn
13	FFEN	R/W	0h	SDFIFO Enable 0: Disable FIFO operation 1: Enable FIFO operation Note: When FIFO is disabled, FIFO contents are cleared Reset type: SYSRSn
12	FFIEN	R/W	0h	SDFIFO data ready Interrupt Enable Reset type: SYSRSn
11	RESERVED	R-0	0h	Reserved
10-6	SDFFST	R	0h	SDFIFO Status 00000 FIFO empty 00001 FIFO has 1 word . . . . 10000 FIFO has 16 words Reset type: SYSRSn
5	RESERVED	R-0	0h	Reserved
4-0	SDFFIL	R/W	0h	SDFIFO interrupt level bits The FIFO will generate an interrupt when the FIFO status (SDFFST) >= FIFO level (SDFFIL) Reset type: SYSRSn

19.12.2.32 SDSYNC2 Register (Offset = 2Eh) [Reset = 43Fh]

SDSYNC2 is shown in Figure 19-46 and described in Table 19-43.

Return to the Summary Table.

SD Filter Sync control for Ch2

Figure 19-46. SDSYNC2 Register

15	14	13	12	11	10	9	8
RESERVED					WTSCLREN	FFSYNCCLREN	WTSYNCLR
R-0h					R/W-1h	R/W-0h	R-0/W-0h
7	6	5	4	3	2	1	0
WTSYNFLG	WTSYNCEN	SYNCSEL					
R-0h	R/W-0h	R/W-3Fh					

Table 19-43. SDSYNC2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R-0	0h	Reserved
10	WTSCLREN	R/W	1h	WTSYNFLG Clear-on-FIFOINT Enable 0: WTSYNFLG can only be cleared manually (using WTSYNCLR bit) 1: WTSYNFLG is cleared automatically on SDFFINT Reset type: SYSRSn
9	FFSYNCCLREN	R/W	0h	FIFO Clear-on-SDSYNC Enable 0: SDFIFO is not automaticaly cleared upon receiving SDSYNC 1: SDFIFO is automaticaly cleared upon receiving SDSYNC Reset type: SYSRSn
8	WTSYNCLR	R-0/W	0h	Wait-for-Sync Flag Clear (always reads 0) 0: Write of 0 has no affect 1: Write of 1 clears WTSYNFLG Reset type: SYSRSn
7	WTSYNFLG	R	0h	Wait-for-Sync Flag 0: SDSYNC event has not occurred 1: SDSYNC event occurred. Reset type: SYSRSn
6	WTSYNCEN	R/W	0h	Wait-for-Sync Enable 0: Incoming Data written to SDFIFO on every Data-Ready (DR) Event 1: Incoming Data written to SDFIFO on DR event only after SDSYNC event occurs Reset type: SYSRSn
5-0	SYNCSEL	R/W	3Fh	Defines source for the SDSYNC Input on this channel Refer SDSYNCx.SYNCSEL table Reset type: SYSRSn

19.12.2.33 SDFLT2CMPL2 Register (Offset = 2Fh) [Reset = 0h]

SDFLT2CMPL2 is shown in Figure 19-47 and described in Table 19-44.

Return to the Summary Table.

Second low level threhold for CH2

Figure 19-47. SDFLT2CMPL2 Register

15	14	13	12	11	10	9	8
RESERVED	LLT2						
R-0h	R/W-0h						
7	6	5	4	3	2	1	0
LLT2							
R/W-0h							

Table 19-44. SDFLT2CMPL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	LLT2	R/W	0h	Second Unsigned low-level threshold for the comparator filter output. Reset type: SYSRSn

19.12.2.34 SDCTLPARM3 Register (Offset = 30h) [Reset = 0h]

SDCTLPARM3 is shown in Figure 19-48 and described in Table 19-45.

Return to the Summary Table.

Control Parameter Register for Ch3

Figure 19-48. SDCTLPARM3 Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	SDDATASYNC	RESERVED	SDCLKSYNC	SDCLKSEL	RESERVED	MOD	
R-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	

Table 19-45. SDCTLPARM3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	RESERVED	R-0	0h	Reserved
6	SDDATASYNC	R/W	0h	0: SD Data is not passed through a synchronizer. 1: SD Data is passed through a synchronizer. Reset type: SYSRSn
5	RESERVED	R-0	0h	Reserved
4	SDCLKSYNC	R/W	0h	0: SD Clock is not passed through a synchronizer. 1: SD Clock is passed through a synchronizer. Reset type: SYSRSn
3	SDCLKSEL	R/W	0h	SD3 Clock source select. 0: Clock source to SDFM filter is its channel clock. 1: Clock source to SDFM filter is SD1 filter clock. Reset type: SYSRSn
2	RESERVED	R	0h	Reserved
1-0	MOD	R/W	0h	Modulator clock modes 0: Mode 0: Modulator clock running at 1x data rate 1: Reserved 2: Reserved 3: Reserved Reset type: SYSRSn

19.12.2.35 SDDFPARM3 Register (Offset = 31h) [Reset = 0h]

SDDFPARM3 is shown in Figure 19-49 and described in Table 19-46.

Return to the Summary Table.

Data Filter Parameter Register for Ch3

Figure 19-49. SDDFPARM3 Register

15	14	13	12	11	10	9	8
RESERVED			SDSYNCEN	SST		AE	FEN
R-0h			R/W-0h	R/W-0h		R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DOSR							
R/W-0h							

Table 19-46. SDDFPARM3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12	SDSYNCEN	R/W	0h	PWM synchronization (SDSYNC) of data filter 0: PWM synchronization of data filter is disabled 1: PWM synchronization of data filter is enabled Note: SDSYNCx.SYNCSEL bits define which PWM signal is used to synchronize PWMs Reset type: SYSRSn
11-10	SST	R/W	0h	Data filter structure 00: Data filter runs with a Sincfast structure 01: Data filter runs with a Sinc1 structure 10: Data filter runs with a Sinc2 structure 11: Data filter runs with a Sinc3 structure Reset type: SYSRSn
9	AE	R/W	0h	Data filter Acknowledge Enable 0: Acknowledge flag is disabled for the particular filter 1: Acknowledge flag is enabled for the particular filter Reset type: SYSRSn
8	FEN	R/W	0h	Filter Enable 0: The data filter is disabled and no data is produced 1: The data filter is enabled and data are produced in the data filter Note: When filter is disabled, DOSR counter held in reset, filter data erased. Also resets FIFO pointers and clears the FIFO Reset type: SYSRSn
7-0	DOSR	R/W	0h	Data filter Oversampling ratio The actual oversampling ratio of data filter is DOSR + 1 These bits set the oversampling ratio of the data filter. 0x0FF represents an oversampling ratio of 256. Reset type: SYSRSn

19.12.2.36 SDDPARM3 Register (Offset = 32h) [Reset = 0h]

SDDPARM3 is shown in Figure 19-50 and described in Table 19-47.

Return to the Summary Table.

Data Parameter Register for Ch3

Figure 19-50. SDDPARM3 Register

15	14	13	12	11	10	9	8
SH					DR	RESERVED	
R/W-0h					R/W-0h	R-0h	
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

Table 19-47. SDDPARM3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	SH	R/W	0h	Shift Control These bits indicate by how many bits the 16-bit window is shifted up when 16-bit data representation is chosen. Reset type: SYSRSn
10	DR	R/W	0h	Data filter Data representation 0: Data stored in 16b 2's complement 1: Data stored in 32b 2's complement Reset type: SYSRSn
9-0	RESERVED	R-0	0h	Reserved

19.12.2.37 SDFLT3CMPH1 Register (Offset = 33h) [Reset = 7FFFh]

SDFLT3CMPH1 is shown in Figure 19-51 and described in Table 19-48.

Return to the Summary Table.

High-level Threshold Register for Ch3

Figure 19-51. SDFLT3CMPH1 Register

15	14	13	12	11	10	9	8
RESERVED	HLT						
R-0h	R/W-7FFFh						
7	6	5	4	3	2	1	0
HLT							
R/W-7FFFh							

Table 19-48. SDFLT3CMPH1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	HLT	R/W	7FFFh	Unsigned high-level threshold for the comparator filter output. Reset type: SYSRSn

19.12.2.38 SDFLT3CMPL1 Register (Offset = 34h) [Reset = 0h]

SDFLT3CMPL1 is shown in Figure 19-52 and described in Table 19-49.

Return to the Summary Table.

Low-level Threshold Register for Ch3

Figure 19-52. SDFLT3CMPL1 Register

15	14	13	12	11	10	9	8
RESERVED	LLT						
R-0h	R/W-0h						
7	6	5	4	3	2	1	0
LLT							
R/W-0h							

Table 19-49. SDFLT3CMPL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	LLT	R/W	0h	Unsigned low-level threshold for the comparator filter output. Reset type: SYSRSn

19.12.2.39 SDCPARM3 Register (Offset = 35h) [Reset = 2000h]

SDCPARM3 is shown in Figure 19-53 and described in Table 19-50.

Return to the Summary Table.

Comparator Filter Parameter Register for Ch3

Figure 19-53. SDCPARM3 Register

15	14	13	12	11	10	9	8
CEVT2SEL		CEN	CEVT1SEL		HZEN	MFIE	CS1_CS0
R/W-0h		R/W-1h	R/W-0h		R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CS1_CS0	EN_CEVT2	EN_CEVT1	COSR				
R/W-0h	R/W-0h	R/W-0h	R/W-0h				

Table 19-50. SDCPARM3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	CEVT2SEL	R/W	0h	Comparator Event2 Select 00: COMPL1 01: COMPL1 OR COMPH1 10: COMPL2 11: COMPL2 OR COMPH2 Reset type: SYSRSn
13	CEN	R/W	1h	Comparator Filter enable 0: Disable comparator filter 1: Enable comparator filter Reset type: SYSRSn
12-11	CEVT1SEL	R/W	0h	Comparator Event1 Select 00: COMPH1 01: COMPL1 OR COMPH1 10: COMPH2 11: COMPL2 OR COMPH2 Reset type: SYSRSn
10	HZEN	R/W	0h	High level (Z) Threshold crossing output enable 0: Disable Higher level Threshold (Z) crossing 1: Enable Higher level Threhold (Z) crossing Reset type: SYSRSn
9	MFIE	R/W	0h	Modulator Failure Interrupt Enable 0: Disable modulator failure interrupt and its flag 1: Enable modulator failure interrupt and its flag Reset type: SYSRSn
8-7	CS1_CS0	R/W	0h	Comparator filter structure 00: Comparator filter runs with a sincfast structure 01: Comparator filter runs with a Sinc1 structure 10: Comparator filter runs with a Sinc2 structure 11: Comparator filter runs with a Sinc3 structure Reset type: SYSRSn
6	EN_CEVT2	R/W	0h	CEVT2 interrupt enable 0: Disable CEVT2 interrupt 1: Enable CEVT2 interrupt Reset type: SYSRSn
5	EN_CEVT1	R/W	0h	CEVT1 interrupt enable 0: Disable CEVT1 interrupt 1: Enable CEVT1 interrupt Reset type: SYSRSn

Table 19-50. SDCPARM3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-0	COSR	R/W	0h	Comparator Oversampling ratio. The actual rate is COSR + 1. These bits set the oversampling ratio of the filter. 0x1F represents an oversampling ratio of 32 Reset type: SYSRSn

19.12.2.40 SDDATA3 Register (Offset = 36h) [Reset = 0h]

SDDATA3 is shown in Figure 19-54 and described in Table 19-51.

Return to the Summary Table.

Data Filter Data Register (16 or 32bit) for Ch3

Figure 19-54. SDDATA3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA32HI																DATA16															
R-0h																R-0h															

Table 19-51. SDDATA3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DATA32HI	R	0h	Hi-order 16b in 32b mode, 16-bit Data in 16b mode Reset type: SYSRSn
15-0	DATA16	R	0h	Lo-order 16b in 32b mode Reset type: SYSRSn

19.12.2.41 SDDATFIFO3 Register (Offset = 38h) [Reset = 0h]

SDDATFIFO3 is shown in Figure 19-55 and described in Table 19-52.

Return to the Summary Table.

Filter Data FIFO Output(32b) for Ch3

Figure 19-55. SDDATFIFO3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA32HI																DATA16															
R-0h																R-0h															

Table 19-52. SDDATFIFO3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DATA32HI	R	0h	Hi-order 16b in 32b mode, 16-bit Data in 16b mode Reset type: SYSRSn
15-0	DATA16	R	0h	Lo-order 16b in 32b mode Reset type: SYSRSn

19.12.2.42 SDCDATA3 Register (Offset = 3Ah) [Reset = 0h]

SDCDATA3 is shown in Figure 19-56 and described in Table 19-53.

Return to the Summary Table.

Comparator Filter Data Register (16b) for Ch3

Figure 19-56. SDCDATA3 Register

15	14	13	12	11	10	9	8
DATA16							
R-0h							
7	6	5	4	3	2	1	0
DATA16							
R-0h							

Table 19-53. SDCDATA3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	DATA16	R	0h	Comparator Data output - 16b only Reset type: SYSRSn

19.12.2.43 SDFLT3CMPH2 Register (Offset = 3Bh) [Reset = 7FFFh]

SDFLT3CMPH2 is shown in Figure 19-57 and described in Table 19-54.

Return to the Summary Table.

Second high level threhold for CH3

Figure 19-57. SDFLT3CMPH2 Register

15	14	13	12	11	10	9	8
RESERVED	HLT2						
R-0h	R/W-7FFFh						
7	6	5	4	3	2	1	0
HLT2							
R/W-7FFFh							

Table 19-54. SDFLT3CMPH2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	HLT2	R/W	7FFFh	Second Unsigned high-level threshold for the comparator filter output. Reset type: SYSRSn

19.12.2.44 SDFLT3CMPHZ Register (Offset = 3Ch) [Reset = 0h]

SDFLT3CMPHZ is shown in Figure 19-58 and described in Table 19-55.

Return to the Summary Table.

High-level (Z) Threshold Register for Ch3

Figure 19-58. SDFLT3CMPHZ Register

15	14	13	12	11	10	9	8
RESERVED	HLTZ						
R-0h	R/W-0h						
7	6	5	4	3	2	1	0
HLTZ							
R/W-0h							

Table 19-55. SDFLT3CMPHZ Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	HLTZ	R/W	0h	Unsigned High-level threshold (Z) for the comparator filter output Primarily intended for detecting "zero"-crossing events. Unlike the primary comparator SDCMPHx, it does not have the ability to generate an interrupt. Reset type: SYSRSn

19.12.2.45 SDFIFOCTL3 Register (Offset = 3Dh) [Reset = 0h]

SDFIFOCTL3 is shown in Figure 19-59 and described in Table 19-56.

Return to the Summary Table.

FIFO Control Register for Ch3

Figure 19-59. SDFIFOCTL3 Register

15	14	13	12	11	10	9	8
OVFIEN	DRINTSEL	FFEN	FFIEN	RESERVED	SDFFST		
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R-0h		
7	6	5	4	3	2	1	0
SDFFST		RESERVED	SDFFIL				
R-0h		R-0h	R/W-0h				

Table 19-56. SDFIFOCTL3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	OVFIEN	R/W	0h	SDFIFO Overflow interrupt enable 0: SDFIFO Overflow condition will not generate an interrupt 1: SDFIFO overflow condition generates an interrupt on SDy_ERR Reset type: SYSRSn
14	DRINTSEL	R/W	0h	Data-Ready Interrupt (DRINT) source select 0 = AF1 (Select non-FIFO data-ready interrupt) 1 = SDFFINT1 (Select FIFO data-ready interrupt) Reset type: SYSRSn
13	FFEN	R/W	0h	SDFIFO Enable 0: Disable FIFO operation 1: Enable FIFO operation Note: When FIFO is disabled, FIFO contents are cleared Reset type: SYSRSn
12	FFIEN	R/W	0h	SDFIFO data ready Interrupt Enable Reset type: SYSRSn
11	RESERVED	R-0	0h	Reserved
10-6	SDFFST	R	0h	SDFIFO Status 00000 FIFO empty 00001 FIFO has 1 word . . . . 10000 FIFO has 16 words Reset type: SYSRSn
5	RESERVED	R-0	0h	Reserved
4-0	SDFFIL	R/W	0h	SDFIFO interrupt level bits The FIFO will generate an interrupt when the FIFO status (SDFFST) >= FIFO level (SDFFIL) Reset type: SYSRSn

19.12.2.46 SDSYNC3 Register (Offset = 3Eh) [Reset = 43Fh]

SDSYNC3 is shown in Figure 19-60 and described in Table 19-57.

Return to the Summary Table.

SD Filter Sync control for Ch3

Figure 19-60. SDSYNC3 Register

15	14	13	12	11	10	9	8
RESERVED					WTSCLREN	FFSYNCCLREN	WTSYNCLR
R-0h					R/W-1h	R/W-0h	R-0/W-0h
7	6	5	4	3	2	1	0
WTSYNFLG	WTSYNCEN	SYNCSEL					
R-0h	R/W-0h	R/W-3Fh					

Table 19-57. SDSYNC3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R-0	0h	Reserved
10	WTSCLREN	R/W	1h	WTSYNFLG Clear-on-FIFOINT Enable 0: WTSYNFLG can only be cleared manually (using WTSYNCLR bit) 1: WTSYNFLG is cleared automatically on SDFFINT Reset type: SYSRSn
9	FFSYNCCLREN	R/W	0h	FIFO Clear-on-SDSYNC Enable 0: SDFIFO is not automaticaly cleared upon receiving SDSYNC 1: SDFIFO is automaticaly cleared upon receiving SDSYNC Reset type: SYSRSn
8	WTSYNCLR	R-0/W	0h	Wait-for-Sync Flag Clear (always reads 0) 0: Write of 0 has no affect 1: Write of 1 clears WTSYNFLG Reset type: SYSRSn
7	WTSYNFLG	R	0h	Wait-for-Sync Flag 0: SDSYNC event has not occurred 1: SDSYNC event occurred. Reset type: SYSRSn
6	WTSYNCEN	R/W	0h	Wait-for-Sync Enable 0: Incoming Data written to SDFIFO on every Data-Ready (DR) Event 1: Incoming Data written to SDFIFO on DR event only after SDSYNC event occurs Reset type: SYSRSn
5-0	SYNCSEL	R/W	3Fh	Defines source for the SDSYNC Input on this channel Refer SDSYNCx.SYNCSEL table Reset type: SYSRSn

19.12.2.47 SDFLT3CMPL2 Register (Offset = 3Fh) [Reset = 0h]

SDFLT3CMPL2 is shown in Figure 19-61 and described in Table 19-58.

Return to the Summary Table.

Second low level threhold for CH3

Figure 19-61. SDFLT3CMPL2 Register

15	14	13	12	11	10	9	8
RESERVED	LLT2						
R-0h	R/W-0h						
7	6	5	4	3	2	1	0
LLT2							
R/W-0h							

Table 19-58. SDFLT3CMPL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	LLT2	R/W	0h	Second Unsigned low-level threshold for the comparator filter output. Reset type: SYSRSn

19.12.2.48 SDCTLPARM4 Register (Offset = 40h) [Reset = 0h]

SDCTLPARM4 is shown in Figure 19-62 and described in Table 19-59.

Return to the Summary Table.

Control Parameter Register for Ch4

Figure 19-62. SDCTLPARM4 Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	SDDATASYNC	RESERVED	SDCLKSYNC	SDCLKSEL	RESERVED	MOD	
R-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	

Table 19-59. SDCTLPARM4 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	RESERVED	R-0	0h	Reserved
6	SDDATASYNC	R/W	0h	0: SD Data is not passed through a synchronizer. 1: SD Data is passed through a synchronizer. Reset type: SYSRSn
5	RESERVED	R-0	0h	Reserved
4	SDCLKSYNC	R/W	0h	0: SD Clock is not passed through a synchronizer. 1: SD Clock is passed through a synchronizer. Reset type: SYSRSn
3	SDCLKSEL	R/W	0h	SD4 Clock source select. 0: Clock source to SDFM filter is its channel clock. 1: Clock source to SDFM filter is SD1 filter clock. Reset type: SYSRSn
2	RESERVED	R	0h	Reserved
1-0	MOD	R/W	0h	Modulator clock modes 0: Mode 0: Modulator clock running at 1x data rate 1: Reserved 2: Reserved 3: Reserved Reset type: SYSRSn

19.12.2.49 SDDFPARM4 Register (Offset = 41h) [Reset = 0h]

SDDFPARM4 is shown in Figure 19-63 and described in Table 19-60.

Return to the Summary Table.

Data Filter Parameter Register for Ch4

Figure 19-63. SDDFPARM4 Register

15	14	13	12	11	10	9	8
RESERVED			SDSYNCEN	SST		AE	FEN
R-0h			R/W-0h	R/W-0h		R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DOSR							
R/W-0h							

Table 19-60. SDDFPARM4 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12	SDSYNCEN	R/W	0h	PWM synchronization (SDSYNC) of data filter 0: PWM synchronization of data filter is disabled 1: PWM synchronization of data filter is enabled Note: SDSYNCx.SYNCSEL bits define which PWM signal is used to synchronize PWMs Reset type: SYSRSn
11-10	SST	R/W	0h	Data filter structure 00: Data filter runs with a Sincfast structure 01: Data filter runs with a Sinc1 structure 10: Data filter runs with a Sinc2 structure 11: Data filter runs with a Sinc3 structure Reset type: SYSRSn
9	AE	R/W	0h	Data filter Acknowledge Enable 0: Acknowledge flag is disabled for the particular filter 1: Acknowledge flag is enabled for the particular filter Reset type: SYSRSn
8	FEN	R/W	0h	Filter Enable 0: The data filter is disabled and no data is produced 1: The data filter is enabled and data are produced in the data filter Note: When filter is disabled, DOSR counter held in reset, filter data erased. Also resets FIFO pointers and clears the FIFO Reset type: SYSRSn
7-0	DOSR	R/W	0h	Data filter Oversampling ratio The actual oversampling ratio of data filter is DOSR + 1 These bits set the oversampling ratio of the data filter. 0x0FF represents an oversampling ratio of 256. Reset type: SYSRSn

19.12.2.50 SDDPARM4 Register (Offset = 42h) [Reset = 0h]

SDDPARM4 is shown in Figure 19-64 and described in Table 19-61.

Return to the Summary Table.

Data Parameter Register for Ch4

Figure 19-64. SDDPARM4 Register

15	14	13	12	11	10	9	8
SH					DR	RESERVED	
R/W-0h					R/W-0h	R-0h	
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

Table 19-61. SDDPARM4 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	SH	R/W	0h	Shift Control These bits indicate by how many bits the 16-bit window is shifted up when 16-bit data representation is chosen. Reset type: SYSRSn
10	DR	R/W	0h	Data filter Data representation 0: Data stored in 16b 2's complement 1: Data stored in 32b 2's complement Reset type: SYSRSn
9-0	RESERVED	R-0	0h	Reserved

19.12.2.51 SDFLT4CMPH1 Register (Offset = 43h) [Reset = 7FFFh]

SDFLT4CMPH1 is shown in Figure 19-65 and described in Table 19-62.

Return to the Summary Table.

High-level Threshold Register for Ch4

Figure 19-65. SDFLT4CMPH1 Register

15	14	13	12	11	10	9	8
RESERVED	HLT						
R-0h	R/W-7FFFh						
7	6	5	4	3	2	1	0
HLT							
R/W-7FFFh							

Table 19-62. SDFLT4CMPH1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	HLT	R/W	7FFFh	Unsigned high-level threshold for the comparator filter output. Reset type: SYSRSn

19.12.2.52 SDFLT4CMPL1 Register (Offset = 44h) [Reset = 0h]

SDFLT4CMPL1 is shown in Figure 19-66 and described in Table 19-63.

Return to the Summary Table.

Low-level Threshold Register for Ch4

Figure 19-66. SDFLT4CMPL1 Register

15	14	13	12	11	10	9	8
RESERVED	LLT						
R-0h	R/W-0h						
7	6	5	4	3	2	1	0
LLT							
R/W-0h							

Table 19-63. SDFLT4CMPL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	LLT	R/W	0h	Unsigned low-level threshold for the comparator filter output. Reset type: SYSRSn

19.12.2.53 SDCPARM4 Register (Offset = 45h) [Reset = 2000h]

SDCPARM4 is shown in Figure 19-67 and described in Table 19-64.

Return to the Summary Table.

Comparator Filter Parameter Register for Ch4

Figure 19-67. SDCPARM4 Register

15	14	13	12	11	10	9	8
CEVT2SEL		CEN	CEVT1SEL		HZEN	MFIE	CS1_CS0
R/W-0h		R/W-1h	R/W-0h		R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CS1_CS0	EN_CEVT2	EN_CEVT1	COSR				
R/W-0h	R/W-0h	R/W-0h	R/W-0h				

Table 19-64. SDCPARM4 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	CEVT2SEL	R/W	0h	Comparator Event2 Select 00: COMPL1 01: COMPL1 OR COMPH1 10: COMPL2 11: COMPL2 OR COMPH2 Reset type: SYSRSn
13	CEN	R/W	1h	Comparator Filter enable 0: Disable comparator filter 1: Enable comparator filter Reset type: SYSRSn
12-11	CEVT1SEL	R/W	0h	Comparator Event1 Select 00: COMPH1 01: COMPL1 OR COMPH1 10: COMPH2 11: COMPL2 OR COMPH2 Reset type: SYSRSn
10	HZEN	R/W	0h	High level (Z) Threshold crossing output enable 0: Disable Higher level Threshold (Z) crossing 1: Enable Higher level Threhold (Z) crossing Reset type: SYSRSn
9	MFIE	R/W	0h	Modulator Failure Interrupt Enable 0: Disable modulator failure interrupt and its flag 1: Enable modulator failure interrupt and its flag Reset type: SYSRSn
8-7	CS1_CS0	R/W	0h	Comparator filter structure 00: Comparator filter runs with a sincfast structure 01: Comparator filter runs with a Sinc1 structure 10: Comparator filter runs with a Sinc2 structure 11: Comparator filter runs with a Sinc3 structure Reset type: SYSRSn
6	EN_CEVT2	R/W	0h	CEVT2 interrupt enable 0: Disable CEVT2 interrupt 1: Enable CEVT2 interrupt Reset type: SYSRSn
5	EN_CEVT1	R/W	0h	CEVT1 interrupt enable 0: Disable CEVT1 interrupt 1: Enable CEVT1 interrupt Reset type: SYSRSn

Table 19-64. SDCPARM4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-0	COSR	R/W	0h	Comparator Oversampling ratio. The actual rate is COSR + 1. These bits set the oversampling ratio of the filter. 0x1F represents an oversampling ratio of 32 Reset type: SYSRSn

19.12.2.54 SDDATA4 Register (Offset = 46h) [Reset = 0h]

SDDATA4 is shown in Figure 19-68 and described in Table 19-65.

Return to the Summary Table.

Data Filter Data Register (16 or 32bit) for Ch4

Figure 19-68. SDDATA4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA32HI																DATA16															
R-0h																R-0h															

Table 19-65. SDDATA4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DATA32HI	R	0h	Hi-order 16b in 32b mode, 16-bit Data in 16b mode Reset type: SYSRSn
15-0	DATA16	R	0h	Lo-order 16b in 32b mode Reset type: SYSRSn

19.12.2.55 SDDATFIFO4 Register (Offset = 48h) [Reset = 0h]

SDDATFIFO4 is shown in Figure 19-69 and described in Table 19-66.

Return to the Summary Table.

Filter Data FIFO Output(32b) for Ch4

Figure 19-69. SDDATFIFO4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA32HI																DATA16															
R-0h																R-0h															

Table 19-66. SDDATFIFO4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DATA32HI	R	0h	Hi-order 16b in 32b mode, 16-bit Data in 16b mode Reset type: SYSRSn
15-0	DATA16	R	0h	Lo-order 16b in 32b mode Reset type: SYSRSn

19.12.2.56 SDCDATA4 Register (Offset = 4Ah) [Reset = 0h]

SDCDATA4 is shown in Figure 19-70 and described in Table 19-67.

Return to the Summary Table.

Comparator Filter Data Register (16b) for Ch4

Figure 19-70. SDCDATA4 Register

15	14	13	12	11	10	9	8
DATA16							
R-0h							
7	6	5	4	3	2	1	0
DATA16							
R-0h							

Table 19-67. SDCDATA4 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	DATA16	R	0h	Comparator Data output - 16b only Reset type: SYSRSn

19.12.2.57 SDFLT4CMPH2 Register (Offset = 4Bh) [Reset = 7FFFh]

SDFLT4CMPH2 is shown in Figure 19-71 and described in Table 19-68.

Return to the Summary Table.

Second high level threhold for CH4

Figure 19-71. SDFLT4CMPH2 Register

15	14	13	12	11	10	9	8
RESERVED	HLT2						
R-0h	R/W-7FFFh						
7	6	5	4	3	2	1	0
HLT2							
R/W-7FFFh							

Table 19-68. SDFLT4CMPH2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	HLT2	R/W	7FFFh	Second Unsigned high-level threshold for the comparator filter output. Reset type: SYSRSn

19.12.2.58 SDFLT4CMPHZ Register (Offset = 4Ch) [Reset = 0h]

SDFLT4CMPHZ is shown in Figure 19-72 and described in Table 19-69.

Return to the Summary Table.

High-level (Z) Threshold Register for Ch4

Figure 19-72. SDFLT4CMPHZ Register

15	14	13	12	11	10	9	8
RESERVED	HLTZ						
R-0h	R/W-0h						
7	6	5	4	3	2	1	0
HLTZ							
R/W-0h							

Table 19-69. SDFLT4CMPHZ Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	HLTZ	R/W	0h	Unsigned High-level threshold (Z) for the comparator filter output Primarily intended for detecting "zero"-crossing events. Unlike the primary comparator SDCMPHx, it does not have the ability to generate an interrupt. Reset type: SYSRSn

19.12.2.59 SDFIFOCTL4 Register (Offset = 4Dh) [Reset = 0h]

SDFIFOCTL4 is shown in Figure 19-73 and described in Table 19-70.

Return to the Summary Table.

FIFO Control Register for Ch4

Figure 19-73. SDFIFOCTL4 Register

15	14	13	12	11	10	9	8
OVFIEN	DRINTSEL	FFEN	FFIEN	RESERVED	SDFFST		
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R-0h		
7	6	5	4	3	2	1	0
SDFFST		RESERVED	SDFFIL				
R-0h		R-0h	R/W-0h				

Table 19-70. SDFIFOCTL4 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	OVFIEN	R/W	0h	SDFIFO Overflow interrupt enable 0: SDFIFO Overflow condition will not generate an interrupt 1: SDFIFO overflow condition generates an interrupt on SDy_ERR Reset type: SYSRSn
14	DRINTSEL	R/W	0h	Data-Ready Interrupt (DRINT) source select 0 = AF1 (Select non-FIFO data-ready interrupt) 1 = SDFFINT1 (Select FIFO data-ready interrupt) Reset type: SYSRSn
13	FFEN	R/W	0h	SDFIFO Enable 0: Disable FIFO operation 1: Enable FIFO operation Note: When FIFO is disabled, FIFO contents are cleared Reset type: SYSRSn
12	FFIEN	R/W	0h	SDFIFO data ready Interrupt Enable Reset type: SYSRSn
11	RESERVED	R-0	0h	Reserved
10-6	SDFFST	R	0h	SDFIFO Status 00000 FIFO empty 00001 FIFO has 1 word . . . . 10000 FIFO has 16 words Reset type: SYSRSn
5	RESERVED	R-0	0h	Reserved
4-0	SDFFIL	R/W	0h	SDFIFO interrupt level bits The FIFO will generate an interrupt when the FIFO status (SDFFST) >= FIFO level (SDFFIL) Reset type: SYSRSn

19.12.2.60 SDSYNC4 Register (Offset = 4Eh) [Reset = 43Fh]

SDSYNC4 is shown in Figure 19-74 and described in Table 19-71.

Return to the Summary Table.

SD Filter Sync control for Ch4

Figure 19-74. SDSYNC4 Register

15	14	13	12	11	10	9	8
RESERVED					WTSCLREN	FFSYNCCLREN	WTSYNCLR
R-0h					R/W-1h	R/W-0h	R-0/W-0h
7	6	5	4	3	2	1	0
WTSYNFLG	WTSYNCEN	SYNCSEL					
R-0h	R/W-0h	R/W-3Fh					

Table 19-71. SDSYNC4 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R-0	0h	Reserved
10	WTSCLREN	R/W	1h	WTSYNFLG Clear-on-FIFOINT Enable 0: WTSYNFLG can only be cleared manually (using WTSYNCLR bit) 1: WTSYNFLG is cleared automatically on SDFFINT Reset type: SYSRSn
9	FFSYNCCLREN	R/W	0h	FIFO Clear-on-SDSYNC Enable 0: SDFIFO is not automaticaly cleared upon receiving SDSYNC 1: SDFIFO is automaticaly cleared upon receiving SDSYNC Reset type: SYSRSn
8	WTSYNCLR	R-0/W	0h	Wait-for-Sync Flag Clear (always reads 0) 0: Write of 0 has no affect 1: Write of 1 clears WTSYNFLG Reset type: SYSRSn
7	WTSYNFLG	R	0h	Wait-for-Sync Flag 0: SDSYNC event has not occurred 1: SDSYNC event occurred. Reset type: SYSRSn
6	WTSYNCEN	R/W	0h	Wait-for-Sync Enable 0: Incoming Data written to SDFIFO on every Data-Ready (DR) Event 1: Incoming Data written to SDFIFO on DR event only after SDSYNC event occurs Reset type: SYSRSn
5-0	SYNCSEL	R/W	3Fh	Defines source for the SDSYNC Input on this channel Refer SDSYNCx.SYNCSEL table Reset type: SYSRSn

19.12.2.61 SDFLT4CMPL2 Register (Offset = 4Fh) [Reset = 0h]

SDFLT4CMPL2 is shown in Figure 19-75 and described in Table 19-72.

Return to the Summary Table.

Second low level threhold for CH4

Figure 19-75. SDFLT4CMPL2 Register

15	14	13	12	11	10	9	8
RESERVED	LLT2						
R-0h	R/W-0h						
7	6	5	4	3	2	1	0
LLT2							
R/W-0h							

Table 19-72. SDFLT4CMPL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R-0	0h	Reserved
14-0	LLT2	R/W	0h	Second Unsigned low-level threshold for the comparator filter output. Reset type: SYSRSn

19.12.2.62 SDCOMP1CTL Register (Offset = 60h) [Reset = 0h]

SDCOMP1CTL is shown in Figure 19-76 and described in Table 19-73.

Return to the Summary Table.

SD Comparator event filter1 Control Register

Figure 19-76. SDCOMP1CTL Register

15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED		CEVT2DIGFILTSEL		RESERVED	RESERVED
R-0h	R-0h	R-0h		R/W-0h		R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED		CEVT1DIGFILTSEL		RESERVED	RESERVED
R-0h	R-0h	R-0h		R/W-0h		R-0h	R-0h

Table 19-73. SDCOMP1CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	RESERVED	R	0h	Reserved
13-12	RESERVED	R	0h	Reserved
11-10	CEVT2DIGFILTSEL	R/W	0h	High comparator COMPH source select. 0 CEVT2 output drives COMPLOUT 1 Reserved 2 Output of digital filter drives COMPLOUT 3 Reserved Reset type: SYSRSn
9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	RESERVED	R	0h	Reserved
6	RESERVED	R	0h	Reserved
5-4	RESERVED	R	0h	Reserved
3-2	CEVT1DIGFILTSEL	R/W	0h	High comparator COMPH source select. 0 CEVT1 output drives COMPHOUT 1 Reserved 2 Output of digital filter drives COMPHOUT 3 Reserved Reset type: SYSRSn
1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

19.12.2.63 SDCOMP1EVT2FLTCTL Register (Offset = 61h) [Reset = 0h]

SDCOMP1EVT2FLTCTL is shown in Figure 19-77 and described in Table 19-74.

Return to the Summary Table.

COMPL/CEVT2 Digital filter1 Control Register

Figure 19-77. SDCOMP1EVT2FLTCTL Register

15	14	13	12	11	10	9	8
FILINIT	RESERVED	THRESH					SAMPWIN
R-0/W1S-0h	R-0h	R/W-0h					R/W-0h
7	6	5	4	3	2	1	0
SAMPWIN				RESERVED			
R/W-0h				R-0h			

Table 19-74. SDCOMP1EVT2FLTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	FILINIT	R-0/W1S	0h	Low filter initialization. 0 No effect 1 Initialize all samples to the filter input value Reset type: SYSRSn
14	RESERVED	R	0h	Reserved
13-9	THRESH	R/W	0h	Low filter majority voting threshold. At least THRESH samples of the opposite state must appear within the sample window in order for the output to change state. Reset type: SYSRSn
8-4	SAMPWIN	R/W	0h	Low filter sample window size. Number of samples to monitor is SAMPWIN+1. Reset type: SYSRSn
3-0	RESERVED	R	0h	Reserved

19.12.2.64 SDCOMP1EVT2FLTCLKCTL Register (Offset = 62h) [Reset = 0h]

SDCOMP1EVT2FLTCLKCTL is shown in Figure 19-78 and described in Table 19-75.

Return to the Summary Table.

COMPL/CEVT2 Digital filter1 Clock Control Register

Figure 19-78. SDCOMP1EVT2FLTCLKCTL Register

15	14	13	12	11	10	9	8
RESERVED						CLKPRESCALE	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
CLKPRESCALE							
R/W-0h							

Table 19-75. SDCOMP1EVT2FLTCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	CLKPRESCALE	R/W	0h	Low filter sample clock prescale. Number of system clocks between samples. Reset type: SYSRSn

19.12.2.65 SDCOMP1EVT1FLTCTL Register (Offset = 63h) [Reset = 0h]

SDCOMP1EVT1FLTCTL is shown in Figure 19-79 and described in Table 19-76.

Return to the Summary Table.

COMPH/CEVT1 Digital filter1 Control Register

Figure 19-79. SDCOMP1EVT1FLTCTL Register

15	14	13	12	11	10	9	8
FILINIT	RESERVED	THRESH					SAMPWIN
R-0/W1S-0h	R-0h	R/W-0h					R/W-0h
7	6	5	4	3	2	1	0
SAMPWIN				RESERVED			
R/W-0h				R-0h			

Table 19-76. SDCOMP1EVT1FLTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	FILINIT	R-0/W1S	0h	High filter initialization. 0 No effect 1 Initialize all samples to the filter input value Reset type: SYSRSn
14	RESERVED	R	0h	Reserved
13-9	THRESH	R/W	0h	High filter majority voting threshold. At least THRESH samples of the opposite state must appear within the sample window in order for the output to change state. Reset type: SYSRSn
8-4	SAMPWIN	R/W	0h	High filter sample window size. Number of samples to monitor is SAMPWIN+1. Reset type: SYSRSn
3-0	RESERVED	R	0h	Reserved

19.12.2.66 SDCOMP1EVT1FLTCLKCTL Register (Offset = 64h) [Reset = 0h]

SDCOMP1EVT1FLTCLKCTL is shown in Figure 19-80 and described in Table 19-77.

Return to the Summary Table.

COMPH/CEVT1 Digital filter1 Clock Control Register

Figure 19-80. SDCOMP1EVT1FLTCLKCTL Register

15	14	13	12	11	10	9	8
RESERVED						CLKPRESCALE	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
CLKPRESCALE							
R/W-0h							

Table 19-77. SDCOMP1EVT1FLTCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	CLKPRESCALE	R/W	0h	High filter sample clock prescale. Number of system clocks between samples. Reset type: SYSRSn

19.12.2.67 SDCOMP1LOCK Register (Offset = 67h) [Reset = 0h]

SDCOMP1LOCK is shown in Figure 19-81 and described in Table 19-78.

Return to the Summary Table.

SD compartor event filter1 Lock Register

Figure 19-81. SDCOMP1LOCK Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			RESERVED	COMP	RESERVED	RESERVED	SDCOMP1CTL
R-0h			R-0h	R/WSonce-0h	R-0h	R-0h	R/WSonce-0h

Table 19-78. SDCOMP1LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3	COMP	R/WSonce	0h	Lock write-access to the SDCOMP1EVT1/2FLTTCTL and COMP1FILCLKCTL registers. 0 SDCOMP1EVT1/2FLTCTL and SDCOMP1EVT1/2FLTCLKCTL registers are not locked. Write 0 to this bit has no effect. 1 SDCOMP1EVT1/2FLTCTL and SDCOMP1EVT1/2FLTCLKCTL registers are locked. Only a system reset can clear this bit. Reset type: SYSRSn
2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved
0	SDCOMP1CTL	R/WSonce	0h	Lock write-access to the SDCOMP1CTL register. 0 SDCOMP1CTL register is not locked. Write 0 to this bit has no effect. 1 SDCOMP1CTL register is locked. Only a system reset can clear this bit. Reset type: SYSRSn

19.12.2.68 SDCOMP2CTL Register (Offset = 68h) [Reset = 0h]

SDCOMP2CTL is shown in Figure 19-82 and described in Table 19-79.

Return to the Summary Table.

SD Comparator event filter2 Control Register

Figure 19-82. SDCOMP2CTL Register

15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED		CEVT2DIGFILTSEL		RESERVED	RESERVED
R-0h	R-0h	R-0h		R/W-0h		R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED		CEVT1DIGFILTSEL		RESERVED	RESERVED
R-0h	R-0h	R-0h		R/W-0h		R-0h	R-0h

Table 19-79. SDCOMP2CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	RESERVED	R	0h	Reserved
13-12	RESERVED	R	0h	Reserved
11-10	CEVT2DIGFILTSEL	R/W	0h	High comparator COMPH source select. 0 CEVT2 output drives COMPLOUT 1 Reserved 2 Output of digital filter drives COMPLOUT 3 Reserved Reset type: SYSRSn
9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	RESERVED	R	0h	Reserved
6	RESERVED	R	0h	Reserved
5-4	RESERVED	R	0h	Reserved
3-2	CEVT1DIGFILTSEL	R/W	0h	High comparator COMPH source select. 0 CEVT1 output drives COMPHOUT 1 Reserved 2 Output of digital filter drives COMPHOUT 3 Reserved Reset type: SYSRSn
1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

19.12.2.69 SDCOMP2EVT2FLTCTL Register (Offset = 69h) [Reset = 0h]

SDCOMP2EVT2FLTCTL is shown in Figure 19-83 and described in Table 19-80.

Return to the Summary Table.

COMPL/CEVT2 Digital filter2 Control Register

Figure 19-83. SDCOMP2EVT2FLTCTL Register

15	14	13	12	11	10	9	8
FILINIT	RESERVED	THRESH					SAMPWIN
R-0/W1S-0h	R-0h	R/W-0h					R/W-0h
7	6	5	4	3	2	1	0
SAMPWIN				RESERVED			
R/W-0h				R-0h			

Table 19-80. SDCOMP2EVT2FLTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	FILINIT	R-0/W1S	0h	Low filter initialization. 0 No effect 1 Initialize all samples to the filter input value Reset type: SYSRSn
14	RESERVED	R	0h	Reserved
13-9	THRESH	R/W	0h	Low filter majority voting threshold. At least THRESH samples of the opposite state must appear within the sample window in order for the output to change state. Reset type: SYSRSn
8-4	SAMPWIN	R/W	0h	Low filter sample window size. Number of samples to monitor is SAMPWIN+1. Reset type: SYSRSn
3-0	RESERVED	R	0h	Reserved

19.12.2.70 SDCOMP2EVT2FLTCLKCTL Register (Offset = 6Ah) [Reset = 0h]

SDCOMP2EVT2FLTCLKCTL is shown in Figure 19-84 and described in Table 19-81.

Return to the Summary Table.

COMPL/CEVT2 Digital filter2 Clock Control Register

Figure 19-84. SDCOMP2EVT2FLTCLKCTL Register

15	14	13	12	11	10	9	8
RESERVED						CLKPRESCALE	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
CLKPRESCALE							
R/W-0h							

Table 19-81. SDCOMP2EVT2FLTCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	CLKPRESCALE	R/W	0h	Low filter sample clock prescale. Number of system clocks between samples. Reset type: SYSRSn

19.12.2.71 SDCOMP2EVT1FLTCTL Register (Offset = 6Bh) [Reset = 0h]

SDCOMP2EVT1FLTCTL is shown in Figure 19-85 and described in Table 19-82.

Return to the Summary Table.

COMPH/CEVT1 Digital filter2 Control Register

Figure 19-85. SDCOMP2EVT1FLTCTL Register

15	14	13	12	11	10	9	8
FILINIT	RESERVED	THRESH					SAMPWIN
R-0/W1S-0h	R-0h	R/W-0h					R/W-0h
7	6	5	4	3	2	1	0
SAMPWIN				RESERVED			
R/W-0h				R-0h			

Table 19-82. SDCOMP2EVT1FLTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	FILINIT	R-0/W1S	0h	High filter initialization. 0 No effect 1 Initialize all samples to the filter input value Reset type: SYSRSn
14	RESERVED	R	0h	Reserved
13-9	THRESH	R/W	0h	High filter majority voting threshold. At least THRESH samples of the opposite state must appear within the sample window in order for the output to change state. Reset type: SYSRSn
8-4	SAMPWIN	R/W	0h	High filter sample window size. Number of samples to monitor is SAMPWIN+1. Reset type: SYSRSn
3-0	RESERVED	R	0h	Reserved

19.12.2.72 SDCOMP2EVT1FLTCLKCTL Register (Offset = 6Ch) [Reset = 0h]

SDCOMP2EVT1FLTCLKCTL is shown in Figure 19-86 and described in Table 19-83.

Return to the Summary Table.

COMPH/CEVT1 Digital filter2 Clock Control Register

Figure 19-86. SDCOMP2EVT1FLTCLKCTL Register

15	14	13	12	11	10	9	8
RESERVED						CLKPRESCALE	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
CLKPRESCALE							
R/W-0h							

Table 19-83. SDCOMP2EVT1FLTCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	CLKPRESCALE	R/W	0h	High filter sample clock prescale. Number of system clocks between samples. Reset type: SYSRSn

19.12.2.73 SDCOMP2LOCK Register (Offset = 6Fh) [Reset = 0h]

SDCOMP2LOCK is shown in Figure 19-87 and described in Table 19-84.

Return to the Summary Table.

SD compartor event filter2 Lock Register

Figure 19-87. SDCOMP2LOCK Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			RESERVED	COMP	RESERVED	RESERVED	SDCOMP2CTL
R-0h			R-0h	R/WSonce-0h	R-0h	R-0h	R/WSonce-0h

Table 19-84. SDCOMP2LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3	COMP	R/WSonce	0h	Lock write-access to the SDCOMP2EVT1/2FLTTCTL and COMP2FILCLKCTL registers. 0 SDCOMP2EVT1/2FLTCTL and SDCOMP2EVT1/2FLTCLKCTL registers are not locked. Write 0 to this bit has no effect. 1 SDCOMP2EVT1/2FLTCTL and SDCOMP2EVT1/2FLTCLKCTL registers are locked. Only a system reset can clear this bit. Reset type: SYSRSn
2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved
0	SDCOMP2CTL	R/WSonce	0h	Lock write-access to the SDCOMP2CTL register. 0 SDCOMP2CTL register is not locked. Write 0 to this bit has no effect. 1 SDCOMP2CTL register is locked. Only a system reset can clear this bit. Reset type: SYSRSn

19.12.2.74 SDCOMP3CTL Register (Offset = 70h) [Reset = 0h]

SDCOMP3CTL is shown in Figure 19-88 and described in Table 19-85.

Return to the Summary Table.

SD Comparator event filter3 Control Register

Figure 19-88. SDCOMP3CTL Register

15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED		CEVT2DIGFILTSEL		RESERVED	RESERVED
R-0h	R-0h	R-0h		R/W-0h		R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED		CEVT1DIGFILTSEL		RESERVED	RESERVED
R-0h	R-0h	R-0h		R/W-0h		R-0h	R-0h

Table 19-85. SDCOMP3CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	RESERVED	R	0h	Reserved
13-12	RESERVED	R	0h	Reserved
11-10	CEVT2DIGFILTSEL	R/W	0h	High comparator COMPH source select. 0 CEVT2 output drives COMPLOUT 1 Reserved 2 Output of digital filter drives COMPLOUT 3 Reserved Reset type: SYSRSn
9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	RESERVED	R	0h	Reserved
6	RESERVED	R	0h	Reserved
5-4	RESERVED	R	0h	Reserved
3-2	CEVT1DIGFILTSEL	R/W	0h	High comparator COMPH source select. 0 CEVT1 output drives COMPHOUT 1 Reserved 2 Output of digital filter drives COMPHOUT 3 Reserved Reset type: SYSRSn
1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

19.12.2.75 SDCOMP3EVT2FLTCTL Register (Offset = 71h) [Reset = 0h]

SDCOMP3EVT2FLTCTL is shown in Figure 19-89 and described in Table 19-86.

Return to the Summary Table.

COMPL/CEVT2 Digital filter3 Control Register

Figure 19-89. SDCOMP3EVT2FLTCTL Register

15	14	13	12	11	10	9	8
FILINIT	RESERVED	THRESH					SAMPWIN
R-0/W1S-0h	R-0h	R/W-0h					R/W-0h
7	6	5	4	3	2	1	0
SAMPWIN				RESERVED			
R/W-0h				R-0h			

Table 19-86. SDCOMP3EVT2FLTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	FILINIT	R-0/W1S	0h	Low filter initialization. 0 No effect 1 Initialize all samples to the filter input value Reset type: SYSRSn
14	RESERVED	R	0h	Reserved
13-9	THRESH	R/W	0h	Low filter majority voting threshold. At least THRESH samples of the opposite state must appear within the sample window in order for the output to change state. Reset type: SYSRSn
8-4	SAMPWIN	R/W	0h	Low filter sample window size. Number of samples to monitor is SAMPWIN+1. Reset type: SYSRSn
3-0	RESERVED	R	0h	Reserved

19.12.2.76 SDCOMP3EVT2FLTCLKCTL Register (Offset = 72h) [Reset = 0h]

SDCOMP3EVT2FLTCLKCTL is shown in Figure 19-90 and described in Table 19-87.

Return to the Summary Table.

COMPL/CEVT2 Digital filter3 Clock Control Register

Figure 19-90. SDCOMP3EVT2FLTCLKCTL Register

15	14	13	12	11	10	9	8
RESERVED						CLKPRESCALE	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
CLKPRESCALE							
R/W-0h							

Table 19-87. SDCOMP3EVT2FLTCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	CLKPRESCALE	R/W	0h	Low filter sample clock prescale. Number of system clocks between samples. Reset type: SYSRSn

19.12.2.77 SDCOMP3EVT1FLTCTL Register (Offset = 73h) [Reset = 0h]

SDCOMP3EVT1FLTCTL is shown in Figure 19-91 and described in Table 19-88.

Return to the Summary Table.

COMPH/CEVT1 Digital filter3 Control Register

Figure 19-91. SDCOMP3EVT1FLTCTL Register

15	14	13	12	11	10	9	8
FILINIT	RESERVED	THRESH					SAMPWIN
R-0/W1S-0h	R-0h	R/W-0h					R/W-0h
7	6	5	4	3	2	1	0
SAMPWIN				RESERVED			
R/W-0h				R-0h			

Table 19-88. SDCOMP3EVT1FLTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	FILINIT	R-0/W1S	0h	High filter initialization. 0 No effect 1 Initialize all samples to the filter input value Reset type: SYSRSn
14	RESERVED	R	0h	Reserved
13-9	THRESH	R/W	0h	High filter majority voting threshold. At least THRESH samples of the opposite state must appear within the sample window in order for the output to change state. Reset type: SYSRSn
8-4	SAMPWIN	R/W	0h	High filter sample window size. Number of samples to monitor is SAMPWIN+1. Reset type: SYSRSn
3-0	RESERVED	R	0h	Reserved

19.12.2.78 SDCOMP3EVT1FLTCLKCTL Register (Offset = 74h) [Reset = 0h]

SDCOMP3EVT1FLTCLKCTL is shown in Figure 19-92 and described in Table 19-89.

Return to the Summary Table.

COMPH/CEVT1 Digital filter3 Clock Control Register

Figure 19-92. SDCOMP3EVT1FLTCLKCTL Register

15	14	13	12	11	10	9	8
RESERVED						CLKPRESCALE	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
CLKPRESCALE							
R/W-0h							

Table 19-89. SDCOMP3EVT1FLTCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	CLKPRESCALE	R/W	0h	High filter sample clock prescale. Number of system clocks between samples. Reset type: SYSRSn

19.12.2.79 SDCOMP3LOCK Register (Offset = 77h) [Reset = 0h]

SDCOMP3LOCK is shown in Figure 19-93 and described in Table 19-90.

Return to the Summary Table.

SD compartor event filter3 Lock Register

Figure 19-93. SDCOMP3LOCK Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			RESERVED	COMP	RESERVED	RESERVED	SDCOMP3CTL
R-0h			R-0h	R/WSonce-0h	R-0h	R-0h	R/WSonce-0h

Table 19-90. SDCOMP3LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3	COMP	R/WSonce	0h	Lock write-access to the SDCOMP3EVT1/2FLTTCTL and COMP3FILCLKCTL registers. 0 SDCOMP3EVT1/2FLTCTL and SDCOMP3EVT1/2FLTCLKCTL registers are not locked. Write 0 to this bit has no effect. 1 SDCOMP3EVT1/2FLTCTL and SDCOMP3EVT1/2FLTCLKCTL registers are locked. Only a system reset can clear this bit. Reset type: SYSRSn
2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved
0	SDCOMP3CTL	R/WSonce	0h	Lock write-access to the SDCOMP3CTL register. 0 SDCOMP3CTL register is not locked. Write 0 to this bit has no effect. 1 SDCOMP3CTL register is locked. Only a system reset can clear this bit. Reset type: SYSRSn

19.12.2.80 SDCOMP4CTL Register (Offset = 78h) [Reset = 0h]

SDCOMP4CTL is shown in Figure 19-94 and described in Table 19-91.

Return to the Summary Table.

SD Comparator event filter4 Control Register

Figure 19-94. SDCOMP4CTL Register

15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED		CEVT2DIGFILTSEL		RESERVED	RESERVED
R-0h	R-0h	R-0h		R/W-0h		R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	RESERVED	RESERVED		CEVT1DIGFILTSEL		RESERVED	RESERVED
R-0h	R-0h	R-0h		R/W-0h		R-0h	R-0h

Table 19-91. SDCOMP4CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	RESERVED	R	0h	Reserved
13-12	RESERVED	R	0h	Reserved
11-10	CEVT2DIGFILTSEL	R/W	0h	High comparator COMPH source select. 0 CEVT2 output drives COMPLOUT 1 Reserved 2 Output of digital filter drives COMPLOUT 3 Reserved Reset type: SYSRSn
9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	RESERVED	R	0h	Reserved
6	RESERVED	R	0h	Reserved
5-4	RESERVED	R	0h	Reserved
3-2	CEVT1DIGFILTSEL	R/W	0h	High comparator COMPH source select. 0 CEVT1 output drives COMPHOUT 1 Reserved 2 Output of digital filter drives COMPHOUT 3 Reserved Reset type: SYSRSn
1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

19.12.2.81 SDCOMP4EVT2FLTCTL Register (Offset = 79h) [Reset = 0h]

SDCOMP4EVT2FLTCTL is shown in Figure 19-95 and described in Table 19-92.

Return to the Summary Table.

COMPL/CEVT2 Digital filter4 Control Register

Figure 19-95. SDCOMP4EVT2FLTCTL Register

15	14	13	12	11	10	9	8
FILINIT	RESERVED	THRESH					SAMPWIN
R-0/W1S-0h	R-0h	R/W-0h					R/W-0h
7	6	5	4	3	2	1	0
SAMPWIN				RESERVED			
R/W-0h				R-0h			

Table 19-92. SDCOMP4EVT2FLTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	FILINIT	R-0/W1S	0h	Low filter initialization. 0 No effect 1 Initialize all samples to the filter input value Reset type: SYSRSn
14	RESERVED	R	0h	Reserved
13-9	THRESH	R/W	0h	Low filter majority voting threshold. At least THRESH samples of the opposite state must appear within the sample window in order for the output to change state. Reset type: SYSRSn
8-4	SAMPWIN	R/W	0h	Low filter sample window size. Number of samples to monitor is SAMPWIN+1. Reset type: SYSRSn
3-0	RESERVED	R	0h	Reserved

19.12.2.82 SDCOMP4EVT2FLTCLKCTL Register (Offset = 7Ah) [Reset = 0h]

SDCOMP4EVT2FLTCLKCTL is shown in Figure 19-96 and described in Table 19-93.

Return to the Summary Table.

COMPL/CEVT2 Digital filter4 Clock Control Register

Figure 19-96. SDCOMP4EVT2FLTCLKCTL Register

15	14	13	12	11	10	9	8
RESERVED						CLKPRESCALE	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
CLKPRESCALE							
R/W-0h							

Table 19-93. SDCOMP4EVT2FLTCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	CLKPRESCALE	R/W	0h	Low filter sample clock prescale. Number of system clocks between samples. Reset type: SYSRSn

19.12.2.83 SDCOMP4EVT1FLTCTL Register (Offset = 7Bh) [Reset = 0h]

SDCOMP4EVT1FLTCTL is shown in Figure 19-97 and described in Table 19-94.

Return to the Summary Table.

COMPH/CEVT1 Digital filter4 Control Register

Figure 19-97. SDCOMP4EVT1FLTCTL Register

15	14	13	12	11	10	9	8
FILINIT	RESERVED	THRESH					SAMPWIN
R-0/W1S-0h	R-0h	R/W-0h					R/W-0h
7	6	5	4	3	2	1	0
SAMPWIN				RESERVED			
R/W-0h				R-0h			

Table 19-94. SDCOMP4EVT1FLTCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	FILINIT	R-0/W1S	0h	High filter initialization. 0 No effect 1 Initialize all samples to the filter input value Reset type: SYSRSn
14	RESERVED	R	0h	Reserved
13-9	THRESH	R/W	0h	High filter majority voting threshold. At least THRESH samples of the opposite state must appear within the sample window in order for the output to change state. Reset type: SYSRSn
8-4	SAMPWIN	R/W	0h	High filter sample window size. Number of samples to monitor is SAMPWIN+1. Reset type: SYSRSn
3-0	RESERVED	R	0h	Reserved

19.12.2.84 SDCOMP4EVT1FLTCLKCTL Register (Offset = 7Ch) [Reset = 0h]

SDCOMP4EVT1FLTCLKCTL is shown in Figure 19-98 and described in Table 19-95.

Return to the Summary Table.

COMPH/CEVT1 Digital filter4 Clock Control Register

Figure 19-98. SDCOMP4EVT1FLTCLKCTL Register

15	14	13	12	11	10	9	8
RESERVED						CLKPRESCALE	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
CLKPRESCALE							
R/W-0h							

Table 19-95. SDCOMP4EVT1FLTCLKCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	CLKPRESCALE	R/W	0h	High filter sample clock prescale. Number of system clocks between samples. Reset type: SYSRSn

19.12.2.85 SDCOMP4LOCK Register (Offset = 7Fh) [Reset = 0h]

SDCOMP4LOCK is shown in Figure 19-99 and described in Table 19-96.

Return to the Summary Table.

SD compartor event filter4 Lock Register

Figure 19-99. SDCOMP4LOCK Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			RESERVED	COMP	RESERVED	RESERVED	SDCOMP4CTL
R-0h			R-0h	R/WSonce-0h	R-0h	R-0h	R/WSonce-0h

Table 19-96. SDCOMP4LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R	0h	Reserved
4	RESERVED	R	0h	Reserved
3	COMP	R/WSonce	0h	Lock write-access to the SDCOMP4EVT1/2FLTTCTL and COMP4FILCLKCTL registers. 0 SDCOMP4EVT1/2FLTCTL and SDCOMP4EVT1/2FLTCLKCTL registers are not locked. Write 0 to this bit has no effect. 1 SDCOMP4EVT1/2FLTCTL and SDCOMP4EVT1/2FLTCLKCTL registers are locked. Only a system reset can clear this bit. Reset type: SYSRSn
2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved
0	SDCOMP4CTL	R/WSonce	0h	Lock write-access to the SDCOMP4CTL register. 0 SDCOMP4CTL register is not locked. Write 0 to this bit has no effect. 1 SDCOMP4CTL register is locked. Only a system reset can clear this bit. Reset type: SYSRSn

19.12.3 SDFM Registers to Driverlib Functions

Table 19-97. SDFM Registers to Driverlib Functions

File	Driverlib Function
SDIFLG	
sdfm.h	SDFM_getThresholdStatus
sdfm.h	SDFM_getModulatorStatus
sdfm.h	SDFM_getNewFilterDataStatus
sdfm.h	SDFM_getFIFOOverflowStatus
sdfm.h	SDFM_getFIFOISRStatus
sdfm.h	SDFM_getIsrStatus
sdfm.h	SDFM_clearInterruptFlag
SDIFLGCLR	
sdfm.h	SDFM_clearInterruptFlag
SDCTL	
sdfm.h	SDFM_clearZeroCrossTripStatus
sdfm.h	SDFM_setupModulatorClock
sdfm.h	SDFM_enableMasterInterrupt

Table 19-97. SDFM Registers to Driverlib Functions (continued)

File	Driverlib Function
sdfm.h	SDFM_disableMasterInterrupt
sdfm.h	SDFM_selectClockSource
sdfm.h	SDFM_enableSynchronizer
sdfm.h	SDFM_disableSynchronizer
SDMFILEN	
sdfm.h	SDFM_enableMasterFilter
sdfm.h	SDFM_disableMasterFilter
SDSTATUS	
sdfm.h	SDFM_getZeroCrossTripStatus
SDCTLPARM1	
sdfm.h	SDFM_setupModulatorClock
sdfm.h	SDFM_selectClockSource
sdfm.h	SDFM_enableSynchronizer
sdfm.h	SDFM_disableSynchronizer
SDDFPARM1	
sdfm.h	SDFM_enableExternalReset
sdfm.h	SDFM_disableExternalReset
sdfm.h	SDFM_enableFilter
sdfm.h	SDFM_disableFilter
sdfm.h	SDFM_setFilterType
sdfm.h	SDFM_setFilterOverSamplingRatio
sdfm.h	SDFM_enableInterrupt
sdfm.h	SDFM_disableInterrupt
SDDPARM1	
sdfm.h	SDFM_setOutputDataFormat
sdfm.h	SDFM_setDataShiftValue
SDFLT1CMPH1	
sdfm.h	SDFM_setCompFilterHighThreshold
SDFLT1CMPL1	
sdfm.h	SDFM_setCompFilterLowThreshold
SDCPARM1	
sdfm.h	SDFM_enableComparator
sdfm.h	SDFM_disableComparator
sdfm.h	SDFM_selectCompEventSource
sdfm.h	SDFM_enableZeroCrossEdgeDetect
sdfm.h	SDFM_disableZeroCrossEdgeDetect
sdfm.h	SDFM_enableInterrupt
sdfm.h	SDFM_disableInterrupt
sdfm.h	SDFM_setComparatorFilterType
sdfm.h	SDFM_setCompFilterOverSamplingRatio
SDDATA1	
sdfm.h	SDFM_getFilterData
SDDATFIFO1	
sdfm.h	SDFM_getFIFOData
SDCDATA1	

Table 19-97. SDFM Registers to Driverlib Functions (continued)

File	Driverlib Function
sdfm.h	SDFM_getComparatorSincData
SDFLT1CMPH2	
-	
SDFLT1CMPHZ	
sdfm.h	SDFM_setCompFilterZeroCrossThreshold
SDFIFOCTL1	
sdfm.h	SDFM_enableFIFOBuffer
sdfm.h	SDFM_disableFIFOBuffer
sdfm.h	SDFM_enableInterrupt
sdfm.h	SDFM_disableInterrupt
sdfm.h	SDFM_getFIFODataCount
sdfm.h	SDFM_setFIFOInterruptLevel
sdfm.h	SDFM_setDataReadyInterruptSource
SDSYNC1	
sdfm.h	SDFM_getWaitForSyncStatus
sdfm.h	SDFM_clearWaitForSyncFlag
sdfm.h	SDFM_enableWaitForSync
sdfm.h	SDFM_disableWaitForSync
sdfm.h	SDFM_setPWMSyncSource
sdfm.h	SDFM_setFIFOClearOnSyncMode
sdfm.h	SDFM_setWaitForSyncClearMode
SDFLT1CMPL2	
-	
SDCTLPARM2	
-	See SDCTLPARM1
SDDFPARM2	
-	See SDDFPARM1
SDDPARM2	
-	See SDDPARM1
SDFLT2CMPH1	
-	
SDFLT2CMPL1	
-	
SDCPARM2	
-	See SDCPARM1
SDDATA2	
-	See SDDATA1
SDDATFIFO2	
-	
SDCDATA2	
-	
SDFLT2CMPH2	
-	
SDFLT2CMPHZ	
-	

Table 19-97. SDFM Registers to Driverlib Functions (continued)

File	Driverlib Function
SDFIFOCTL2	
-	
SDSYNC2	
-	
SDFLT2CMPL2	
-	
SDCTLPARM3	
-	See SDCTLPARM1
SDDFPARM3	
-	See SDDFPARM1
SDDPARM3	
-	See SDDPARM1
SDFLT3CMPH1	
-	
SDFLT3CMPL1	
-	
SDCPARM3	
-	See SDCPARM1
SDDATA3	
-	See SDDATA1
SDDATFIFO3	
-	
SDCDATA3	
-	
SDFLT3CMPH2	
-	
SDFLT3CMPHZ	
-	
SDFIFOCTL3	
-	
SDSYNC3	
-	
SDFLT3CMPL2	
-	
SDCTLPARM4	
-	See SDCTLPARM1
SDDFPARM4	
-	See SDDFPARM1
SDDPARM4	
-	See SDDPARM1
SDFLT4CMPH1	
-	
SDFLT4CMPL1	
-	
SDCPARM4	

Table 19-97. SDFM Registers to Driverlib Functions (continued)

File	Driverlib Function
-	See SDCPARM1
SDDATA4	
-	See SDDATA1
SDDATFIFO4	
-	
SDCDATA4	
-	
SDFLT4CMPH2	
-	
SDFLT4CMPHZ	
-	
SDFIFOCTL4	
-	
SDSYNC4	
-	
SDFLT4CMPL2	
-	
SDCOMP1CTL	
sdfm.h	SDFM_selectCompEventHighSource
sdfm.h	SDFM_selectCompEventLowSource
SDCOMP1EVT2FLTCTL	
sdfm.c	SDFM_configCompEventLowFilter
sdfm.h	SDFM_initCompEventLowFilter
SDCOMP1EVT2FLTCLKCTL	
sdfm.c	SDFM_configCompEventLowFilter
SDCOMP1EVT1FLTCTL	
sdfm.c	SDFM_configCompEventHighFilter
sdfm.h	SDFM_initCompEventHighFilter
SDCOMP1EVT1FLTCLKCTL	
sdfm.c	SDFM_configCompEventHighFilter
SDCOMP1LOCK	
sdfm.h	SDFM_lockCompEventFilterConfig
SDCOMP2CTL	
-	
SDCOMP2EVT2FLTCTL	
-	
SDCOMP2EVT2FLTCLKCTL	
-	
SDCOMP2EVT1FLTCTL	
-	
SDCOMP2EVT1FLTCLKCTL	
-	
SDCOMP2LOCK	
-	
SDCOMP3CTL	

Table 19-97. SDFM Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
SDCOMP3EVT2FLTCTL	
-	
SDCOMP3EVT2FLTCLKCTL	
-	
SDCOMP3EVT1FLTCTL	
-	
SDCOMP3EVT1FLTCLKCTL	
-	
SDCOMP3LOCK	
-	
SDCOMP4CTL	
-	
SDCOMP4EVT2FLTCTL	
-	
SDCOMP4EVT2FLTCLKCTL	
-	
SDCOMP4EVT1FLTCTL	
-	
SDCOMP4EVT1FLTCLKCTL	
-	
SDCOMP4LOCK	
-	

Chapter 20

Enhanced Pulse Width Modulator (ePWM)

The enhanced pulse width modulator (ePWM) peripheral is a key element in controlling many of the power electronic systems found in both commercial and industrial equipment. These systems include digital motor control, switch mode power supply control, uninterruptible power supplies (UPS), and other forms of power conversion. The ePWM peripheral can also perform a digital to analog (DAC) function, where the duty cycle is equivalent to a DAC analog value; it is sometimes referred to as a power DAC.

This chapter is applicable for ePWM type 4 with added register protection capability. See the *C2000 Real-Time Control Peripheral Reference Guide* for a list of all devices with an ePWM module of the same type, to determine the differences between the types, and for a list of device-specific differences within a type.

Further information can be found in the *Flexible PWMs Enable Multi-Axis Drives, Multi-Level Inverters Technical Brief*.

20.1 Introduction

This chapter includes an overview and information about each submodule:

- Time-Base Submodule
- Counter Compare Submodule
- Action Qualifier Submodule
- Dead-Band Generator Submodule
- PWM Chopper (PC) Submodule
- Trip Zone Submodule
- Event Trigger Submodule
- Digital Compare Submodule

The ePWM Type 4 is functionally compatible to Type 2 (a Type 3 does not exist). Type 4 has the following enhancements in addition to the Type 2 features:

- **Register Address Map**

  Additional registers are required for new features on ePWM Type 4. The ePWM register address space has been remapped for better alignment and easy usage.
- **Delayed Trip Functionality**

  Changes have been added to achieve deadband insertion capabilities to support, for example, delayed trip functionality needed for peak current mode control type application scenarios. This has been accomplished by allowing comparator events to go into the Action Qualifier as a trigger event (Events T1 and T2). If comparator T1 / T2 events are used to edit the PWM, changes to the PWM waveform will not take place immediately. Instead, they will synchronize to the next TBCLK.
- **Dead-Band Generator Submodule Enhancements**

  Shadowing of the DBCTL register to allow dynamic configuration changes.
- **One Shot and Global Load of Registers**

  The ePWM Type 4 allows one shot and global load capability from shadow to active registers to avoid partial loads in, for example, multiphase applications. It also allows a programmable prescale of shadow to active load events. ePWM Type 4 Global Load can simplify ePWM software by removing interrupts and ensuring that all registers are loaded at the same time.
- **Trip Zone Submodule Enhancements**

  Independent flags have been added to reflect the trip status for each of the TZ sources. Changes have been made to the trip zone submodule to support certain power converter switching techniques like valley switching.
- **Digital Compare Submodule Enhancements**

  Blanking window filter register width has been increased from 8 to 16 bits. DCCAP functionality has been enhanced to provide more programmability.
- **PWM SYNC Related Enhancements**

  The ePWM Type 4 allows PWM SYNCOUT generation based on CMPC and CMPD events. These events can also be used for PWMSYNC pulse selection.

The ePWM Type 2 is fully compatible to Type 1. Type 2 has the following enhancements in addition to the Type 1 features:

- **High Resolution Dead-Band Capability**

  High resolution capability is added to dead-band RED and FED in half-cycle clocking mode.
- **Dead-Band Generator Submodule Enhancements**

  The ePWM Type 2 has features to enable both RED and FED on either PWM outputs. Provides increased dead band with 14-bit counters and dead-band / dead-band high-resolution registers are shadowed

- **High Resolution Extension available on ePWMxB outputs**

  Provides the ability to enable high-resolution period and duty cycle control on ePWMxB outputs. This is discussed in more detail in Section 20.15.
- **Counter Compare Submodule Enhancements**

  The ePWM Type 2 allows Interrupts and SOC events to be generated by additional counter compares CMPC and CMPD.
- **Event Trigger Submodule Enhancements**

  Prescaling logic to issue interrupt requests and ADC start of conversion expanded up to every 15 events. It allows software initialization of event counters on SYNC event.
- **Digital Compare Submodule Enhancements**

  Digital Compare Trip Select logic [DCTRIPSEL] has up to 12 external trip sources selected by the Input X-BAR logic in addition to an ability to OR all of them (up to 14 [external and internal sources]) to create the respective DCxEVTs.
- **Simultaneous Writes to TBPRD and CMPx Registers**

  This feature allows writes to TBPRD, CMPA:CMPAHR, CMPB:CMPBHR, CMPC and CMPD of any ePWM module to be tied to any other ePWM module, and also allows all ePWM modules to be tied to a particular ePWM module if desired.
- **Shadow to Active Load on SYNC of TBPRD and CMP Registers**

  This feature supports simultaneous writes of TBPRD and CMPA/B/C/D registers.

An effective PWM peripheral must be able to generate complex pulse width waveforms with minimal CPU overhead or intervention. It must be highly programmable and very flexible while being easy to understand and use. The ePWM unit described here addresses these requirements by allocating all needed timing and control resources on a per PWM channel basis. Cross coupling or sharing of resources has been avoided; instead, the ePWM is built up from smaller single channel submodules with separate resources that can operate together as required to form a system. This modular approach results in an orthogonal architecture and provides a more transparent view of the peripheral structure, helping users to understand its operation quickly.

In this document, the letter x within a signal or submodule name is used to indicate a generic ePWM instance on a device. For example, output signals EPWMxA and EPWMxB refer to the output signals from the ePWMx instance. Thus, EPWM1A and EPWM1B belong to ePWM1 and likewise EPWM4A and EPWM4B belong to ePWM4.

Type0 to Type1 Enhancements

- **Increased Dead-Band Resolution**

  Dead-band clocking has been enhanced to allow half-cycle clocking to double resolution.
- **Enhanced Interrupt and SOC Generation**

  Interrupts and ADC start-of-conversion can now be generated on both the TBCTR == zero and TBCTR == period events. This feature enables dual edge PWM control. Additionally, the ADC start-of-conversion can be generated from an event defined in the digital compare submodule.
- **High Resolution Period Capability**

  Provides the ability to enable high-resolution period. This is discussed in more detail in the device-specific *High-Resolution Pulse Width Modulator (HRPWM)* document..
- **Digital Compare Submodule**

  The digital compare submodule enhances the event triggering and trip zone submodules by providing filtering, blanking and improved trip functionality to digital compare signals. Such features are essential for peak current mode control and for support of analog comparators.

Note

The name of the sync signal that goes to the CMPSS has been updated from PWMSYNC to EPWMSYNCPER (SYNCPER/PWMSYNCPER/EPWMxSYNCPER) to avoid confusion with the other EPWM sync signals EPWMSYNCI and EPWMSYNCO. For a description of these signals, see Table 20-2.

20.1.1 EPWM Related Collateral

Foundational Materials

- C2000 Academy - Control Peripherals
- *Real-Time Control Reference Guide*
  - Refer to the EPWM section

Getting Started Materials

- *Flexible PWMs Enable Multi-Axis Drives, Multi-Level Inverters Application Report*
- Getting Started with the C2000 ePWM Module (Video)
- *Using PWM Output as a Digital-to-Analog Converter on a TMS320F280x Digital Signal Control Application Report*
  - Chapters 1 to 6 are Fundamental material, derivations, and explanations that are useful for learning about how PWM can be used to implement a DAC. Subsequent chapters are Getting Started and Expert material for implementing in a system.
- *Using the Enhanced Pulse Width Modulator (ePWM) Module Application Report*

Expert Materials

- C2000 real-time microcontrollers - Reference designs
  - See TI designs related to specific end applications used.
- *CRM/ZVS PFC Implementation Based on C2000 Type-4 PWM Module Application Report*
- *Leverage New Type ePWM Features for Multiple Phase Control Application Report*

20.1.2 Submodule Overview

The ePWM module represents one complete PWM channel composed of two PWM outputs: EPWMxA and EPWMxB. Multiple ePWM modules are instanced within a device as shown in Figure 20-1. Each ePWM instance is identical with one exception. Some instances include a hardware extension that allows more precise control of the PWM outputs. This extension is the high-resolution pulse width modulator (HRPWM) and is described in Section 20.15. See the device-specific data manual to determine which ePWM instances include this feature. Each ePWM module is indicated by a numerical value starting with 1. For example, ePWM1 is the first instance and ePWM3 is the third instance in the system and ePWMx indicates any instance.

The ePWM modules are chained together by way of a clock synchronization scheme that allows them to operate as a single system when required. Additionally, this synchronization scheme can be extended to the capture peripheral submodules (eCAP). The number of submodules is device-dependent and based on target application needs. Submodules can also operate standalone.

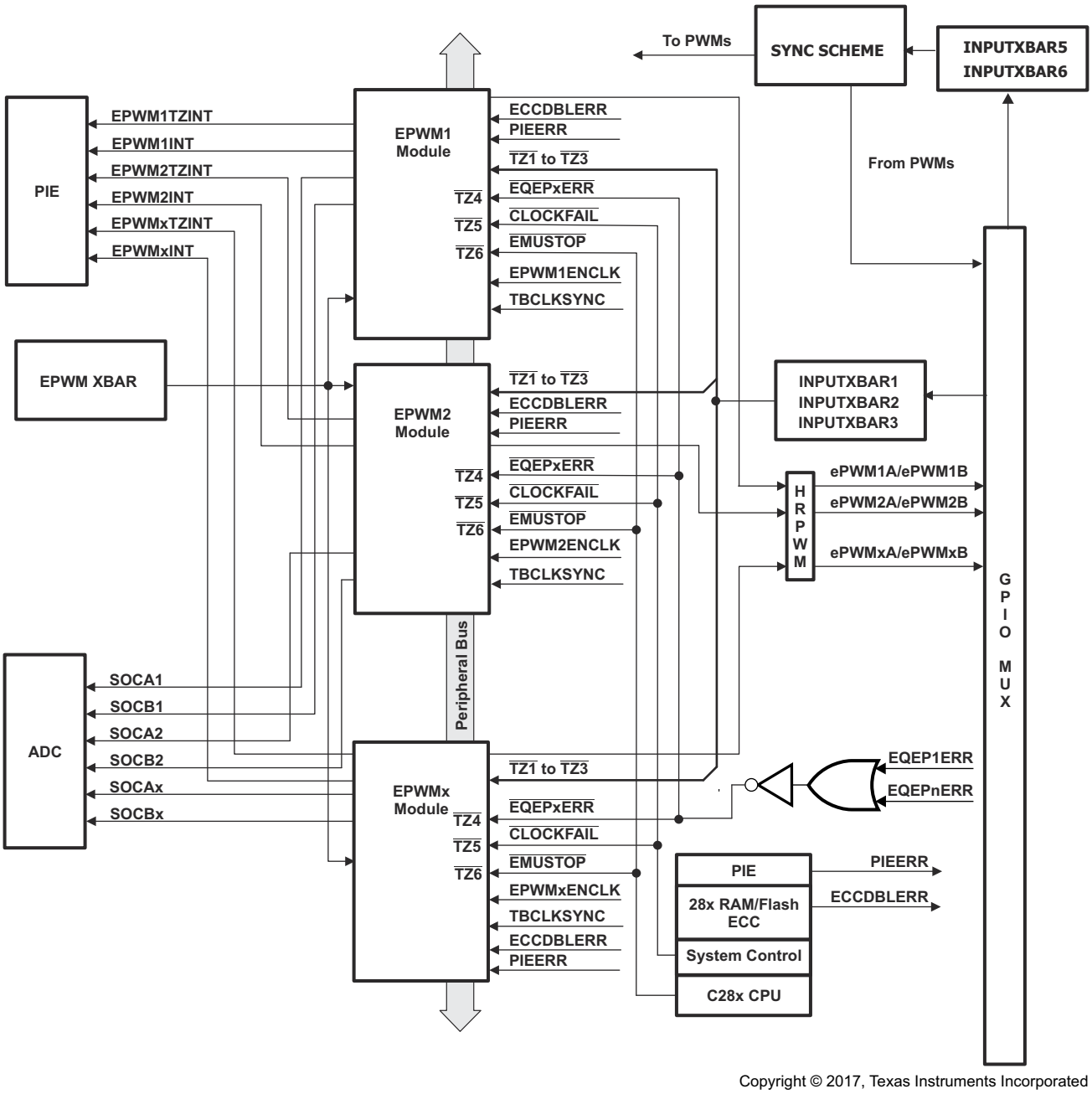
Each ePWM module supports the following features:

- Dedicated 16-bit time-base counter with period and frequency control
- Two PWM outputs (EPWMxA and EPWMxB) that can be used in the following configurations:
  - Two independent PWM outputs with single-edge operation
  - Two independent PWM outputs with dual-edge symmetric operation
  - One independent PWM output with dual-edge asymmetric operation
- Asynchronous override control of PWM signals through software.
- Programmable phase-control support for lag or lead operation relative to other ePWM modules.
- Hardware-locked (synchronized) phase relationship on a cycle-by-cycle basis.
- Dead-band generation with independent rising and falling edge delay control.

- Programmable trip zone allocation of both cycle-by-cycle trip and one-shot trip on fault conditions.
- A trip condition can force either high, low, or high-impedance state logic levels at PWM outputs.
- All events can trigger both CPU interrupts and ADC start of conversion (SOC)
- Programmable event prescaling minimizes CPU overhead on interrupts.
- PWM chopping by high-frequency carrier signal, useful for pulse transformer gate drives.

Each ePWM module is connected to the input/output signals shown in Figure 20-1. The signals are described in detail in subsequent sections.

The order in which the ePWM modules are connected may differ from what is shown in Figure 20-1. See Section 20.4.3.3 for the synchronization scheme for a particular device. Each ePWM module consists of eight submodules and is connected within a system by way of the signals shown in Figure 20-2.



A. This signal exists only on devices with an eQEP submodule.

Figure 20-1. Multiple ePWM Modules

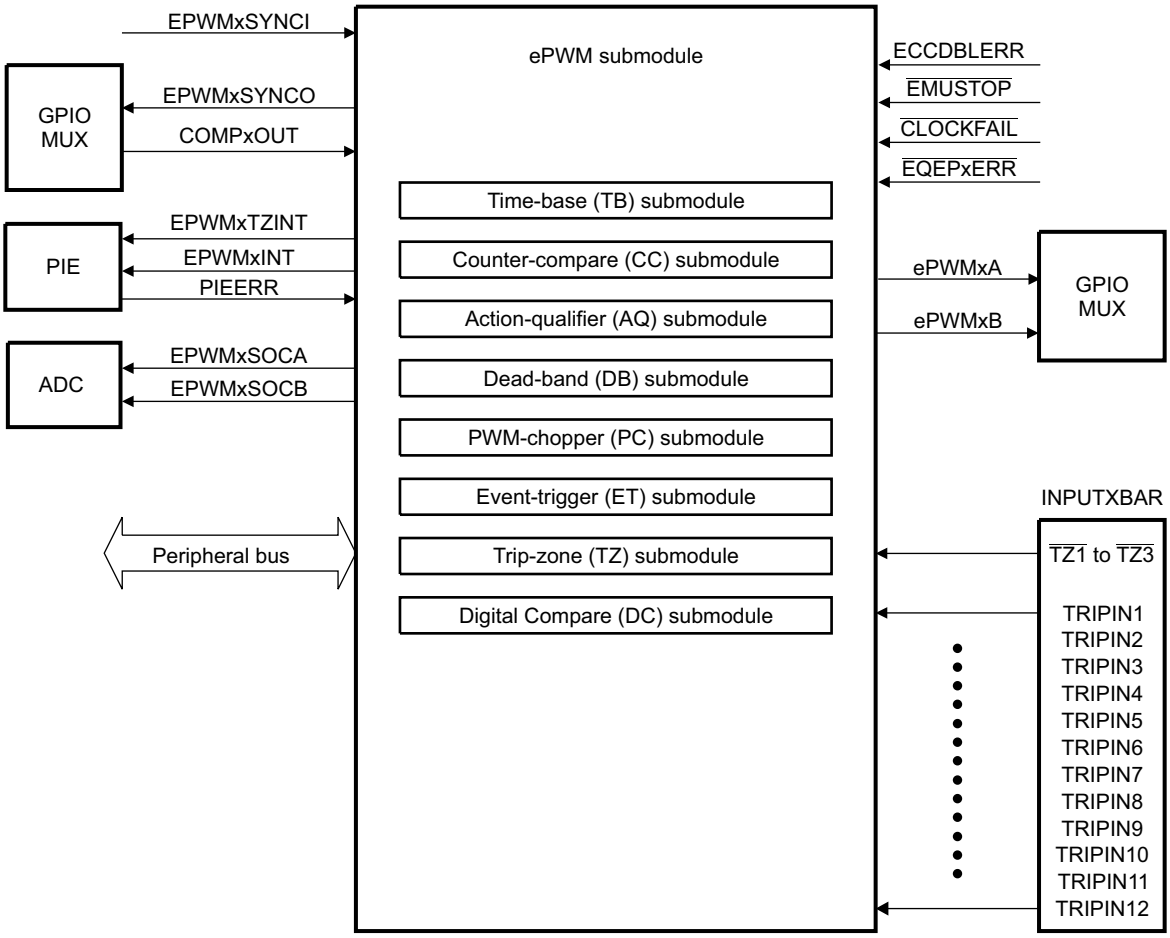


Figure 20-2. Submodules and Signal Connections for an ePWM Module

Figure 20-3 shows more internal details of a single ePWM module. The main signals used by the ePWM module are:

- **PWM output signals (EPWMxA and EPWMxB)**

  The PWM output signals are made available external to the device through the GPIO peripheral described in the *System Control and Interrupts* chapter.
- **Trip-zone signals ($\overline{TZ1}$ to $\overline{TZ6}$)**

  These input signals alert the ePWM module of fault conditions external to the ePWM module. Each submodule on a device can be configured to either use or ignore any of the trip-zone signals. The $\overline{TZ1}$ to $\overline{TZ3}$ trip-zone signals can be configured as asynchronous inputs through the GPIO peripheral using the Input X-BAR logic, refer to Figure 20-51. $\overline{TZ4}$ is connected to an inverted EQEPx error signal (EQEPxERR), which can be generated from any one of the EQEP submodule (for those devices with an EQEP module). $\overline{TZ5}$ is connected to the system clock fail logic, and $\overline{TZ6}$ is connected to the $\overline{EMUSTOP}$ output from the CPU. This allows you to configure a trip action when the clock fails or the CPU halts.
- **Time-base synchronization input (EPWMxSYNCI), output (EPWMxSYNCO), and peripheral (EPWMxSYNCPER) signals**

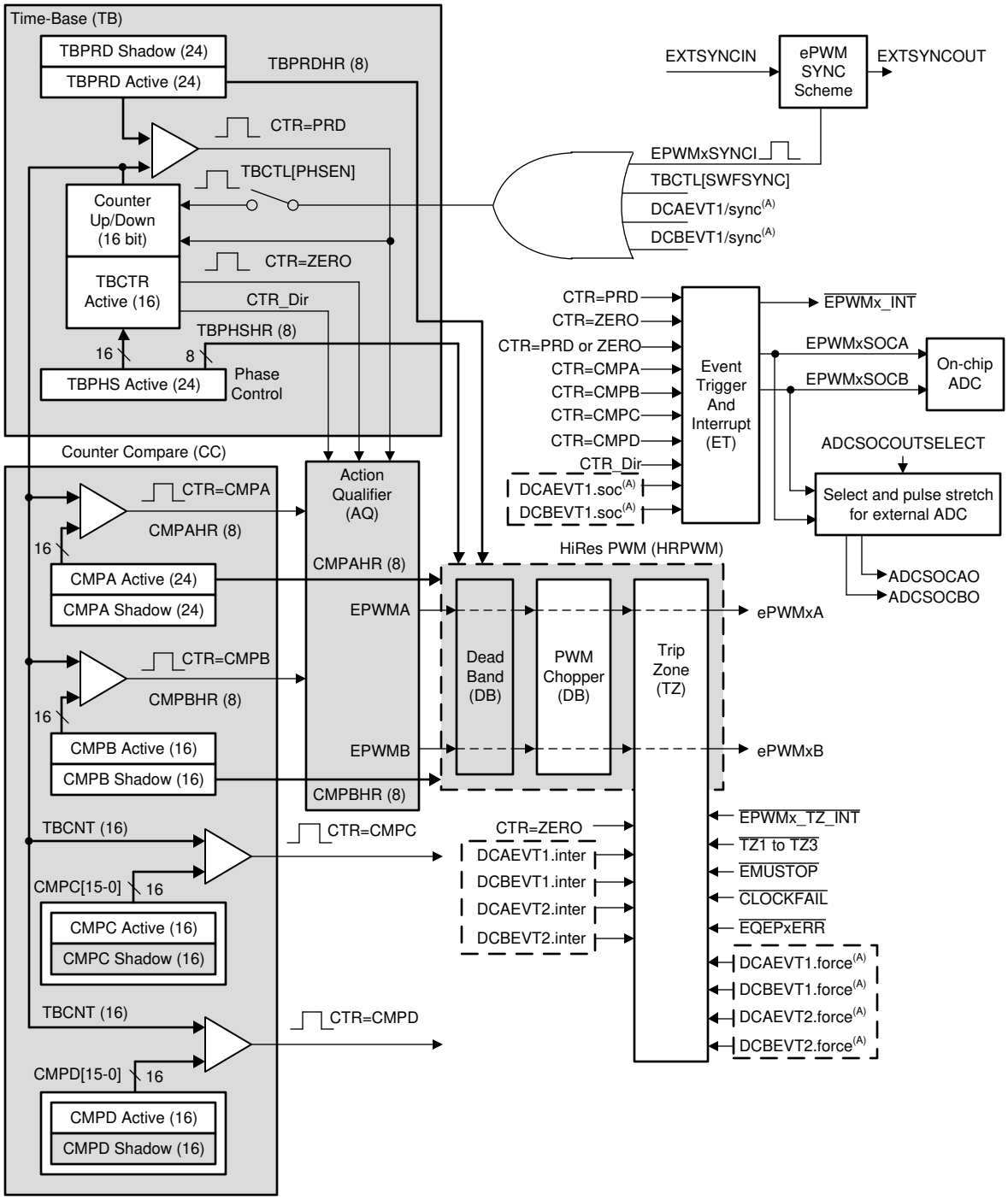
  Each ePWM module can be synchronized with other ePWM modules or other peripherals, using EPWMSYNCINSEL. Each ePWM module can also generate a synchronization output signal. The source of the EPWMxSYNCOUT can be selected and enabled by EPWMSYNCOUTEN and TBCTL2.OSHTSYNCMODE. For more information, see Section 20.4.3.3.

  Each ePWM module also generates another PWMSYNC signal called EPWMxSYNCPER. EPWMxSYNCPER goes to the GPDAC and CMPSS for synchronization purposes. It is configured using the HRPCTL register but has no relation with the HRPWM. For more information on how EPWMxSYNCPER is used by the GPDAC and CMPSS, see their respective chapters.
- **ADC start-of-conversion signals (EPWMxSOCA and EPWMxSOCB)**

  Each ePWM module has two ADC start of conversion signals. Any ePWM module can trigger a start of conversion. Whichever event triggers the start of conversion is configured in the event-trigger submodule of the ePWM.
- **Comparator output signals (COMPxOUT)**

  Output signals from the comparator module can be fed through the Input X-BAR to one or all of the 12 trip inputs [TRIPIN1 - TRIPIN12] and in conjunction with the trip zone signals can generate digital compare events.
- **Peripheral bus**

  The peripheral bus is 32-bits wide and allows both 16-bit and 32-bit writes to the ePWM register file.



A. These events are generated by the ePWM Digital Compare (DC) submodule based on the levels of the TRIPIN inputs.

Figure 20-3. ePWM Modules and Critical Internal Signal Interconnects

20.2 Configuring Device Pins

To connect the device input pins to the module, the Input X-BAR must be used. Some examples of when an external signal may be needed are TZx, TRIPx, and EXTSYNCIN. Any GPIO on the device can be configured as an input. The GPIO input qualification should be set to asynchronous mode by setting the appropriate GPxQSEL register bits to 11b. The internal pullups can be configured in the GPyPUD register. Since the GPIO mode is used, the GPyINV register can invert the signals. Additionally, some TRIPx (TRIP4-12 excluding TRIP6) signals must be routed through the ePWM X-Bar in addition to the Input X-Bar.

The GPIO mux registers must be configured for this peripheral. To avoid glitches on the pins, the GPyGMUX bits must be configured first (while keeping the corresponding GPyMUX bits at the default of zero), followed by writing the GPyMUX register to the desired value.

See the *GPIO* chapter for more details on GPIO mux, GPIO settings, and XBAR configuration.

20.3 ePWM Modules Overview

Eight submodules are included in every ePWM peripheral. Each of these submodules performs specific tasks that can be configured by software.

Table 20-1 lists the eight key submodules together with a list of their main configuration parameters. For example, if you need to adjust or control the duty cycle of a PWM waveform, then you should see the counter-compare submodule in Section 20.5 for relevant details.

Table 20-1. Submodule Configuration Parameters

Submodule	Configuration Parameter or Option
Time Base (TB)	• Scale the time-base clock (TBCLK) relative to the ePWM clock (EPWMCLK). • Configure the PWM time-base counter (TBCTR) frequency or period. • Set the mode for the time-base counter: – count-up mode: used for asymmetric PWM – count-down mode: used for asymmetric PWM – count-up-and-down mode: used for symmetric PWM • Configure the time-base phase relative to another ePWM module. • Synchronize the time-base counter between modules through hardware or software. • Configure the direction (up or down) of the time-base counter after a synchronization event. • Simultaneous writes to the TBPRD registers on all PWM's corresponding to the configuration on EPWMXLINK. • Configure how the time-base counter will behave when the device is halted by an emulator. • Specify the source for the synchronization output of the ePWM module • Configure one shot and global load of registers in this module.
Counter Compare (CC)	• Specify the PWM duty cycle for output EPWMxA and/or output EPWMxB • Specify the time at which switching events occur on the EPWMxA or EPWMxB output • Specify the programmable delay for interrupt and SOC generation with additional comparators • Simultaneous writes to the CMPA, CMPB, CMPC, CMPD registers on all PWM's corresponding to the configuration on EPWMXLINK. • Configure one shot and global load of registers in this module.
Action Qualifier (AQ)	• Specify the type of action taken when a time-base counter-compare, trip-zone submodule, or comparator event occurs: – No action taken – Output EPWMxA and/or EPWMxB switched high – Output EPWMxA and/or EPWMxB switched low – Output EPWMxA and/or EPWMxB toggled • Force the PWM output state through software control • Configure and control the PWM dead band through software • Configure one shot and global load of registers in this module.

Table 20-1. Submodule Configuration Parameters (continued)

Submodule	Configuration Parameter or Option
Dead Band (DB)	• Control of traditional complementary dead-band relationship between upper and lower switches • Specify the output rising-edge-delay value • Specify the output falling-edge delay value • Bypass the dead-band module entirely. In this case the PWM waveform is passed through without modification. • Option to enable half-cycle clocking for double resolution. • Allow ePWMxB phase shifting with respect to the ePWMxA output. • Configure one shot and global load of registers in this module.
PWM Chopper (PC)	• Create a chopping (carrier) frequency. • Pulse width of the first pulse in the chopped pulse train. • Duty cycle of the second and subsequent pulses. • Bypass the PWM chopper module entirely. In this case the PWM waveform is passed through without modification.
Trip Zone (TZ)	• Configure the ePWM module to react to one, all, or none of the trip-zone signals or digital compare events. • Specify the trip action taken when a fault occurs: – Force EPWMxA and/or EPWMxB high – Force EPWMxA and/or EPWMxB low – Force EPWMxA and/or EPWMxB to a high-impedance state – Configure EPWMxA and/or EPWMxB to ignore any trip condition. • Configure how often the ePWM will react to each trip-zone signal: – One-shot – Cycle-by-cycle • Enable the trip-zone to initiate an interrupt. • Bypass the trip-zone module entirely. • Programmable option for cycle-by-cycle trip clear • If desired, independently configure trip actions taken when time-base counter is counting down.
Event Trigger (ET)	• Enable the ePWM events that will trigger an interrupt. • Enable ePWM events that will trigger an ADC start-of-conversion event. • Specify the rate at which events cause triggers (every occurrence or every 2nd or up to 15th occurrence) • Poll, set, or clear event flags
Digital Compare (DC)	• Enables comparator (COMP) module outputs and trip zone signals which are configured using the Input X-BAR to create events and filtered events • Specify event-filtering options to capture TBCTR counter, generate blanking window, or insert delay in PWM output or time-base counter based on captured value.

20.4 Time-Base (TB) Submodule

Each ePWM module has its own time-base submodule that determines all of the event timing for the ePWM module. Built-in synchronization logic allows the time-base of multiple ePWM modules to work together as a single system. Figure 20-4 illustrates the time-base module's place within the ePWM.

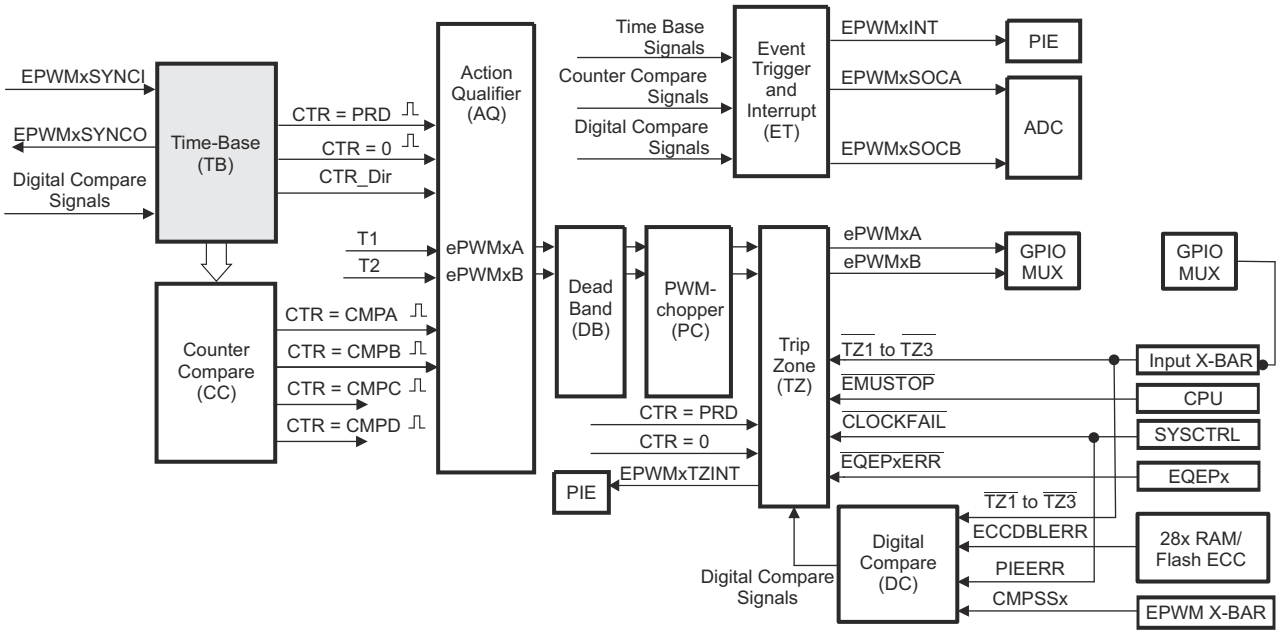


Figure 20-4. Time-Base Submodule

20.4.1 Purpose of the Time-Base Submodule

You can configure the time-base submodule for the following:

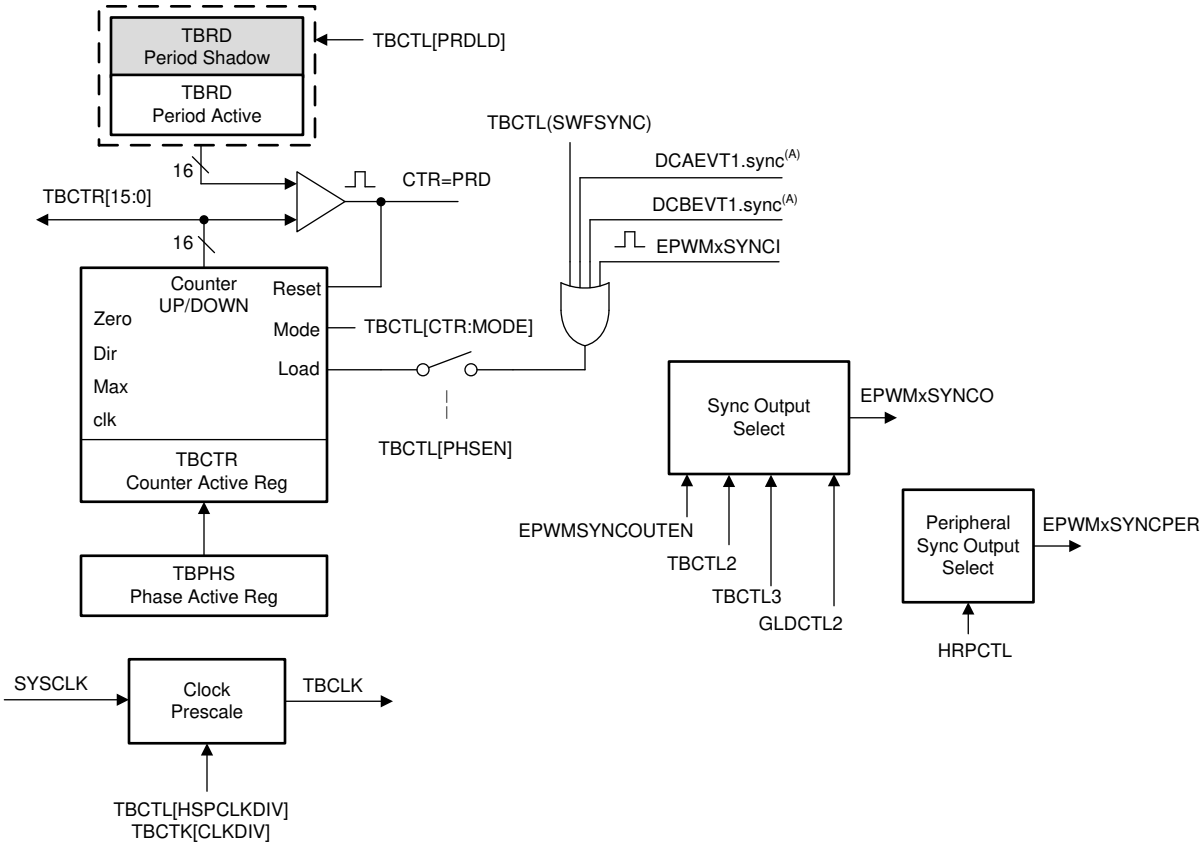
- Specify the ePWM time-base counter (TBCTR) frequency or period to control how often events occur.
- Manage time-base synchronization with other ePWM modules.
- Maintain a phase relationship with other ePWM modules.
- Set the time-base counter to count-up, count-down, or count-up-and-down mode.
- Generate the following events:
  - CTR = PRD: Time-base counter equal to the specified period (TBCTR = TBPRD).
  - CTR = Zero: Time-base counter equal to zero (TBCTR = 0x00).
- Configure the rate of the time-base clock; a prescaled version of the ePWM clock (EPWMCLK). This allows the time-base counter to increment/decrement at a slower rate.

Note

If it is required by the application code to updated the TBCTR value through software, while the TBCTR is counting, it is important to note that the Time-Base module will need at least 1 TBCLK cycle for the Time-Base related events to be realized. Hence the TBCTR should either be written with TBCTR = PRD-1 instead of TBCTR = PRD (in case the counter is counting up) and should be written as TBCTR = 1 instead of TBCTR = 0(in case the counter is counting down) for the events to be realized.

20.4.2 Controlling and Monitoring the Time-Base Submodule

The block diagram in Figure 20-5 shows the critical signals and registers of the time-base submodule. Table 20-2 provides descriptions of the key signals associated with the time-base submodule.



A. These signals are generated by the digital compare (DC) submodule.

Figure 20-5. Time-Base Submodule Signals and Registers

Table 20-2. Key Time-Base Signals

Signal	Description
EPWMxSYNCI	Time-base synchronization input. Input pulse used to synchronize the time-base counter with the counter of other ePWM modules. For more information on all of the signals available for synchronization, see EPWMSYNCINSEL. For information on the synchronization order of a particular device, see Section 20.4.3.3.
EPWMxSYNCO	Time-base synchronization output. This output pulse is used to synchronize the counter of other ePWM modules. Using EPWMSYNCOUTEN, TBCTL2, TBCTL3 and GLDCTL2, the source of the output pulse is selected.
EPWMxSYNCPER	Time-base peripheral synchronization output. This output signal is used to synchronize the CMPSS to the EPWM. It can be configured using the HRPCTL register. Note that this signal has no relation with the HRPWM.
CTR = PRD	Time-base counter equal to the specified period. This signal is generated whenever the counter value is equal to the active period register value. That is when TBCTR = TBPRD.
CTR = Zero	Time-base counter equal to zero This signal is generated whenever the counter value is zero. That is when TBCTR equals 0x00.
CTR = CMPB	Time-base counter equal to active counter-compare B register (TBCTR = CMPB). This event is generated by the counter-compare submodule and used by the synchronization out logic
CTR_dir	Time-base counter direction. Indicates the current direction of the ePWM's time-base counter. This signal is high when the counter is increasing and low when it is decreasing.
CTR_max	Time-base counter equal max value. (TBCTR = 0xFFFF) Generated event when the TBCTR value reaches its maximum value. This signal is only used only as a status bit
TBCLK	Time-base clock. This is a prescaled version of the ePWM clock (EPWMCLK) and is used by all submodules within the ePWM. This clock determines the rate at which time-base counter increments or decrements.

20.4.3 Calculating PWM Period and Frequency

The frequency of PWM events is controlled by the time-base period (TBPRD) register and the mode of the time-base counter. Figure 20-6 shows the period (T_{pwm}) and frequency (F_{pwm}) relationships for the up-count, down-count, and up-down-count time-base counter modes when the period is set to 4 (TBPRD = 4). The time increment for each step is defined by the time-base clock (TBCLK) which is a prescaled version of the ePWM clock (EPWMCLK).

The time-base counter has three modes of operation selected by the time-base control register (TBCTL):

- **Up-Down-Count Mode:**

  In up-down-count mode, the time-base counter starts from zero and increments until the period (TBPRD) value is reached. When the period value is reached, the time-base counter then decrements until it reaches zero. At this point the counter repeats the pattern and begins to increment.
- **Up-Count Mode:**

  In this mode, the time-base counter starts from zero and increments until it reaches the value in the period register (TBPRD). When the period value is reached, the time-base counter resets to zero and begins to increment once again.
- **Down-Count Mode:**

  In down-count mode, the time-base counter starts from the period (TBPRD) value and decrements until it reaches zero. When it reaches zero, the time-base counter is reset to the period value and it begins to decrement once again.

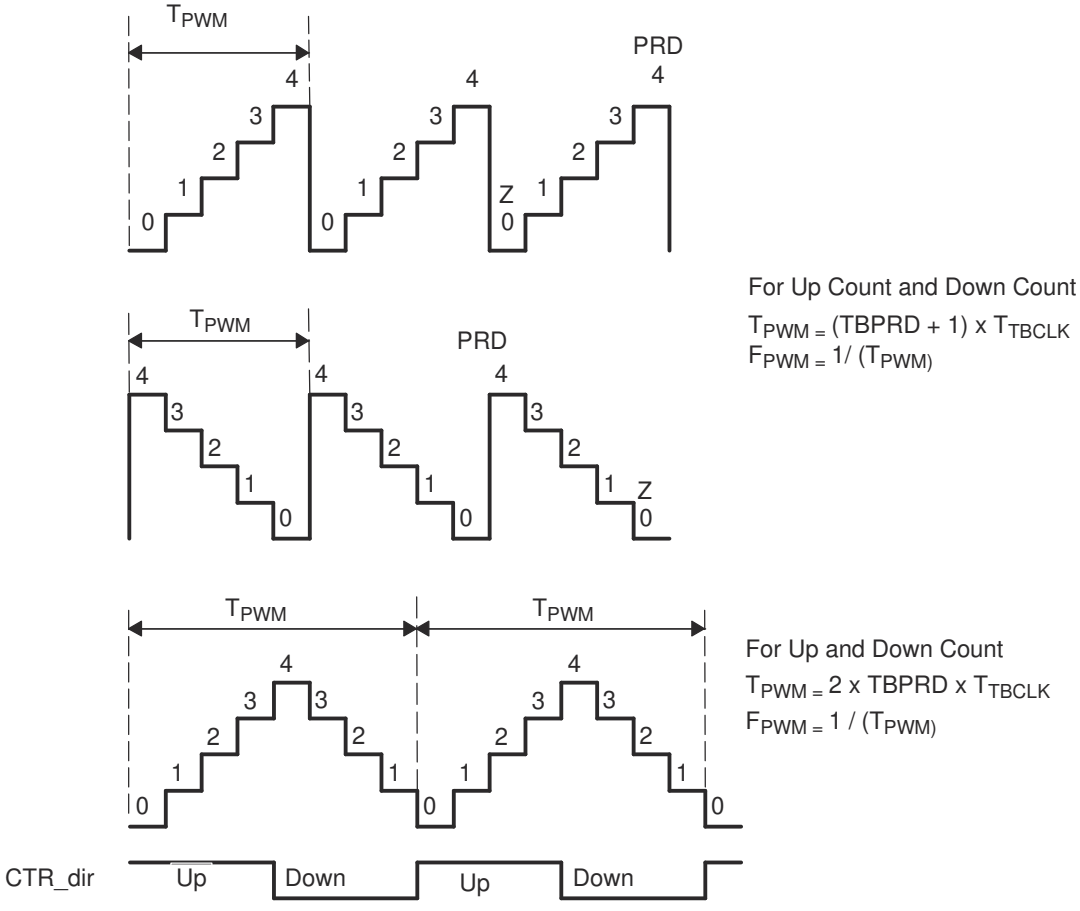


Figure 20-6. Time-Base Frequency and Period

20.4.3.1 Time-Base Period Shadow Register

The time-base period register (TBPRD) has a shadow register. Shadowing allows the register update to be synchronized with the hardware. The following definitions are used to describe all shadow registers in the ePWM module:

- **Active Register**

  The active register controls the hardware and is responsible for actions that the hardware causes or invokes.
- **Shadow Register**

  The shadow register buffers or provides a temporary holding location for the active register. It has no direct effect on any control hardware. At a strategic point in time the shadow register's content is transferred to the active register. This prevents corruption or spurious operation due to the register being asynchronously modified by software.

The memory address of the shadow period register is the same as the active register. Which register is written to or read from is determined by the TBCTL[PRDLD] bit. This bit enables and disables the TBPRD shadow register as follows:

- **Time-Base Period Shadow Mode:**

  The TBPRD shadow register is enabled when TBCTL[PRDLD] = 0. Reads from and writes to the TBPRD memory address go to the shadow register. The shadow register contents are transferred to the active register (TBPRD (Active) ← TBPRD (shadow)) when the time-base counter equals zero (TBCTR = 0x00) and/or a sync event as determined by the TBCTL2[PRDLDSYNC] bit. The PRDLDSYNC bit is valid only if TBCTL[PRDLD] = 0. By default the TBPRD shadow register is enabled. The sources for the SYNC input is explained in Section 20.4.3.3.

  The global load control mechanism can also be used with the time-base period register by configuring the appropriate bits in the global load configuration register (GLDCFG). When global load mode is selected the transfer of contents from shadow register to active register, for all registers that have this mode enabled, occurs at the same event as defined by the configuration bits in Global Shadow to Active Load Control Register (GLDCTL). Global load control mechanism is explained in Section 20.4.7.
- **Time-Base Period Immediate Load Mode:**

  If immediate load mode is selected (TBCTL[PRDLD] = 1), then a read from or a write to the TBPRD memory address goes directly to the active register.

20.4.3.2 Time-Base Clock Synchronization

The TBCLKSYNC bit in the peripheral clock enable registers allows all users to globally synchronize all enabled ePWM modules to the time-base clock (TBCLK). When set, all enabled ePWM module clocks are started with the first rising edge of TBCLK aligned. For perfectly synchronized TBCLKs, the prescalers for each ePWM module must be set identically.

The proper procedure for enabling ePWM clocks is as follows:

1. Enable ePWM module clocks in the PCLKCRx register
2. Set TBCLKSYNC= 0
3. Configure ePWM modules
4. Set TBCLKSYNC= 1

20.4.3.3 Time-Base Counter Synchronization

The ePWM type 4 introduces a new synchronization scheme that allows for increased flexibility of synchronization of the ePWM modules. Each ePWM module has a synchronization input (SYNCI), a synchronization output (SYNCO) and a peripheral synchronization output (SYNCPER). In Figure 20-7, EXTSYNC1 is sourced from INPUTXBAR5 and EXTSYNC2 is sourced from INPUTXBAR6, which can be configured to select any GPIO as the synchronization input. Refer to Table 20-3 for a list of all sync inputs including INPUTXBAR5 and INPUTXBAR6. Figure 20-8 shows the sources that can be used for EXTSYNCOUT.

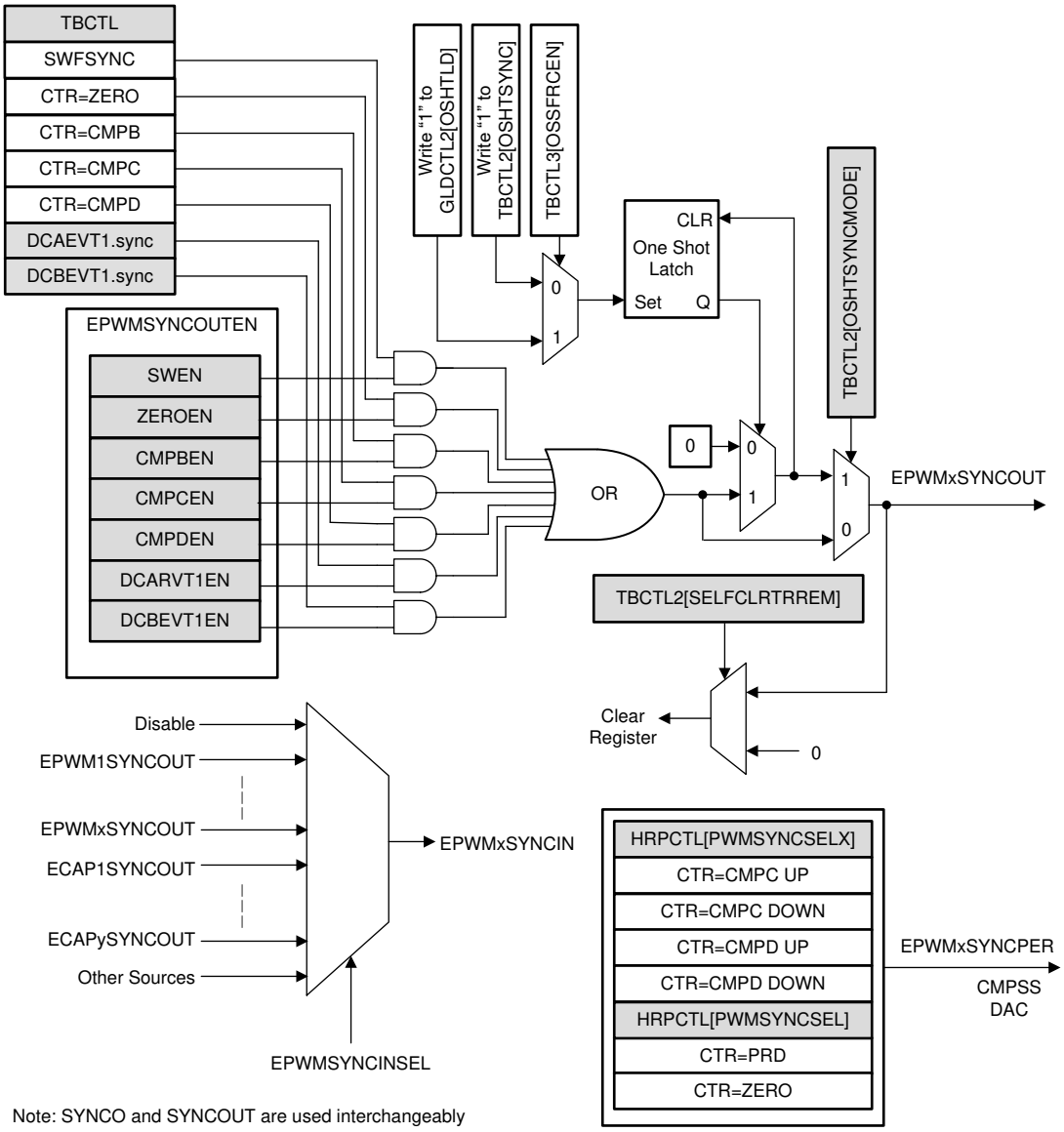


Figure 20-7. Time-Base Counter Synchronization Scheme

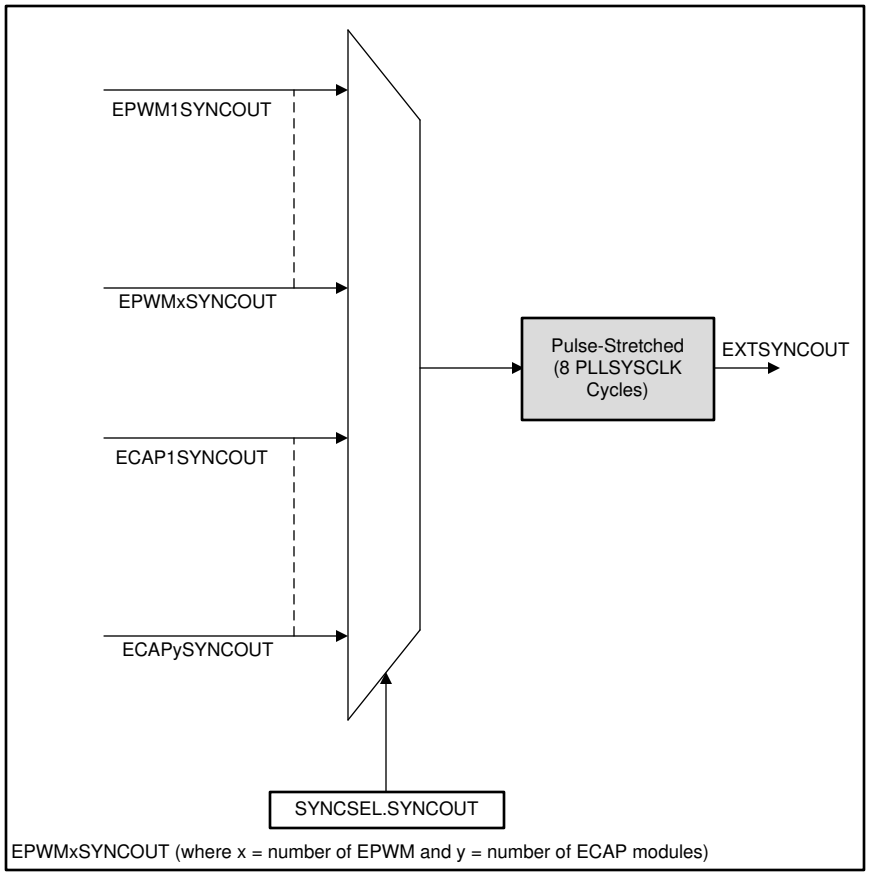


Figure 20-8. ePWM External SYNC Output

Note

See the data manual for the number of ePWM and eCAP modules available on your specific device.

Each ePWM module can be configured to use or ignore the synchronization input. If the TBCTL[PHSEN] bit is set, then the time-base counter (TBCTR) of the ePWM module will be automatically loaded with the phase register (TBPHS) contents when one of the following conditions occur:

- **EPWMxSYNCI: Synchronization Input Pulse:**

  The value of the phase register is loaded into the counter register when an input synchronization pulse is detected (TBPHS → TBCTR). This operation occurs on the next valid time-base clock (TBCLK) edge.
- **Software Forced Synchronization Pulse:**

  Writing a 1 to the TBCTL[SWFSYNC] control bit invokes a software forced synchronization. This pulse is ORed with the synchronization input signal, and therefore has the same effect as a pulse on EPWMxSYNCI.
- **Digital Compare Event Synchronization Pulse:**

  DCAEVT1 and DCBEVT1 digital compare events can be configured to generate synchronization pulses which have the same affect as EPWMxSYNCI.

Note

If the EPWMxSYNCI signal is held HIGH, the sync will NOT continuously occur. The EPWMxSYNCI is rising edge activated.

This feature enables the ePWM module to be automatically synchronized to the time base of another ePWM module. Lead or lag phase control can be added to the waveforms generated by different ePWM modules to synchronize them. In up-down-count mode, the TBCTL[PSHDIR] bit configures the direction of the time-base counter immediately after a synchronization event. The new direction is independent of the direction prior to the synchronization event. The PHSDIR bit is ignored in count-up or count-down modes. See Figure 20-9 through Figure 20-12 for examples.

Clearing the TBCTL[PHSEN] bit configures the ePWM to ignore the synchronization input pulse.

20.4.3.4 ePWM SYNC Selection

Table 20-3 specifies the sources for the ePWM SYNC input and output.

Table 20-3. ePWM SYNC Selection

EPWMSYNCINSEL.SEL, ECAPSYNCINSEL.SEL	SYNC Source
0x0	Reserved
0x1	EPWM1.SYNCOUT
0x2	EPWM2.SYNCOUT
0x3	EPWM3.SYNCOUT
0x4	EPWM4.SYNCOUT
0x5	EPWM5.SYNCOUT
0x6	EPWM6.SYNCOUT
0x7	EPWM7.SYNCOUT
0x8	Reserved
0x9	Reserved
0xA	Reserved
0xB	Reserved
0xC	Reserved
0xD	Reserved
0xE	Reserved
0xF	Reserved
0x10	Reserved
0x11	ECAP1.SYNCOUT
0x12	ECAP2.SYNCOUT
0x13	
0x14	Reserved
0x15	Reserved
0x16	Reserved
0x17	Reserved
0x18	INPUT-XBAR.INPUT5
0x19	INPUT-XBAR.INPUT6
0x1A	Reserved
0x1B	Reserved
0x1C	Reserved
0x1D	Reserved
0x1E	Reserved
0x1F	Reserved

20.4.4 Phase Locking the Time-Base Clocks of Multiple ePWM Modules

20.4.5 Simultaneous Writes to TBPRD and CMPx Registers Between ePWM Modules

For variable frequency applications, there is a need for simultaneous writes of TBPRD and CMPx registers between ePWM modules. This prevents situations where a CTR = 0 or CTR = PRD pulse forces a shadow to active load of these registers before all registers are updated between ePWM modules (resulting in some registers being loaded from new shadow values while others are loaded from old shadow values). To support this, an ePWM register linking scheme for TBPRD:TBPRDHR, CMPA:CMPAHR, CMPB:CMPBHR, CMPC, and CMPD registers between PWM modules has been added.

For a particular ePWM module # A , user code writes “B+1”, to the linked register bit-field in EPWMXLINK. “B” is the ePWM module # being linked to (that is, writes to the ePWM module “B” TBPRD:TBPRDHR, CMPA:CMPAHR, CMPB:CMPBHR, or CMPC will simultaneously be written to corresponding register in ePWM module “A”). For instance if ePWM3 EPWMXLINK register is configured so that CMPA:CMPAHR are linked to ePWM1, then a write to CMPA:CMPAHR in ePWM 1 will simultaneously write the same value to CMPA:CMPAHR in ePWM3. If ePWM4 also has its CMPA:CMPAHR registers linked to ePWM1, then a write to ePWM 1 will write the same value to the CMPA:CMPAHR registers in both ePWM3 and ePWM4.

The register description for EPWMXLINK clearly explains the linked register bit-field values for corresponding ePWM. An example of using the EPWMXLINK is linking ePWM2 CMPA with CMPA of ePWM1. In this case, a write to CMPA of ePWM1 will also change the CMPA value for ePWM2.

20.4.6 Time-Base Counter Modes and Timing Waveforms

The time-base counter operates in one of four modes:

- Up-count mode that is asymmetrical
- Down-count mode that is asymmetrical
- Up-down-count that is symmetrical
- Frozen where the time-base counter is held constant at the current value

To illustrate the operation of the first three modes, the following timing diagrams show when events are generated and how the time-base responds to an EPWMxSYNCI signal.

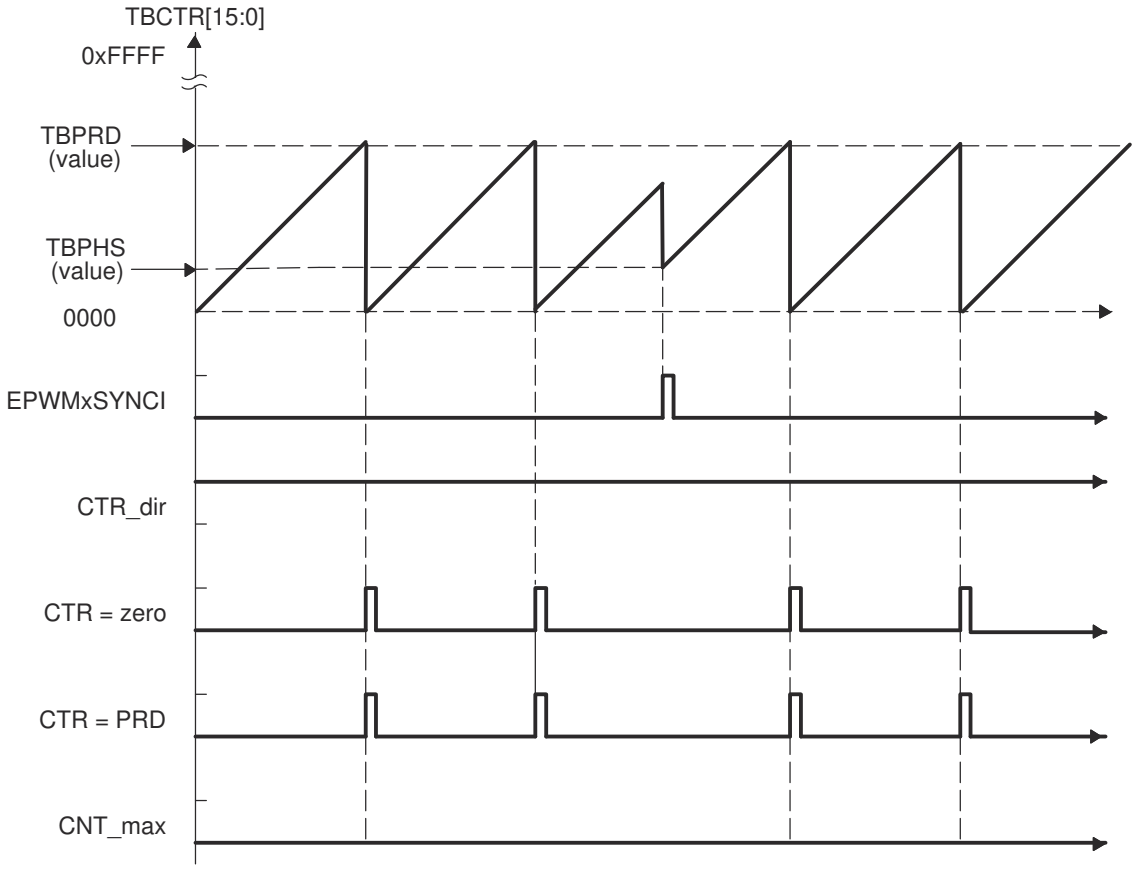


Figure 20-9. Time-Base Up-Count Mode Waveforms

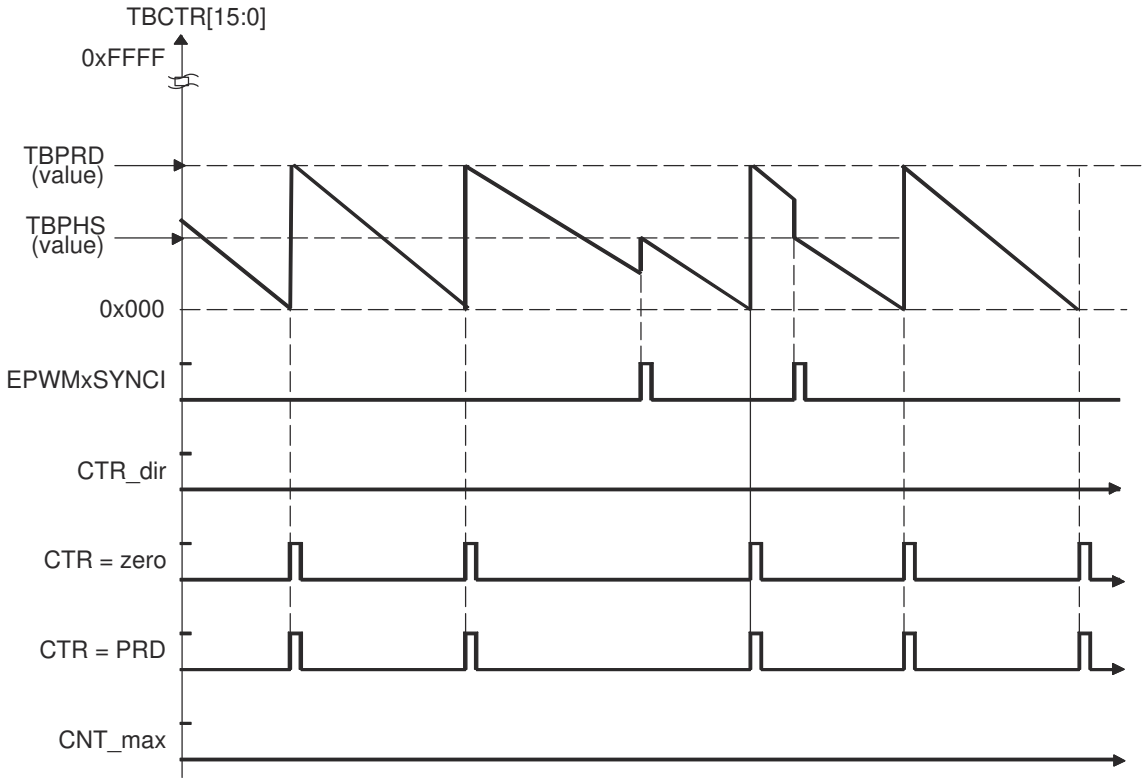


Figure 20-10. Time-Base Down-Count Mode Waveforms

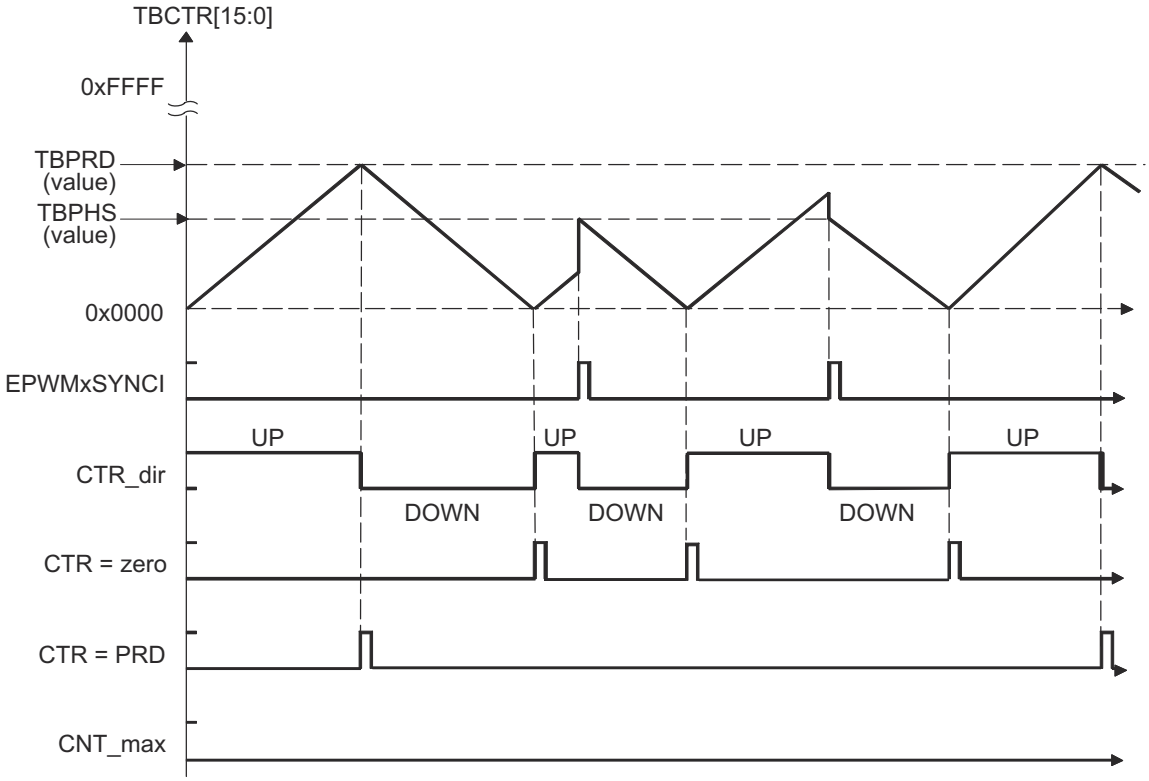


Figure 20-11. Time-Base Up-Down-Count Waveforms, TBCTL[PHSDIR = 0] Count Down On Synchronization Event

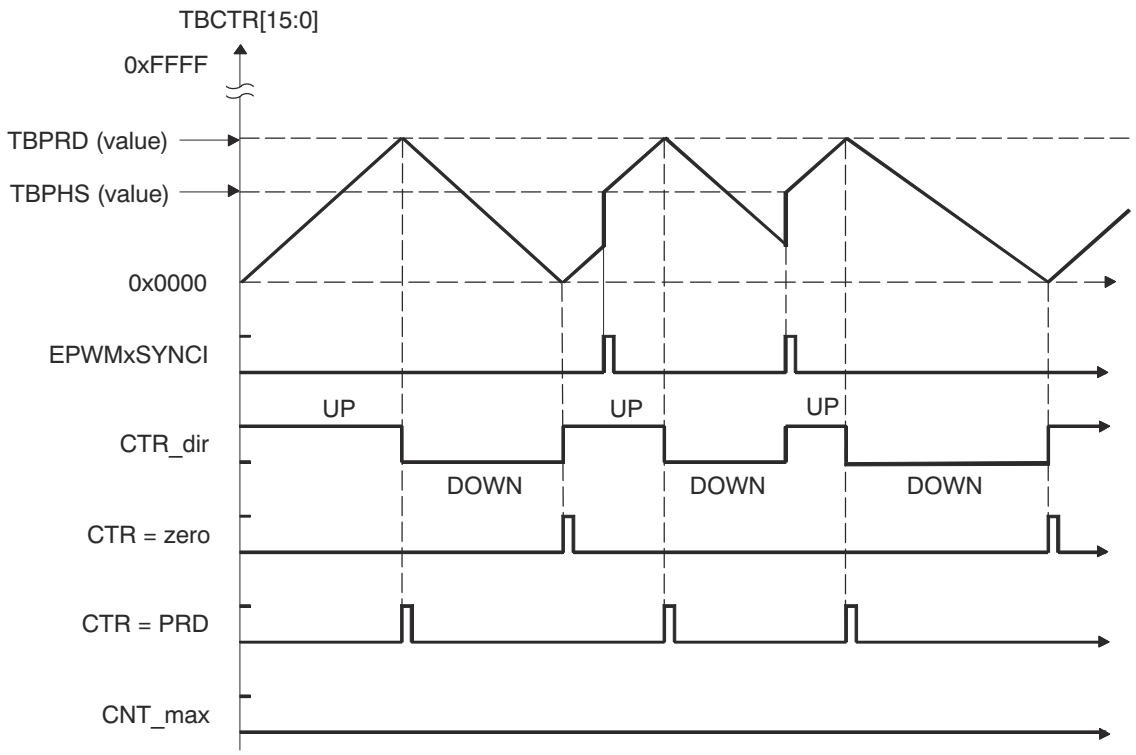


Figure 20-12. Time-Base Up-Down Count Waveforms, TBCTL[PHSDIR = 1] Count Up On Synchronization Event

20.4.7 Global Load

Figure 20-13 illustrates the signals and registers associated with the global load feature.

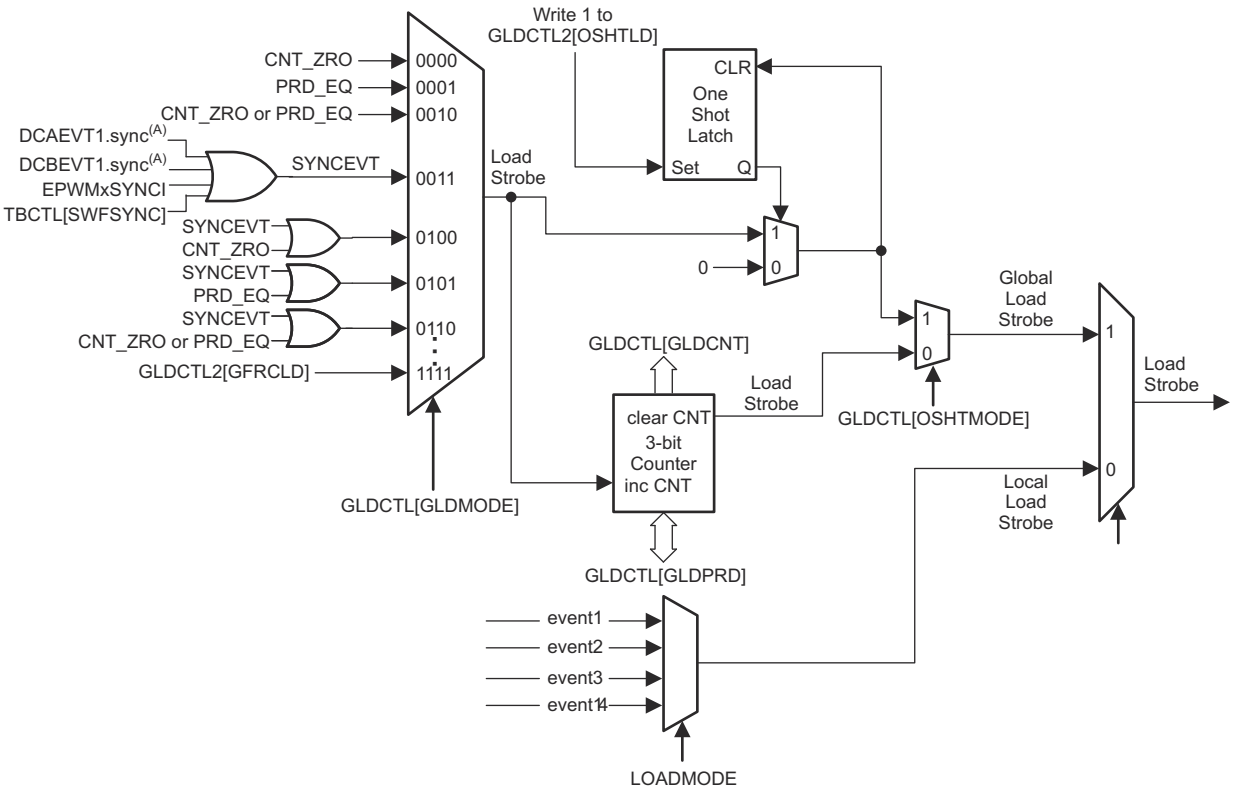


Figure 20-13. Global Load: Signals and Registers

Note

The SYNCEVT signal is only propagated through when PHSEN is SET.

When this feature is enabled, the transfer of contents from the shadow register to the active register, for all registers that have this mode enabled, occurs at the same event as defined by the configuration bits in Global Shadow to Active Load Control Register (GLDCTL[GLDMODE]). When GLDCTL[GLD] = ’1’, shadow to active load event selection bits for individual shadowed registers are ignored and global load mode takes effect for the corresponding registers enabled by GLDCFG[REGx].

When GLDCTL[GLD] = ’1’ and GLDCFG[REGx] = ‘0’ global load mode does not affect the corresponding register (REGx). Shadow to active load event selection bits for individual shadowed registers decide how the transfer of contents from shadow register to active register takes place.

20.4.7.1 Global Load Pulse Pre-Scalar

This feature provides the capability to choose shadow to active transfers to happen once in ‘N’ occurrences of selected global load pulse (GLDCTL[GLDMODE]). This pre-scale functionality is not available for registers that cannot or are not configured to use the global load mechanism (that is, GLDCTL[GLD] = ’0’ or GLDCFG[REGx] = ‘0’).

20.4.7.2 One-Shot Load Mode

This feature allows users to cause the shadow register to active register transfers to occur once. When GLDCTL2[OSHTLD] = '1' the shadow to active register transfer, for registers that are configured to use the global load mechanism, takes place on the event selected by GLDCTL[GLDMODE].

Software force loading of contents from shadow register to active register is possible by using GLDCTL2[GFRCLD]. The GLDCTL2 register can also be linked across multiple PWM modules by using EPWMXLINK[GLDCTL2LINK]. This, along with the one-shot load mode feature discussed above, provides a method to correctly update multiple PWM registers in one or more PWM modules at certain PWM events or, if desired, in the same clock cycle. This is very useful in variable frequency applications and/or multi-phase interleaved applications.

Note

One-shot load mode should **NOT** be used when high-resolution mode is enabled.

20.4.7.3 One-Shot Sync Mode

You can enable the one-shot sync mode to generate a SYNCOUT pulse by configuring **TBCLT2[OSHTSYNCMODE]** and setting the **TBCLT2[OSHTSYNC]** bit as shown in Figure 20-14.

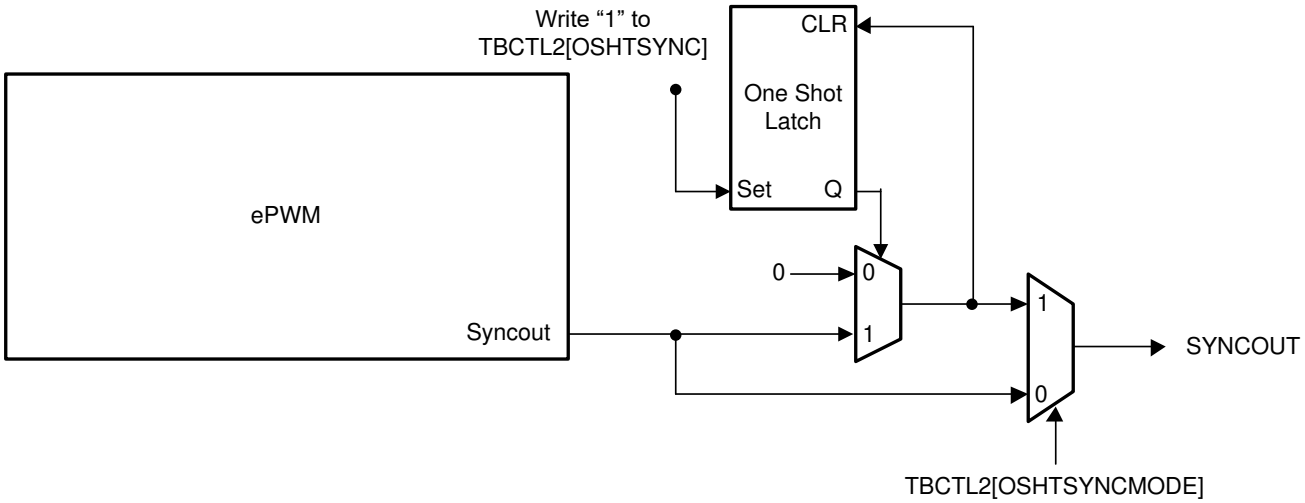


Figure 20-14. One-Shot Sync Mode

20.5 Counter-Compare (CC) Submodule

Figure 20-15 illustrates the counter-compare submodule within the ePWM.

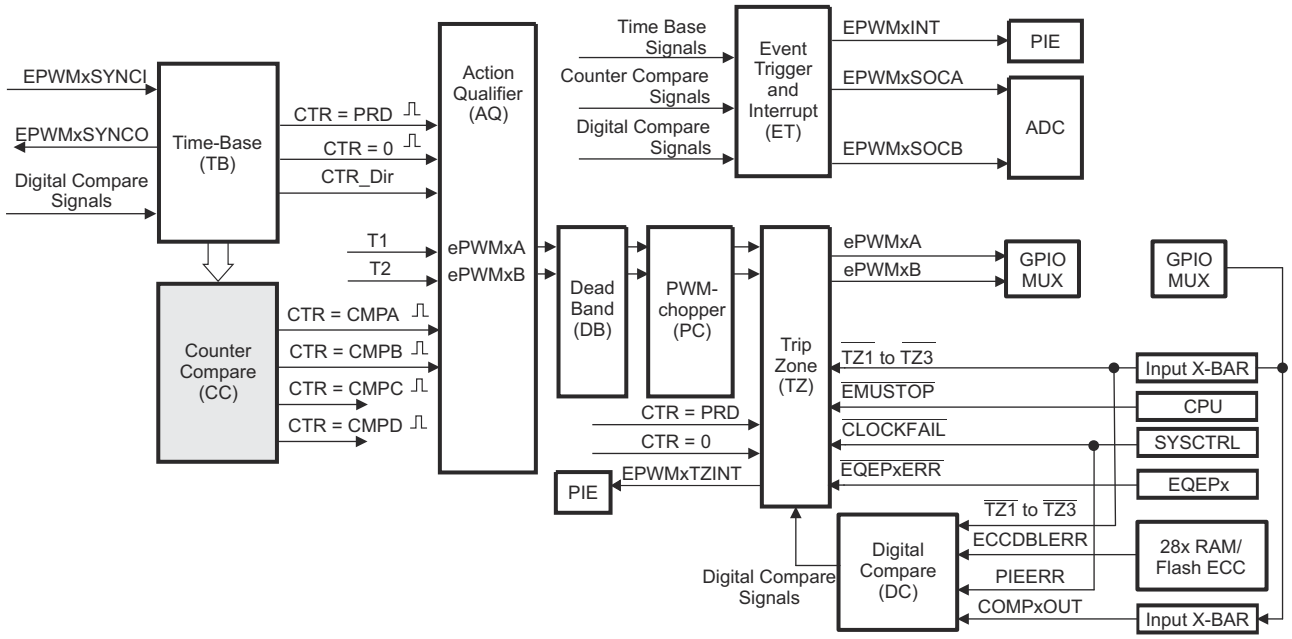


Figure 20-15. Counter-Compare Submodule

20.5.1 Purpose of the Counter-Compare Submodule

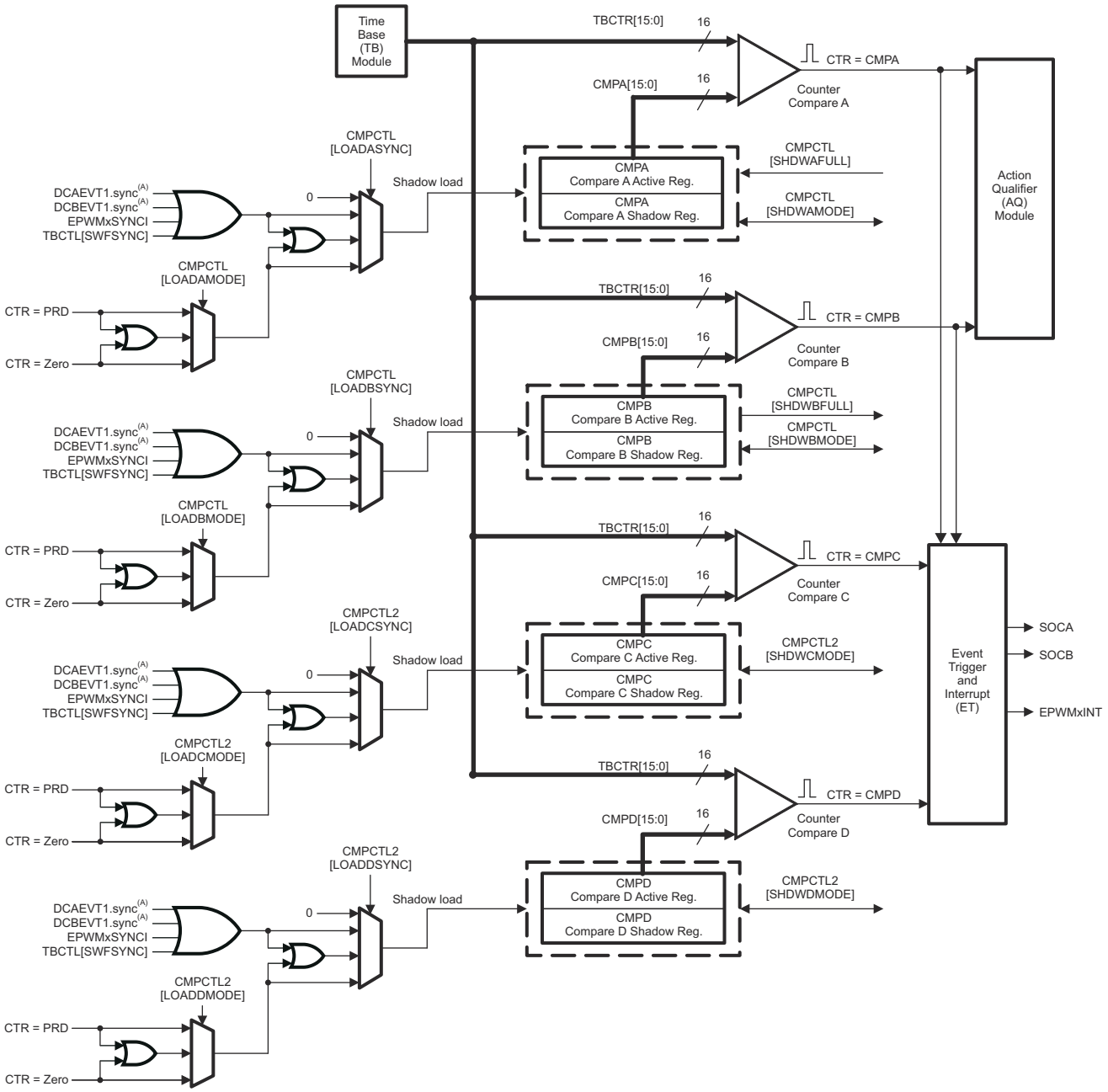
The counter-compare submodule takes as input the time-base counter value. This value is continuously compared to the counter-compare A (CMPA), counter-compare B (CMPB), counter-compare C (CMPC), and counter-compare D (CMPD) registers. When the time-base counter is equal to one of the compare registers, the counter-compare unit generates an appropriate event.

The counter-compare:

- Generates events based on programmable time stamps using the CMPA, CMPB, CMPC, and CMPD registers:
  - CTR = CMPA: Time-base counter equals counter-compare A register (TBCTR = CMPA)
  - CTR = CMPB: Time-base counter equals counter-compare B register (TBCTR = CMPB)
  - CTR = CMPC: Time-base counter equals counter-compare C register (TBCTR = CMPC)
  - CTR = CMPD: Time-base counter equals counter-compare D register (TBCTR = CMPD)
- Controls the PWM duty cycle, if the action-qualifier submodule is configured appropriately using counter-compare A (CMPA) and counter-compare B (CMPB)
- Shadows new compare values to prevent corruption or glitches during the active PWM cycle

20.5.2 Controlling and Monitoring the Counter-Compare Submodule

The counter-compare submodule operation is shown in Figure 20-16.



A. These events are generated by the ePWM digital compare (DC) submodule based on the levels of the TRIPIN inputs (for example, CMPSSx and $\overline{TZ}$ signals).

Figure 20-16. Detailed View of the Counter-Compare Submodule

20.5.3 Operational Highlights for the Counter-Compare Submodule

The counter-compare submodule is responsible for generating events which can be used in the action-qualifier and/or event-trigger submodules. There are four independent compare events:

1. CTR = CMPA: Time-base counter equal to counter-compare A register (TBCTR = CMPA).
2. CTR = CMPB: Time-base counter equal to counter-compare B register (TBCTR = CMPB).
3. CTR = CMPC: Time-base counter equal to counter-compare C register (TBCTR = CMPC). This event can be used to generate an event in the event trigger submodule only.
4. CTR = CMPD: Time-base counter equal to counter-compare D register (TBCTR = CMPD). This event can be used to generate an event in the event trigger submodule only

For up-count or down-count mode, each event occurs only once per cycle. For up-down count mode each event occurs twice per cycle if the compare value is between 0x00-TBPRD and once per cycle if the compare value is equal to 0x00 or equal to TBPRD. These events are fed into the action-qualifier submodule where they are qualified by the counter direction and converted into actions if enabled. Refer to Section 20.6.1 for more details.

The counter-compare registers CMPA and CMPB each have an associated shadow register. Shadowing provides a way to keep updates to the registers synchronized with the hardware. When shadowing is used, updates to the active registers only occur at strategic points. This prevents corruption or spurious operation due to the register being asynchronously modified by software. The memory address of the active register and the shadow register is identical. The register that is written to or read from is determined by the CMPCTL[SHDWAMODE] and CMPCTL[SHDWBMODE] bits. These bits enable and disable the CMPC shadow register and CMPD shadow register, respectively. The behavior of the two load modes is:

Shadow Mode:

The shadow mode for the CMPA is enabled by clearing the CMPCTL[SHDWAMODE] bit and the shadow register for CMPB is enabled by clearing the CMPCTL[SHDWBMODE] bit. Shadow mode is enabled by default for both CMPA and CMPB.

If the shadow register is enabled then the content of the shadow register is transferred to the active register on one of the following events as specified by the CMPCTL[LOADAMODE], CMPCTL[LOADBMODE], CMPCTL[LOADASYNC], and CMPCTL[LOADBSYNC] register bits:

- CTR = PRD: Time-base counter equal to the period (TBCTR = TBPRD).
- CTR = Zero: Time-base counter equal to zero (TBCTR = 0x00)
- Both CTR = PRD and CTR = Zero
- SYNC event caused by DCAEVT1 or DCBEVT1 or EPWMxSYNCI or TBCTL[SWFSYNC]
- Both SYNC event or a selection made by LOADAMODE/LOADBMODE

Only the active register contents are used by the counter-compare submodule to generate events to be sent to the action-qualifier.

Immediate Mode:

If immediate load mode is selected (that is, CMPCTL[SHDWAMODE] = 1 or CMPCTL[SHDWBMODE] = 1), then a read from or a write to the register will go directly to the active register.

Additional Comparators

The counter-compare submodule on ePWMs type 2 and later are responsible for generating two additional independent compare events based on two compare registers, which is fed to Event Trigger submodule:

1. CTR = CMPC: Time-base counter equal to counter-compare C register (TBCTR = CMPC).
2. CTR = CMPD: Time-base counter equal to counter-compare D register (TBCTR = CMPD).

The counter-compare registers CMPC and CMPD each have an associated shadow register. By default this register is shadowed. The memory address of the active register and the shadow register is identical. The value in the active CMPC and CMPD register is compared to the time-base counter (TBCTR). When the values are equal, the counter compare module generates a “time-base counter equal to counter compare C or counter compare D ” event respectively. Shadowing of this register is enabled and disabled by the CMPCTL2[SHDWCMODE] and CMPCTL2[SHDWDMODE] bit. These bits enable and disable the CMPC shadow register and CMPD shadow register respectively. The behavior of the two load modes is described below:

Shadow Mode:

The shadow mode for the CMPC is enabled by clearing the CMPCTL2[SHDWCMODE] bit and the shadow register for CMPD is enabled by clearing the CMPCTL2[SHDWDMODE] bit. Shadow mode is enabled by default for both CMPC and CMPD.

If the shadow register is enabled then the content of the shadow register is transferred to the active register on one of the following events as specified by the CMPCTL2[LOADCMODE], CMPCTL2[LOADDMODE], CMPCTL2[LOADCSYNC], and CMPCTL2[LOADDSYNC] register bits:

- CTR = PRD: Time-base counter equal to the period (TBCTR = TBPRD).
- CTR = Zero: Time-base counter equal to zero (TBCTR = 0x00)
- Both CTR = PRD and CTR = Zero
- SYNC event caused by DCAEVT1 or DCBEVT1 or EPWMxSYNCI or TBCTL[SWFSYNC]
- Both SYNC event or a selection made by LOADCMODE/LOADDMODE

Only the active register contents are used by the counter-compare submodule to generate events to be sent to the action-qualifier.

Immediate Load Mode:

If the immediate load mode is selected (that is, CMPCTL2[SHDWCMODE] = 1 or CMPCTL2[SHDWDMODE] = 1), then a read from or a write to the register will go directly to the active register.

Global Load Support

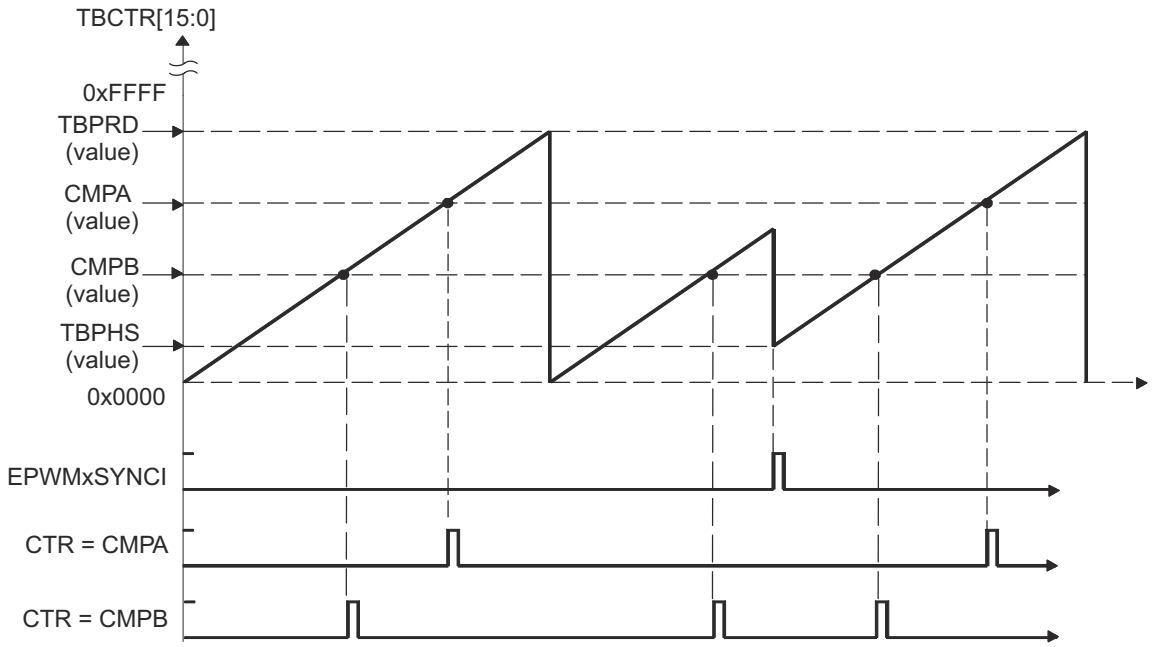
The global load control mechanism can also be used for all counter-compare registers by configuring the appropriate bits in the global load configuration register (GLDCFG). When the global load mode is selected the transfer of contents from shadow register to active register, for all registers that have this mode enabled, occurs at the same event as defined by the configuration bits in the Global Shadow to Active Load Control Register (GLDCTL). The global load control mechanism is explained in Section 20.4.7.

20.5.4 Count Mode Timing Waveforms

The counter-compare module can generate compare events in all three count modes:

- Up-count mode: used to generate an asymmetrical PWM waveform.
- Down-count mode: used to generate an asymmetrical PWM waveform.
- Up-down-count mode: used to generate a symmetrical PWM waveform.

To best illustrate the operation of the first three modes, the timing diagrams in Figure 20-17 through Figure 20-20 show when events are generated and how the EPWMxSYNCI signal interacts.



An EPWMxSYNCI external synchronization event can cause a discontinuity in the TBCTR count sequence. This can lead to a compare event being skipped. This skipping is considered normal operation and must be taken into account.

Figure 20-17. Counter-Compare Event Waveforms in Up-Count Mode

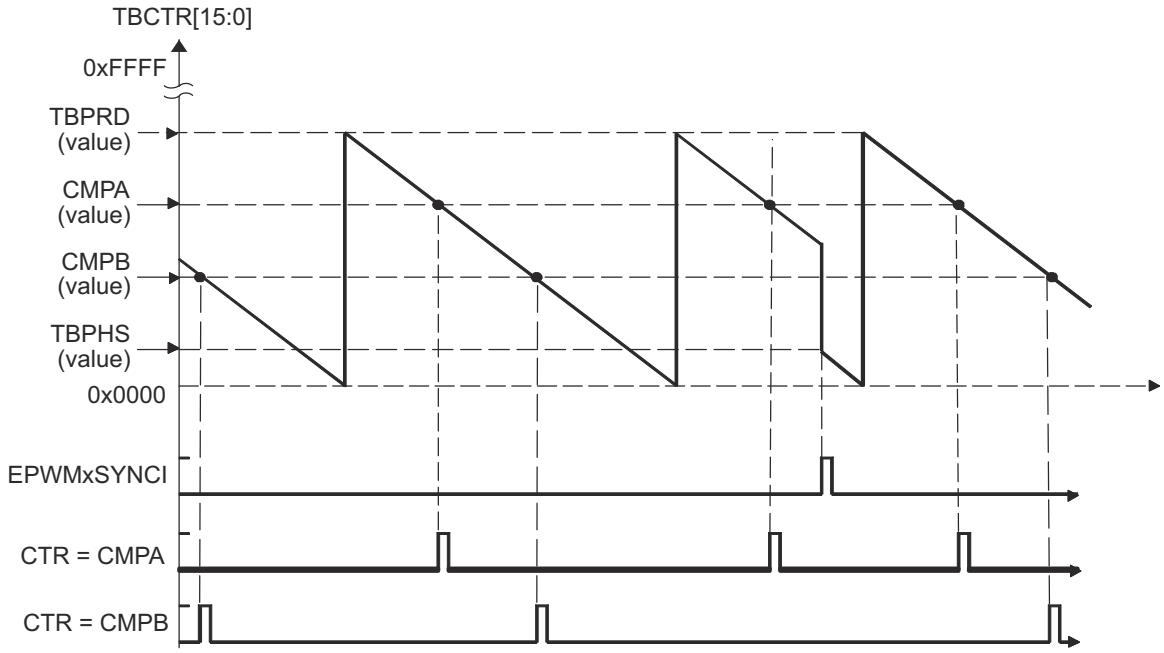


Figure 20-18. Counter-Compare Events in Down-Count Mode

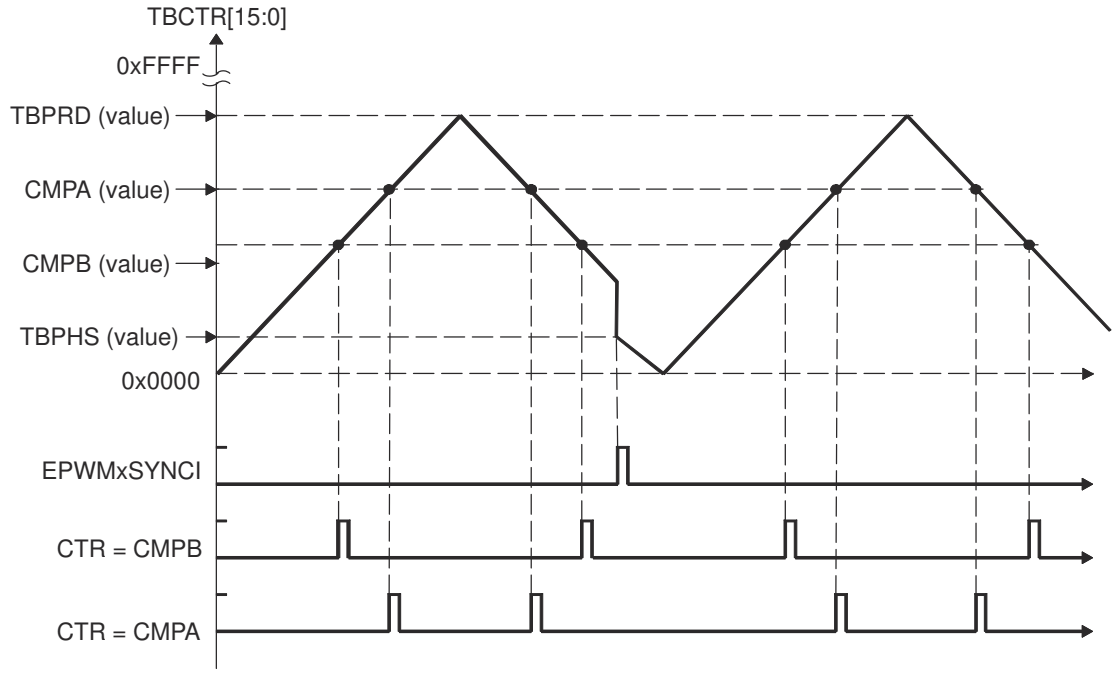


Figure 20-19. Counter-Compare Events In Up-Down-Count Mode, TBCTL[PHSDIR = 0] Count Down On Synchronization Event

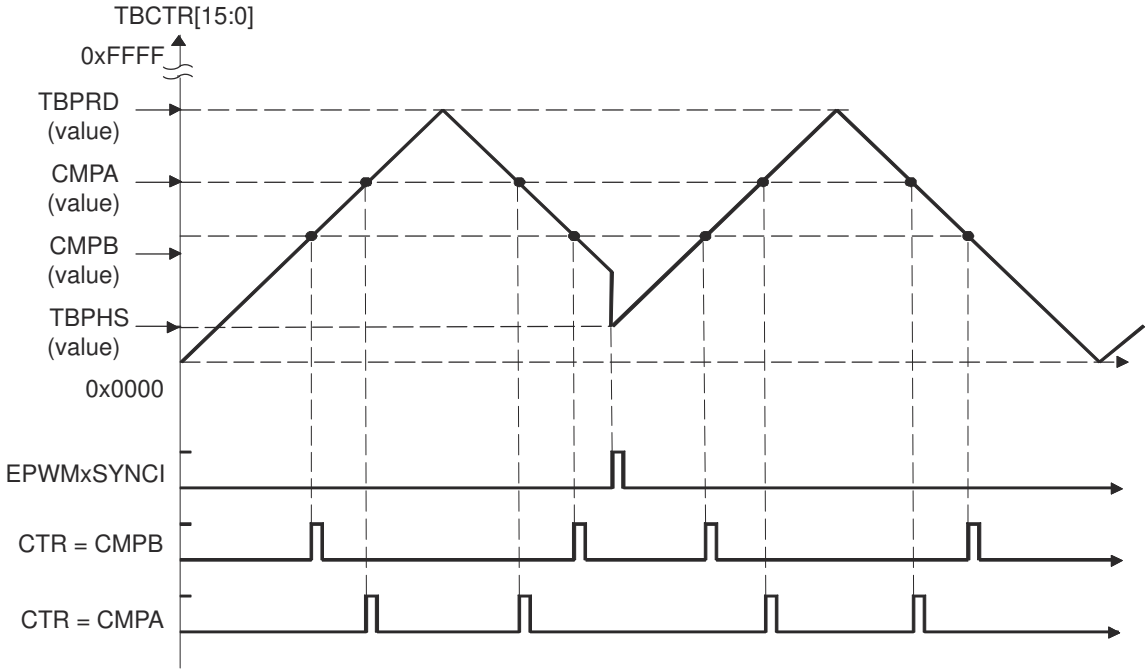


Figure 20-20. Counter-Compare Events In Up-Down-Count Mode, TBCTL[PHSDIR = 1] Count Up On Synchronization Event

20.6 Action-Qualifier (AQ) Submodule

The action-qualifier submodule has the most important role in waveform construction and PWM generation. It decides which events are converted into various action types, thereby producing the required switched waveforms at the EPWMxA and EPWMxB outputs.

Figure 20-21 shows the action-qualifier (AQ) submodule in the ePWM system.

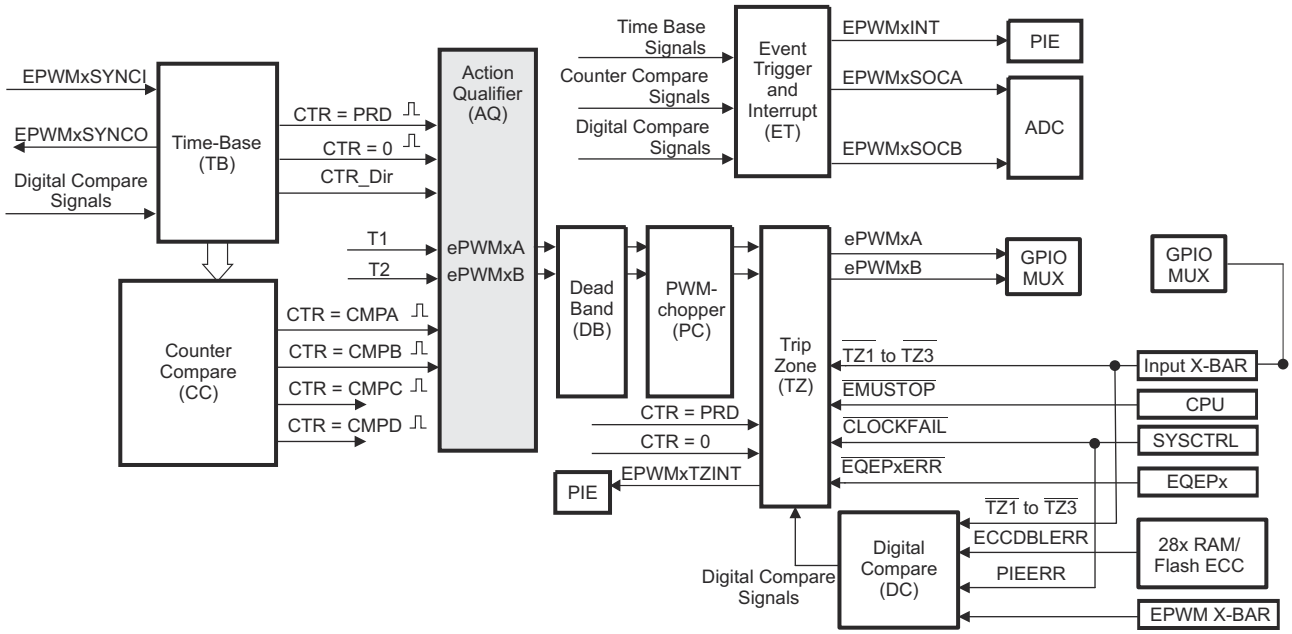


Figure 20-21. Action-Qualifier Submodule

20.6.1 Purpose of the Action-Qualifier Submodule

The action-qualifier submodule is responsible for the following:

- Qualifying and generating actions (set, clear, toggle) based on the following events:
  - CTR = PRD: Time-base counter equal to the period (TBCTR = TBPRD).
  - CTR = Zero: Time-base counter equal to zero (TBCTR = 0x00)
  - CTR = CMPA: Time-base counter equal to the counter-compare A register (TBCTR = CMPA)
  - CTR = CMPB: Time-base counter equal to the counter-compare B register (TBCTR = CMPB)
- T1, T2 events: Trigger events based on comparator, trip or syncin events
- Managing priority when these events occur concurrently
- Providing independent control of events when the time-base counter is increasing and when it is decreasing

20.6.2 Action-Qualifier Submodule Control and Status Register Definitions

The action-qualifier submodule operation is shown in Figure 20-22 and monitored by way of the registers in Section 20.17.

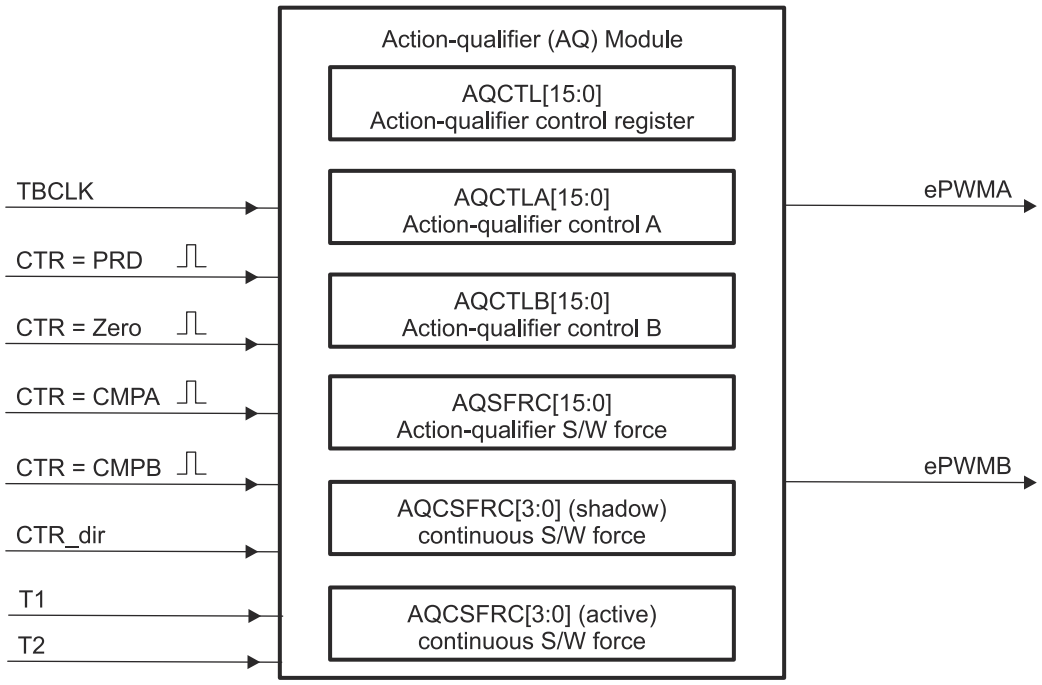


Figure 20-22. Action-Qualifier Submodule Inputs and Outputs

For convenience, the possible input events are summarized again in Table 20-4.

Table 20-4. Action-Qualifier Submodule Possible Input Events

Signal	Description	Registers Compared
CTR = PRD	Time-base counter equal to the period value	TBCTR = TBPRD
CTR = Zero	Time-base counter equal to zero	TBCTR = 0x00
CTR = CMPA	Time-base counter equal to the counter-compare A	TBCTR = CMPA
CTR = CMPB	Time-base counter equal to the counter-compare B	TBCTR = CMPB
T1 event	Based on comparator, trip or syncin events	None
T2 event	Based on comparator, trip or syncin events	None
Software forced event	Asynchronous event initiated by software	

The software forced action is a useful asynchronous event. This control is handled by the AQSFRC and AQCSFRC registers.

Note

If the CSFA is not used in shadow mode, the RLDCSF bit must be configured to disable shadow mode.

The action-qualifier submodule controls how the two outputs EPWMxA and EPWMxB behave when a particular event occurs. The event inputs to the action-qualifier submodule are further qualified by the counter direction (up or down). This allows for independent action on outputs on both the count-up and count-down phases.

The possible actions imposed on outputs EPWMxA and EPWMxB are:

- **Set High:**

  Set output EPWMxA or EPWMxB to a high level.
- **Clear Low:**

  Set output EPWMxA or EPWMxB to a low level.
- **Toggle:**

  If EPWMxA or EPWMxB is currently pulled high, then pull the output low. If EPWMxA or EPWMxB is currently pulled low, then pull the output high.
- **Do Nothing:**

  Keep outputs EPWMxA and EPWMxB at same level as currently set. Although the "Do Nothing" option prevents an event from causing an action on the EPWMxA and EPWMxB outputs, this event can still trigger interrupts and ADC start of conversion. See the description in Section 20.10 for details.

Actions are specified independently for either output (EPWMxA or EPWMxB). Any or all events can be configured to generate actions on a given output. For example, both CTR = CMPA and CTR = CMPB can operate on output EPWMxA. All qualifier actions are configured via the control registers found at the end of this section.

For clarity, the illustrations in this chapter use a set of symbolic actions. These symbols are summarized in Figure 20-23. Each symbol represents an action as a marker in time. Some actions are fixed in time (zero and period) while the CMPA and CMPB actions are moveable and their time positions are programmed by way of the counter-compare A and B registers, respectively. To turn off or disable an action, use the "Do Nothing option"; it is the default at reset.

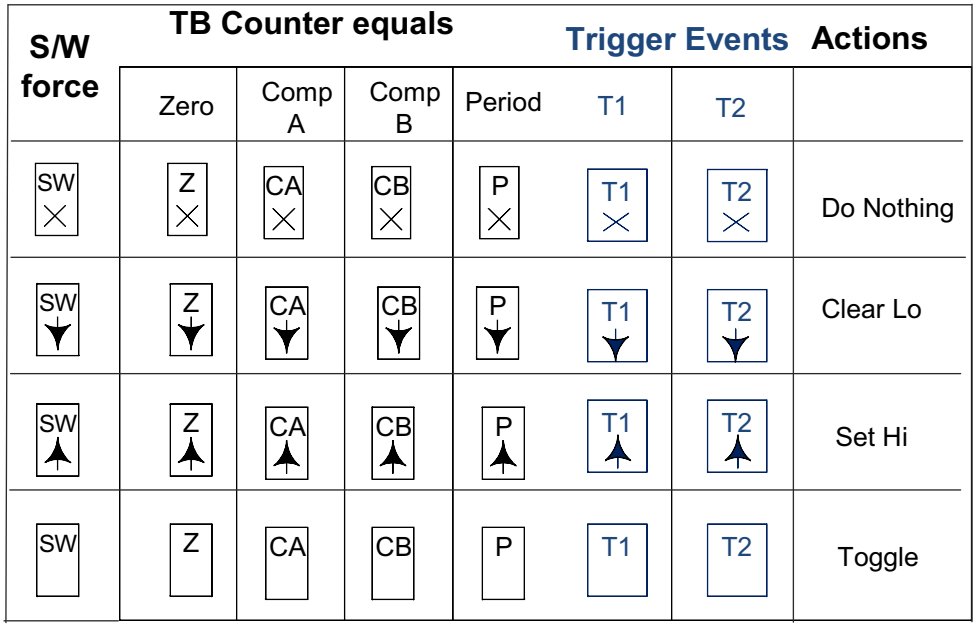


Figure 20-23. Possible Action-Qualifier Actions for EPWMxA and EPWMxB Outputs

The Action Qualifier Trigger Event Source Selection register (AQTSRCSEL) is used to select the source for T1 and T2 events. T1/T2 selection and configuration of a trip/digital-compare event in Action Qualifier submodule is independent of the configuration of that event in the Trip-Zone submodule. A particular trip event may or may not be configured to cause trip action in the Trip Zone submodule, but the same event can be used by the Action Qualifier to generate T1/T2 for controlling PWM generation.

20.6.3 Action-Qualifier Event Priority

It is possible for the ePWM action qualifier to receive more than one event at the same time. In this case events are assigned a priority by the hardware. The general rule is events occurring later in time have a higher priority and software forced events always have the highest priority. The event priority levels for up-down-count mode are shown in Table 20-5. A priority level of 1 is the highest priority and level 10 is the lowest. The priority changes slightly depending on the direction of TBCTR.

Table 20-5. Action-Qualifier Event Priority for Up-Down-Count Mode

Priority Level	Event If TBCTR is Incrementing TBCTR = Zero up to TBCTR = TBPRD	Event If TBCTR is Decrementing TBCTR = TBPRD down to TBCTR = 1
1 (Highest)	Software forced event	Software forced event
2	T1 on up-count (T1U)	T1 on down-count (T1D)
3	T2 on up-count (T2U)	T2 on down-count (T2D)
4	Counter equals CMPB on up-count (CBU)	Counter equals CMPB on down-count (CBD)
5	Counter equals CMPA on up-count (CAU)	Counter equals CMPA on down-count (CAD)
6	Counter equals zero	Counter equals period (TBPRD)
7	T1 on down-count (T1D)	T1 on up-count (T1U)
8	T2 on down-count (T2D)	T2 on up-count (T2U)
9	Counter equals CMPB on down-count (CBD)	Counter equals CMPB on up-count (CBU)
10 (Lowest)	Counter equals CMPA on down-count (CAD)	Counter equals CMPA on up-count (CBU)

Table 20-6 shows the action-qualifier priority for up-count mode. In this case, the counter direction is always defined as up and therefore, down-count events will never be taken.

Table 20-6. Action-Qualifier Event Priority for Up-Count Mode

Priority Level	Event
1 (Highest)	Software forced event
2	Counter equal to period (TBPRD)
3	T1 on up-count (T1U)
4	T2 on up-count (T2U)
5	Counter equal to CMPB on up-count (CBU)
6	Counter equal to CMPA on up-count (CAU)
7 (Lowest)	Counter equal to Zero

Table 20-7 shows the action-qualifier priority for down-count mode. In this case, the counter direction is always defined as down and thus up-count events will never be taken.

Table 20-7. Action-Qualifier Event Priority for Down-Count Mode

Priority Level	Event
1 (Highest)	Software forced event
2	Counter equal to Zero
3	T1 on down-count (T1D)
4	T2 on down-count (T2D)
5	Counter equal to CMPB on down-count (CBD)
6	Counter equal to CMPA on down-count (CAD)
7 (Lowest)	Counter equal to period (TBPRD)

It is possible to set the compare value greater than the period. In this case the action will take place as shown in Table 20-8.

Table 20-8. Behavior if CMPA/CMPB is Greater than the Period

Counter Mode	Compare on Up-Count Event CAD/CBD	Compare on Down-Count Event CAD/CBD
Up-Count Mode	If CMPA/CMPB ≤ TBPRD period, then the event occurs on a compare match (TBCTR=CMPA or CMPB).	Never occurs.
	If CMPA/CMPB > TBPRD, then the event will not occur.	
Down-Count Mode	Never occurs.	If CMPA/CMPB < TBPRD, the event will occur on a compare match (TBCTR=CMPA or CMPB).
		If CMPA/CMPB ≥ TBPRD, the event will occur on a period match (TBCTR=TBPRD).
Up-Down-Count Mode	If CMPA/CMPB < TBPRD and the counter is incrementing, the event occurs on a compare match (TBCTR=CMPA or CMPB).	If CMPA/CMPB < TBPRD and the counter is decrementing, the event occurs on a compare match (TBCTR=CMPA or CMPB).
	If CMPA/CMPB is ≥ TBPRD, the event will occur on a period match (TBCTR = TBPRD).	If CMPA/CMPB ≥ TBPRD, the event occurs on a period match (TBCTR=TBPRD).

20.6.4 AQCTLA and AQCTLB Shadow Mode Operations

To enable Action Qualifier mode changes which must occur at the end of a period even when the phase changes, shadowing of the AQCTLA and AQCTLB registers has been added on ePWMs type 2 and later. Additionally, shadow to active load on SYNC of these registers is supported as well. Shadowing of this register is enabled and disabled by the AQCTL[SHDWAQAMODE] and AQCTL[SHDWAQBMODE] bits. These bits enable and disable the AQCTLA shadow register and AQCTLB shadow register, respectively. The behavior of the two load modes is:

Shadow Mode

The shadow mode for the AQCTLA is enabled by setting the AQCTL[SHDWAQAMODE] bit, and the shadow register for AQCTLB is enabled by setting the AQCTL[SHDWAQBMODE] bit. Shadow mode is disabled by default for both AQCTLA and AQCTLB

If the shadow register is enabled, then the content of the shadow register is transferred to the active register on one of the following events as specified by the AQCTL[LDAQAMODE], AQCTL[LDAQBMODE], AQCTL[LDAQASYNC], and AQCTL[LDAQBSYNC] register bits:

- CTR = PRD: Time-base counter equal to the period (TBCTR = TBPRD).
- CTR = Zero: Time-base counter equal to zero (TBCTR = 0x00)
- Both CTR = PRD and CTR = Zero
- SYNC event caused by DCAEVT1 or DCBEVT1 or EPWMxSYNCI or TBCTL[SWFSYNC]
- Both SYNC event or a selection made by LDAQAMODE/LDAQBMODE

Global Load Support

Global load control mechanism can also be used for AQCTLA:AQCTLA2, AQCTLB:AQCTLB2, and AQCSFRC registers by configuring the appropriate bits in the global load configuration register (GLDCFG). When global load mode is selected, the transfer of contents from shadow register to active register for all registers that have this mode enabled, occurs at the same event as defined by the configuration bits in the Global Shadow to Active Load Control Register (GLDCTL). The global load control mechanism is explained in Section 20.4.7.

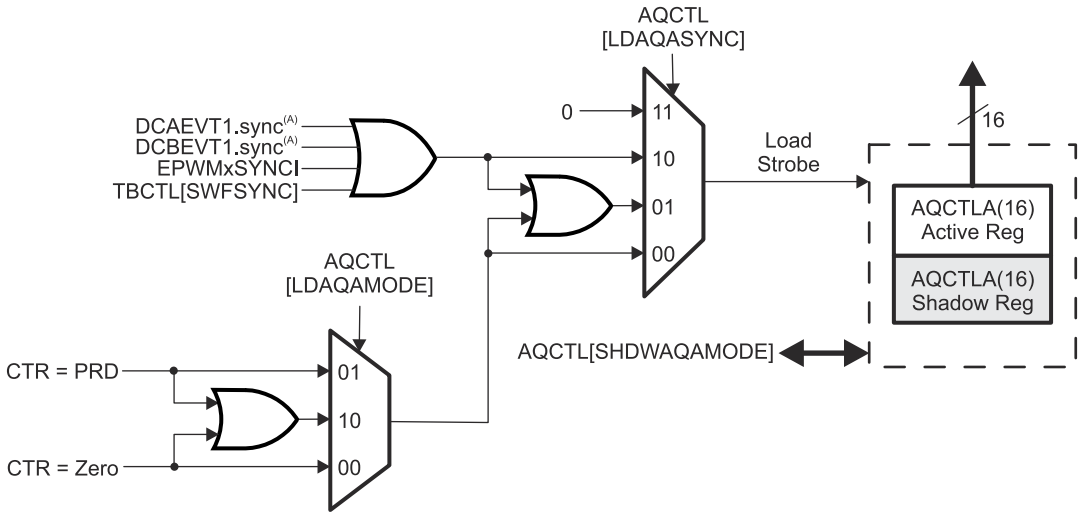
Immediate Load Mode

If immediate load mode is selected (that is, AQCTL[SHDWAQAMODE] = 0 or AQCTL[SHDWAQBMODE] = 0), then a read from or a write to the register will go directly to the active register. See Figure 20-24 and Figure 20-25.

Note

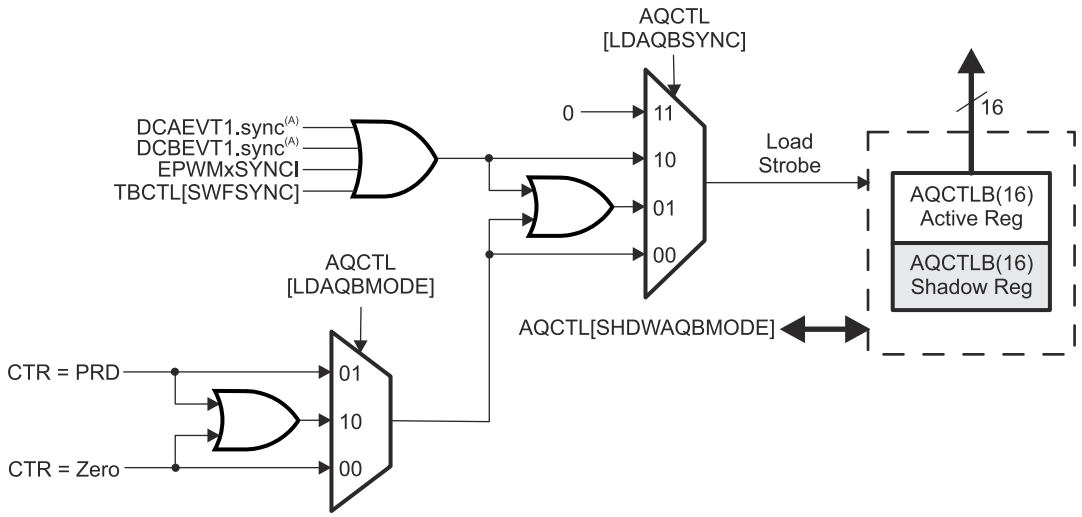
Shadow to Active Load of Action Qualifier Output A/B Control Register [AQCTLA and AQCTLB] on CMPA = 0 or CMPB = 0 boundary

If the Counter-Compare A Register (CMPA) or Counter-Compare B Register (CMPB) is set to a value of 0 and the action qualifier action on AQCTLA and AQCTLB is configured to occur in the same instant as a shadow to active load (that is, CMPA=0 and AQCTLA shadow to active load on TBCTR=0 using AQCTL register LDAQAMODE and LDAQAMODE bits), then both events enter contention and it is recommended to use a Non-Zero Counter-Compare when using Shadow to Active Load of Action Qualifier Output A/B Control Register on TBCTR = 0 boundary.



A. These events are generated by the ePWM digital compare (DC) submodule based on the levels of the TRIPIN inputs (for example, CMPSSx and $\overline{TZ}$ signals).

Figure 20-24. AQCTL[SHDWAQAMODE]



A. These events are generated by the ePWM digital compare (DC) submodule based on the levels of the TRIPIN inputs (for example, CMPSSx and $\overline{TZ}$ signals).

Figure 20-25. AQCTL[SHDWAQBMODE]

20.6.5 Waveforms for Common Configurations

Note

The waveforms in this document show the behavior of the ePWMs for a static compare register value. In a running system, the active compare registers (CMPA and CMPB) are typically updated from their respective shadow registers once every period. The user specifies when the update will take place; either when the time-base counter reaches zero or when the time-base counter reaches period. There are some cases when the action based on the new value can be delayed by one period or the action based on the old value can take effect for an extra period. Some PWM configurations avoid this situation. These include, but are not limited to, the following:

Use up-down-count mode to generate a symmetric PWM:

- If you load CMPA/CMPB on zero, then use CMPA/CMPB values greater than or equal to 1.
- If you load CMPA/CMPB on period, then use CMPA/CMPB values less than or equal to TBPRD-1.

  This means there will always be a pulse of at least one TBCLK cycle in a PWM period which, when very short, tend to be ignored by the system.

Use up-down-count mode to generate an asymmetric PWM:

- To achieve 50%-0% asymmetric PWM use the following configuration: Load CMPA/CMPB on period and use the period action to clear the PWM and a compare-up action to set the PWM. Modulate the compare value from 0 to TBPRD to achieve 50%-0% PWM duty.

When using up-count mode to generate an asymmetric PWM:

- To achieve 0-100% asymmetric PWM use the following configuration: Load CMPA/CMPB on TBPRD. Use the Zero action to set the PWM and a compare-up action to clear the PWM. Modulate the compare value from 0 to TBPRD+1 to achieve 0-100% PWM duty.

When using up-count mode to generate an asymmetric PWM with deadband enabled:

- To achieve 0%-100% PWM use the following configuration: When the CMPA value is too close to 0 or PRD such that the following conditions are met (CMPX < Deadband) or (CMPX > PRD – Deadband) the actions specified by the AQCTL register for CMPX will not take effect. To avoid this, the AQCTL settings will have to be altered under these conditions only to generate either high or low pulses for both CAU or CAD events (both set or both clear). Ensure that this software update is occurring synchronous to the PWM carrier cycle, and shadow mode is enabled.

When using up-down-count mode to generate an asymmetric PWM with deadband enabled:

- To achieve 0%-100% PWM use the following configuration: When the CMPA value is too close to 0 or PRD such that the following conditions are met (CMPX < Deadband/2) or (CMPX > PRD – (Deadband)/2) the actions specified by the AQCTL register for CMPX will not take effect. To avoid this, the AQCTL settings will have to be altered under these conditions only to generate either high or low pulses for both CAU or CAD events (both set or both clear). Ensure that this software update is occurring synchronous to the PWM carrier cycle, and shadow mode is enabled.

See the *Using Enhanced Pulse Width Modulator (ePWM) Module for 0-100% Duty Cycle Control Application Report*.

Figure 20-26 shows how a symmetric PWM waveform can be generated using the up-down-count mode of the TBCTR. In this mode, 0%-100% DC modulation is achieved by using equal compare matches on the up count and down count portions of the waveform. In the example shown, CMPA is used to make the comparison. When the counter is incrementing the CMPA match will pull the PWM output high. Likewise, when the counter is decrementing the compare match will pull the PWM signal low. When CMPA = 0, the PWM signal is high for the entire period giving a 100% duty waveform. When CMPA = TBPRD, the PWM signal is low achieving 0% duty.

When using this configuration in practice, if you load CMPA/CMPB on zero, then use CMPA/CMPB values greater than or equal to 1. If you load CMPA/CMPB on period, then use CMPA/CMPB values less than or equal

to TBPRD-1. This means there will always be a pulse of at least one TBCLK cycle in a PWM period which, when very short, tend to be ignored by the system.

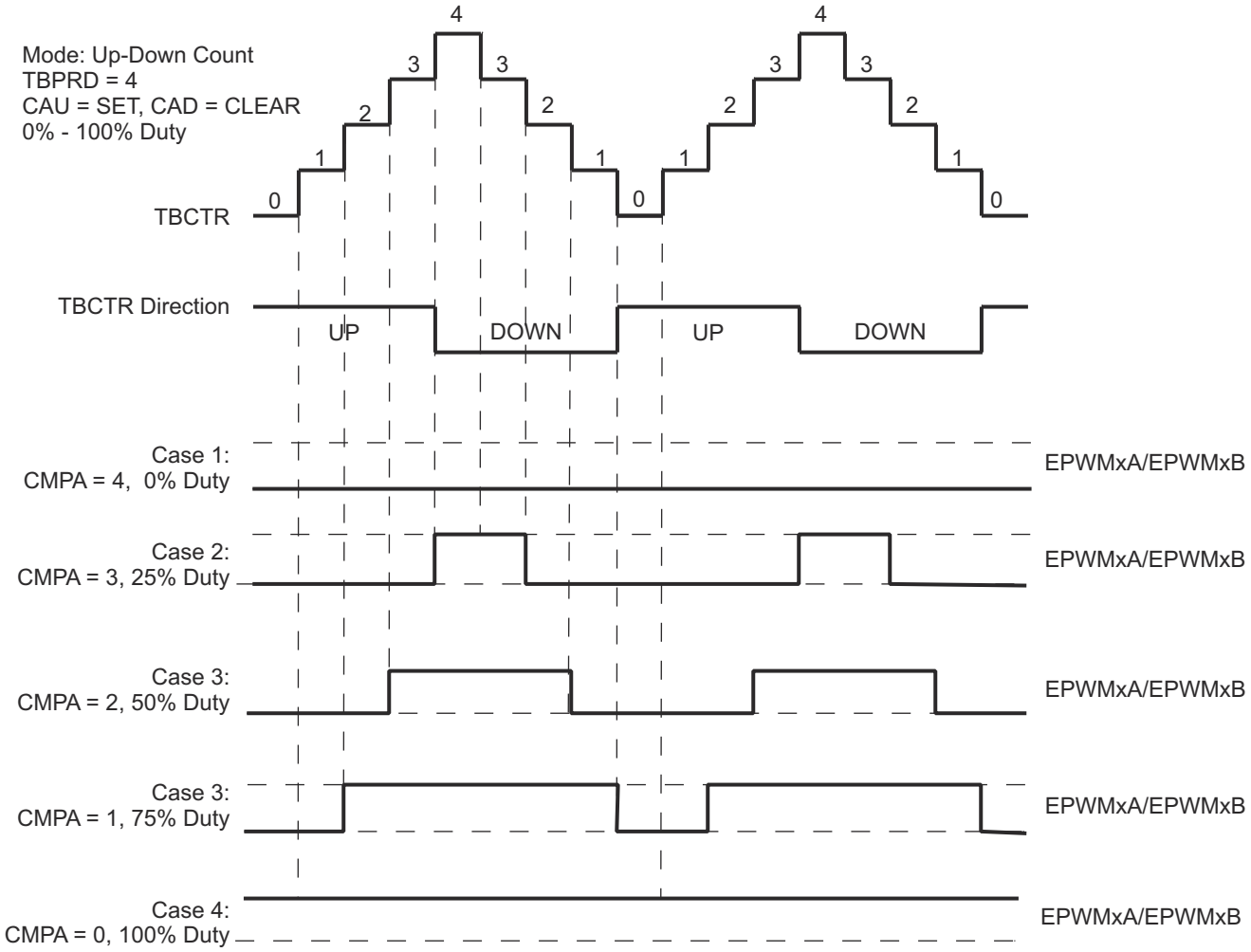
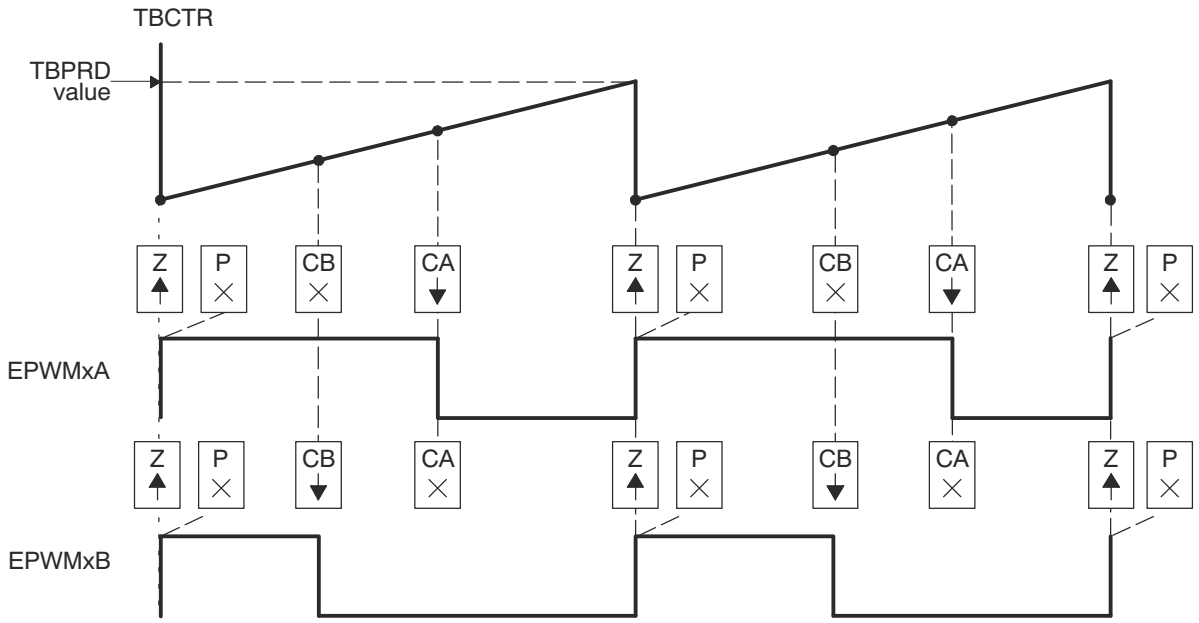


Figure 20-26. Up-Down-Count Mode Symmetrical Waveform

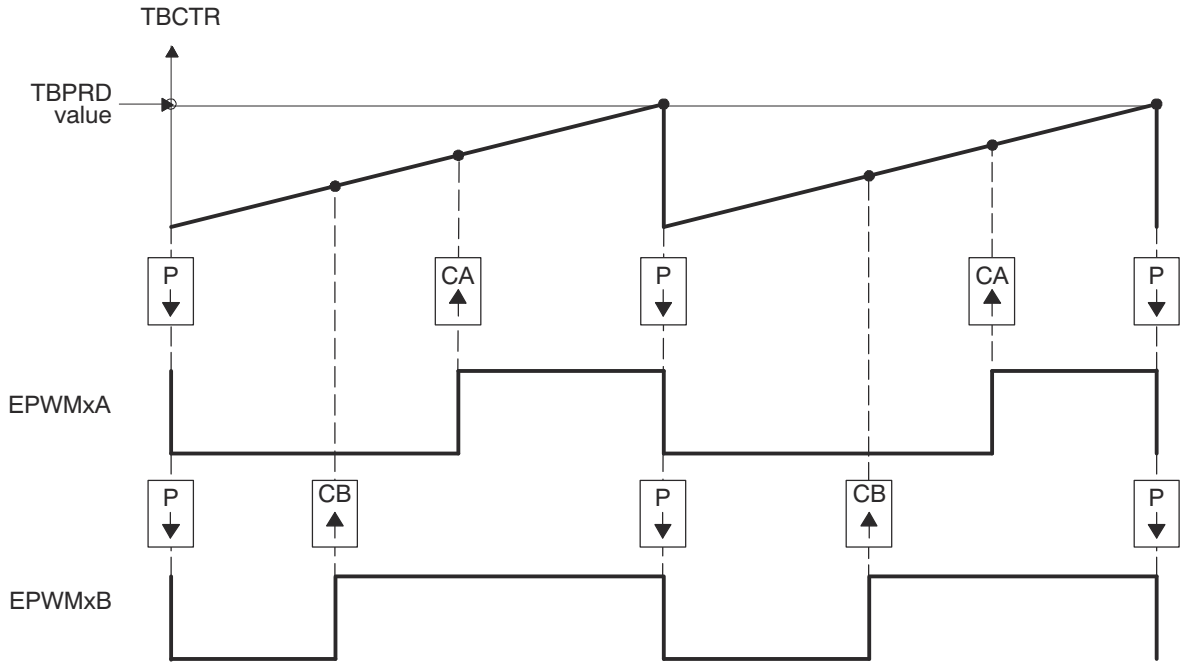
The PWM waveforms in Figure 20-27 through Figure 20-32 show some common action-qualifier configurations. Some conventions used in the figures and examples are as follows:

- TBPRD, CMPA, and CMPB refer to the value written in their respective registers. The active register, not the shadow register, is used by the hardware.
- CMPx, refers to either CMPA or CMPB.
- EPWMxA and EPWMxB refer to the output signals from ePWMx
- Up-Down means Count-up-and-down mode, Up means up-count mode and Dwn means down-count mode
- Sym = Symmetric, Asym = Asymmetric



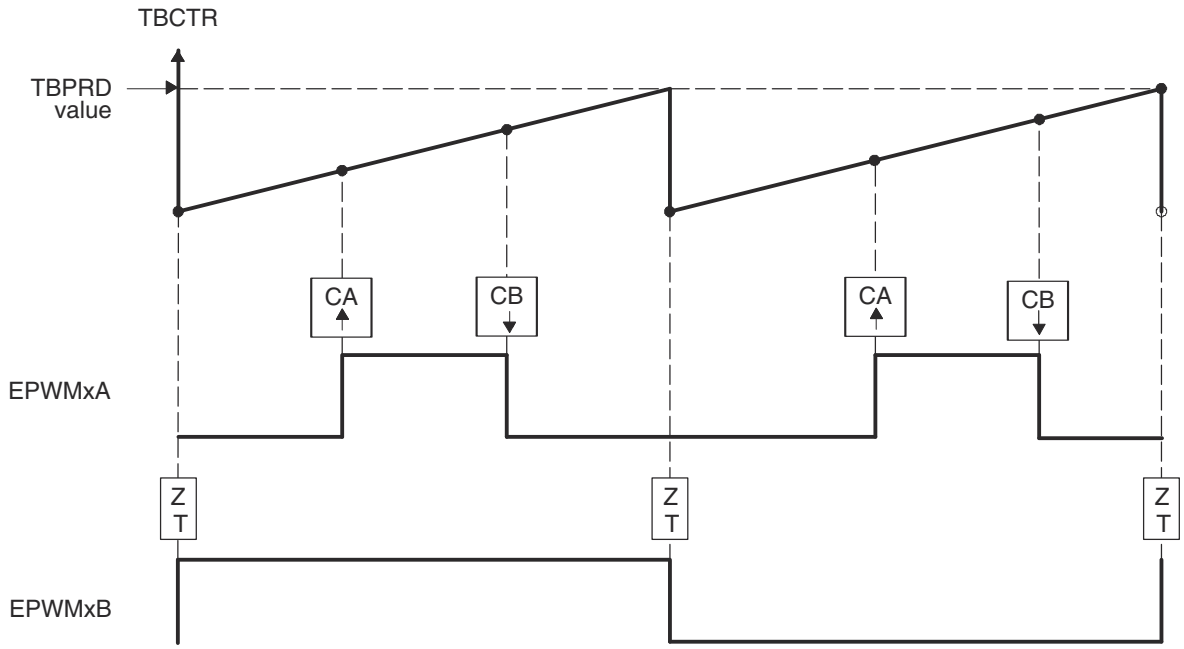
A. PWM period = (TBPRD + 1) × T_{TBCLK}
B. Duty modulation for EPWMxA is set by CMPA, and is active high (that is, high time duty proportional to CMPA).
C. Duty modulation for EPWMxB is set by CMPB and is active high (that is, high time duty proportional to CMPB).
D. The "Do Nothing" actions (X) are shown for completeness, but will not be shown on subsequent diagrams.
E. Actions at zero and period, although appearing to occur concurrently, are actually separated by one TBCLK period. TBCTR wraps from period to 0000.

Figure 20-27. Up, Single Edge Asymmetric Waveform, With Independent Modulation on EPWMxA and EPWMxB—Active High



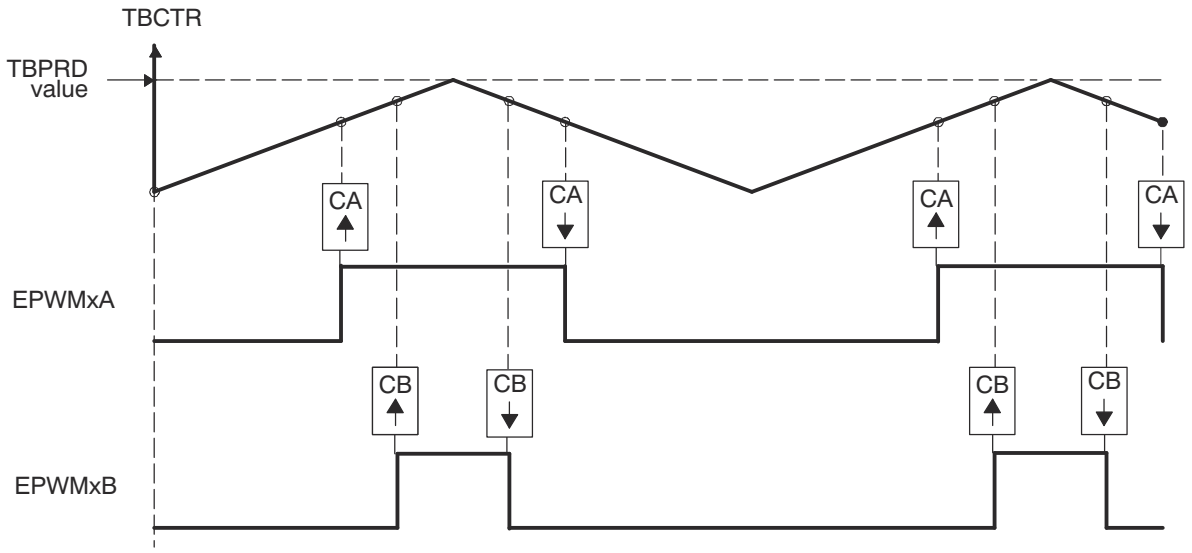
A. PWM period = (TBPRD + 1) × T_{TBCLK}
B. Duty modulation for EPWMxA is set by CMPA, and is active low (that is, the low time duty is proportional to CMPA).
C. Duty modulation for EPWMxB is set by CMPB and is active low (that is, the low time duty is proportional to CMPB).
D. Actions at zero and period, although appearing to occur concurrently, are actually separated by one TBCLK period. TBCTR wraps from period to 0000.

Figure 20-28. Up, Single Edge Asymmetric Waveform With Independent Modulation on EPWMxA and EPWMxB—Active Low



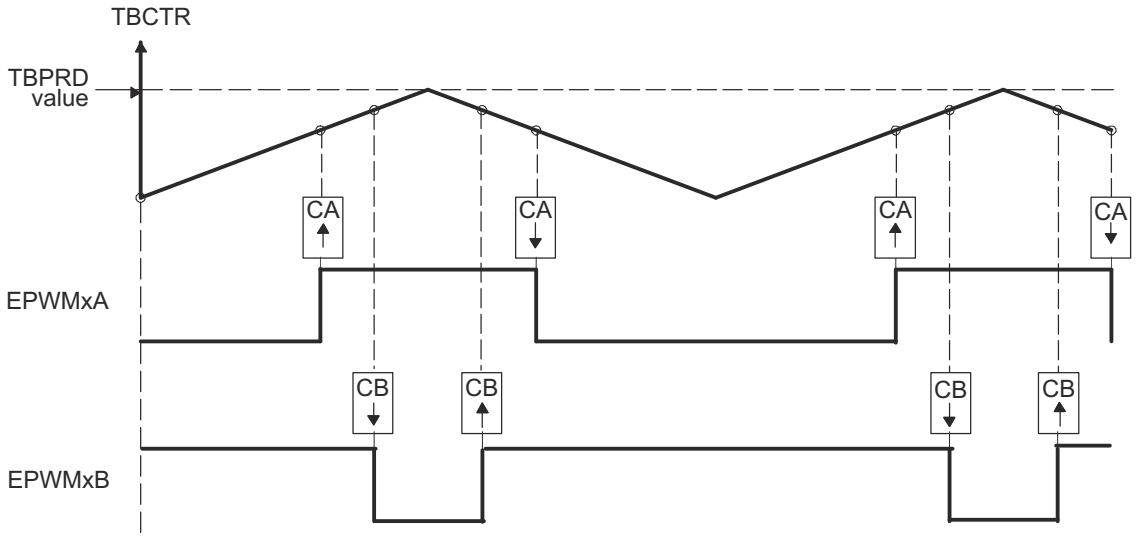
A. PWM frequency = 1/((TBPRD + 1) × T_{TBCLK})
B. Pulse can be placed anywhere within the PWM cycle (0000 - TBPRD)
C. High time duty proportional to (CMPB - CMPA)

Figure 20-29. Up-Count, Pulse Placement Asymmetric Waveform With Independent Modulation on EPWMxA



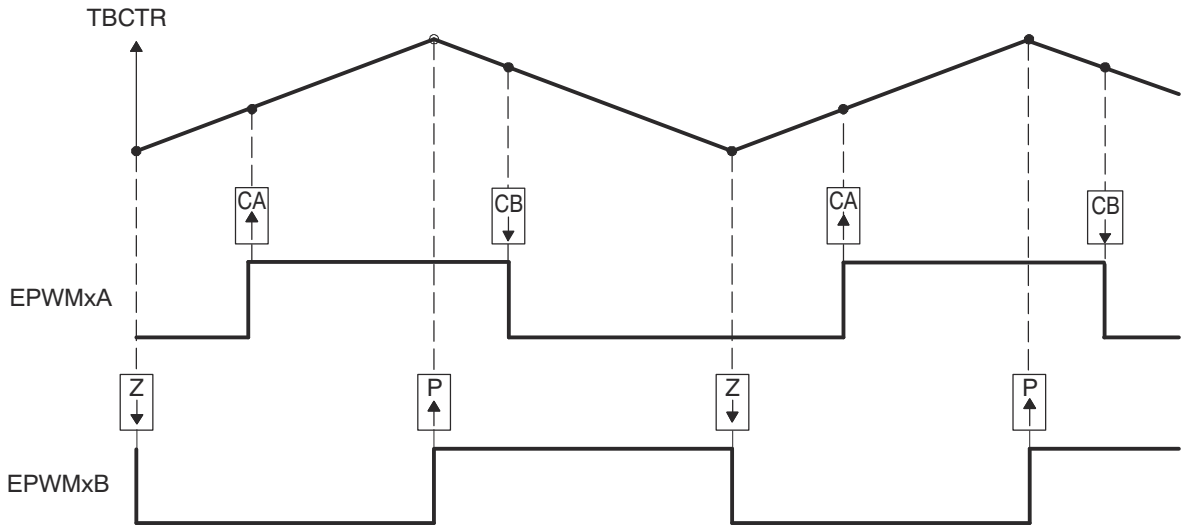
A. PWM period = 2 x TBPRD × T_{TBCLK}
B. Duty modulation for EPWMxA is set by CMPA, and is active low (that is, the low time duty is proportional to CMPA).
C. Duty modulation for EPWMxB is set by CMPB and is active low (that is, the low time duty is proportional to CMPB).
D. Outputs EPWMxA and EPWMxB can drive independent power switches.

Figure 20-30. Up-Down-Count, Dual Edge Symmetric Waveform, With Independent Modulation on EPWMxA and EPWMxB — Active Low



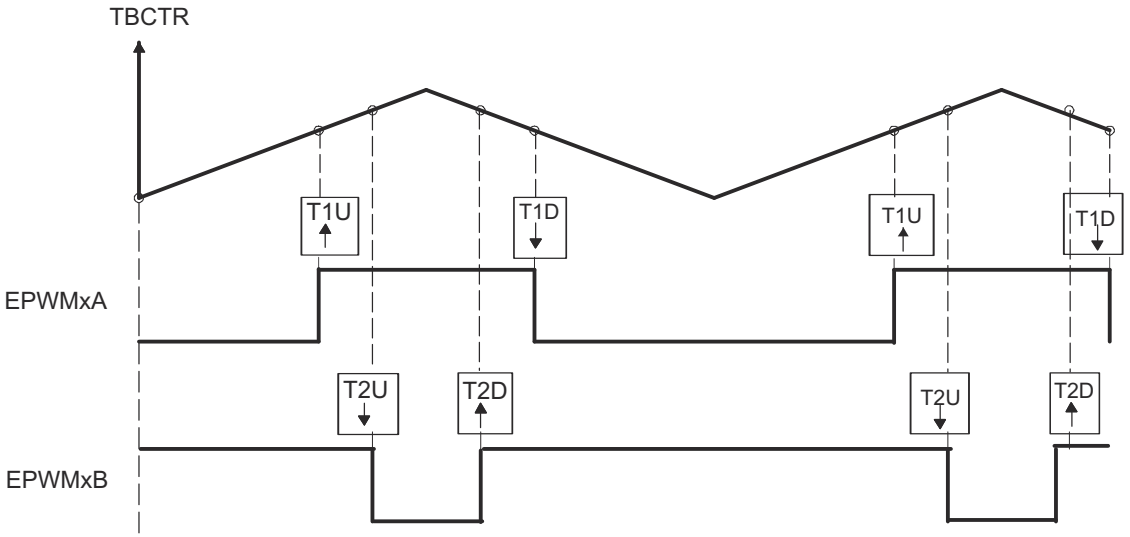
A. PWM period = 2 × TBPRD × T_{TBCLK}
B. Duty modulation for EPWMxA is set by CMPA, and is active low, that is, low time duty proportional to CMPA.
C. Duty modulation for EPWMxB is set by CMPB and is active high, that is, high time duty proportional to CMPB.
D. Outputs EPWMx can drive upper/lower (complementary) power switches.
E. Dead-band = CMPB - CMPA (fully programmable edge placement by software). Note the dead-band module is also available if the more classical edge delay method is required.

Figure 20-31. Up-Down-Count, Dual Edge Symmetric Waveform, With Independent Modulation on EPWMxA and EPWMxB — Complementary



A. PWM period = 2 × TBPRD × TBCLK
B. Rising edge and falling edge can be asymmetrically positioned within a PWM cycle. This allows for pulse placement techniques.
C. Duty modulation for EPWMxA is set by CMPA and CMPB.
D. Low time duty for EPWMxA is proportional to (CMPA + CMPB).
E. To change this example to active high, CMPA and CMPB actions need to be inverted (that is, Clear on CMPA, Set on CMPB).
F. Duty modulation for EPWMxB is fixed at 50% (utilizes spare action resources for EPWMxB).

Figure 20-32. Up-Down-Count, Dual Edge Asymmetric Waveform, With Independent Modulation on EPWMxA—Active Low



A. PWM period = 2 × TBPRD × TTBCLK
B. Independent T1 event actions when counter is counting up and when it is counting down are used to generate EPWMxA output.
C. Independent T2 event actions when counter is counting up and when it is counting down are used to generate EPWMxB output.
D. TZ1 is selected as the source for T1.
E. TZ2 is selected as the source for T2.

Figure 20-33. Up-Down-Count, PWM Waveform Generation Utilizing T1 and T2 Events

20.7 Dead-Band Generator (DB) Submodule

Figure 20-34 illustrates the dead-band submodule within the ePWM module.

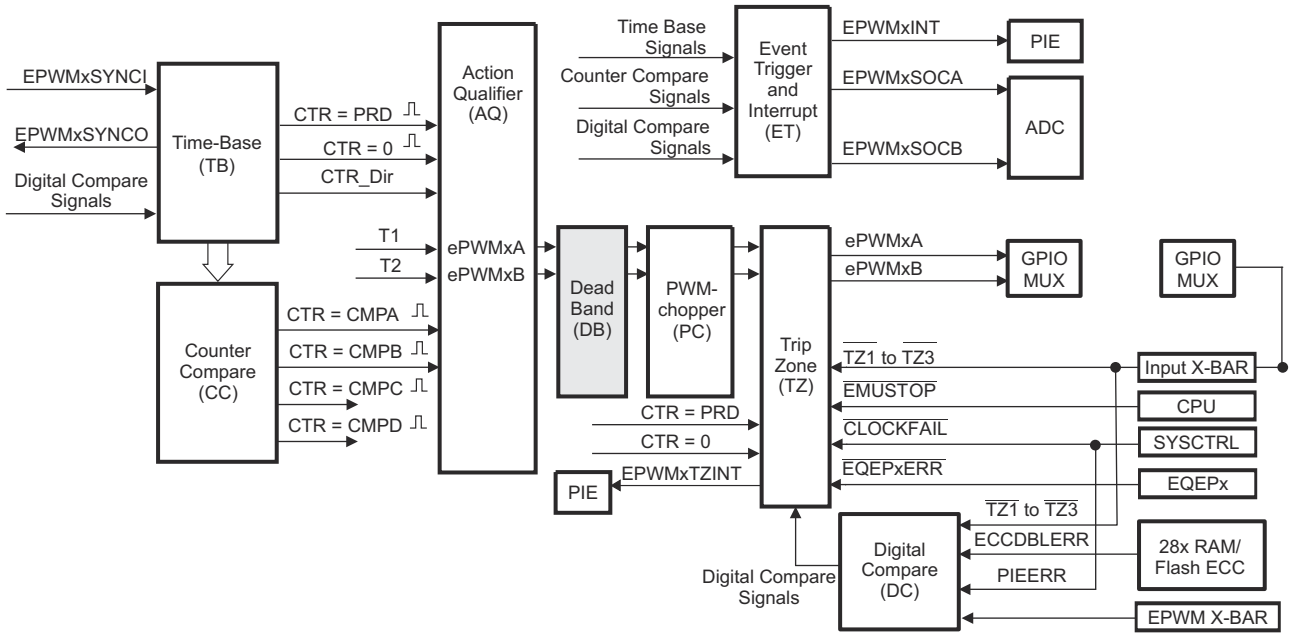


Figure 20-34. Dead_Band Submodule

20.7.1 Purpose of the Dead-Band Submodule

The action-qualifier (AQ) module section discussed how it is possible to generate the required dead band by having full control over edge placement using both the CMPA and CMPB resources of the ePWM module. However, if the more classical edge delay-based dead band with polarity control is required, then the dead-band submodule described here should be used.

The key functions of the dead-band module are:

- Generating appropriate signal pairs (EPWMxA and EPWMxB) with dead-band relationship from a single EPWMxA input
- Programming signal pairs for:
  - Active high (AH)
  - Active low (AL)
  - Active high complementary (AHC)
  - Active low complementary (ALC)
- Adding programmable delay to rising edges (RED)
- Adding programmable delay to falling edges (FED)
- Can be totally bypassed from the signal path (note dotted lines in diagram)

20.7.2 Dead-band Submodule Additional Operating Modes

On type 1 ePWM RED could appear on one channel output and FED could appear on the other channel output.

The following list shows the distinct difference between type 1 and type 4 modules with respect to dead-band operating modes:

- By adding S6, S7, and S8 in Figure 20-35, RED and FED can appear on both the A-channel and B-channel outputs. Additionally, both RED and FED together can be applied to either the A-channel or B-channel outputs to allow B-channel phase shifting with respect to the A-channel.

Note

Phase shifting B-channel with respect to the A-channel using the dead-band submodule additional operating modes has limitations with respect to the choice of RED and FED delay with respect to the operating duty cycle of the ePWMxA and ePWMxB outputs.

- The dead-band counters have also been increased to 14 bits
- Dead-band and dead-band High-resolution registers are now shadowed
- High-resolution dead-band RED and FED have been enabled using the DBREDHR and DBFEDHR registers

Note

The PWM chopper will not be enabled when high-resolution dead band is enabled.

High-resolution dead-band RED and FED requires Half-Cycle clocking mode (DBCTL[HALFCYCLE] = 1).

Cannot have both RED and FED together applied to both ePWMxA and ePWMxB. RED and FED together can be applied only to either OutA OR OutB.

Phase shifting B-channel with respect to the A-channel: When PWMxB is derived from PWMxA using the DEDB_MODE bit and by delaying rising edge and falling edge by the phase shift amount. When the duty cycle value on PWMxA is less than this phase shift amount, PWMxA's falling edge has precedence over the delayed rising edge for PWMxB. It is recommended to make sure the duty cycle value of the current waveform fed to the dead-band module is greater than the required phase shift amount.

The Type 4 action qualifier and dead-band outputs of the ePWM module are delayed by one TBCLK cycle in comparison to the Type 2 ePWM module, although the Type 4 behavior is the same as the Type 3 PWM. Both PWMA and PWMB signals are delayed under all circumstances.

Shadow Mode

The shadow mode for the DBRED is enabled by setting the DBCTL[SHDWDBREDMODE] bit and the shadow register for DBFED is enabled by setting the DBCTL [SHDWDBFEDMODE] bit. Shadow mode is disabled by default for both DBRED and DBFED

If the shadow register is enabled, then the content of the shadow register is transferred to the active register on one of the following events as specified by the DBCTL [LOADREDMODE] and DBCTL [LOADFEDMODE] register bits:

- CTR = PRD: Time-base counter equal to the period (TBCTR = TBPRD).
- CTR = Zero: Time-base counter equal to zero (TBCTR = 0x00)
- Both CTR = PRD and CTR = Zero

The DBCTL register can be shadowed. The shadow mode for DBCTL is enabled by setting the DBCTL2[SHDWDBCTLMODE] bit. If the shadow register is enabled then the content of the shadow register is transferred to the active register on one of the following events as specified by the DBCTL2[LOADDBCTLMODE] register bit:

- CTR = PRD: Time-base counter equal to the period (TBCTR = TBPRD)
- CTR = Zero: Time-base counter equal to zero (TBCTR = 0x00)
- Both CTR = PRD and CTR = Zero

Global Load Support

Global load control mechanism can also be used for DBRED:DBREDHR, DBFED:DBFEDHR, and DBCTL registers by configuring the appropriate bits in the global load configuration register (GLDCFG). When global load mode is selected the transfer of contents from shadow register to active register, for all registers that have this mode enabled, occurs at the same event as defined by the configuration bits in the Global Shadow to Active Load Control Register (GLDCTL). The Global load control mechanism is explained in Section 20.4.7.

Note

When DBRED/DBFED active is loaded with a new shadow value while DB counters are counting, the new DBRED/DBFED value only affects the NEXT PWMx edge and not the current edge.

A Dead-band value of zero should not be used when the Global Shadow to Active Load is set to occur at CTR=ZERO. Similarly, a Dead-band value of PRD should not be used when the Global Shadow to Active Load is set to occur at CTR=PRD.

TBPRDHR should not be used with Global load. If high resolution period must be changed in the application, users must write to the individual period registers from an ePWM ISR (The ISR must be synchronous with the PWM switching period), where the Global Load One-Shot bit is also written to.

20.7.3 Operational Highlights for the Dead-Band Submodule

The configuration options for the dead-band submodule are shown in Figure 20-35.

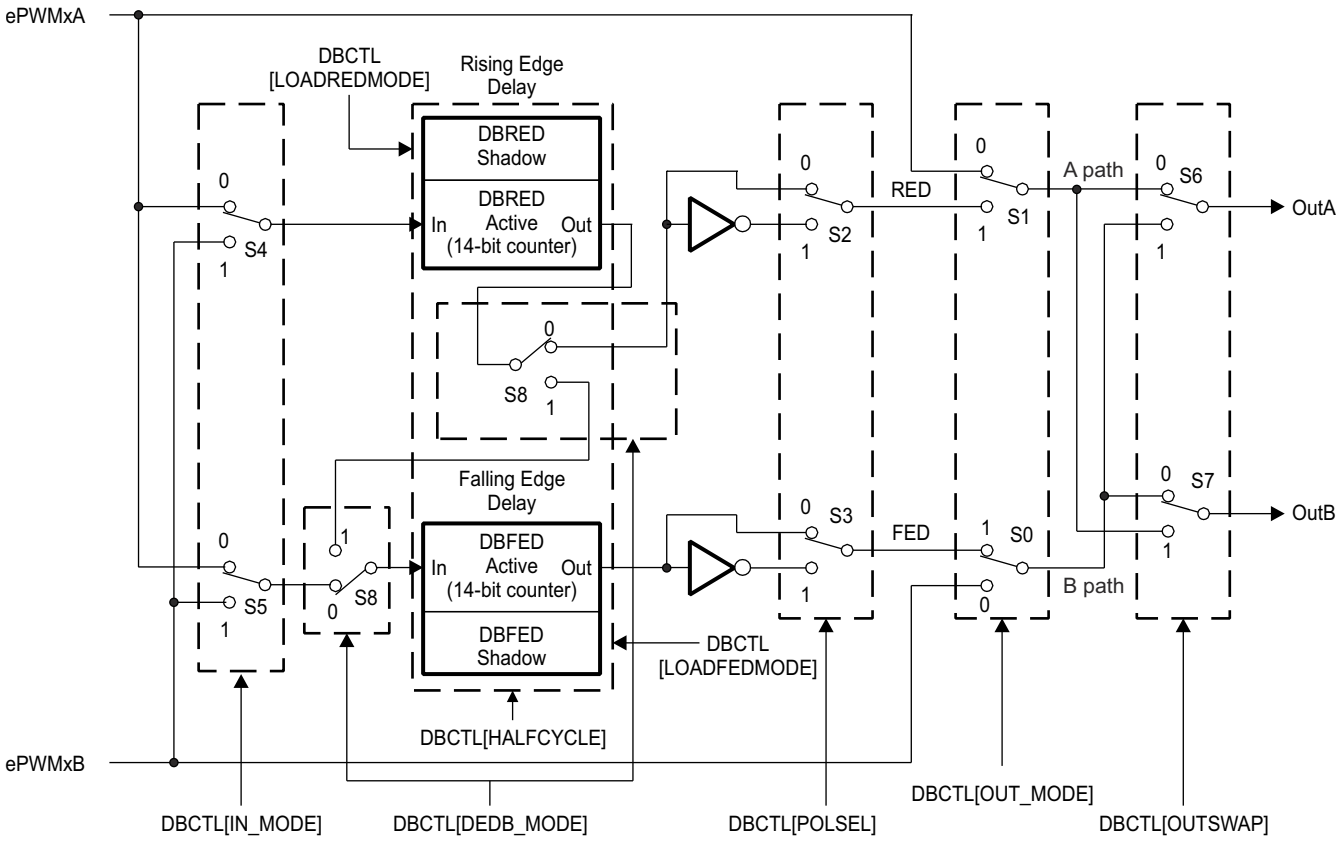


Figure 20-35. Configuration Options for the Dead-Band Submodule

Although all combinations are supported, not all are typical usage modes. Table 20-9 documents some classical dead-band configurations. These modes assume that the DBCTL[IN_MODE] is configured such that EPWMxA In is the source for both falling-edge and rising-edge delay. Enhanced, or non-traditional modes can be achieved by changing the input signal source. The modes shown in Table 20-9 fall into the following categories:

- **Mode 1: Bypass both falling-edge delay (FED) and rising-edge delay (RED)**

  Allows you to fully disable the dead-band submodule from the PWM signal path.
- **Mode 2-5: Classical Dead-Band Polarity Settings:**

  These represent typical polarity configurations that should address all the active high/low modes required by available industry power switch gate drivers. The waveforms for these typical cases are shown in Figure 20-36. Note that to generate equivalent waveforms to Figure 20-36, configure the action-qualifier submodule to generate the signal as shown for EPWMxA.
- **Mode 6: Bypass rising-edge-delay and Mode 7: Bypass falling-edge-delay**

  Finally the last two entries in Table 20-9 show combinations where either the falling-edge-delay (FED) or rising-edge-delay (RED) blocks are bypassed.

Figure 20-36 shows waveforms for typical cases where 0% < duty < 100%.

Table 20-9. Classical Dead-Band Operating Modes

Mode	Mode Description	DBCTL[POLSEL]		DBCTL[OUT_MODE]	
		S3	S2	S1	S0
1	EPWMxA and EPWMxB Passed Through (No Delay)	X	X	0	0
2	Active High Complementary (AHC)	1	0	1	1
3	Active Low Complementary (ALC)	0	1	1	1
4	Active High (AH)	0	0	1	1
5	Active Low (AL)	1	1	1	1
6	EPWMxA Out = EPWMxA In (No Delay) EPWMxB Out = EPWMxA In with Falling Edge Delay	0 or 1	0 or 1	0	1
7	EPWMxA Out = EPWMxA In with Rising Edge Delay EPWMxB Out = EPWMxB In with No Delay	0 or 1	0 or 1	1	0

Table 20-10. Additional Dead-Band Operating Modes

Mode Description	DBCTL[DEDB-MODE]	DBCTL[OUTSWAP]	
	S8	S6	S7
EPWMxA and EPWMxB signals are as defined by OUT-MODE bits.	0	0	0
EPWMxA = A-path as defined by OUT-MODE bits. EPWMxB = A-path as defined by OUT-MODE bits (rising edge delay or delay-bypassed A-signal path)	0	0	1
EPWMxA = B-path as defined by OUT-MODE bits (falling edge delay or delay-bypassed B-signal path) EPWMxB = B-path as defined by OUT-MODE bits	0	1	0
EPWMxA = B-path as defined by OUT-MODE bits (falling edge delay or delay-bypassed B-signal path) EPWMxB = A-path as defined by OUT-MODE bits (rising edge delay or delay-bypassed A-signal path)	0	1	1
Rising edge delay applied to EPWMxA / EPWMxB as selected by S4 switch (IN-MODE bits) on A signal path only. Falling edge delay applied to EPWMxA / EPWMxB as selected by S5 switch (IN-MODE bits) on B signal path only.	0	X	X
Rising edge delay and falling edge delay applied to source selected by S4 switch (IN-MODE bits) and output to B signal path only.(1)	1	X	X

(1) When this bit is set to 1, user should always either set OUT_MODE bits such that Apath = InA ***or*** OUTSWAP bits such that EPWMxA=Bpath. Otherwise, EPWMxA will be invalid.

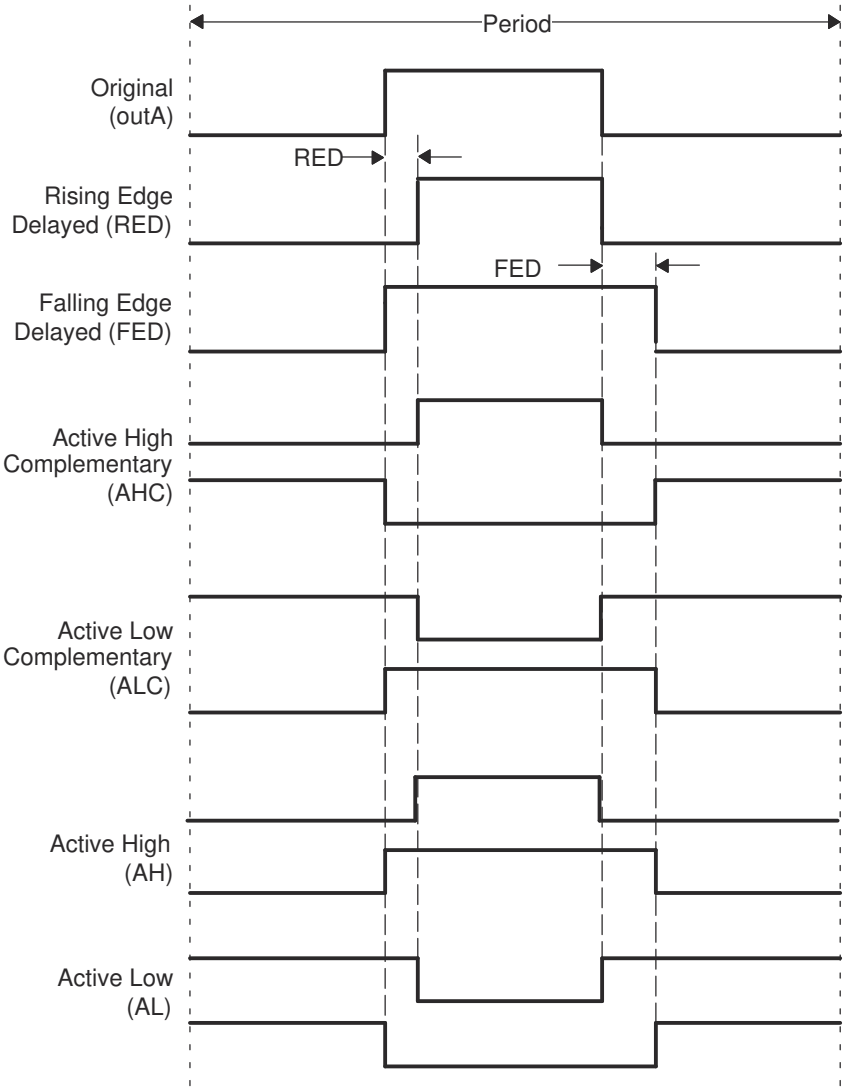


Figure 20-36. Dead-Band Waveforms for Typical Cases (0% < Duty < 100%)

The dead-band submodule supports independent values for rising-edge (RED) and falling-edge (FED) delays. The amount of delay is programmed using the DBRED and DBFED registers. These are 10-bit registers and their value represents the number of time-base clock, TBCLK, periods by which a signal edge is delayed. For example, the formula to calculate falling-edge-delay and rising-edge-delay is:

$FED = DBFED \times T_{TBCLK}$

$RED = DBRED \times T_{TBCLK}$

Where T_{TBCLK} is the period of TBCLK, the prescaled version of EPWMCLK.

For convenience, delay values for various TBCLK options are shown in Table 20-11. The ePWM input clock frequency that these delay values been computed by is 100 MHz.

Table 20-11. Dead-Band Delay Values in µS as a Function of DBFED and DBRED

Dead-Band Value	Dead-Band Delay in µS		
DBFED, DBRED	TBCLK = EPWMCLK/1	TBCLK = EPWMCLK /2	TBCLK = EPWMCLK/4
1	0.01 µS	0.02 µS	0.04 µS
5	0.05 µS	0.10 µS	0.20 µS
10	0.10 µS	0.20 µS	0.40 µS
100	1.00 µS	2.00 µS	4.00 µS
200	2.00 µS	4.00 µS	8.00 µS
400	4.00 µS	8.00 µS	16.00 µS
500	5.00 µS	10.00 µS	20.00 µS
600	6.00 µS	12.00 µS	24.00 µS
700	7.00 µS	14.00 µS	28.00 µS
800	8.00 µS	16.00 µS	32.00 µS
900	9.00 µS	18.00 µS	36.00 µS
1000	10.00 µS	20.00 µS	40.00 µS

When half-cycle clocking is enabled, the formula to calculate the falling-edge-delay and rising-edge-delay becomes:

$FED = DBFED \times T_{TBCLK}/2$

$RED = DBRED \times T_{TBCLK}/2$

20.8 PWM Chopper (PC) Submodule

Figure 20-37 illustrates the PWM chopper (PC) submodule within the ePWM module.

The PWM chopper submodule allows a high-frequency carrier signal to modulate the PWM waveform generated by the action-qualifier and dead-band submodules. This capability is important if you need pulse transformer-based gate drivers to control the power switching elements.

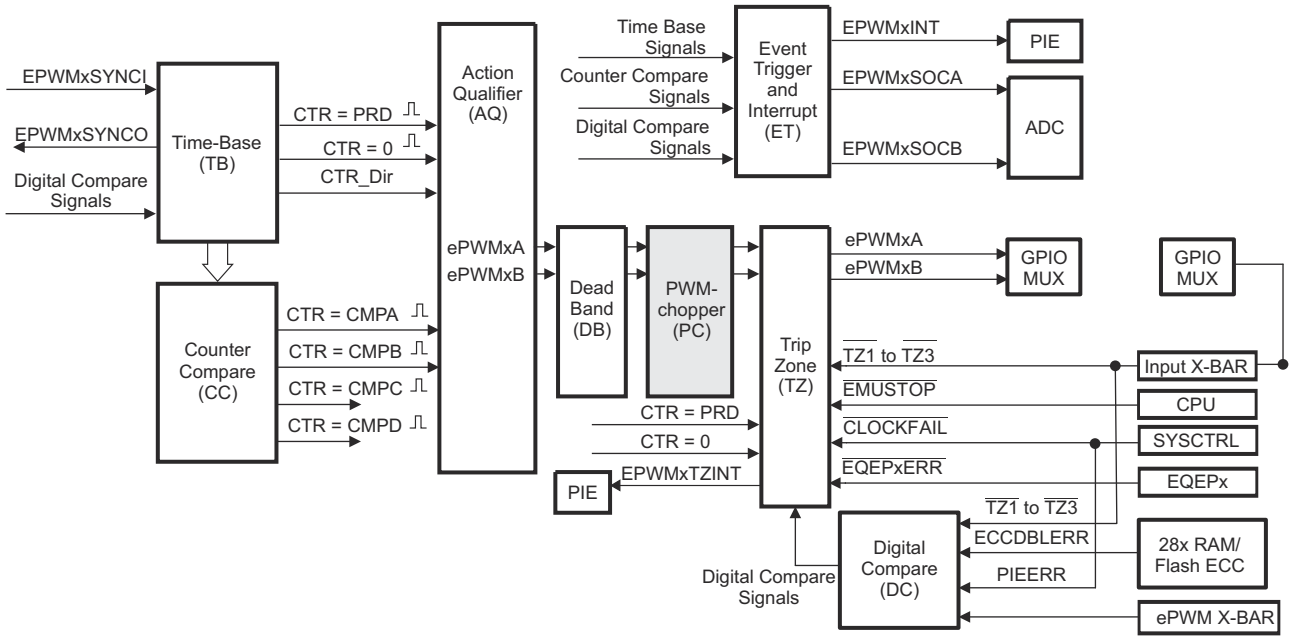


Figure 20-37. PWM Chopper Submodule

20.8.1 Purpose of the PWM Chopper Submodule

The key functions of the PWM chopper submodule are:

- Programmable chopping (carrier) frequency
- Programmable pulse width of first pulse
- Programmable duty cycle of second and subsequent pulses
- Can be fully bypassed if not required

20.8.2 Operational Highlights for the PWM Chopper Submodule

Figure 20-38 shows the operational details of the PWM chopper submodule. The carrier clock is derived from EPWMCLK. Its frequency and duty cycle are controlled via the CHPFREQ and CHPDUTY bits in the PCCTL register. The one-shot block is a feature that provides a high energy first pulse to ensure hard and fast power switch turn on, while the subsequent pulses sustain pulses, ensuring the power switch remains on. The one-shot width is programmed via the OSHTWTH bits. The PWM chopper submodule can be fully disabled (bypassed) via the CHPEN bit.

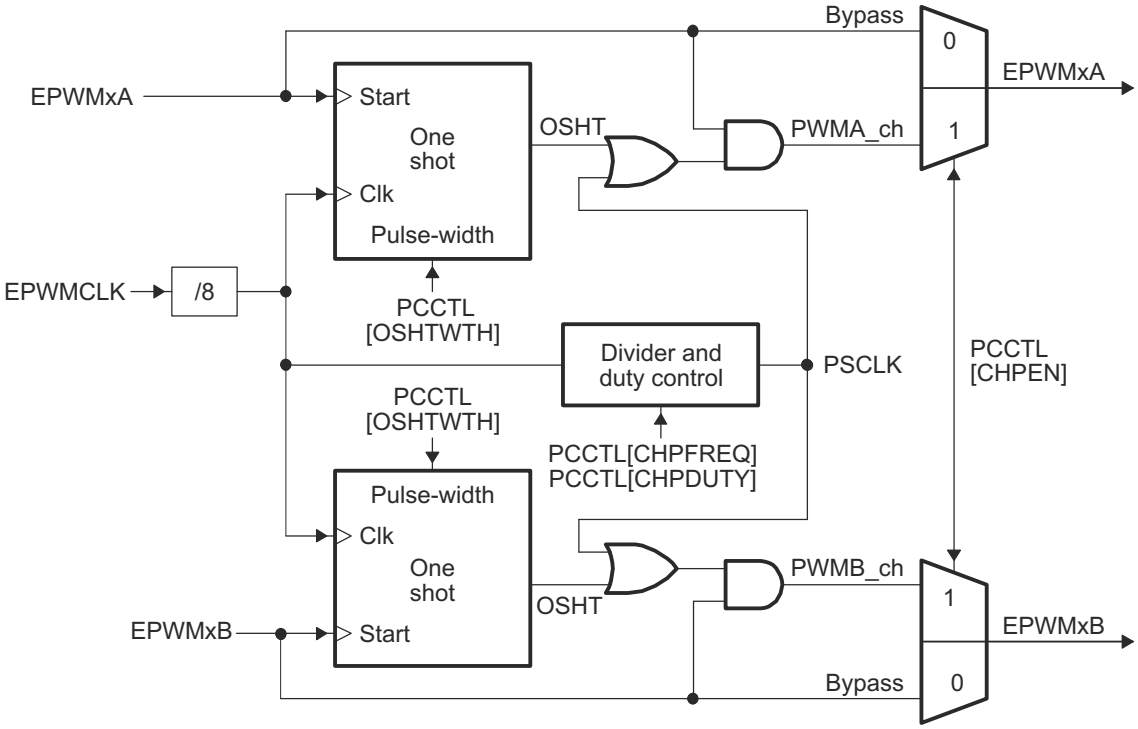


Figure 20-38. PWM Chopper Submodule Operational Details

20.8.3 Waveforms

Figure 20-39 shows simplified waveforms of the chopping action only; one-shot and duty-cycle control are not shown. Details of the one-shot and duty-cycle control are discussed in the following sections.

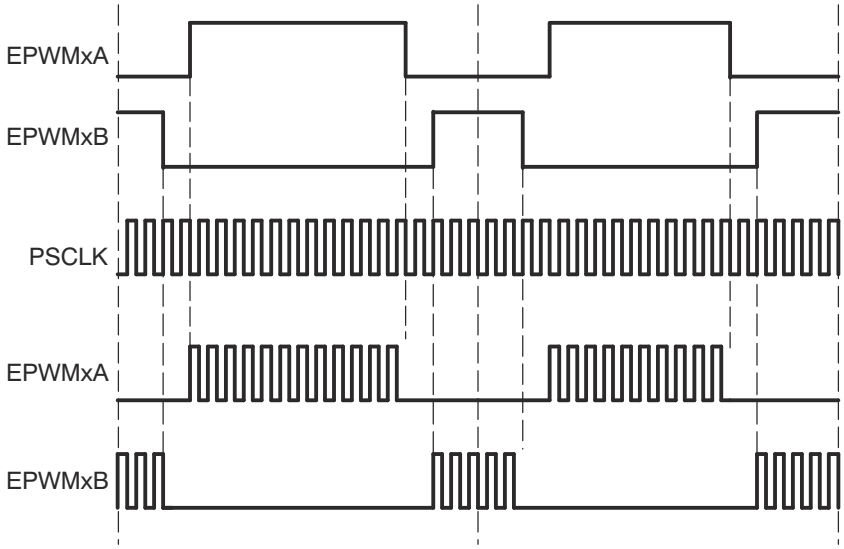


Figure 20-39. Simple PWM Chopper Submodule Waveforms Showing Chopping Action Only

20.8.3.1 One-Shot Pulse

The width of the first pulse can be programmed to any of 16 possible pulse width values. The width or period of the first pulse is given by:

$T_{1stpulse} = T_{EPWMCLK} \times 8 \times OSHTWTH$

Where $T_{EPWMCLK}$ is the period of the system clock (EPWMCLK) and OSHTWTH is the four control bits (value from 1 to 16)

Figure 20-40 shows the first and subsequent sustaining pulses and Table 20-12 gives the possible pulse width values for a EPWMCLK = 80 MHz.

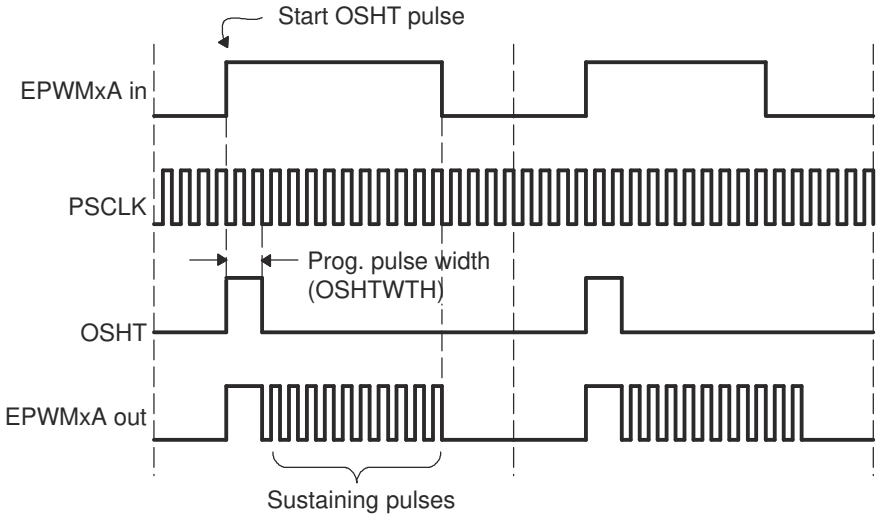


Figure 20-40. PWM Chopper Submodule Waveforms Showing the First Pulse and Subsequent Sustaining Pulses

Table 20-12. Possible Pulse Width Values for EPWMCLK = 80 MHz

OSHTWTHz (hex)	Pulse Width (nS)
0	100
1	200
2	300
3	400
4	500
5	600
6	700
7	800
8	900
9	1000
A	1100
B	1200
C	1300
D	1400
E	1500
F	1600

20.8.3.2 Duty Cycle Control

Pulse transformer-based gate drive designs need to comprehend the magnetic properties or characteristics of the transformer and associated circuitry. Saturation is one such consideration. To assist the gate drive designer, the duty cycles of the second and subsequent pulses have been made programmable. These sustaining pulses ensure the correct drive strength and polarity is maintained on the power switch gate during the on period, and hence a programmable duty cycle allows a design to be tuned or optimized via software control.

Figure 20-41 shows the duty cycle control that is possible by programming the CHPDUTY bits. One of seven possible duty ratios can be selected ranging from 12.5% to 87.5%.

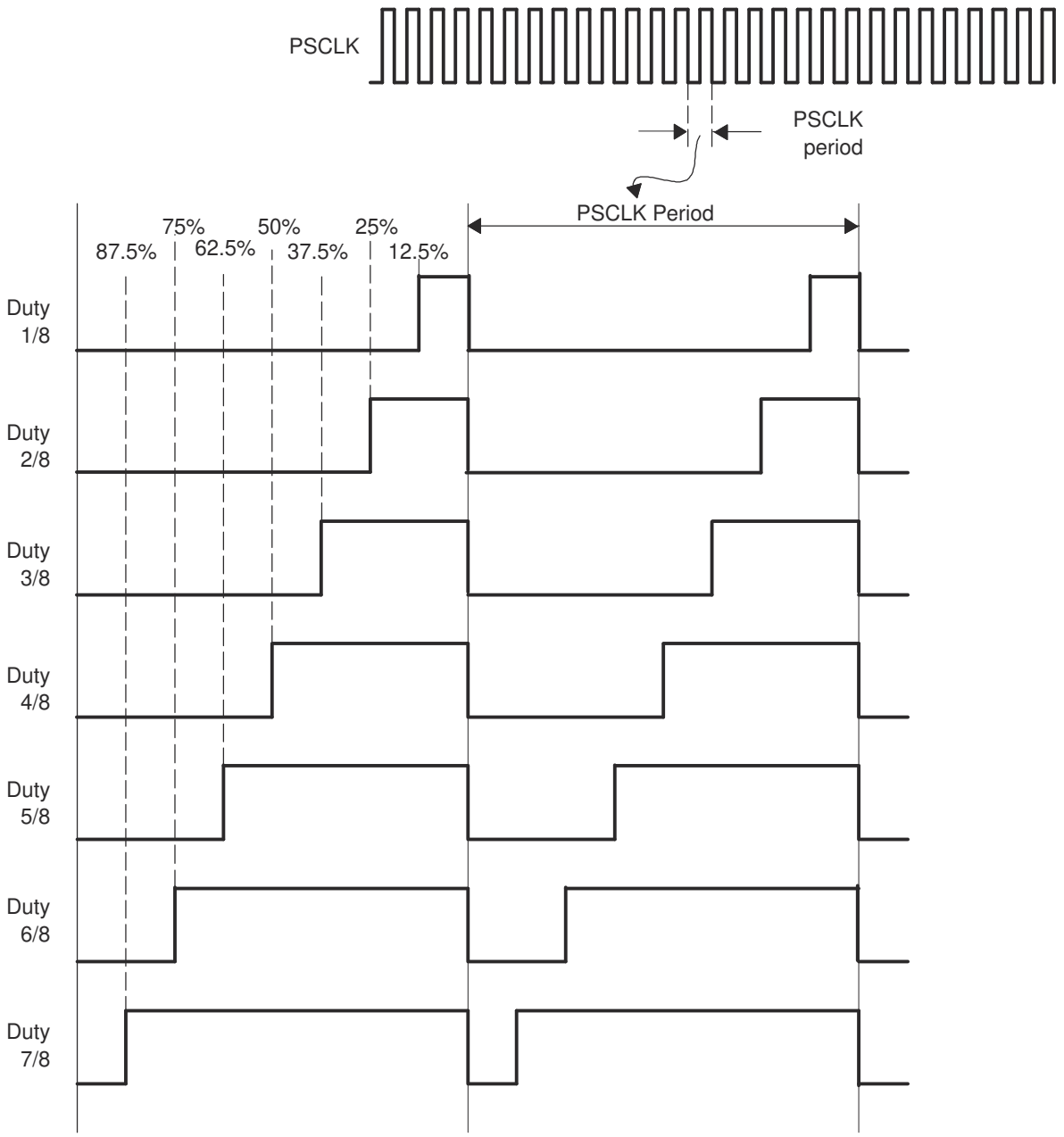


Figure 20-41. PWM Chopper Submodule Waveforms Showing the Pulse Width (Duty Cycle) Control of Sustaining Pulses

20.9 Trip-Zone (TZ) Submodule

Figure 20-42 shows how the trip-zone (TZ) submodule fits within the ePWM module.

Each ePWM module is connected to six $\overline{TZn}$ signals ($\overline{TZ1}$ to $\overline{TZ6}$). $\overline{TZ1}$ to $\overline{TZ3}$ are sourced from the GPIO mux. $\overline{TZ4}$ is sourced from an inverted EQEPxERR signal on those devices with an EQEP module. $\overline{TZ5}$ is connected to the system clock fail logic, and $\overline{TZ6}$ is sourced from the EMUSTOP output from the CPU. These signals indicate external fault or trip conditions, and the ePWM outputs can be programmed to respond accordingly when faults occur.

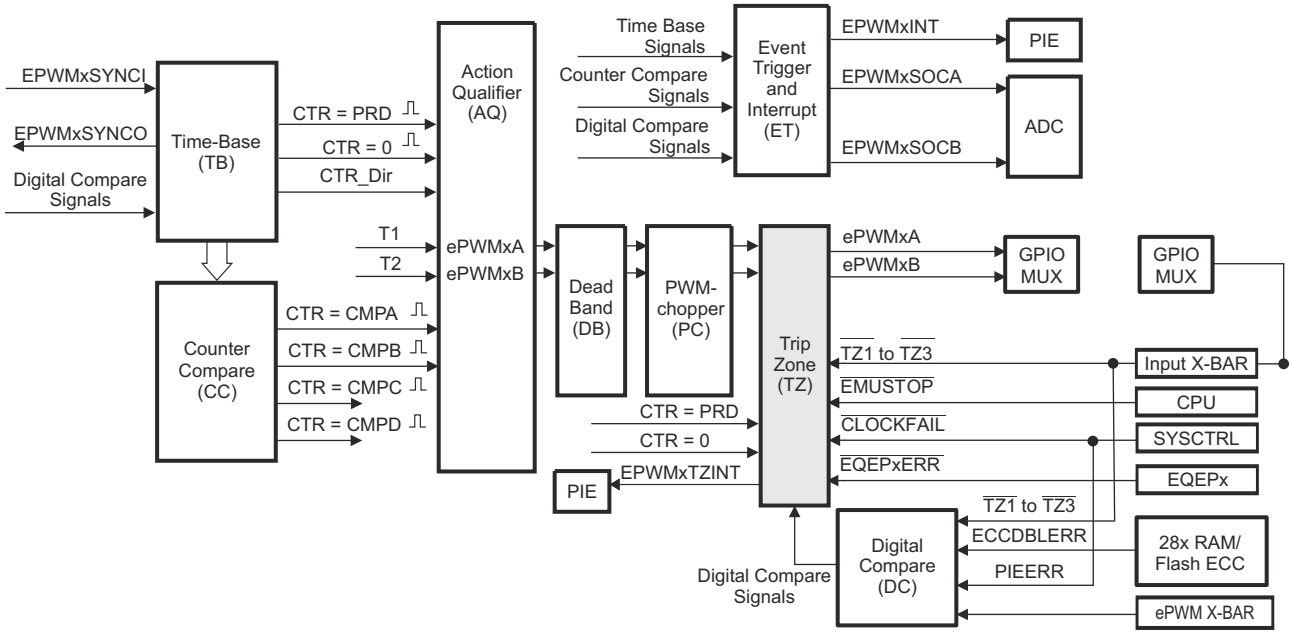


Figure 20-42. Trip-Zone Submodule

20.9.1 Purpose of the Trip-Zone Submodule

The key functions of the trip-zone submodule are:

- Trip inputs $\overline{TZ1}$ to $\overline{TZ6}$ can be flexibly mapped to any ePWM module.
- Upon a fault condition, outputs EPWMxA and EPWMxB can be forced to one of the following:
  - High
  - Low
  - High-impedance
  - No action taken
- Support for one-shot trip (OSHT) for major short circuits or over-current conditions.
- Support for cycle-by-cycle tripping (CBC) for current limiting operation.
- Support for digital compare tripping (DC) based on state of on-chip analog comparator module outputs and/or $\overline{TZ1}$ to $\overline{TZ3}$ signals.
- Each trip-zone input and digital compare (DC) submodule DCAEVT1/2 or DCBEVT1/2 force event can be allocated to either one-shot or cycle-by-cycle operation.
- Interrupt generation is possible on any trip-zone input.
- Software-forced tripping is also supported.
- The trip-zone submodule can be fully bypassed if it is not required.

20.9.2 Operational Highlights for the Trip-Zone Submodule

The following sections describe the operational highlights and configuration options for the trip-zone submodule.

The trip-zone signals $\overline{TZ1}$ to $\overline{TZ6}$ (also collectively referred to as $\overline{TZn}$) are active-low input signals. When one of these signals goes low, or when a DCAEVT1/2 or DCBEVT1/2 force happens based on the TZDCSEL register event selection, it indicates that a trip event has occurred. Each ePWM module can be individually configured to ignore or use each of the trip-zone signals or DC events. Which trip-zone signals or DC events are used by a particular ePWM module is determined by the TZSEL register for that specific ePWM module. The trip-zone signals may or may not be synchronized to the ePWMclock (EPWMCLK) and digitally filtered within the GPIO MUX block. A minimum of 3*TBCLK low pulse width on $\overline{TZn}$ inputs is sufficient to trigger a fault condition on the ePWM module. If the pulse width is less than this, the trip condition may not be latched by CBC or OST latches. The asynchronous trip makes sure that if clocks are missing for any reason, the outputs can still be tripped by a valid event present on $\overline{TZn}$ inputs. The GPIOs or peripherals must be appropriately configured. For more information, see the *System Control and Interrupts* chapter.

Each $\overline{TZn}$ input can be individually configured to provide either a cycle-by-cycle or one-shot trip event for an ePWM module. DCAEVT1 and DCBEVT1 events can be configured to directly trip an ePWM module or provide a one-shot trip event to the module. Likewise, DCAEVT2 and DCBEVT2 events can also be configured to directly trip an ePWM module or provide a cycle-by-cycle trip event to the module. This configuration is determined by the TZSEL[DCAEVT1/2], TZSEL[DCBEVT1/2], TZSEL[CBCn], and TZSEL[OSHTn] control bits (where n corresponds to the trip input), respectively.

- **Cycle-by-Cycle (CBC):**

  When a cycle-by-cycle trip event occurs, the action specified in the TZCTL[TZA] and TZCTL[TZB] bits is carried out immediately on the EPWMxA and/or EPWMxB outputs. Table 20-13 lists some of the possible actions. Independent actions can be specified based on the occurrence of the event while the counter is counting up and/or while it is counting down by appropriately configuring bits in the TZCTL2 register. Actions specified in the TZCTL2 register take effect only when the ETZE bit in TZCTL2 is set.

  Additionally, when a cycle-by-cycle trip event occurs, the cycle-by-cycle trip event flag (TZFLG[CBC]) is set and a EPWMx_TZINT interrupt is generated if it is enabled in the TZEINT register and PIE peripheral. A corresponding flag for the event that caused the CBC event is also set in register TZCBCFLG.

  If the CBC interrupt is enabled via the TZEINT register, and DCAEVT2 or DCBEVT2 are selected as CBC trip sources via the TZSEL register, it is not necessary to also enable the DCAEVT2 or DCBEVT2 interrupts in the TZEINT register, as the DC events trigger interrupts through the CBC mechanism.

  The specified condition on the inputs is automatically cleared based on the selection made with TZCLR[CBCPULSE] if the trip event is no longer present. Therefore, in this mode, the trip event is cleared or reset every PWM cycle. The TZFLG[CBC] and TZCBCFLG flag bits will remain set until they are manually cleared by writing to the TZCLR[CBC] and TZCBCCLR flag bits. If the cycle-by-cycle trip event is still present when the TZFLG[CBC] and/or TZCBCFLG register bits are cleared, then these bits will again be immediately set..
- **One-Shot (OSHT):**

  When a one-shot trip event occurs, the action specified in the TZCTL[TZA] and TZCTL[TZB] bits is carried out immediately on the EPWMxA and/or EPWMxB output. Table 20-13 lists some of the possible actions. Independent actions can be specified based on the occurrence of the event while the counter is counting up and/or while it is counting down by appropriately configuring bits in TZCTL2 register. Actions specified in TZCTL2 register take effect only when ETZE bit in TZCTL2 is set.

  Additionally, when a one-shot trip event occurs, the one-shot trip event flag (TZFLG[OST]) is set and a EPWMx_TZINT interrupt is generated if it is enabled in the TZEINT register and PIE peripheral. A corresponding flag for the event that caused the OST event is also set in register TZOSTFLG. The one-shot trip condition must be cleared manually by writing to the TZCLR[OST] bit. If desired, TZOSTFLG register bit should be cleared by manually writing to the corresponding bit in the TZOSTCLR register.

If the one-shot interrupt is enabled via the TZEINT register, and DCAEVT1 or DCBEVT1 are selected as OSHT trip sources via the TZSEL register, it is not necessary to also enable the DCAEVT1 or DCBEVT1 interrupts in the TZEINT register, as the DC events trigger interrupts through the OSHT mechanism.

Note

Clear the TZFLG and TZOSTFLG flags after ensuring that the TRIPIN source of the OST has become inactive. Otherwise, if interrupts are enabled, depending on when the flags are cleared, an OST interrupt could occur and the OST flags are zero.

- **Digital Compare Events (DCAEVT1/2 and DCBEVT1/2):**

  A digital compare DCAEVT1/2 or DCBEVT1/2 event is generated based on a combination of the DCAH/DCAL and DCBH/DCBL signals as selected by the TZDCSEL register. The signals which source the DCAH/DCAL and DCBH/DCBL signals are selected via the DCTRIPSEL register and can be either trip zone input pins or analog comparator CMPSSx signals. For more information on the digital compare submodule signals, see Section 20.11.

  When a digital compare event occurs, the action specified in the TZCTL[DCAEVT1/2] and TZCTL[DCBEVT1/2] bits is carried out immediately on the EPWMxA and/or EPWMxB output. Table 20-13 lists the possible actions. Independent actions can be specified based on the occurrence of the event while the counter is counting up and/or while it is counting down by appropriately configuring bits in TZCTLDCA and TZCTLDCB register. Actions specified in TZCTLDCA and TZCTLDCB registers take effect only when ETZE bit in TZCTL2 is set.

  In addition, the relevant DC trip event flag (TZFLG[DCAEVT1/2] / TZFLG[DCBEVT1/2]) is set and a EPWMx_TZINT interrupt is generated if it is enabled in the TZEINT register and PIE peripheral.

  The specified condition on the pins is automatically cleared when the DC trip event is no longer present. The TZFLG[DCAEVT1/2] or TZFLG[DCBEVT1/2] flag bit will remain set until it is manually cleared by writing to the TZCLR[DCAEVT1/2] or TZCLR[DCBEVT1/2] bit. If the DC trip event is still present when the TZFLG[DCAEVT1/2] or TZFLG[DCBEVT1/2] flag is cleared, then it will again be immediately set.

The action taken when a trip event occurs can be configured individually for each of the ePWM output pins by way of the TZCTL, TZCTL2, TZCTLDCA,, and TZCTLDCB register bit fields. Some of the possible actions, shown in Table 20-13, can be taken on a trip event.

Table 20-13. Possible Actions On a Trip Event

TZCTL Register Bitfield Settings	EPWMxA and/or EPWMxB	Comment
0,0	High-Impedance	Tripped
0,1	Force to High State	Tripped
1,0	Force to Low State	Tripped
1,1	No Change	Do Nothing. No change is made to the output.

Example 20-1. Trip-Zone Configurations

Scenario A:
A one-shot trip event on $\overline{TZ1}$ pulls both EPWM1A, EPWM1B low and also forces EPWM2A and EPWM2B high.

- Configure the ePWM1 registers as follows:
  - TZSEL[OSHT1] = 1: enables $\overline{TZ1}$ as a one-shot event source for ePWM1
  - TZCTL[TZA] = 2: EPWM1A will be forced low on a trip event.
  - TZCTL[TZB] = 2: EPWM1B will be forced low on a trip event.
- Configure the ePWM2 registers as follows:
  - TZSEL[OSHT1] = 1: enables $\overline{TZ1}$ as a one-shot event source for ePWM2
  - TZCTL[TZA] = 1: EPWM2A will be forced high on a trip event.
  - TZCTL[TZB] = 1: EPWM2B will be forced high on a trip event.

Scenario B:
A cycle-by-cycle event on $\overline{TZ5}$ pulls both EPWM1A, EPWM1B low.
A one-shot event on $\overline{TZ1}$ or $\overline{TZ6}$ puts EPWM2A into a high impedance state.

- Configure the ePWM1 registers as follows:
  - TZSEL[CBC5] = 1: enables $\overline{TZ5}$ as a one-shot event source for ePWM1
  - TZCTL[TZA] = 2: EPWM1A will be forced low on a trip event.
  - TZCTL[TZB] = 2: EPWM1B will be forced low on a trip event.
- Configure the ePWM2 registers as follows:
  - TZSEL[OSHT1] = 1: enables $\overline{TZ1}$ as a one-shot event source for ePWM2
  - TZSEL[OSHT6] = 1: enables $\overline{TZ6}$ as a one-shot event source for ePWM2
  - TZCTL[TZA] = 0: EPWM2A will be put into a high-impedance state on a trip event.
  - TZCTL[TZB] = 3: EPWM2B will ignore the trip event.

Note

When configuring the GPIOs and INPUT X-BAR/EPWM X-BAR options, be aware that a change in the X-BAR input selections may cause an unwanted event. Therefore, ideally, the user will set up the GPIO and X-BAR input configurations before enabling the ePWM Trip-Zone. If it is a requirement to change the GPIO/X-BAR configurations while the ePWM Trip-Zone is enabled, the user can turn off the TRIPs by clearing the TZSEL register and re-configuring the TRIP selection (TZSEL) after the INPUT XBAR selection is changed.

20.9.3 Generating Trip Event Interrupts

Figure 20-43 and Figure 20-44 illustrate the trip-zone submodule control and interrupt logic, respectively. DCAEVT1/2 and DCBEVT1/2 signals are described in further detail in Section 20.11.

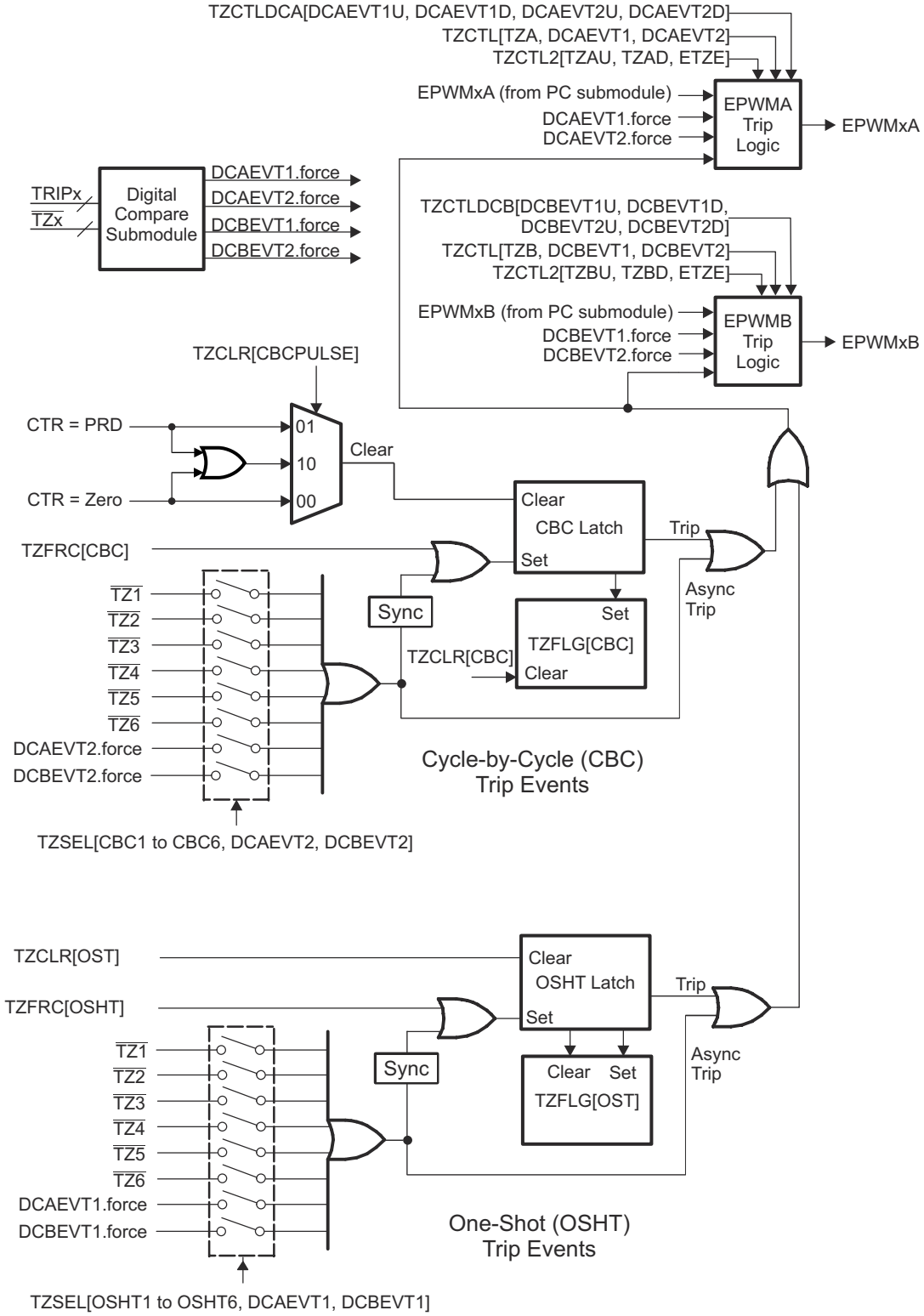


Figure 20-43. Trip-Zone Submodule Mode Control Logic

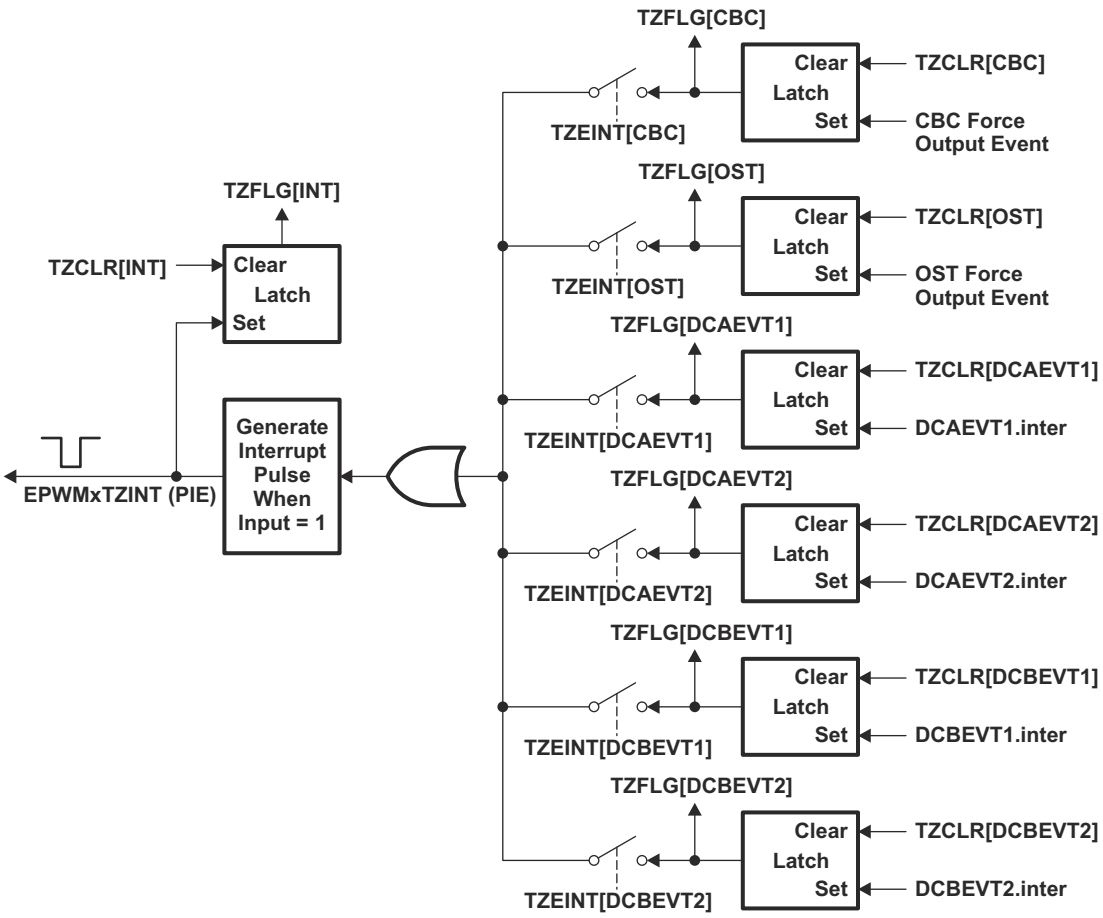


Figure 20-44. Trip-Zone Submodule Interrupt Logic

These individual flags for the CBC, OST and DCxEVTy can be used to detect the source of the EPWMxTZINT Interrupt. When multiple sources are used to generate the EPWMxTZINT interrupt, reading and clearing the flags will user to take different actions based on the specific event.

20.10 Event-Trigger (ET) Submodule

The key functions of the event-trigger submodule are:

- Receives event inputs generated by the time-base, counter-compare, and digital-compare submodules
- Uses the time-base direction information for up/down event qualification
- Uses prescaling logic to issue interrupt requests and ADC start of conversion at:
  - Every event
  - Every second event
  - Up to every fifteenth event
- Provides full visibility of event generation via event counters and flags
- Allows software forcing of Interrupts and ADC start of conversion

The event-trigger submodule manages the events generated by the time-base submodule, the counter-compare submodule, and the digital-compare submodule to generate an interrupt to the CPU and/or a start of conversion pulse to the ADC when a selected event occurs.

Figure 20-45 illustrates where the event-trigger submodule fits within the ePWM system.

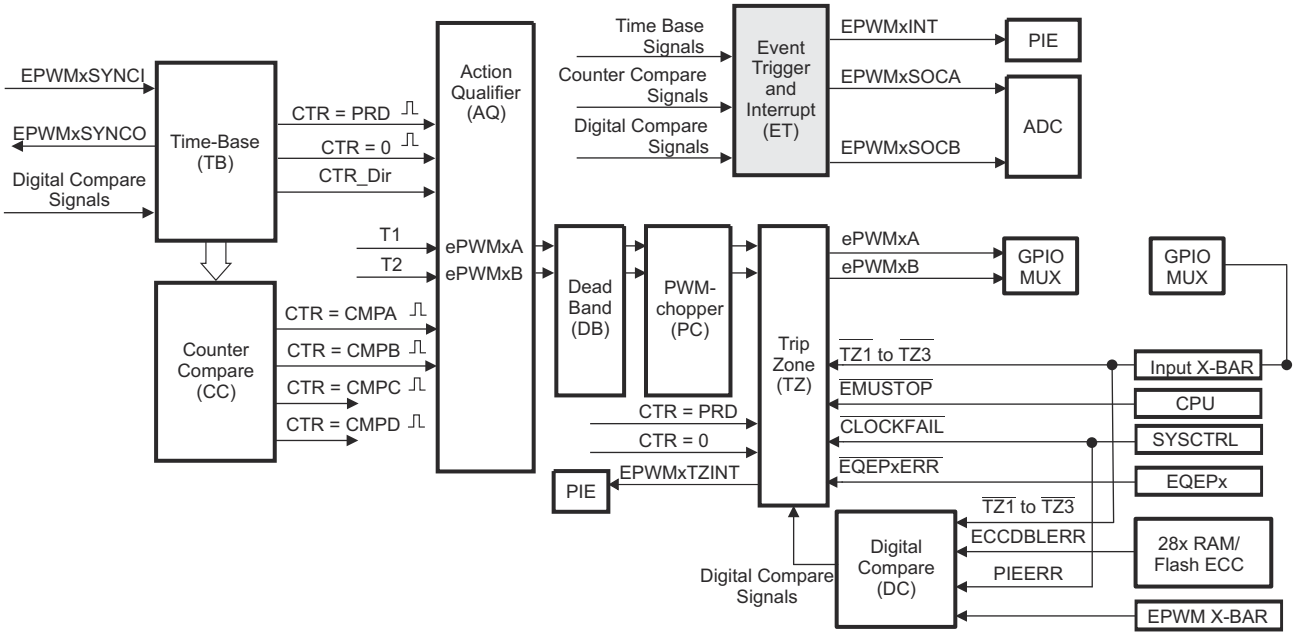


Figure 20-45. Event-Trigger Submodule

20.10.1 Operational Overview of the ePWM Event-Trigger Submodule

The event-trigger submodule monitors various event conditions (shown as inputs on the left side of Figure 20-46) and can be configured to prescale these events before issuing an Interrupt request or an ADC start of conversion. The event-trigger prescaling logic can issue Interrupt requests and ADC start of conversion at:

- Every event
- Every second event
- Up to every fifteenth event

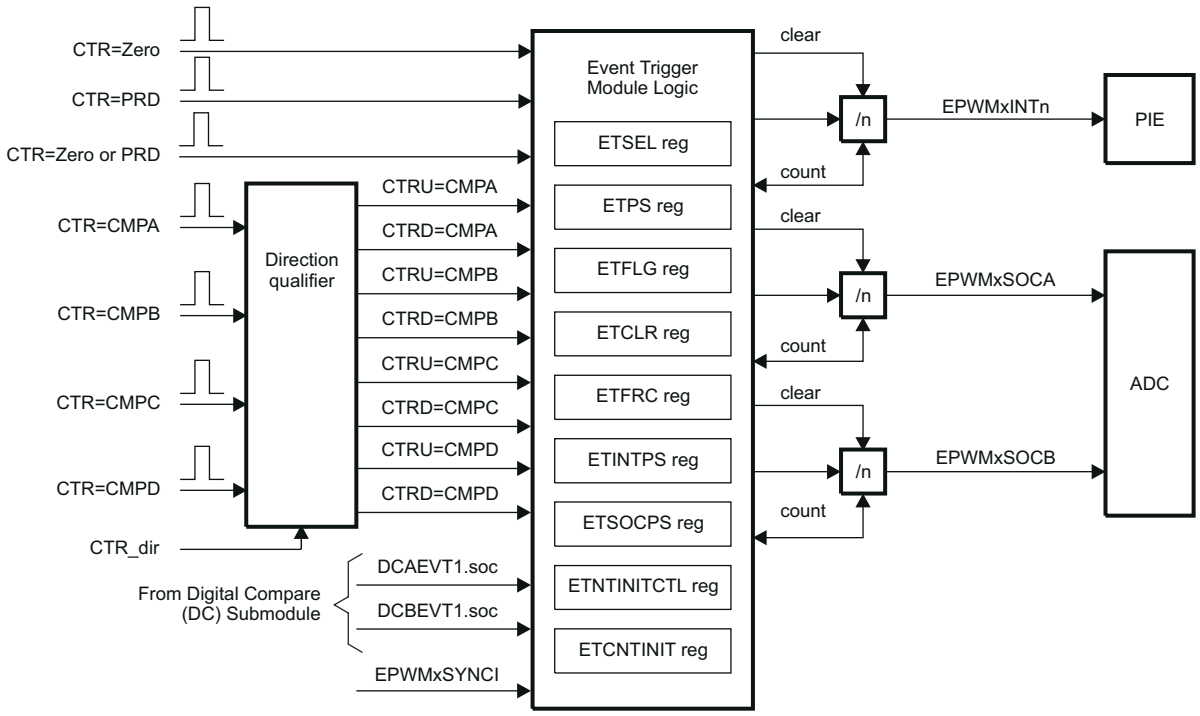


Figure 20-46. Event-Trigger Submodule Showing Event Inputs and Prescaled Outputs

- ETSEL - This selects which of the possible events will trigger an interrupt or start an ADC conversion.
- ETPS - This programs the event prescaling options mentioned above.
- ETFLG - These are flag bits indicating status of the selected and prescaled events.
- ETCLR - These bits allow you to clear the flag bits in the ETFLG register via software.
- ETFRC - These bits allow software forcing of an event. Useful for debugging or software intervention.
- ETINTPS - This programs the interrupt event prescaling options, supporting count and period up to 15 events.
- ETSOCPS - This programs the SOC event prescaling options, supporting count and period up to 15 events.
- ETCNTINITCTL - These bits enable ETCNTINIT initialization via SYNC event OR via software force.
- ETCNTINIT - These bits allow you to initialize INT/SOCA/SOCB counters on SYNC events (or software force) with user programmed value.

A more detailed look at how the various register bits interact with the Interrupt and ADC start of conversion logic are shown in Figure 20-47, Figure 20-48, and Figure 20-49.

Figure 20-47 shows the event-trigger's interrupt generation logic. The interrupt-period (ETPS[INTPRD]) bits specify the number of events required to cause an interrupt pulse to be generated. The choices available are:

- Do not generate an interrupt.
- Generate an interrupt on every event.
- Generate an interrupt on every second event.
- Generate an interrupt on every third event.

On ePWM type 4, in order to enable event generation capability up to 15 events the following changes have been made. The selection made on ETPS[INTPSSEL] bit determines whether ETINTPS register, INTCNT2 and INTPRD2 bit fields determine frequency of events (interrupt once every 0-15 events).

The event that can cause an interrupt is configured by the interrupt selection (ETSEL[INTSEL]) and (ETSEL[INTSELCMP]) bits. The event can be one of the following:

- Time-base counter equal to zero (TBCTR = 0x00).
- Time-base counter equal to period (TBCTR = TBPRD).
- Time-base counter equal to zero or period (TBCTR = 0x00 || TBCTR = TBPRD).
- Time-base counter equal to the compare A register (CMPA) when the timer is incrementing.
- Time-base counter equal to the compare A register (CMPA) when the timer is decrementing.
- Time-base counter equal to the compare B register (CMPB) when the timer is incrementing.
- Time-base counter equal to the compare B register (CMPB) when the timer is decrementing.
- Time-base counter equal to the compare C register (CMPC) when the timer is incrementing.
- Time-base counter equal to the compare C register (CMPC) when the timer is decrementing.
- Time-base counter equal to the compare D register (CMPD) when the timer is incrementing.
- Time-base counter equal to the compare D register (CMPD) when the timer is decrementing.

The number of events that have occurred can be read from the interrupt event counter ETPS[INTCNT] or ETINTPS[INTCNT2] register bits based off of the selection made using ETPS[INTPSSEL]. That is, when the specified event occurs the ETPS[INTCNT] or ETINTPS[INTCNT2] bits are incremented until they reach the value specified by ETPS[INTPRD] or ETINTPS[INTPRD2] determined again by the selection made in ETPS[INTPSSEL]. When ETPS[INTCNT] = ETPS[INTPRD] the counter stops counting and its output is set. The counter is only cleared when an interrupt is sent to the PIE.

When ETPS[INTCNT] reaches ETPS[INTPRD] the following behavior will occur [The below behavior is also applicable to ETINTPS[INTCNT2] and ETINTPS[INTPRD2]:

- If interrupts are enabled, ETSEL[INTEN] = 1 and the interrupt flag is clear, ETFLG[INT] = 0, then an interrupt pulse is generated and the interrupt flag is set, ETFLG[INT] = 1, and the event counter is cleared ETPS[INTCNT] = 0. The counter will begin counting events again.
- If interrupts are disabled, ETSEL[INTEN] = 0, or the interrupt flag is set, ETFLG[INT] = 1, the counter stops counting events when it reaches the period value ETPS[INTCNT] = ETPS[INTPRD].
- If interrupts are enabled, but the interrupt flag is already set, then the counter will hold its output high until the ENTFLG[INT] flag is cleared. This allows for one interrupt to be pending while one is serviced.

Writing '0' to the INTPRD bits will automatically clear the counter INTCNT = 0 and the counter output will be reset (so no interrupts are generated). Writing a 1 to the ETFRC[INT] bit will increment the event counter INTCNT. The counter will behave as described above when INTCNT = INTPRD. When INTPRD = 0, the counter is disabled and hence no events will be detected and the ETFRC[INT] bit is also ignored. The same applies to ETINTPS[INTCNT2] and ETINTPS[INTPRD2]

The above definition means that you can generate an interrupt on every event, on every second event, or on every third event if using the INTCNT and INTPRD. You can generate an interrupt on every event up to 15 events if using the INTCNT2 and INTPRD2.

The INTCNT2 value can be initialized with the value from ETCNTINIT[INTINIT] based on the selection made in ETCNTINITCTL[INTINITEN]. When ETCNTINITCTL[INTINITEN] is set, then it enables initialization of INTCNT2 counter with contents of ETCNTINIT[INTINIT] on a SYNC event or software force determined by ETCNTINITCTL[INTINITFRC].

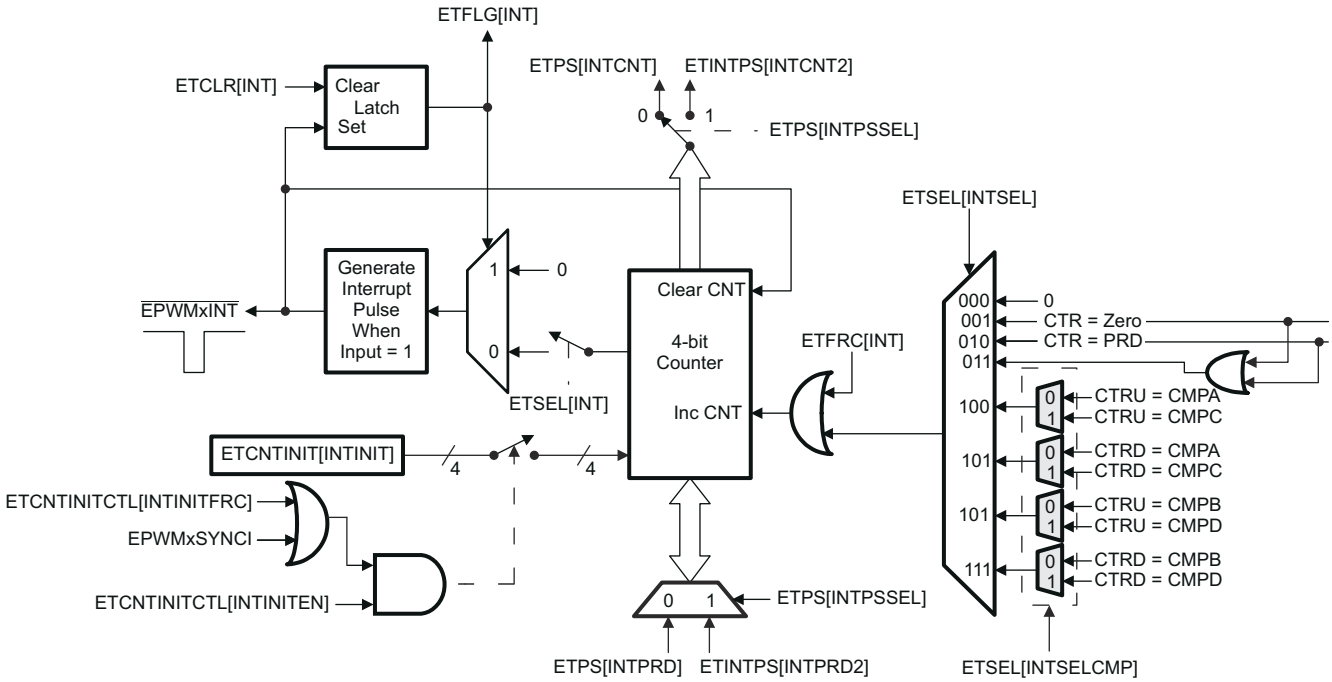
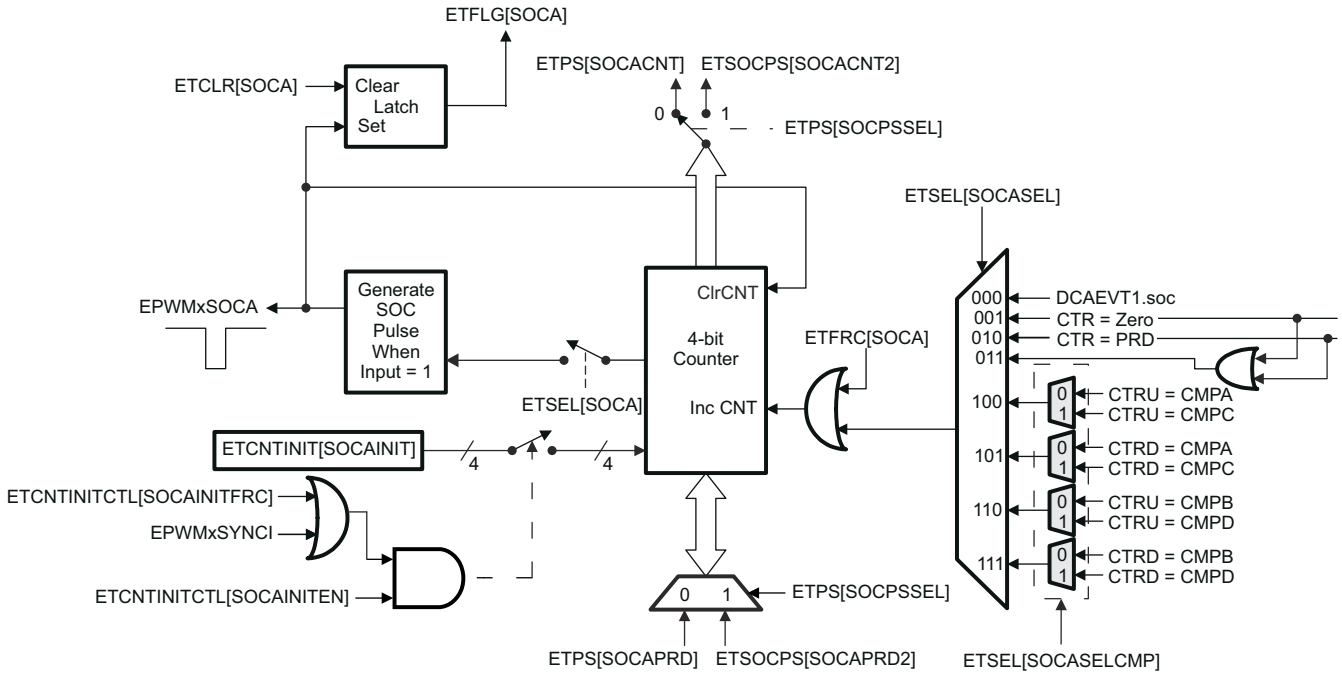


Figure 20-47. Event-Trigger Interrupt Generator

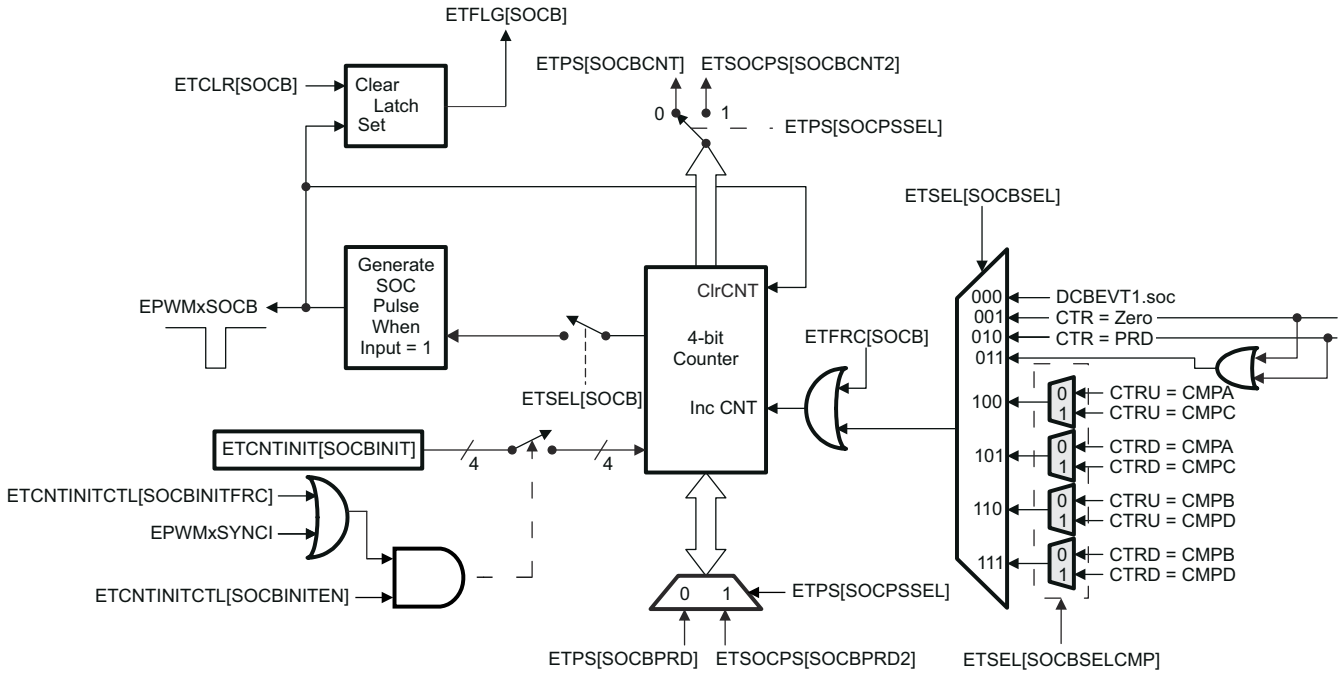
Figure 20-48 shows the operation of the event-trigger's start-of-conversion-A (SOCA) pulse generator. The enhancements include SOCASELCMP and SOCBSELCMP bit fields defined in the ETSEL register enable CMPC and CMPD events respectively to cause a start of conversion. The ETPS[SOCPSSEL] bit field determines whether SOCACNT2 and SOCAPRD2 take control or not. The ETPS[SOCACNT] counter and ETPS[SOCAPRD] period values behave similarly to the interrupt generator except that the pulses are continuously generated. That is, the pulse flag ETFLG[SOCA] is latched when a pulse is generated, but it does not stop further pulse generation. The enable/disable bit ETSEL[SOCAEN] stops pulse generation, but input events can still be counted until the period value is reached as with the interrupt generation logic. The event that will trigger an SOCA and SOCB pulse can be configured separately in the ETSEL[SOCASEL] and ETSEL[SOCBSEL] bits. The possible events are the same events that can be specified for the interrupt generation logic with the addition of the DCAEVT1.soc and DCBEVT1.soc event signals from the digital compare (DC) submodule. The SOCACNT2 initialization scheme is very similar to the interrupt generator with respective enable, value initialize and SYNC or software force options.



NOTE: The DCAEVT1.soc signals are generated by the Digital Compare (DC) submodule in Section 20.11.

Figure 20-48. Event-Trigger SOCA Pulse Generator

Figure 20-49 shows the operation of the event-trigger's start-of-conversion-B (SOCB) pulse generator. The event-trigger's SOCB pulse generator operates the same way as the SOCA.



NOTE: The DCBEVT1.soc signals are generated by the Digital Compare (DC) submodule in Section 20.11.

Figure 20-49. Event-Trigger SOCB Pulse Generator

20.11 Digital Compare (DC) Submodule

Figure 20-50 illustrates where the digital compare (DC) submodule signals interface to other submodules in the ePWM system.

The eCAP input signals are sourced from the Input X-BAR signals as shown in Figure 20-51.

On this device, any of the GPIO pins can be flexibly mapped to be the trip-zone input and/or trip inputs to the trip-zone submodule and digital compare submodule. The Input X-BAR Input Select (INPUTxSELECT) register defines which GPIO pins gets assigned to be the trip-zone inputs / trip inputs.

The digital compare (DC) submodule compares signals external to the ePWM module (for instance, CMPSSx signals from the analog comparators) to directly generate PWM events/actions which then feed to the event-trigger, trip-zone, and time-base submodules. Additionally, blanking window functionality is supported to filter noise or unwanted pulses from the DC event signals.

Note

The user is responsible for driving correct state on the selected pin before enabling clock and configuring the trip input for the respective ePWM peripheral to avoid spurious latch of TRIP signal.

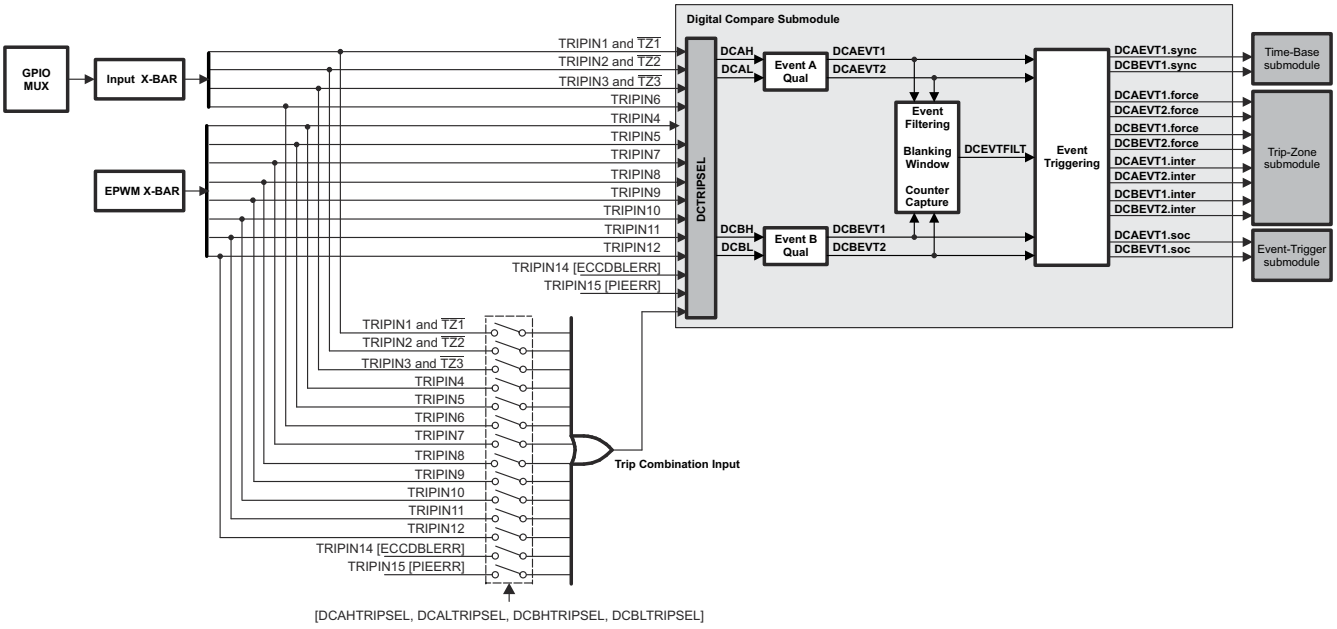


Figure 20-50. Digital-Compare Submodule High-Level Block Diagram

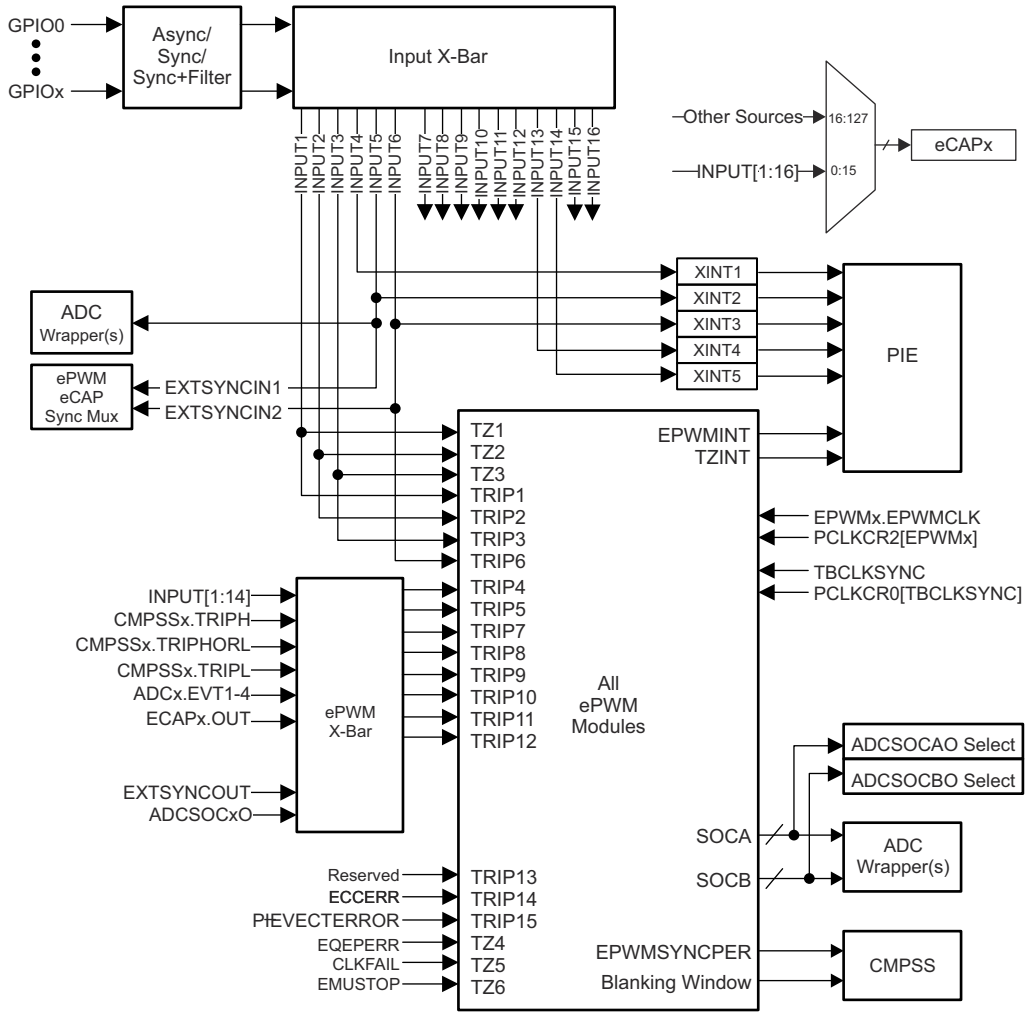


Figure 20-51. GPIO MUX-to-Trip Input Connectivity

20.11.1 Purpose of the Digital Compare Submodule

The key functions of the digital compare submodule are:

- Analog comparator (COMP) module outputs fed though the Input X-BAR logic externally using the GPIO peripheral, internal PIE, ECC error signals, TZ1, $\overline{TZ2}$, and $\overline{TZ3}$ inputs generate Digital Compare A High/Low (DCAH, DCAL) and Digital Compare B High/Low (DCBH, DCBL) signals.
- DCAH/L and DCBH/L signals trigger events which can then either be filtered or fed directly to the trip-zone, event-trigger, and time-base submodules to:
  - generate a trip zone interrupt
  - generate an ADC start of conversion
  - force an event
  - generate a synchronization event for synchronizing the ePWM module TBCTR.
- Event filtering (blanking window logic) can optionally blank the input signal to remove noise.

20.11.2 Enhanced Trip Action Using CMPSS

In order to allow multiple CMPSS at a time to affect DCA/BEVTx events and trip actions, there is a OR logic to bring together ALL trip inputs (up to 15) from sources external to the ePWM module and feed into DCAH, DCAL, DCBH, and DCBL as a “combinational input” using the DCTRIPSEL register. This is configured by selecting “Trip combination input” (value of 0xF) in the DCTRIPSEL register.

The user has a discrete choice of which trip inputs to put through the combinational logic for generating the DCAH, DCAL, DCBH and DCBL signals. This is achieved using the DCAHTRIPSEL, DCALTRIPSEL, DCBHTRIPSEL and DCBLTRIPSEL register selections. Inputs selected for combinational input are passed through to the DCTRIPSEL register.

20.11.3 Using CMPSS to Trip the ePWM on a Cycle-by-Cycle Basis

When using the CMPSS to trip the ePWM on a cycle-by-cycle basis, steps should be taken to prevent an asserted comparator trip state in one PWM cycle from extending into the following cycle. The CMPSS can be used to signal a trip condition to the downstream ePWM modules. For applications like peak current mode control, only one trip event per PWM cycle is expected. Under certain conditions, it is possible for a sustained or late trip event (arriving near the end of a PWM cycle) to carry over into the next PWM cycle if precautions are not taken. If either the CMPSS Digital Filter or the ePWM Digital Compare (DC) submodule is configured to qualify the comparator trip signal, “N” number of clock cycles of qualification will be introduced before the ePWM trip logic can respond to logic changes of the trip signal. Once an ePWM trip condition is qualified, the trip condition will remain active for N clock cycles after the comparator trip signal has de-asserted. If a qualified comparator trip signal remains asserted within N clock cycles prior to the end of a PWM cycle, the trip condition will not be cleared until after the following PWM cycle has started. Thus, the new PWM cycle will detect a trip condition as soon as it begins.

To avoid this undesired trip condition, the user application should take steps to ensure that the qualified trip signal seen by the ePWM trip logic is deasserted prior to the end of each PWM cycle. This can be accomplished through various methods:

- Design the system such that a comparator trip will not be asserted within N clock cycles prior to the end of the PWM cycle.
- Activate blanking of the comparator trip signal via the ePWM event filter at least two clock cycles prior to the PWMSYNCPER signal and continue blanking for at least N clock cycles into the next PWM cycle.
- If the CMPSS COMPxLATCH path is used, clear the COMPxLATCH at least N clock cycles prior to the end of the PWM cycle. The latch can be cleared by software (via COMPSTSCLR) or by generating an early PWMSYNCPER signal. The ePWM modules on this device include the ability to generate PWMSYNCPER upon a CMPC or CMPD match (via HRPCTL) for arbitrary PWMSYNCPER placement within the PWM cycle.

20.11.4 Operation Highlights of the Digital Compare Submodule

The following sections describe the operational highlights and configuration options for the digital compare submodule.

20.11.4.1 Digital Compare Events

As described in Section 20.11.1, trip zone inputs ($\overline{TZ1}$, $\overline{TZ2}$, and $\overline{TZ3}$) and CMPSSx signals from the analog comparator (COMP) module can be selected via the DCTRIPSEL bits to generate the Digital Compare A High and Low (DCAH/L) and Digital Compare B High and Low (DCBH/L) signals. Then, the configuration of the TZDCSEL register qualifies the actions on the selected DCAH/L and DCBH/L signals, which generate the DCAEVT1/2 and DCBEVT1/2 events (Event Qualification A and B).

Note

The $\overline{TZn}$ signals, when used as a DCEVT tripping functions, are treated as a normal input signal and can be defined to be active high or active low inputs. ePWM outputs are asynchronously tripped when either the $\overline{TZn}$, DCAEVTx.force, or DCBEVTx.force signals are active. For the condition to remain latched, a minimum of 3*TBCLK sync pulse width is required. If pulse width is < 3*TBCLK sync pulse width, the trip condition may or may not get latched by CBC or OST latches.

The DCAEVT1/2 and DCBEVT1/2 events can then be filtered to provide a filtered version of the event signals (DCEVTFILT) or the filtering can be bypassed. Filtering is discussed further in Section 20.11.4.2. Either the DCAEVT1/2 and DCBEVT1/2 event signals or the filtered DCEVTFILT event signals can generate a force to the trip zone module, a TZ interrupt, an ADC SOC, or a PWM sync signal.

- **force signal:**

  DCAEVT1/2.force signals force trip zone conditions which either directly influence the output on the EPWMxA pin (via TZCTL, TZCTLDCA, TZCTLDCB register configurations) or, if the DCAEVT1/2 signals are selected as one-shot or cycle-by-cycle trip sources (via the TZSEL register), the DCAEVT1/2.force signals can effect the trip action via the TZCTL or TZCTL2 register configurations. The DCBEVT1/2.force signals behaves similarly, but affect the EPWMxB output pin instead of the EPWMxA output pin.

  The priority of conflicting actions on the TZCTL, TZCTL2, TZCTLDCA and TZCTLDCB registers is as follows (highest priority overrides lower priority):

  Output EPWMxA:
  – TZA (highest) -> DCAEVT1 -> DCAEVT2 (lowest)
  – TZAU (highest) -> DCAEVT1U -> DCAEVT2U (lowest)
  – TZAD (highest) -> DCAEVT1D -> DCAEVT2D (lowest)

  Output EPWMxB:
  – TZB (highest) -> DCBEVT1 -> DCBEVT2 (lowest)
  – TZBU (highest) -> DCBEVT1U -> DCBEVT2U (lowest)
  – TZBD (highest) -> DCBEVT1D -> DCBEVT2D (lowest)
- **interrupt signal:**

  DCAEVT1/2.interrupt signals generate trip zone interrupts to the PIE. To enable the interrupt, the user must set the DCAEVT1, DCAEVT2, DCBEVT1, or DCBEVT2 bits in the TZEINT register. Once one of these events occurs, an EPWMxTZINT interrupt is triggered, and the corresponding bit in the TZCLR register must be set in order to clear the interrupt.
- **soc signal:**

  The DCAEVT1.soc signal interfaces with the event-trigger submodule and can be selected as an event which generates an ADC start-of-conversion-A (SOCA) pulse via the ETSEL[SOCASEL] bit. Likewise, the DCBEVT1.soc signal can be selected as an event which generates an ADC start-of-conversion-B (SOCB) pulse via the ETSEL[SOCBSEL] bit.
- **sync signal:**

  The DCAEVT1.sync and DCBEVT1.sync events are ORed with the EPWMxSYNCI input signal and the TBCTL[SWFSYNC] signal to generate a synchronization pulse to the time-base counter.

Figure 20-52 and Figure 20-53 show how the DCxEVT1, DCxEVT2, or DCEVTFLT signals are processed to generate the digital compare A and B event force, interrupt, soc and sync signals.

In some of the applications like Phase Shifted Full Bridge (PSFB) Converters, it is required that different actions are taken on a CBC trip event and an OST trip event. This can be achieved using the DCxEVT1LAT.

- This latch could be cleared on CNT=0, CTR=PRD, and CNT=0 OR CTR=PRD events based on the setting of DCxCTL.EVTy.LATCLRSEL setting. This is similar to CBC latch clear mechanism.
- DCxEVTy.force signal could be chosen to be either the latched version or the “un-latched” version based on DCxCTL.EVTyLATSEL value.

- The status of DCxEVTyLAT signal can be accessed by reading DCxCTL.EVTyLAT field.

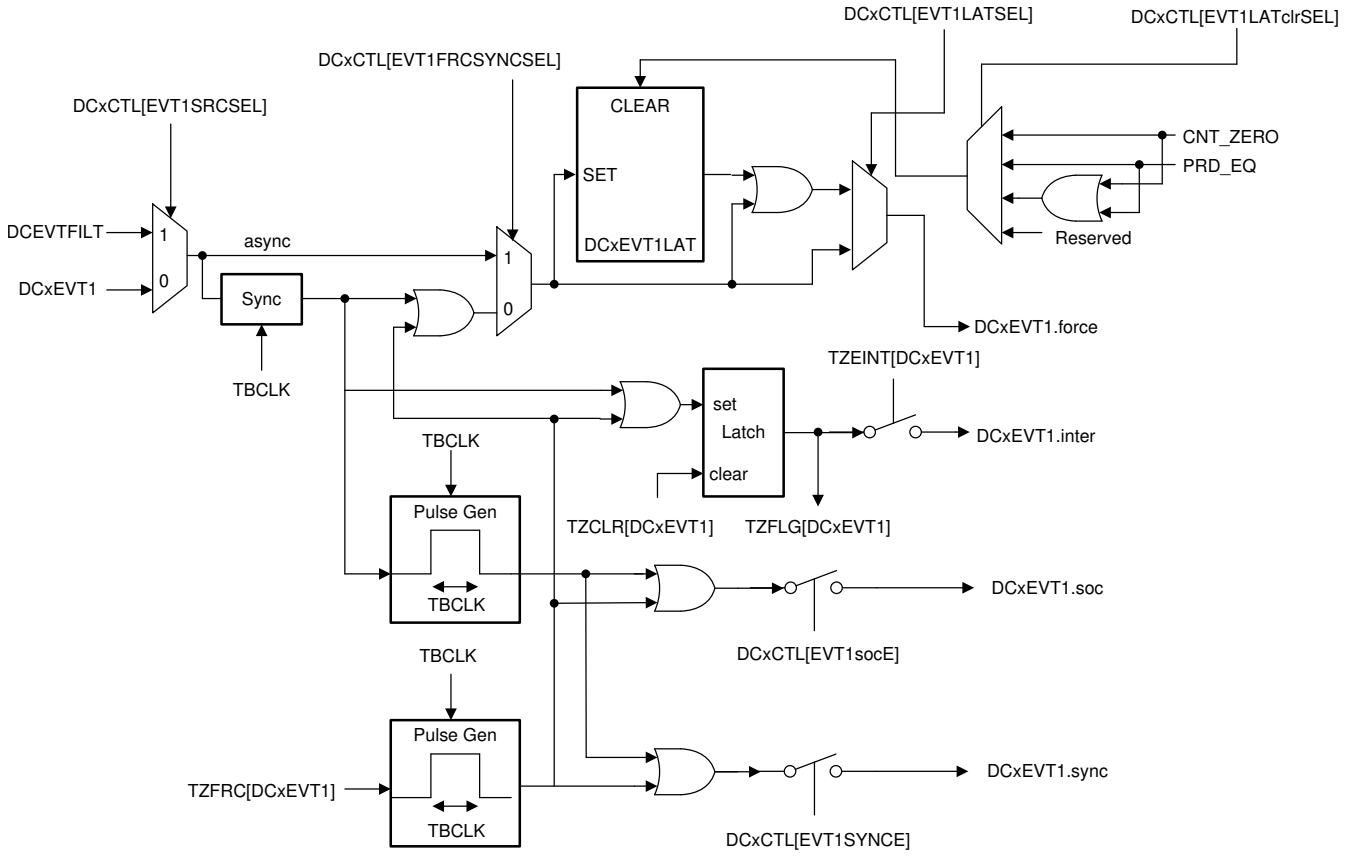


Figure 20-52. DCxEVT1 Event Triggering

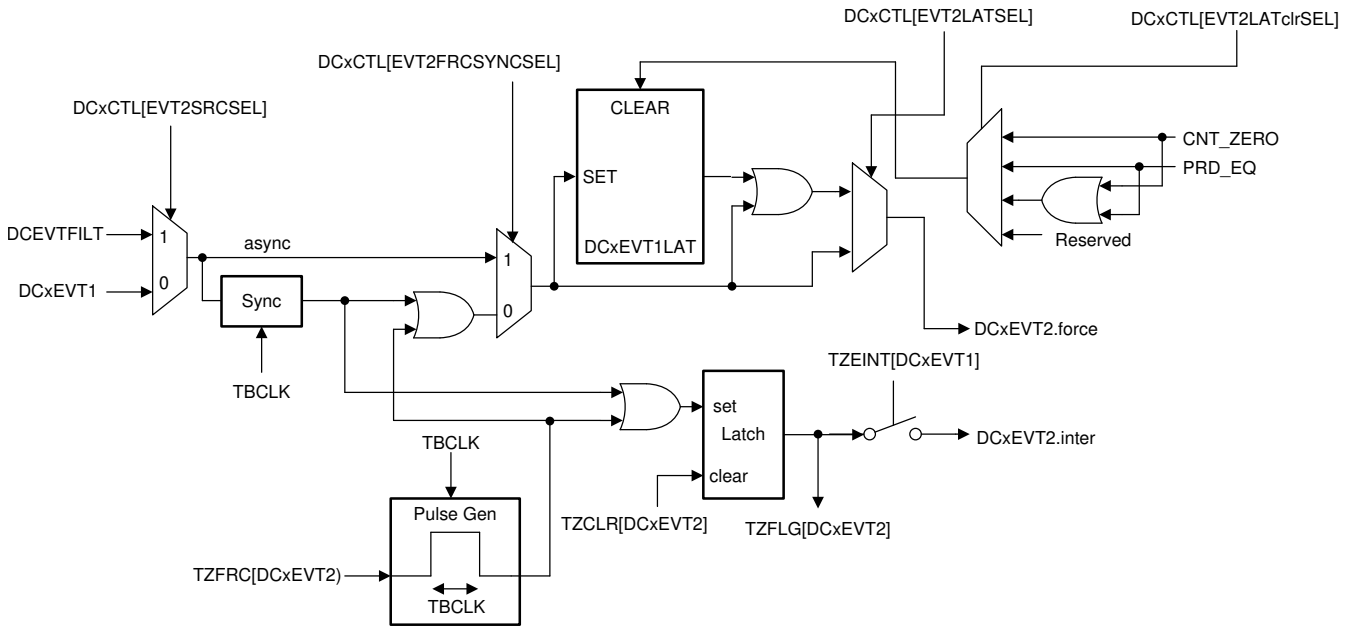


Figure 20-53. DCxEVT2 Event Triggering

20.11.4.2 Event Filtering

The DCAEVT1/2 and DCBEVT1/2 events can be filtered via event filtering logic to remove noise by optionally blanking events for a certain period of time. This is useful for cases where the analog comparator outputs may be selected to trigger DCAEVT1/2 and DCBEVT1/2 events, and the blanking logic is used to filter out potential noise on the signal prior to tripping the PWM outputs or generating an interrupt or ADC start-of-conversion. The event filtering can also capture the TBCTR value of the trip event. Figure 20-54 shows the details of the event filtering logic.

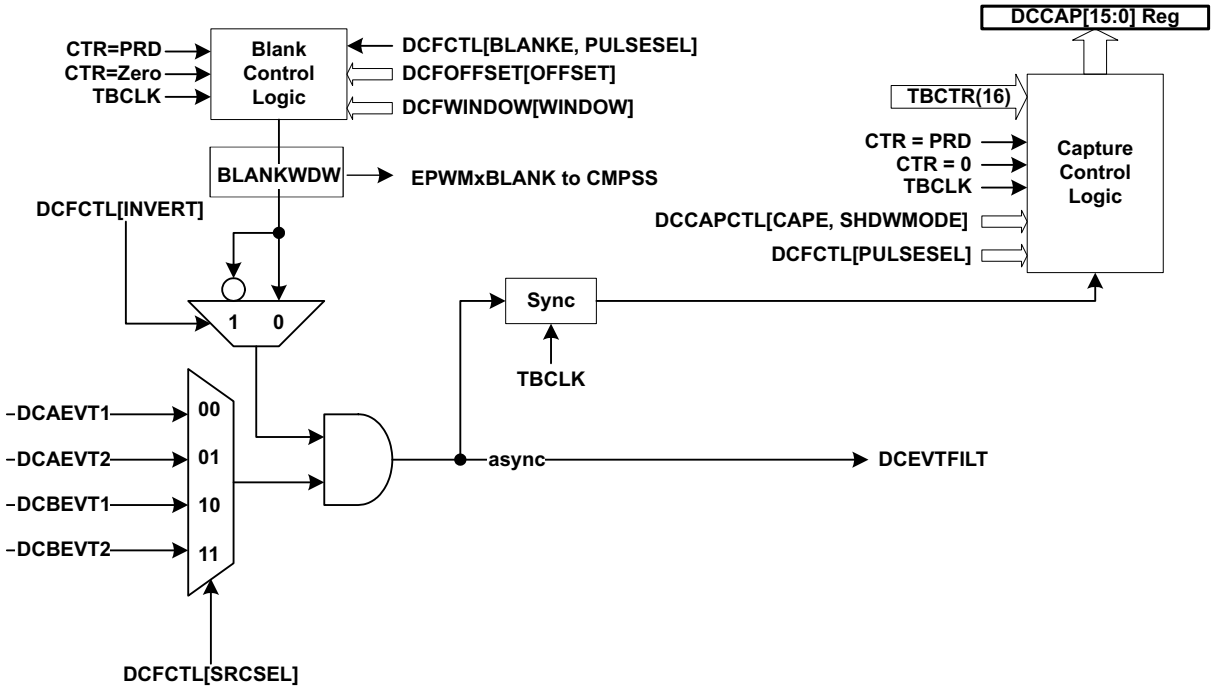


Figure 20-54. Event Filtering

If the blanking logic is enabled, one of the digital compare events – DCAEVT1, DCAEVT2, DCBEVT1, DCBEVT2 – is selected for filtering. The blanking window, which filters out all event occurrences on the signal while it is active, will be aligned to either a CTR = PRD pulse or a CTR = 0 pulse or both CTR = PRD and CTR = 0 (configured by the DCFCTL[PULSESEL] bits). An offset value in TBCLK counts is programmed into the DCFOFFSET register, which determines at what point after the CTR = PRD or CTR = 0 pulse the blanking window starts. The duration of the blanking window, in number of TBCLK counts after the offset counter expires, is written to the DCFWINDOW register by the application. During the blanking window, all events are ignored. Before and after the blanking window ends, events can generate soc, sync, interrupt, and force signals as before.

Note

You must configure the ePWM blanking window appropriately so that the Trip Input stays valid for at least 3 ePWM cycles after the blanking window has expired.

BLANKPULSEMIX Signals

The DCFCTL MUX (available for Blank Control Logic and Capture Control Logic) has new options which allows it to select the BLANKPULSEMIX signal respectively. The BLANKPULSEMIX signal is used if it is selected by DCFCTL[PULSESEL]

20.11.4.3 Valley Switching

Event filtering depicts the valley switching function along with the event filtering logic described in Section 20.11.4.2. This function can be used to achieve programmable valley switching without any additional external circuitry. This module provides an on-chip hardware mechanism that can:

- Capture the oscillation period
- Accurately delay the PWM switching instant
- Allow a programmable number of edges before the delay takes effect
- Provide multiple choices of triggers and events
- Allow easy adaptability for optimum performance under changing system/operating conditions

The DCxEVTy signal needs further processing to support valley switching. Here is a brief description of how valley switching function is enabled:

1. Select one of the DCxEVTy events as input to the valley switching block (DCFCTL[SRCSEL]) with an option to add the blanking window (Blank Control Logic). This is where the comparator output (or external input) above is selected as an input to the valley switching block.
2. Configure the edge filter to capture 'n' rising, falling or both edges through the edge selection logic (DCFCTL[EDGEMODE, EDGECOUNT]).
3. Select the correct event to reset and restart the edge filter (VCAPCTL[TRIGSEL]). Edge capturing event is triggered or armed by this selected edge.
4. Enable valley capture logic (VCAPCTL[VCAPE]).
5. Select the start edge that will indicate the start of capture for oscillation period measurement (VCNTCFG[STARTEDGE]). This is where the 16-bit counter starts counting.
6. Select the stop edge (VCNTCFG[STOPEDGE]) that will indicate the edge at which the 16-bit counter stops counting. The captured counter value (CNTVAL) provides oscillation period information.
   - The STOPEDGE value must always be greater than STARTEDGE value.
7. Configure and apply the captured delay (CNTVAL) to the edge filtered DCxEVTy signal. The CNTVAL value may be applied as is or applied in conjunction with a software programmed value (useful for offset adjustment) (SWVDELVAL) or only a fraction of the delay may be applied with or without SWVDELVAL. This is useful to correctly apply a delay corresponding to the valley point. (VCAPCTL[VDELAYDIV])
8. Configure VCAPCTL[EDGEFILTDLYSEL] to apply hardware delay based on the captured value above.

Once the counter is stopped, counter value is copied into CNTVAL register and counter is reset to zero. No further captures are done until the logic is triggered again by occurrence of event selected by VCAPCTL[TRIGSEL]. In this implementation, the software trigger is used as the source for VCAPCTL[TRIGSEL]. Upon occurrence of the trigger event, irrespective of the current status of the counter, the counter is reset and starts counting from zero upon occurrence of the STARTEDGE. Similarly, upon occurrence of the trigger event, the edge filter is reset and starts counting from zero upon occurrence of the STARTEDGE.

Output from the valley switching block (DCEVTFILT) is then used to synchronize the PWM time-base. The process is shown in Figure 20-55.

Note

A specific application example showcasing the usage of valley switching hardware and software is available on the controlSuite.

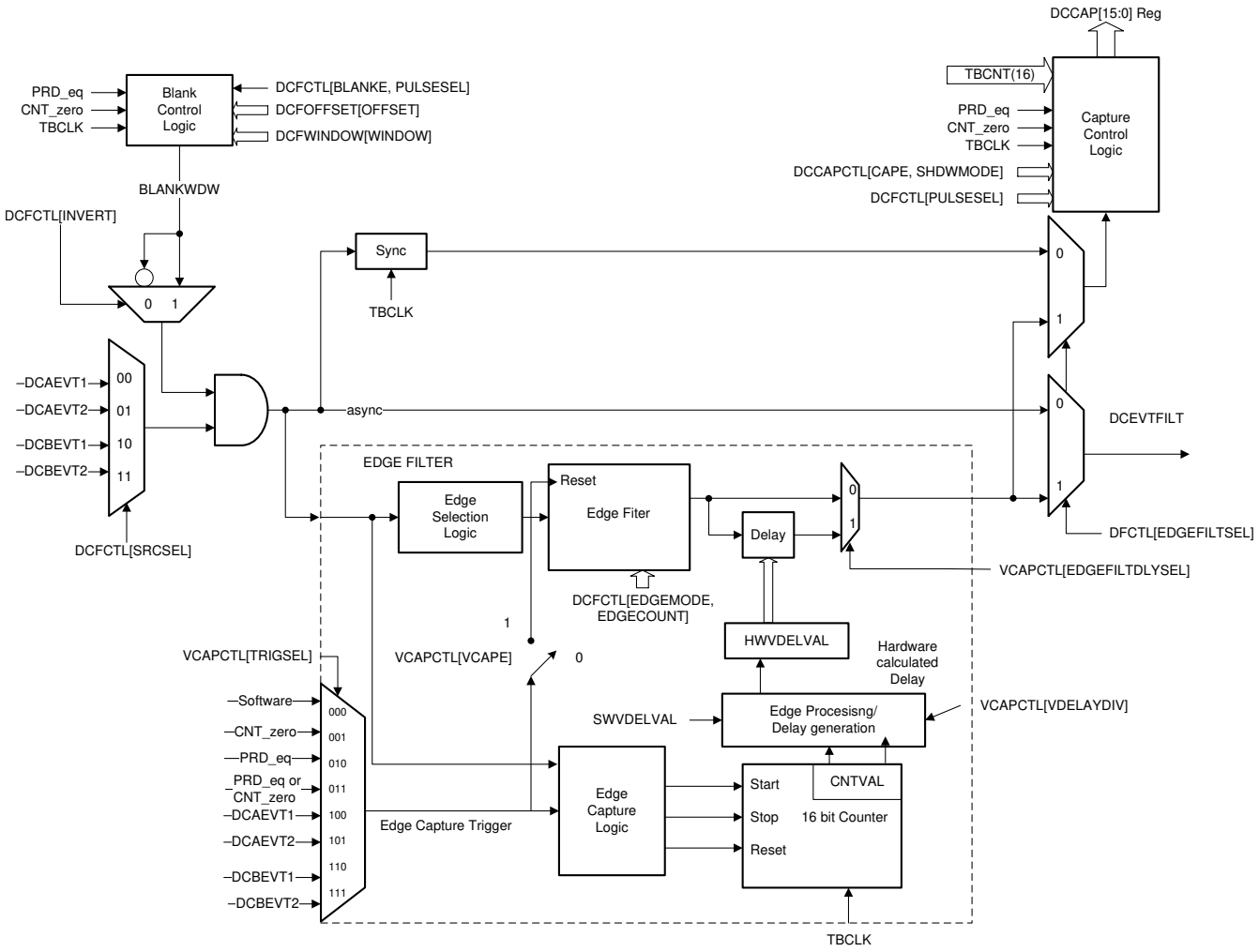


Figure 20-55. Valley Switching

20.12 ePWM Crossbar (X-BAR)

Figure 20-56 shows the architecture of the ePWM Crossbar (X-BAR). This module enables selection of various trigger sources into any of the eight dedicated ETWPM trips inputs, namely the TRIP4, TRIP5, TRIP7, TRIP8, TRIP9, TRIP10, TRIP11, and TRIP12.

Note

Refer to the *Crossbar (X-BAR)* chapter for more information on the X-BAR modules, including X-BAR flags.

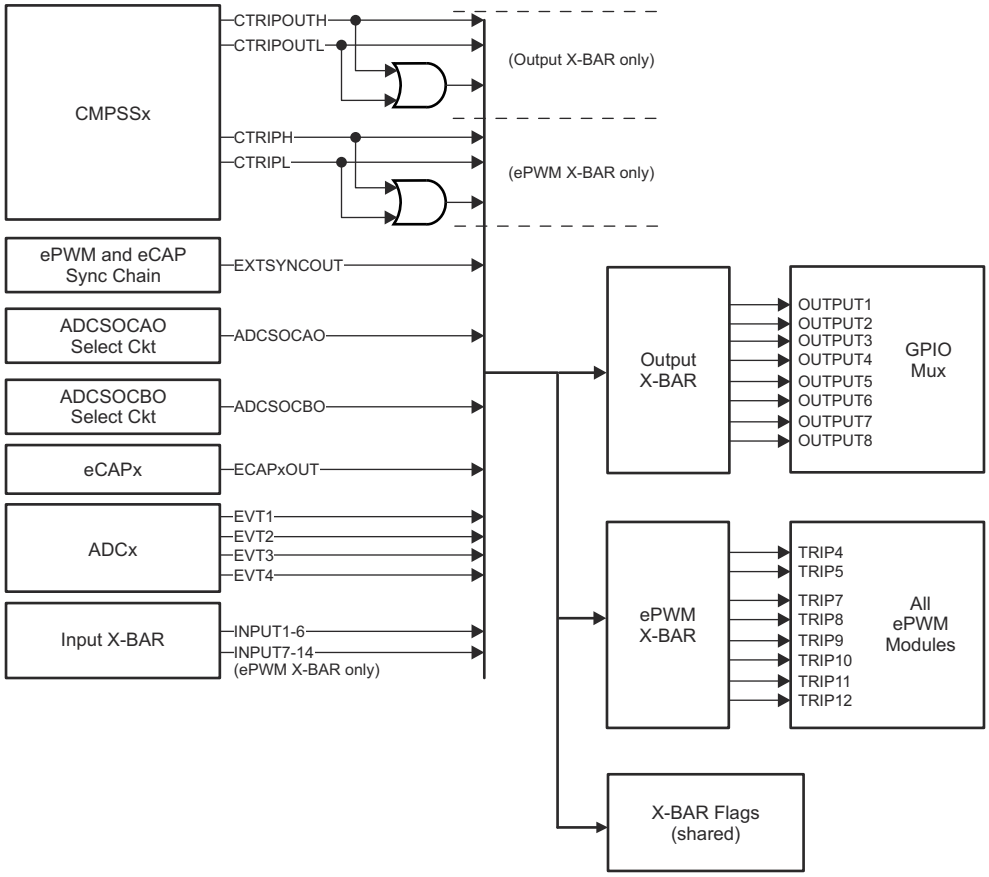


Figure 20-56. ePWM X-BAR

20.13 Applications to Power Topologies

An ePWM module has all the local resources necessary to operate completely as a standalone module or to operate in synchronization with other identical ePWM modules.

20.13.1 Overview of Multiple Modules

Previously in this chapter, all discussions have described the operation of a single module. To facilitate the understanding of multiple modules working together in a system, the ePWM module described in reference is represented by the more simplified block diagram shown in Figure 20-57. This simplified ePWM block shows only the key resources needed to explain how a multiswitch power topology is controlled with multiple ePWM modules working together.

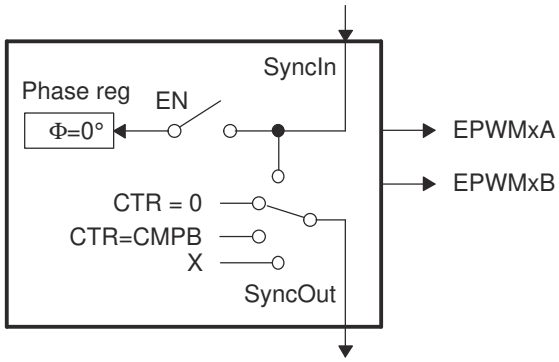


Figure 20-57. Simplified ePWM Module

20.13.2 Key Configuration Capabilities

The key configuration choices available to each module are as follows:

- Options for SyncIn
  - Load own counter with phase register on an incoming sync strobe—enable (EN) switch closed
  - Do nothing or ignore incoming sync strobe—enable switch open
  - Sync flow-through - SyncOut connected to SyncIn
  - Master mode, provides a sync at PWM boundaries—SyncOut connected to CTR = PRD
  - Master mode, provides a sync at any programmable point in time—SyncOut connected to CTR = CMPB
  - Module is in standalone mode and provides No sync to other modules—SyncOut connected to X (disabled)
- Options for SyncOut
  - Sync flow-through - SyncOut connected to SyncIn
  - Master mode, provides a sync at PWM boundaries—SyncOut connected to CTR = PRD
  - Master mode, provides a sync at any programmable point in time—SyncOut connected to CTR = CMPB
  - Module is in standalone mode and provides No sync to other modules—SyncOut connected to X (disabled)

For each choice of SyncOut, a module may also choose to load its own counter with a new phase value on a SyncIn strobe input or choose to ignore it (that is, via the enable switch). Although various combinations are possible, the two most common—master module and slave module modes—are shown in Figure 20-58.

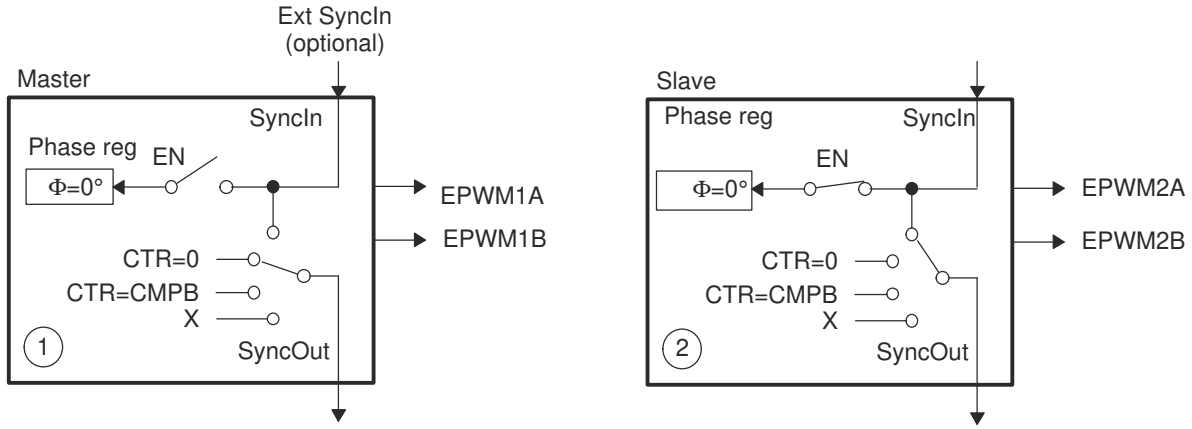
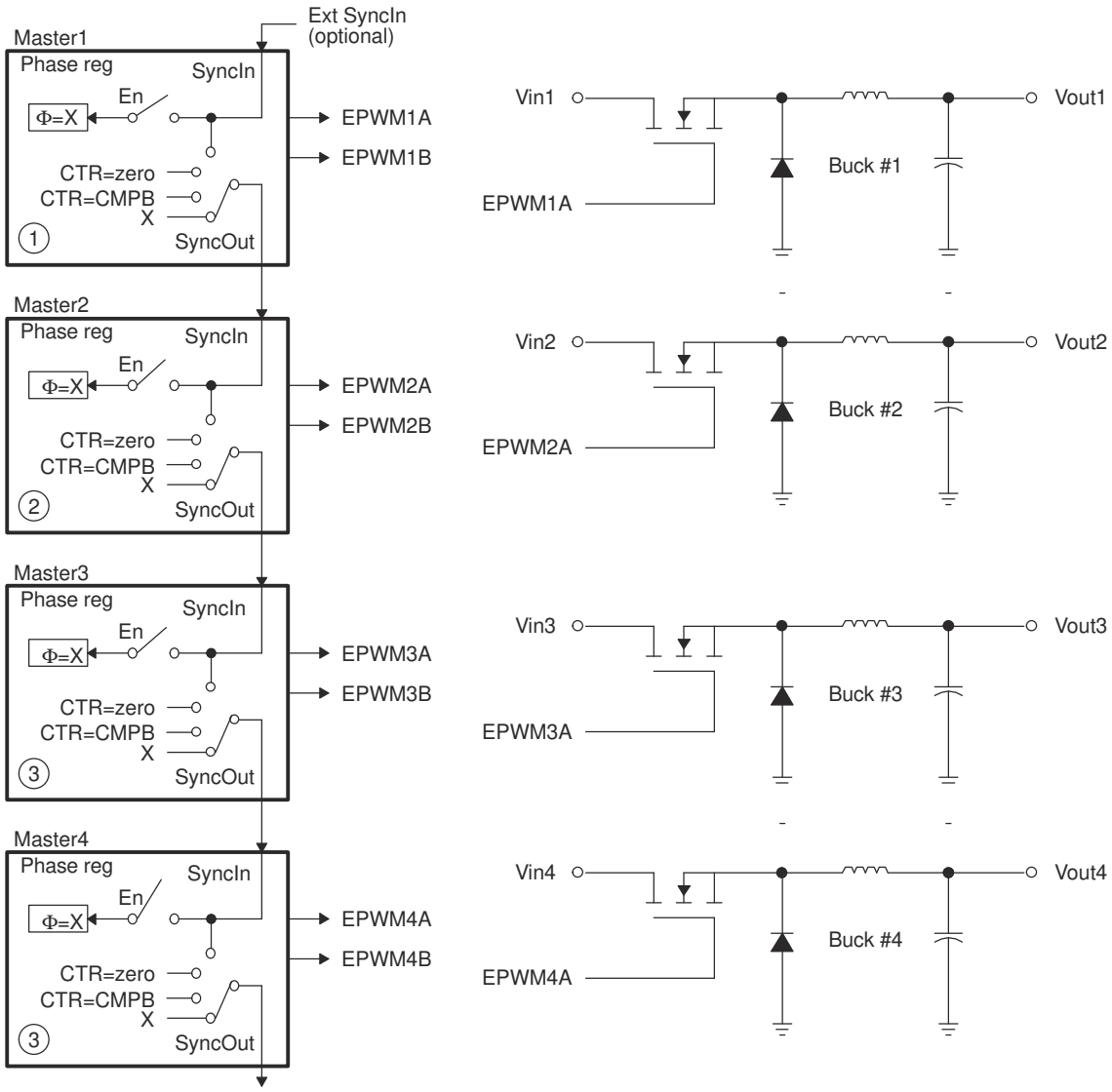


Figure 20-58. EPWM1 Configured as a Typical Master, EPWM2 Configured as a Slave

20.13.3 Controlling Multiple Buck Converters With Independent Frequencies

One of the simplest power converter topologies is the buck. A single ePWM module configured as a master can control two buck stages with the same PWM frequency. If independent frequency control is required for each buck converter, then one ePWM module must be allocated for each converter stage. Figure 20-59 shows four buck stages, each running at independent frequencies. In this case, all four ePWM modules are configured as Masters and no synchronization is used. Figure 20-60 shows the waveforms generated by the setup shown in Figure 20-59; note that only three waveforms are shown, although there are four stages.



A. φ = X indicates value in phase register is a "don't care"

Figure 20-59. Control of Four Buck Stages. Here $F_{PWM1} \neq F_{PWM2} \neq F_{PWM3} \neq F_{PWM4}$

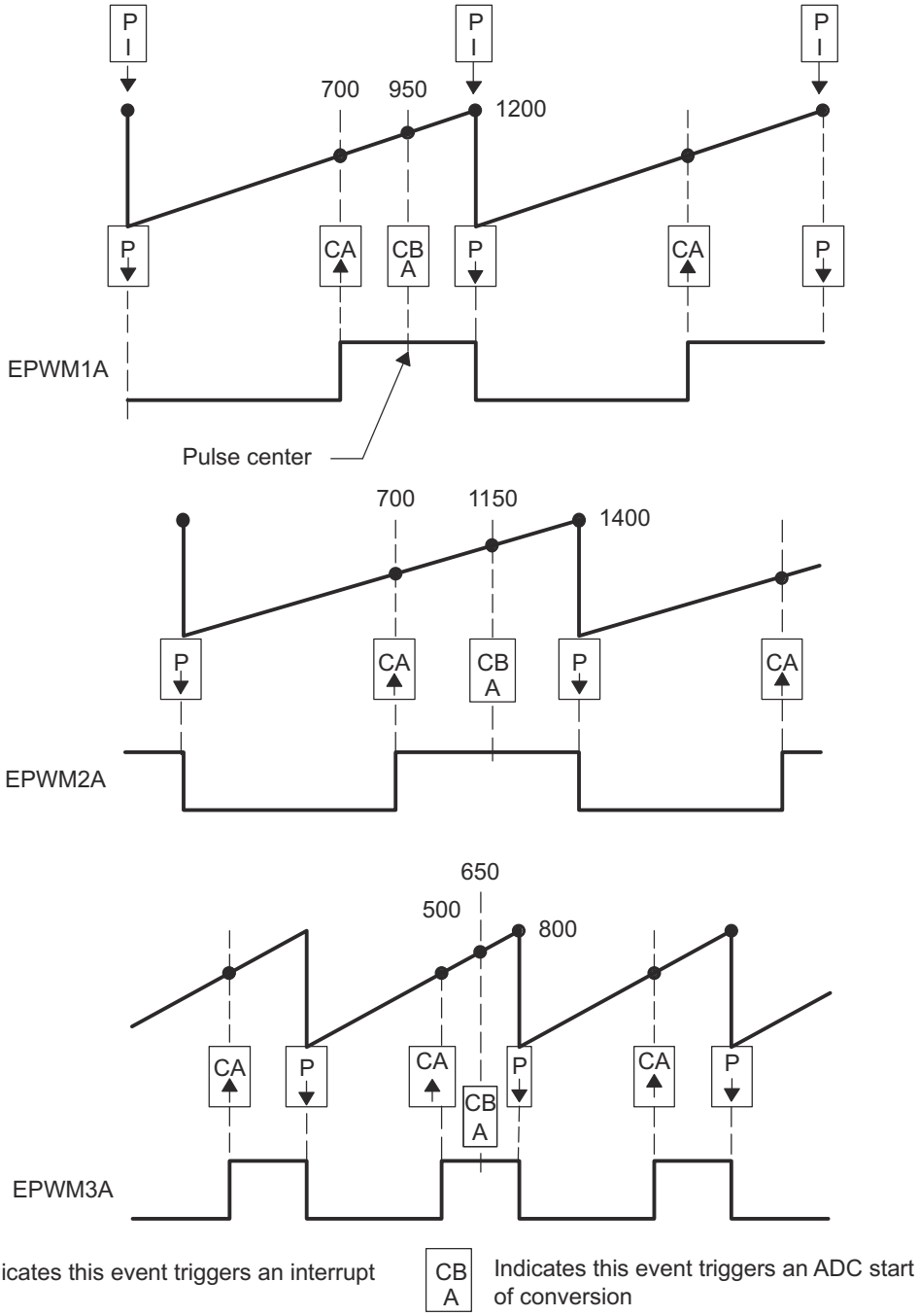
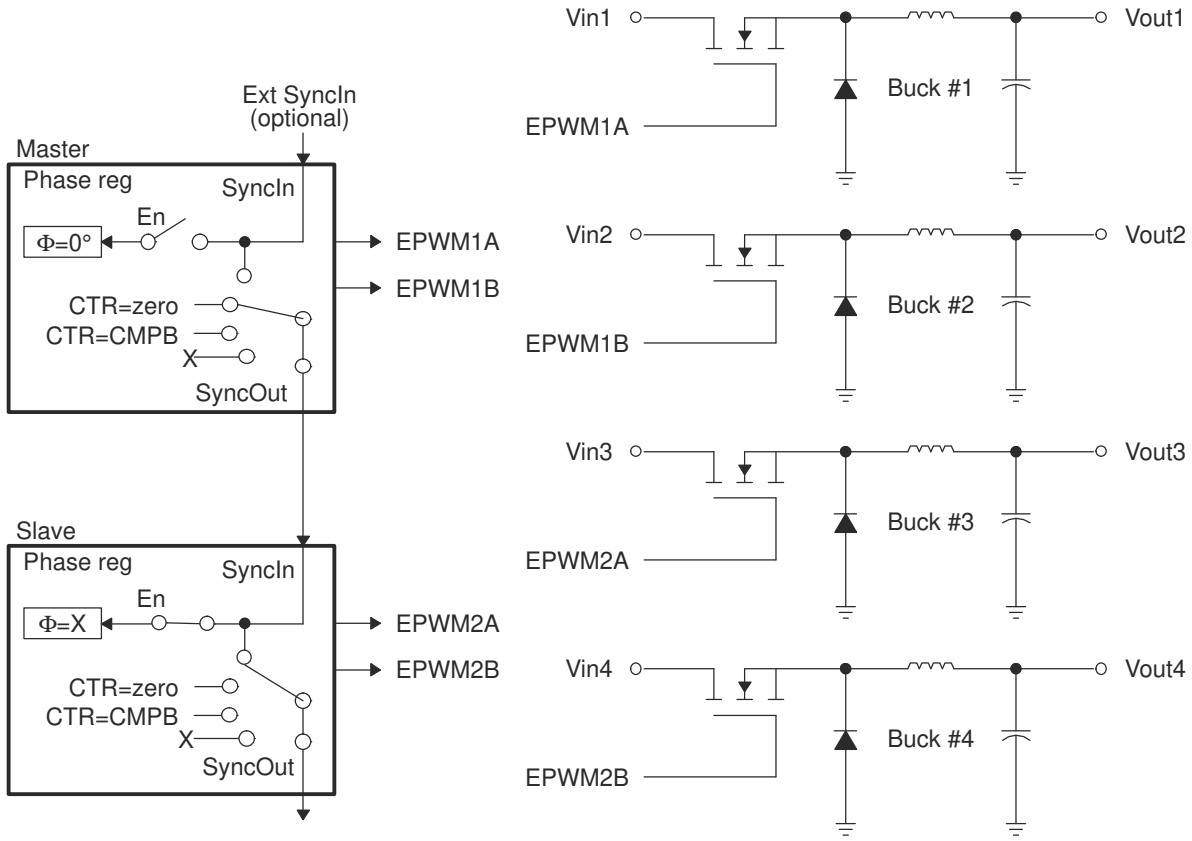


Figure 20-60. Buck Waveforms for Control of Four Buck Stages (Note: Only three bucks shown here)

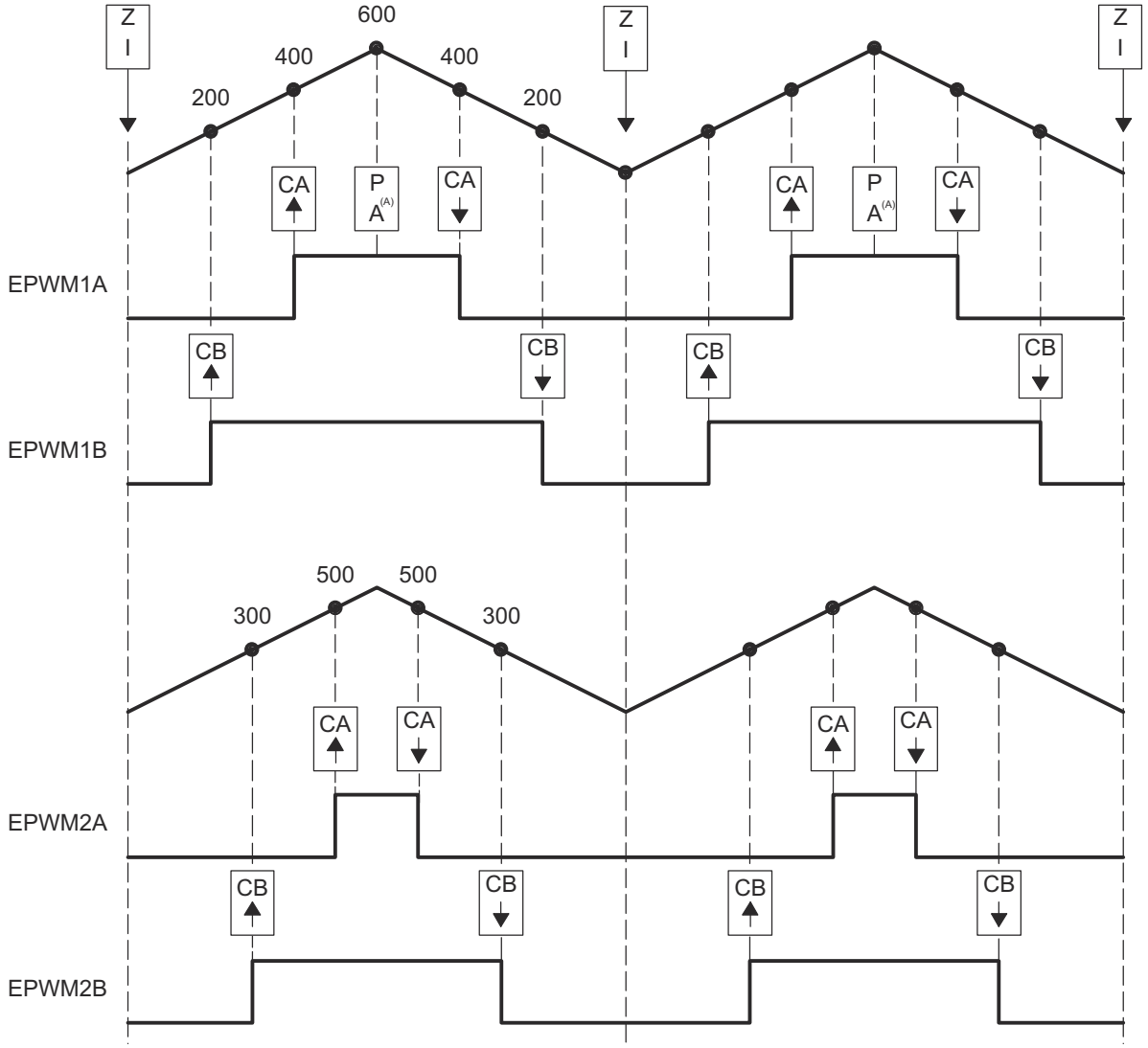
20.13.4 Controlling Multiple Buck Converters With Same Frequencies

If synchronization is a requirement, ePWM module 2 can be configured as a slave and can operate at integer multiple (N) frequencies of module 1. The sync signal from master to slave ensures these modules remain locked. Figure 20-61 shows such a configuration; Figure 20-62 shows the waveforms generated by the configuration.



A. φ = X indicates value in phase register is a "don't care"

Figure 20-61. Control of Four Buck Stages. (Note: F_{PWM2} = N x F_{PWM1})



A. Starts ADC conversion.

Figure 20-62. Buck Waveforms for Control of Four Buck Stages (Note: F_{PWM2} = $F_{PWM1)}$)

20.13.5 Controlling Multiple Half H-Bridge (HHB) Converters

Topologies that require control of multiple switching elements can also be addressed with these same ePWM modules. It is possible to control a Half-H bridge stage with a single ePWM module. This control can be extended to multiple stages. Figure 20-63 shows control of two synchronized Half-H bridge stages where stage 2 can operate at integer multiple (N) frequencies of stage 1. Figure 20-64 shows the waveforms generated by the configuration shown in Figure 20-63.

Module 2 (slave) is configured for Sync flow-through; if required, this configuration allows for a third Half-H bridge to be controlled by PWM module 3 and also, most importantly, to remain in synchronization with master module 1.

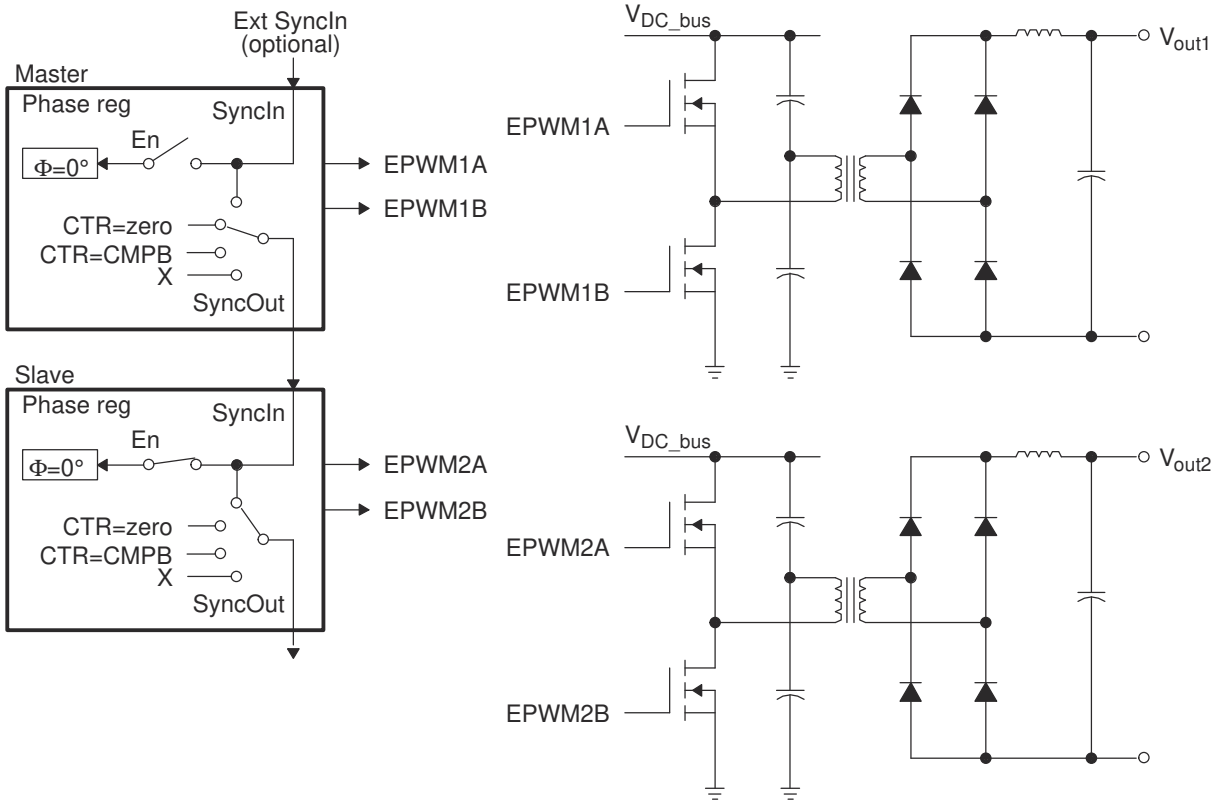


Figure 20-63. Control of Two Half-H Bridge Stages (F_{PWM2} = N x F_{PWM1})

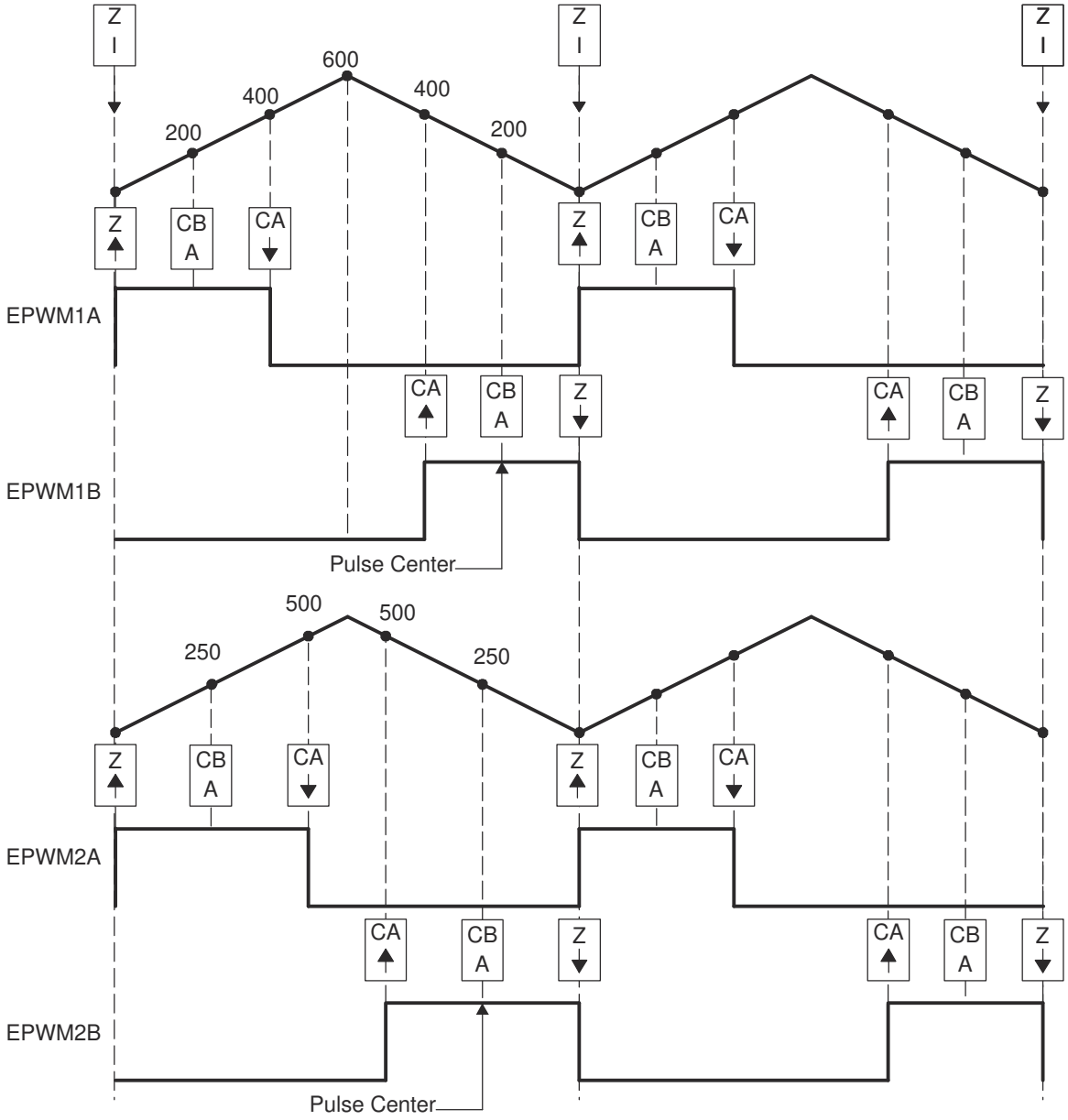


Figure 20-64. Half-H Bridge Waveforms for Control of Two Half-H Bridge Stages (Note: Here $F_{PWM2} = F_{PWM1}$)

20.13.6 Controlling Dual 3-Phase Inverters for Motors (ACI and PMSM)

The idea of multiple modules controlling a single power stage can be extended to the 3-phase Inverter case. In such a case, six switching elements can be controlled using three PWM modules, one for each leg of the inverter. Each leg must switch at the same frequency and all legs must be synchronized. A master + two slaves configuration can easily address this requirement. Figure 20-65 shows how six PWM modules can control two independent 3-phase Inverters; each running a motor.

As in the cases shown in the previous sections, we have a choice of running each inverter at a different frequency (module 1 and module 4 are masters as in Figure 20-65), or both inverters can be synchronized by using one master (module 1) and five slaves. In this case, the frequency of modules 4, 5, and 6 (all equal) can be integer multiples of the frequency for modules 1, 2, 3 (also all equal).

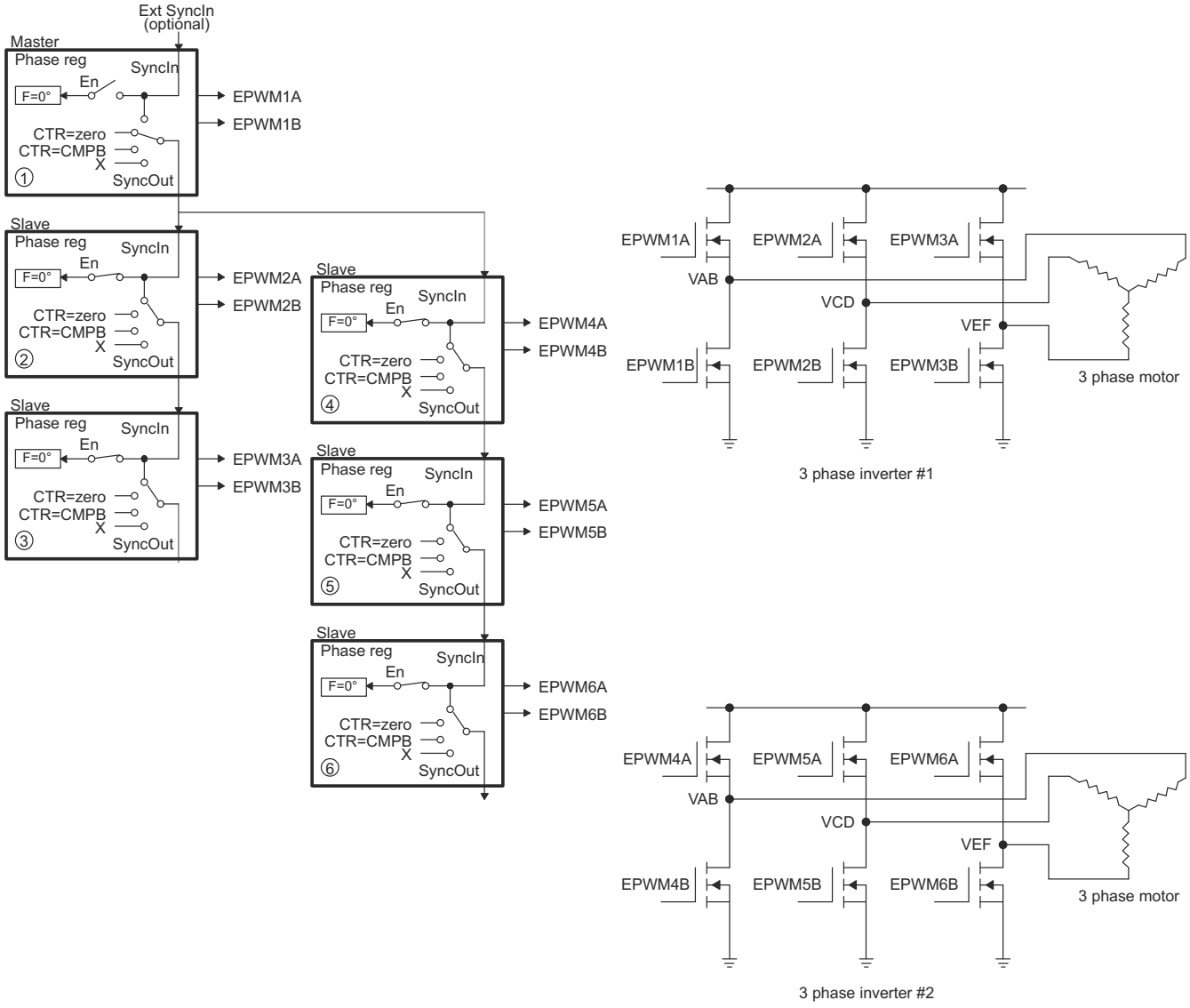


Figure 20-65. Control of Dual 3-Phase Inverter Stages as Is Commonly Used in Motor Control

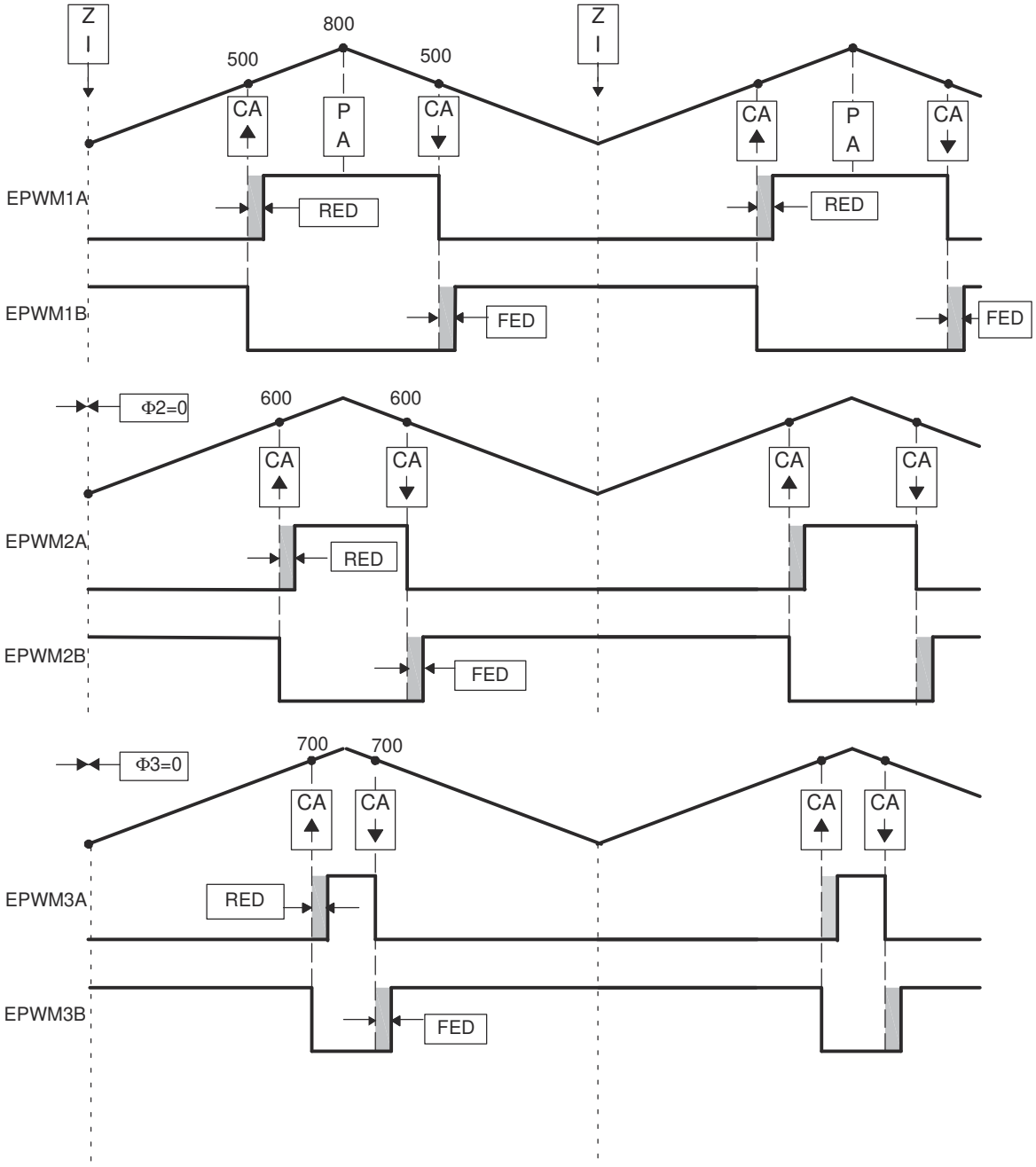


Figure 20-66. 3-Phase Inverter Waveforms for Control of Dual 3-Phase Inverter Stages (Only One Inverter Shown)

20.13.7 Practical Applications Using Phase Control Between PWM Modules

So far, none of the examples have made use of the phase register (TBPHS). It has either been set to zero or its value has been a don't care. However, by programming appropriate values into TBPHS, multiple PWM modules can address another class of power topologies that rely on phase relationship between legs (or stages) for correct operation. As described in the TB module section, a PWM module can be configured to allow a SyncIn pulse to cause the TBPHS register to be loaded into the TBCTR register. To illustrate this concept, Figure 20-67 shows a master and slave module with a phase relationship of 120° (that is, the slave leads the master).

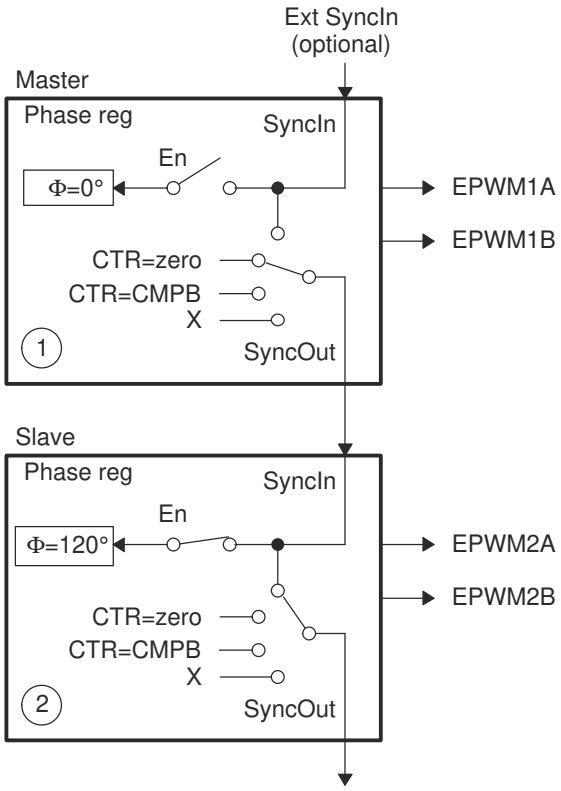


Figure 20-67. Configuring Two PWM Modules for Phase Control

Figure 20-68 shows the associated timing waveforms for this configuration. Here, TBPRD = 600 for both master and slave. For the slave, TBPHS = 200 (that is, 200/600 X 360° = 120°). Whenever the master generates a SyncIn pulse (CTR = PRD), the value of TBPHS = 200 is loaded into the slave TBCTR register so the slave time-base is always leading the master's time-base by 120°.

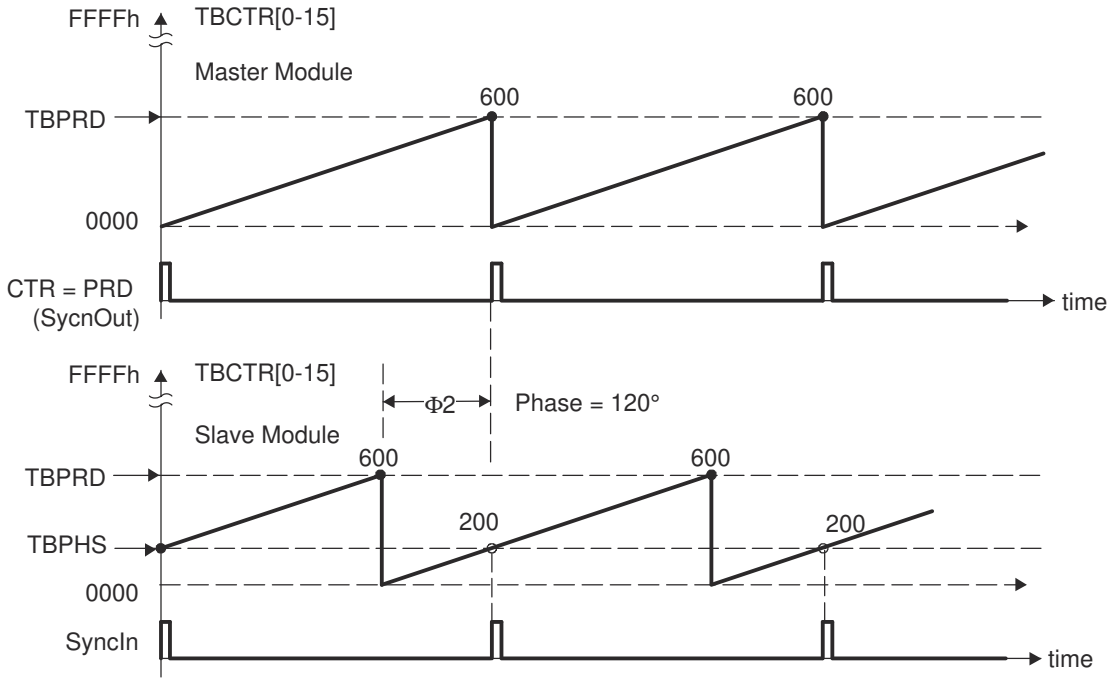


Figure 20-68. Timing Waveforms Associated With Phase Control Between Two Modules

20.13.8 Controlling a 3-Phase Interleaved DC/DC Converter

A popular power topology that makes use of phase-offset between modules is shown in Figure 20-69. This system uses three PWM modules, with module 1 configured as the master. To work, the phase relationship between adjacent modules must be F = 120°. This is achieved by setting the slave TBPHS registers 2 and 3 with values of 1/3 and 2/3 of the period value, respectively. For example, if the period register is loaded with a value of 600 counts, then TBPHS (slave 2) = 200 and TBPHS (slave 3) = 400. Both slave modules are synchronized to the master 1 module.

This concept can be extended to four or more phases, by setting the TBPHS values appropriately. The following formula gives the TBPHS values for N phases:

$$TBPHS(N,M) = (TBPRD/N) \times (M-1) \quad (16)$$

Where:

N = number of phases

M = PWM module number

For example, for the 3-phase case (N=3), TBPRD = 600,

TBPHS(3,2) = (600/3) x (2-1) = 200 (that is, Phase value for Slave module 2)

TBPHS(3,3) = 400 (that is, Phase value for Slave module 3)

Figure 20-70 shows the waveforms for the configuration in Figure 20-69.

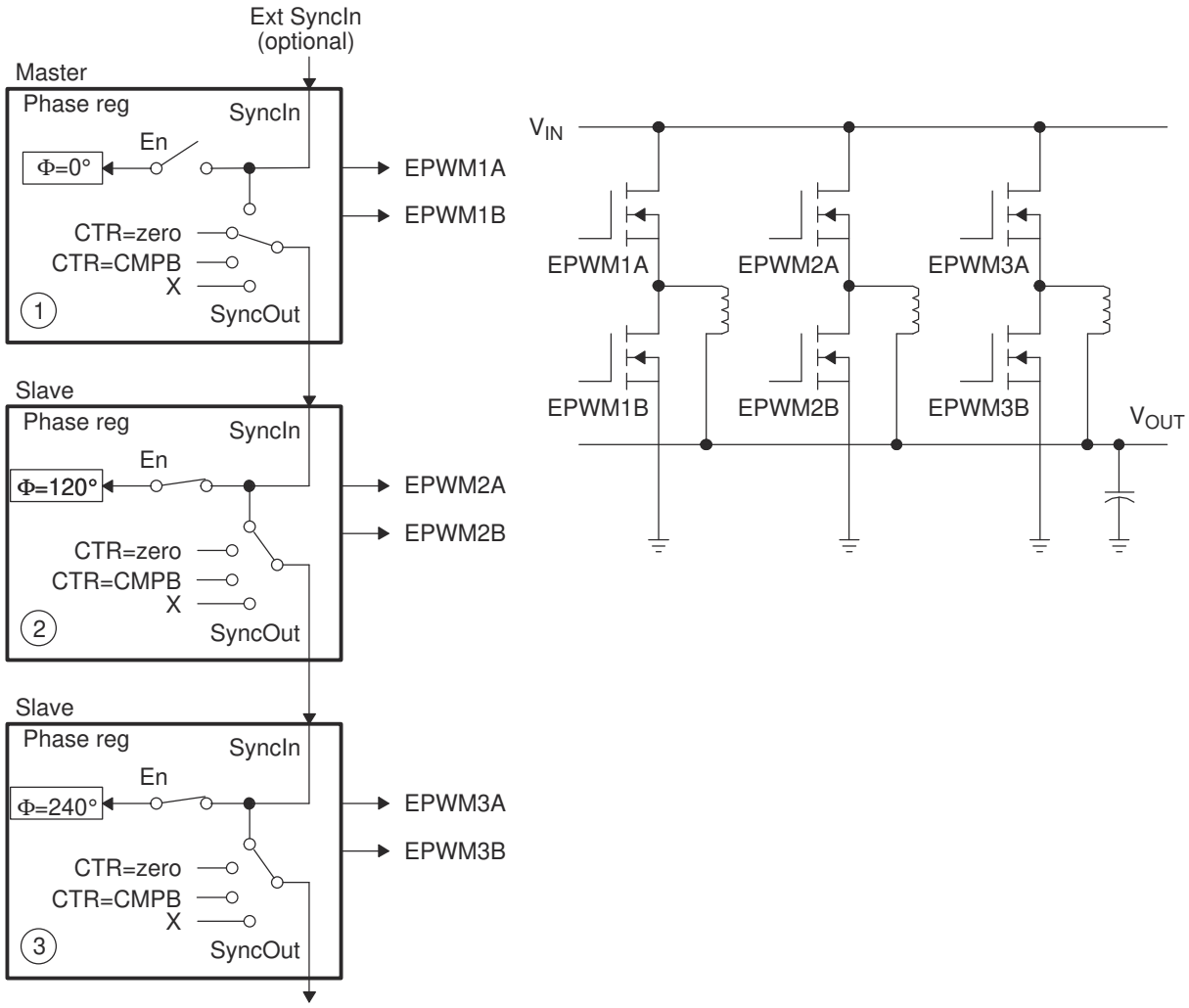


Figure 20-69. Control of a 3-Phase Interleaved DC/DC Converter

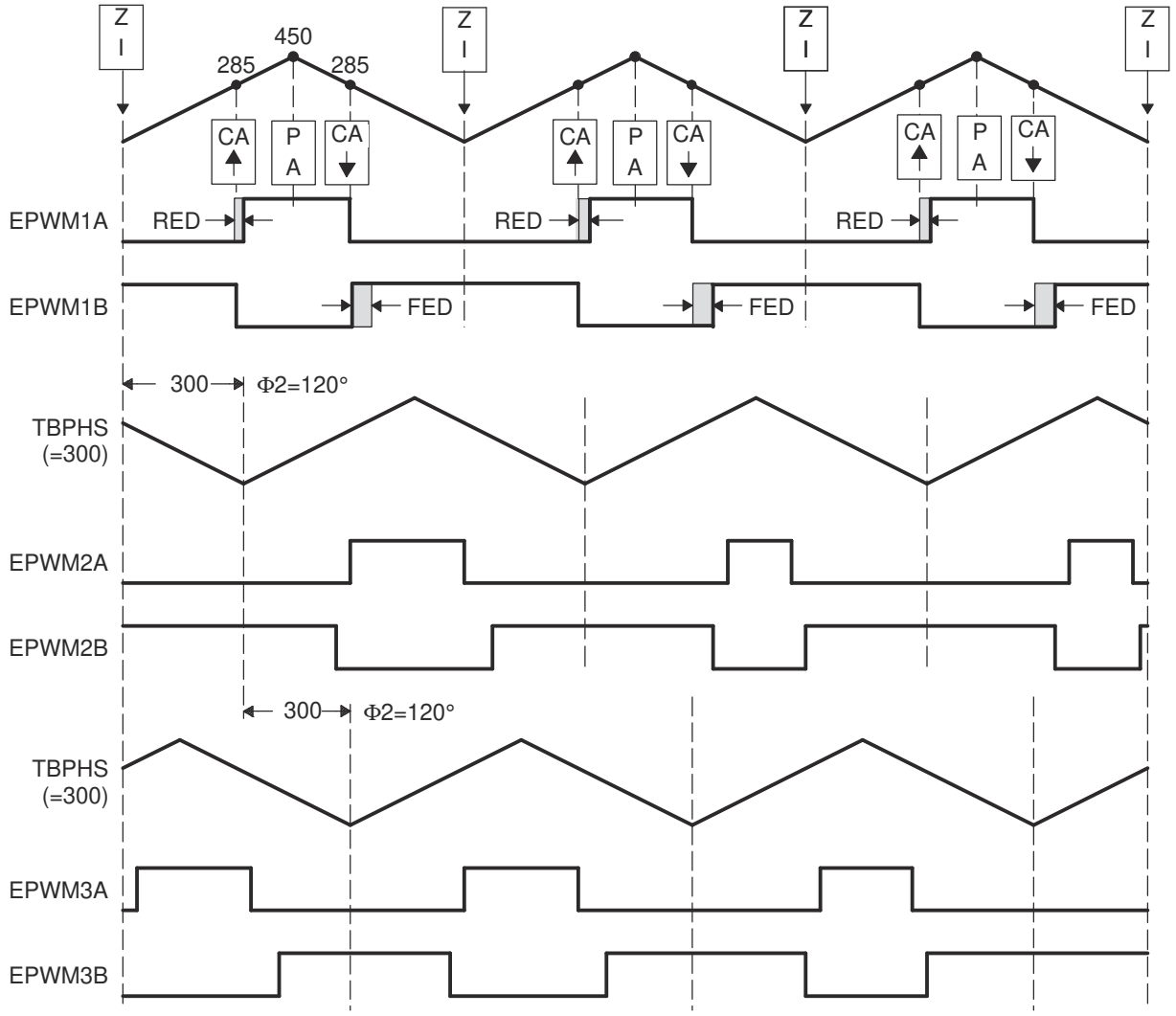


Figure 20-70. 3-Phase Interleaved DC/DC Converter Waveforms for Control of a 3-Phase Interleaved DC/DC Converter

20.13.9 Controlling Zero Voltage Switched Full Bridge (ZVSFB) Converter

The example given in Figure 20-71 assumes a static or constant phase relationship between legs (modules). In such a case, control is achieved by modulating the duty cycle. It is also possible to dynamically change the phase value on a cycle-by-cycle basis. This feature lends itself to controlling a class of power topologies known as *phase-shifted full bridge*, or *zero voltage switched full bridge.* Here the controlled parameter is not duty cycle (this is kept constant at approximately 50 percent); instead it is the phase relationship between legs. Such a system can be implemented by allocating the resources of two PWM modules to control a single power stage, which in turn requires control of four switching elements. Figure 20-72 shows a master/slave module combination synchronized together to control a full H-bridge. In this case, both master and slave modules are required to switch at the same PWM frequency. The phase is controlled by using the slave's phase register (TBPHS). The master's phase register is not used and therefore can be initialized to zero.

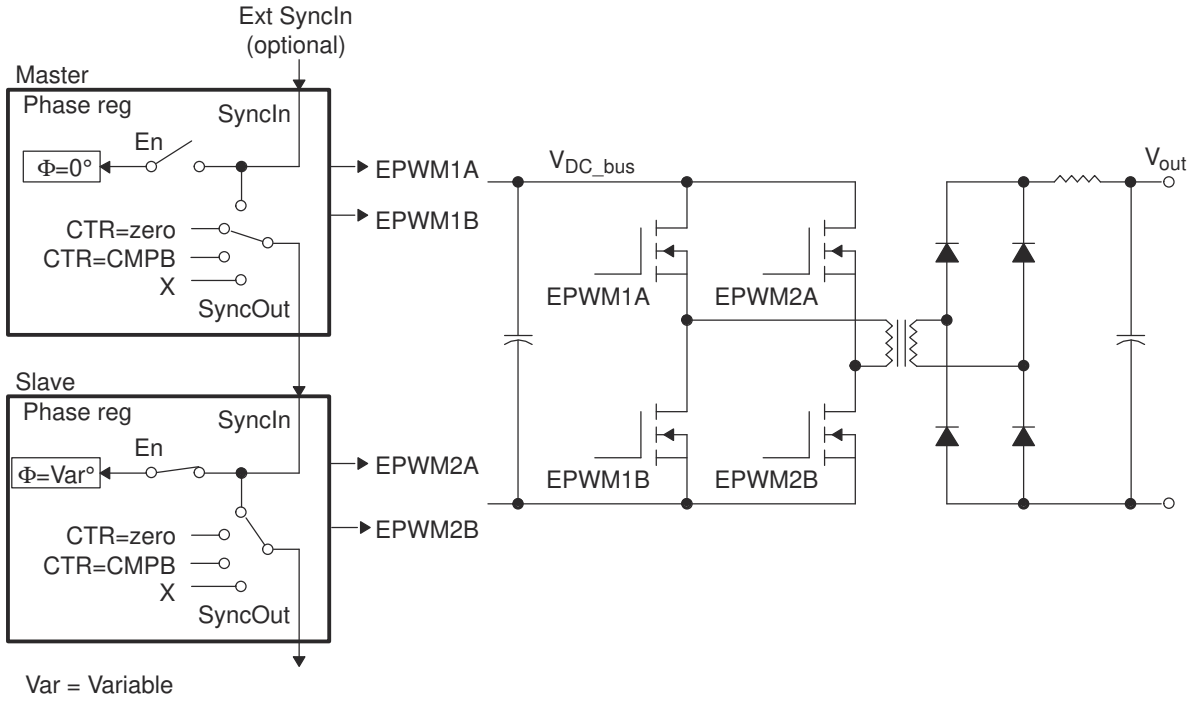


Figure 20-71. Controlling a Full-H Bridge Stage (F_{PWM2} = $F_{PWM1)}$)

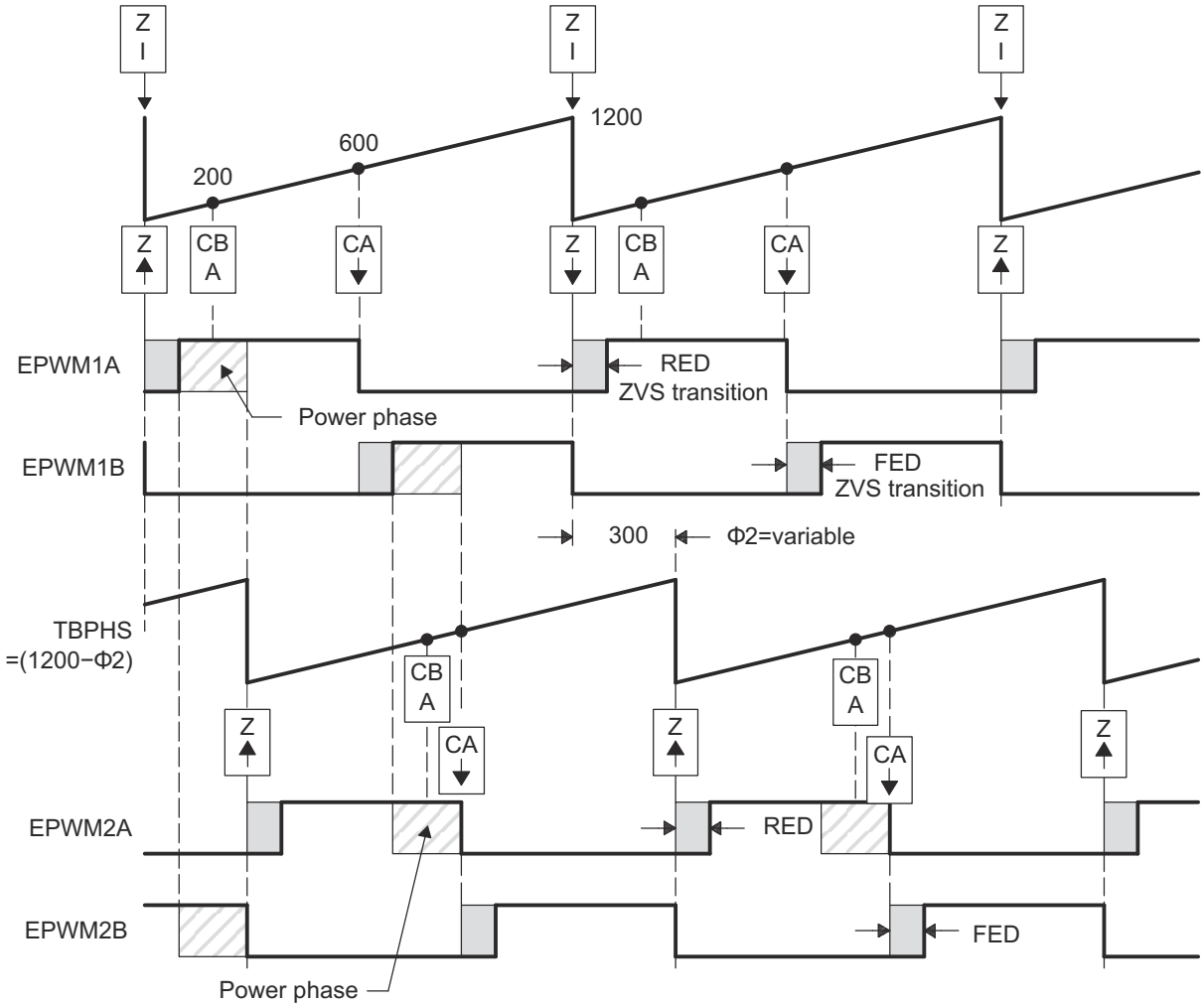


Figure 20-72. ZVS Full-H Bridge Waveforms

20.13.10 Controlling a Peak Current Mode Controlled Buck Module

Peak current control techniques offer a number of benefits like automatic over current limiting, fast correction for input voltage variations and reducing magnetic saturation. Figure 20-73 shows the use of ePWM1A along with the on-chip analog comparator for buck converter topology. The output current is sensed through a current sense resistor and fed to the positive terminal of the on-chip comparator. The internal programmable 12-bit DAC can be used to provide a reference peak current at the negative terminal of the comparator. Alternatively, an external reference could be connected at this input. The comparator output is an input to the Digital compare sub-module. The ePWM module is configured in such a way so as to trip the ePWM1A output as soon as the sensed current reaches the peak reference value. A cycle-by-cycle trip mechanism is used. Figure 20-74 shows the waveforms generated by the configuration.

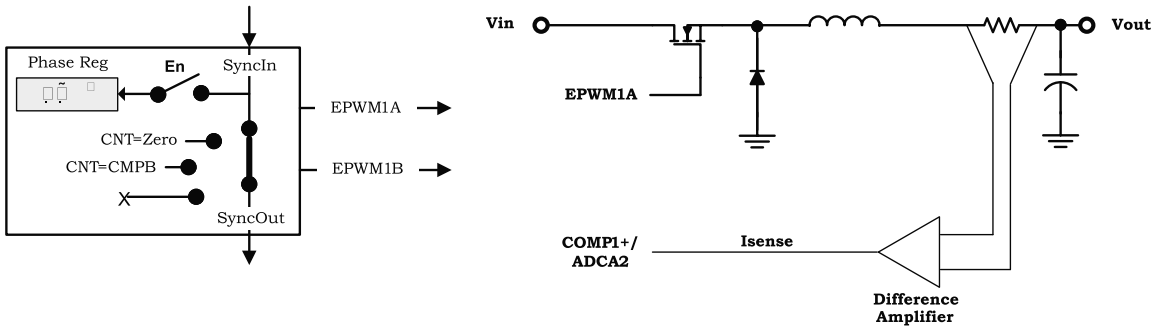


Figure 20-73. Peak Current Mode Control of a Buck Converter

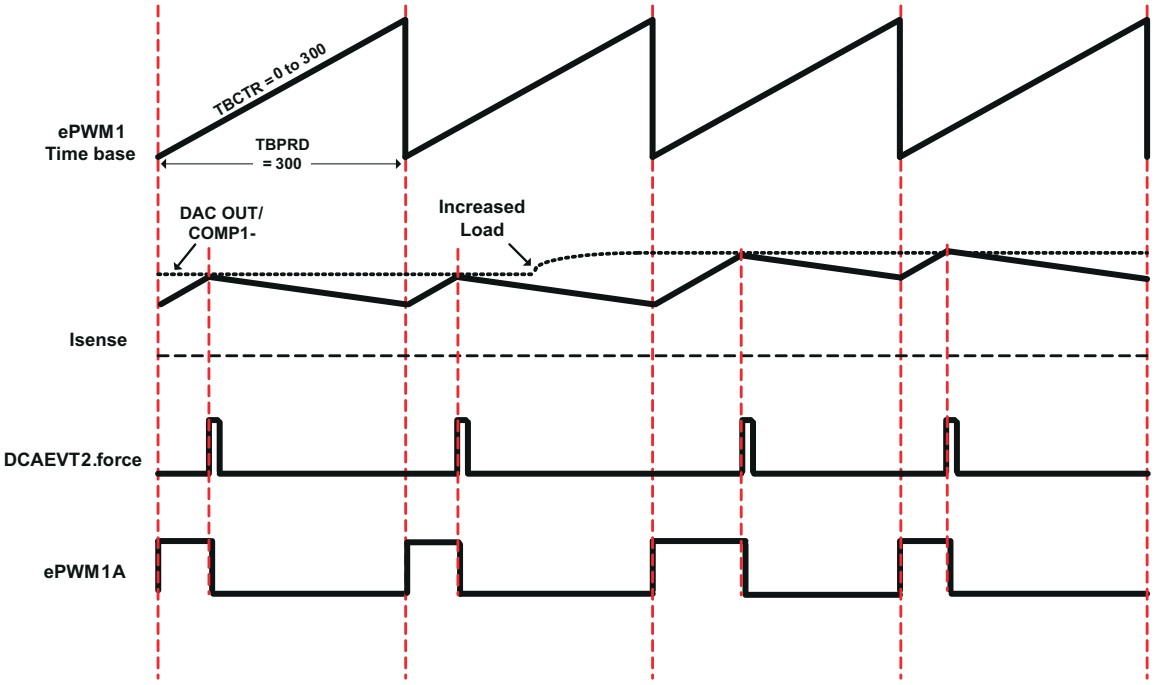
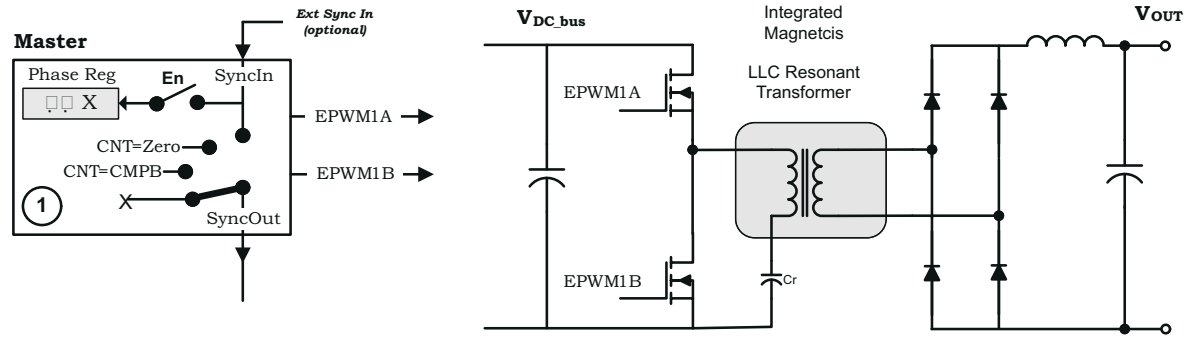


Figure 20-74. Peak Current Mode Control Waveforms for Control of a Buck Converter

20.13.11 Controlling H-Bridge LLC Resonant Converter

Various topologies of resonant converters are well-known in the field of power electronics for many years. In addition to these, H-bridge LLC resonant converter topology has recently gained popularity in many consumer electronics applications where high efficiency and power density are required. In this example, single channel configuration of ePWM1 is detailed, yet the configuration can easily be extended to multichannel. Here the controlled parameter is not duty cycle (this is kept constant at approximately 50 percent); instead it is frequency. Although the deadband is not controlled and kept constant as 300 ns (that is, 30 @100 MHz TBCLK), it is up to user to update it in real time to enhance the efficiency by adjusting enough time delay for soft switching.



NOTE: Θ = X indicates value in phase register is a "don't care"

Figure 20-75. Control of Two Resonant Converter Stages

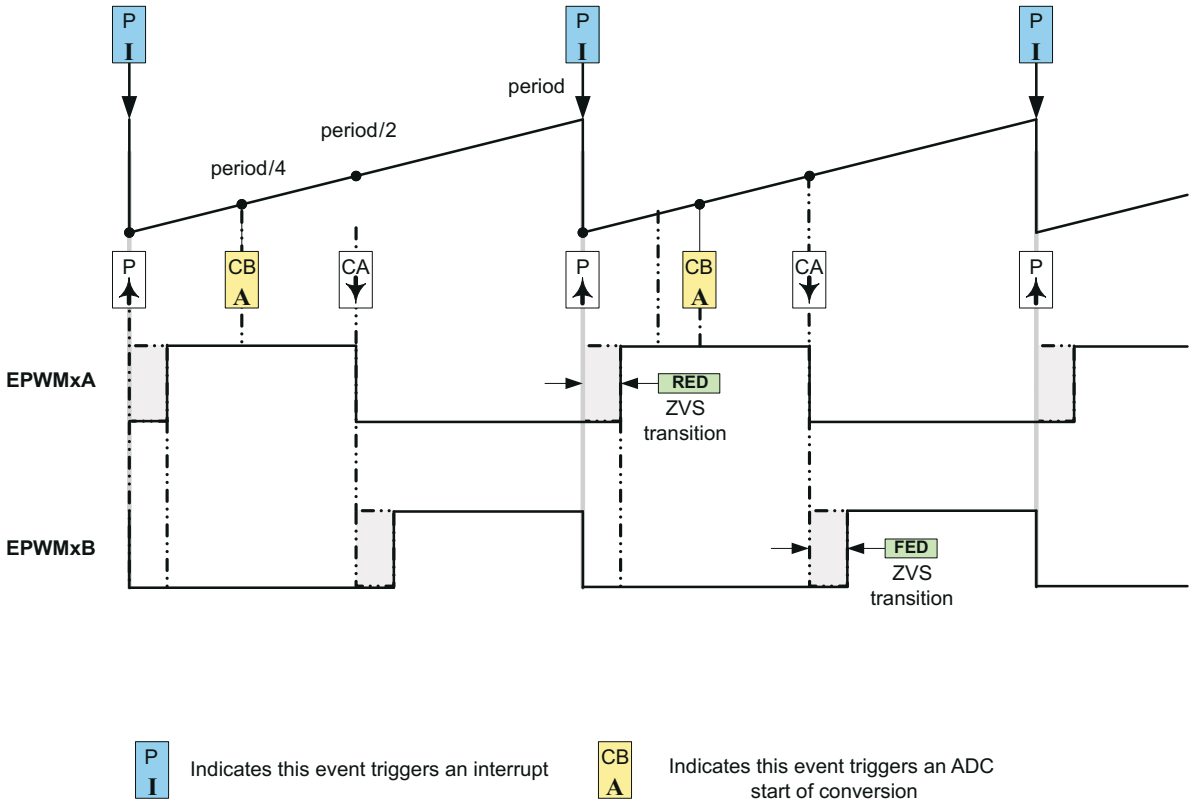


Figure 20-76. H-Bridge LLC Resonant Converter PWM Waveforms

20.14 Register Lock Protection

The register lock protection mechanism is added to protect the critical ePWM registers from being corrupted by accidental writes in case of runaway code. The register EPWMLOCK contains the definition of Lock bits (Table 20-14 shows the lock bits and the corresponding registers). This register also has a KEY field; writes to this register will succeed only if the KEY field is written with a value of 0xa5a5. Refer to the register descriptions for more details.

Table 20-14. Lock Bits and Corresponding Registers

Bit Field	Definition	Registers Locked
HRLOCK	HRPWM Register Set Lock	HRCNFG, HRPWR, HRMSTEP, HRPCTL
GLLOCK	Global Load Register Set Lock	GLDCTL, GLDCFG
TZCFGLOCK	TripZone Register Set Lock	TZSEL, TZDCSEL, TZCTL, TZCTL2, TZCTLDCA, TZCTLDCB, TZEINT
TZCLRLOCK	TripZone Clear Register Set Lock	TZCLR, TZCBCCLR, TZOSTCLR, TZFRC
DCLOCK	Digital Compare Register Set Lock	DCTRIPSEL, DCACTL, DCBCTL, DCFCTL, DCCAPCTL, DCAHTRIPSEL, DCALTRIPSEL, DCBHTRIPSEL, DCBLTRIPSEL

Note

Due to the presence of the KEY field in the same register, only 32-bit writes will succeed if the KEY matches. The 16-bit writes to the upper or lower half of this register will be ignored.

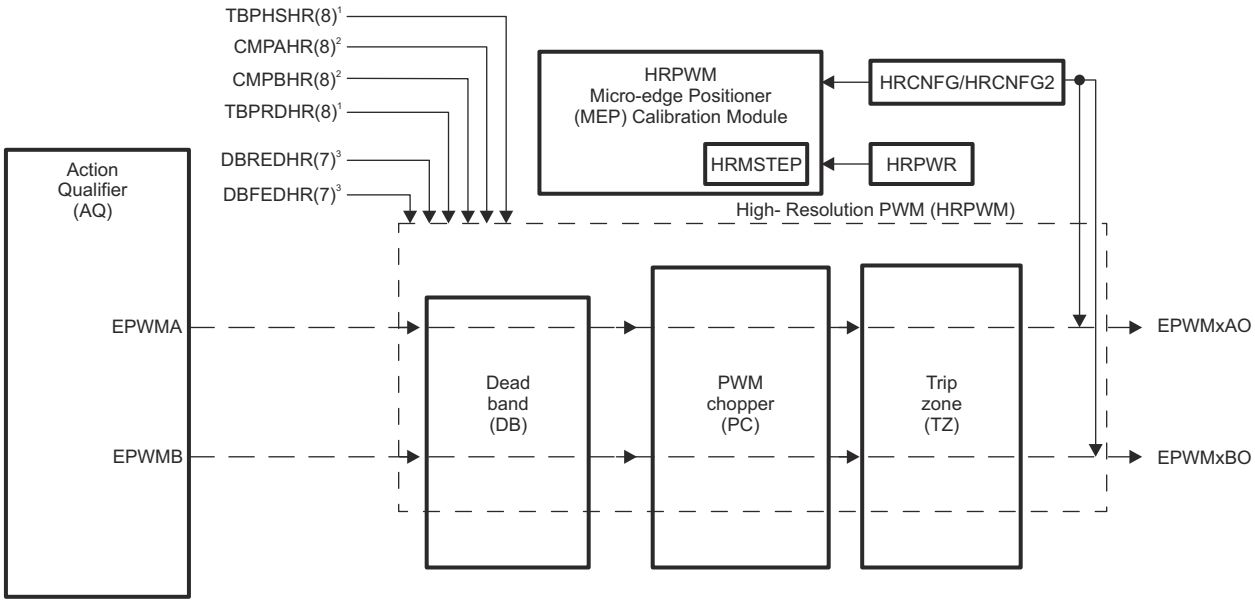
20.15 High-Resolution Pulse Width Modulator (HRPWM)

Figure 20-77 shows a block diagram of the HRPWM. This module extends the time resolution capabilities of the conventionally derived digital pulse width modulator (PWM). HRPWM is typically used when PWM resolution falls below ~ 9-10 bits. The key features of HRPWM are:

- Extended time resolution capability
- Used in both duty cycle and phase-shift control methods
- Finer time granularity control or edge positioning using extensions to the Compare A, Compare B and Phase registers
- Implemented using the A and B signal path of PWM, that is, on the EPWMxA and EPWMxB output
- Dead band high-resolution control for falling and rising edge delay in half cycle clocking operation
- Self-check diagnostics software mode to check if the micro edge positioner (MEP) logic is running optimally
- Enables high resolution output swapping on the EPWMxA and EPWMxB output
- Enables high-resolution output on EPWMxB signal output via inversion of EPWMxA signal output
- Enables high-resolution period, duty and phase control on the EPWMxA and EPWMxB output on devices with an ePWM module

Note

See the device-specific datasheet to determine if your device has an ePWM module with high-resolution period support.



A. From ePWM Time-base (TB) submodule
B. From ePWM counter-compare (CC) submodule
C. From ePWM Deadband (DB) submodule

Figure 20-77. HRPWM Block Diagram

The ePWM peripheral is used to perform a function mathematically equivalent to a digital-to-analog converter (DAC). As shown in Figure 20-78, the effective resolution for conventionally generated PWM is a function of PWM frequency (or period) and system clock frequency.

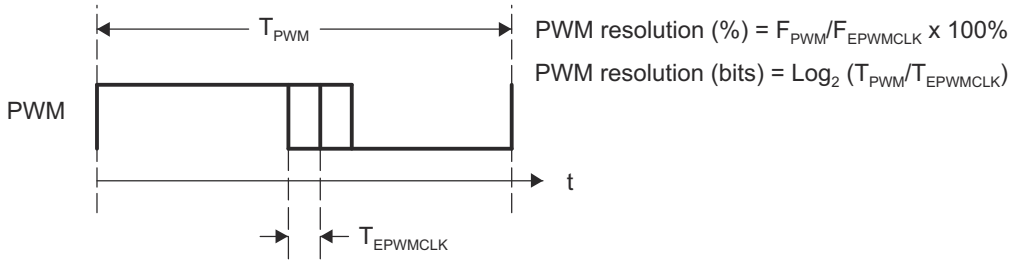


Figure 20-78. Resolution Calculations for Conventionally Generated PWM

If the required PWM operating frequency does not offer sufficient resolution in PWM mode, you may want to consider HRPWM. As an example of improved performance offered by HRPWM, Table 20-15 shows resolution in bits for various PWM frequencies. These values assume a MEP step size of 180 ps. See the device-specific data sheet for typical and maximum performance specifications for the MEP.

Table 20-15. Resolution for PWM and HRPWM

PWM Frequency (kHz)	Regular Resolution (PWM) 100 MHz EPWMCLK		High Resolution (HRPWM)	
	Bits	%	Bits	%
20	12.3	0.02	18.1	0.000
50	11	0.05	16.8	0.001
100	10	0.1	15.8	0.002
150	9.4	0.15	15.2	0.003
200	9	0.2	14.8	0.004
250	8.6	0.25	14.4	0.005
500	7.6	0.5	13.4	0.009
1000	6.6	1	12.4	0.018
1500	6.1	1.5	11.9	0.027
2000	5.6	2	11.4	0.036

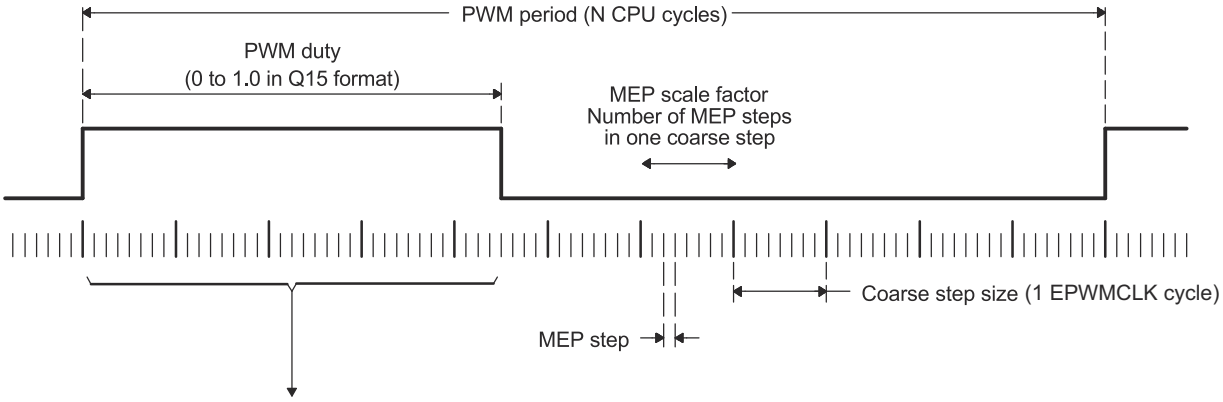
Although each application may differ, typical low frequency PWM operation (below 250 kHz) may not require HRPWM. HRPWM capability is most useful for high frequency PWM requirements of power conversion topologies such as:

- Single-phase buck, boost, and flyback
- Multi-phase buck, boost, and flyback
- Phase-shifted full bridge
- Direct modulation of D-Class power amplifiers

20.15.1 Operational Description of HRPWM

The HRPWM is based on micro-edge positioner (MEP) technology. MEP logic is capable of positioning an edge very finely by sub-dividing one coarse system clock of a conventional PWM generator. The time step accuracy is on the order of 150 ps. See the device-specific data sheet for the typical MEP step size on a particular device. The HRPWM also has a self-check software diagnostics mode to check if the MEP logic is running optimally, under all operating conditions. Details on software diagnostics and functions are in Section 20.15.1.7.

Figure 20-79 shows the relationship between one coarse system clock and edge position in terms of MEP steps, which are controlled via an 8-bit field in the Compare A extension register (CMPAHR). The same operating logic applies to CMPBHR as well.



Number of coarse steps = integer(PWMduty * PWMperiod)
Number of MEP steps = fraction(PWMduty * PWMperiod) * (MEPScaleFactor) + 0.5 (rounding) †

16−bit CMPA register value	= number of coarse steps
16−bit CMPAHR register value	= (number of MEP steps) << 8 (upper 8 bits)

† For MEP range and rounding adjustment. (0x0080 in Q8 format)

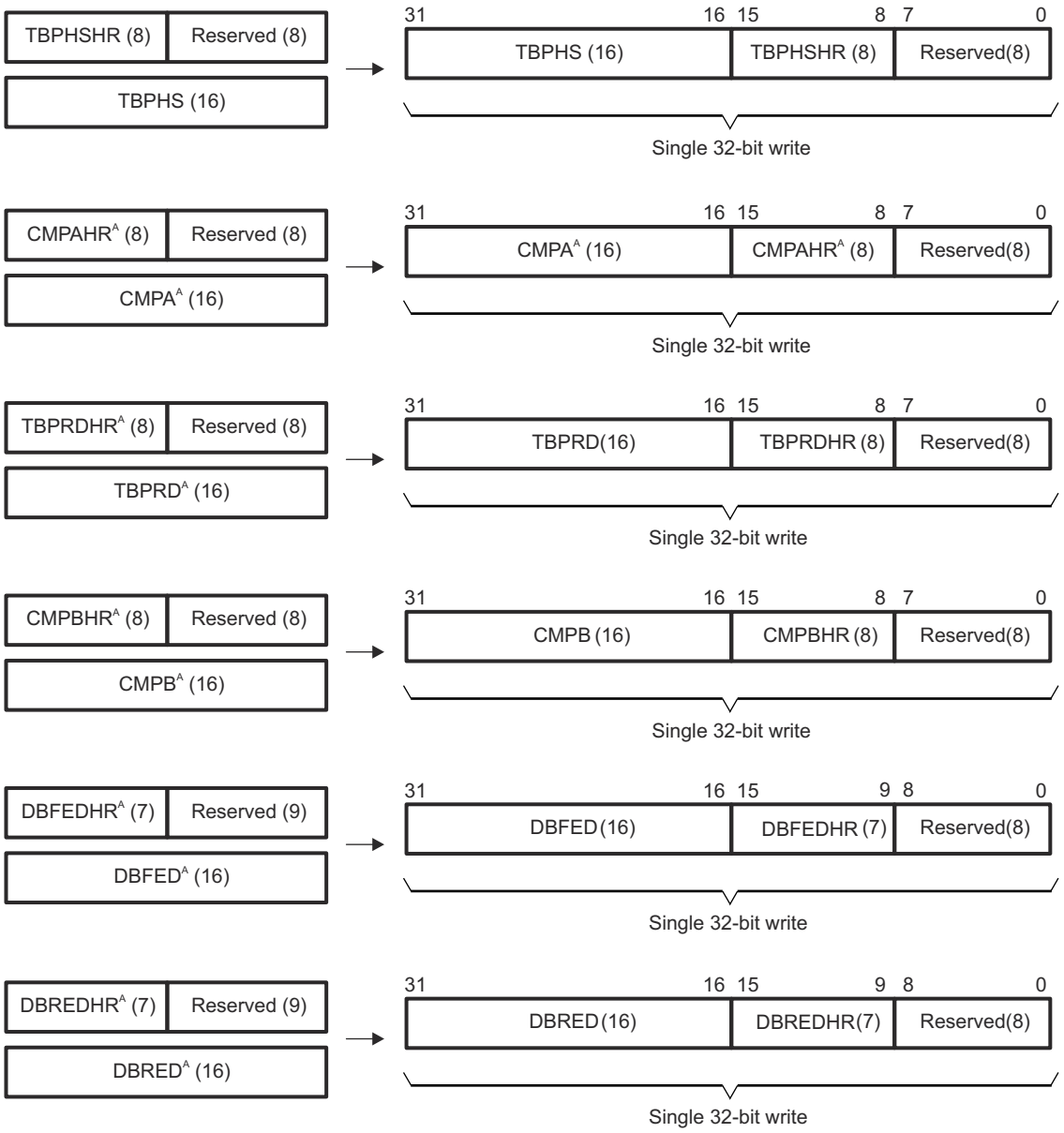
Figure 20-79. Operating Logic Using MEP

To generate an HRPWM waveform, configure the ePWM registers as you would to generate a conventional PWM of a given frequency and polarity. The HRPWM works together with the ePWM registers to extend edge resolution. Although many programming combinations are possible, only a few are needed and practical. These methods are described in Section 20.15.1.8.

20.15.1.1 Controlling the HRPWM Capabilities

The MEP of the HRPWM is controlled by six extension registers. These HRPWM registers are concatenated with the 16-bit TBPHS, TBPRD, CMPA, CMPBM, DBREDM, and DBFEDM registers used to control PWM operation.

- TBPHSHR - Time Base Phase High Resolution Register
- CMPAHR - Counter Compare A High Resolution Register; CMPAHR is for use with the AQ output of Channel A, and is not related to CMPA
- TBPRDHR - Time Base Period High Resolution Register. (available on some devices)
- CMPBHR - Counter Compare B High Resolution Register; CMPBHR is for use with the AQ output of Channel B, and is not related to CMPB
- DBREDHR - Deadband Generator Rising Edge Delay High Resolution Register
- DBFEDHR - Deadband Generator Falling Edge Delay High Resolution Register



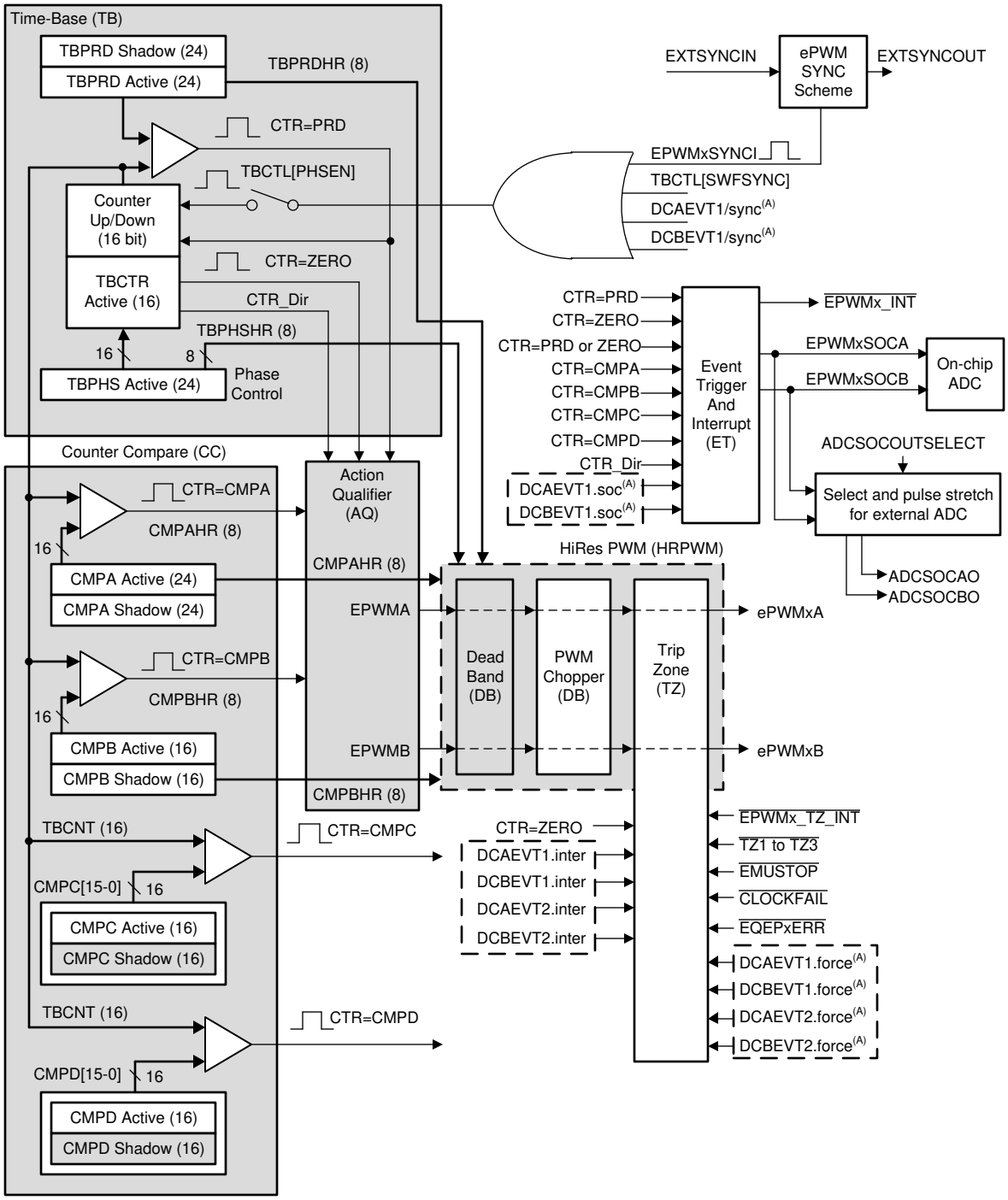
A. Dependant upon your device, these registers may be mirrored and can be written to at two different memory locations. Check the device-specific Technical Reference Manual's *ePWM* chapter for more details on how to read and write to these locations.

Figure 20-80. HRPWM Extension Registers and Memory Configuration

Note

HRPWM capabilities on Deadband Rising Edge Delay and Falling Edge Delay is applicable only during Dead Band half cycle clocking Operation. The number of MEP steps is half in size [bits 15:9] than duty and phase high resolution registers for the same reason

HRPWM capabilities are controlled using the Channel A and B PWM signal path. HRPWM support on the Dead band signal path is available by properly configuring the HRCNFG2 register. Figure 20-81 shows how the HRPWM interfaces with the 8-bit extension registers.



A. These events are generated by the ePWM Digital Compare (DC) submodule based on the levels of the TRIPIN inputs.

Figure 20-81. HRPWM System Interface

20.15.1.2 HRPWM Source Clock

Each HRPWM module is clocked from the respective EPWMxCLK. HRCAL has a separate clock. For example, HRPWM1 is sourced from EPWM1CLK while HRPWM2 is clocked from the EPWM2CLK. Figure 20-82 shows the HRCAL and HRPWM modules are sourced from their respective ePWM clock source.

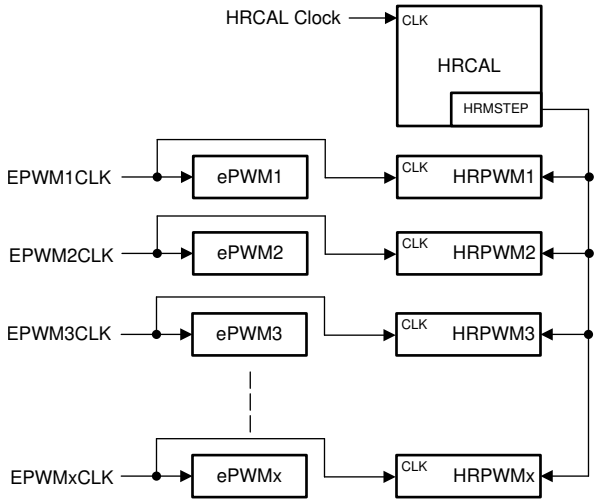


Figure 20-82. HRPWM and HRCAL Source Clock

20.15.1.3 Configuring the HRPWM

Once the ePWM has been configured to provide conventional PWM of a given frequency and polarity, the HRPWM is configured by programming the HRCNFG register in that particular ePWM module's register space. This register provides the following configuration options:

Edge Mode	The MEP can be programmed to provide precise position control on the rising edge (RE), falling edge (FE) or both edges (BE) at the same time. FE and RE are used for power topologies requiring duty cycle control(CMPA or CMPB high-resolution control), while BE is used for topologies requiring phase shifting, for example, phase shifted full bridge (TBPHS or TBPRD high-resolution control).
Control Mode	The MEP is programmed to be controlled either from the CMPAHR / CMPBHR register in case of duty cycle control or the TBPHSHR register (phase control). RE or FE control mode should be used with CMPAHR or CMPBHR register. BE control mode should be used with TBPHSHR register. When the MEP is controlled from the TBPRDHR register (period control) the duty cycle and phase can also be controlled via their respective high-resolution registers.
Shadow Mode	This mode provides the same shadowing (double buffering) option as in regular PWM mode. This option is valid only when operating from the CMPAHR, CMPBHR and TBPRDHR registers and should be chosen to be the same as the regular load option for the CMPA, CMPB register. If TBPHSHR is used, then this option has no effect.
High-Resolution B Signal Control	The B signal path of an ePWM channel can generate a high-resolution output by outputting an inverted version of the high-resolution ePWMxA signal on the ePWMxB pin. A Type 2 or Type 4 HRPWM module can also enable high-resolution features on the B signal path indepedently of the A signal path as well.
Swap ePWMxA and ePWMxB Outputs	This mode enables the swapping of the high resolution A and B outputs. The mode selection allows either A and B Outputs Unchanged or A Output Comes Out On B and B Output Comes Out On A.
Auto-conversion Mode	This mode is used in conjunction with the scale factor optimization (SFO) software only. For a type 4 HRPWM module, below is a description of the Auto-conversion Mode taking CMPAHR as an example. If auto-conversion is enabled, CMPAHR = fraction(PWMduty*PWMperiod)<<8. The scale factor optimization software will calculate the MEP scale factor in background code and automatically update the HRMSTEP register with the calculated number of MEP steps per coarse step. The MEP Calibration Module will then use the values in the HRMSTEP and CMPAHR register to automatically calculate the appropriate number of MEP steps represented by the fractional duty cycle and move the high-resolution ePWM signal edge accordingly. If auto-conversion is disabled, the CMPAHR register behaves like a type 0 HRPWM module and CMPAHR = (fraction(PWMduty * PWMperiod) * MEP Scale Factor + 0.5)<<8). All calculations will need to be performed by user code in this mode, and the HRMSTEP register is ignored. Auto-conversion for high-resolution period has the same behavior as auto-conversion for high-resolution duty cycle. Auto-conversion must always be enabled for high-resolution period mode.

Note

If the HRPWM module is configured in UP-DOWN counter mode, the shadow mode for the HRPWM registers must be set to load on both ZERO AND PERIOD. New values from the user are loaded to the shadow registers only at CTR=ZERO, but the shadow mode of for the registers must be set to both ZERO AND PERIOD. The CTR=PRD event is used for specific internal logic inside the HRPWM module.

Auto-conversion Mode performs the calculation for CMPBHR , DBREDHR and DBFEDHR as well. The scale factor optimization software will calculate the MEP scale factor in background code and automatically update the HRMSTEP register with the calculated number of MEP steps per coarse step. The MEP Calibration Module will then use the values in the HRMSTEP and CMPBHR or DBREDHR / DBFEDHR register to automatically calculate the appropriate number of MEP steps represented by the fractional components and move the high-resolution ePWM signal edge accordingly. If auto-conversion is disabled, CMPBHR behaves same as CMPAHR. CMPBHR = (fraction(PWMduty * PWMperiod) * MEP Scale Factor + 0.5)<<8).

20.15.1.4 Configuring Hi-Res in Deadband Rising Edge and Falling Edge Delay

Once the ePWM has been configured to provide conventional PWM of a given frequency, polarity and deadband enabled in half cycle clocking mode, the high resolution operation on dead band RED and FED lines are enabled by programming the HRCNFG2 register in that particular ePWM module's register space. This register provides the following configuration options:

Edge Mode The MEP can be programmed to provide precise position control on the dead band rising edge (RED), dead band falling edge (FED), or both edges (rising edge of DBRED signal and falling edge of DBFED signal) at the same time.

Control Mode Selects the time event that loads the shadow value in active register for DBRED and DBFED in high resolution mode. You need to select the pulse to match the selection in the ePWM DBCTL[LOADREDMODE] and DBCTL[LOADFEDMODE] bits.

20.15.1.5 Principle of Operation

The MEP logic is capable of placing an edge in one of 255 (8 bits) discrete time steps (see device-specific data sheet for typical MEP step size). The MEP works with the TBM and CCM registers to be certain that time steps are optimally applied and that edge placement accuracy is maintained over a wide range of PWM frequencies, system clock frequencies and other operating conditions. Table 20-16 shows the typical range of operating frequencies supported by the HRPWM.

Table 20-16. Relationship Between MEP Steps, PWM Frequency and Resolution

System (MHz)	MEP Steps Per EPWMCLK (1) (2) (3)	PWM Minimum (Hz) (4)	PWM Maximum (MHz)	Resolution at Maximum (Bits) (5)
60.0	93	916	3.00	10.9
70.0	79	1068	3.50	10.6
80.0	69	1221	4.00	10.4
90.0	62	1373	4.50	10.3
100.0	56	1526	5.00	10.1

(1) TBCLK = EPWMCLK.
(2) Table data based on a MEP time resolution of 180 ps (this is an example value. See the device-specific data sheet for MEP limits)
(3) MEP steps applied = $T_{EPWMCLK}$/180 ps in this example.
(4) PWM minimum frequency is based on a maximum period value,(TBPRD = 65535). PWM mode is asymmetrical up-count.
(5) Resolution in bits is given for the maximum PWM frequency stated.

20.15.1.5.1 Edge Positioning

Note

The following example is presented using the [CMPA:CMPAHR] register combination. The theory of operation and equations is the same if you intend to use [CMPBM:CMPBHRM] for duty cycle control.

In a typical power control loop, a digital controller issues a duty command, usually expressed in a per unit or percentage terms. Assume that for a particular operating point, the demanded duty cycle is 0.405 or 40.5% on time and the required converter PWM frequency is 1.25 MHz. In conventional PWM generation with a system clock of 100 MHz, the duty cycle choices are in the vicinity of 40.5%. As shown in Figure 20-83, a compare value of 32 counts (duty = 40%) is the closest to 40.5% that can be attained. This is equivalent to an edge position of 320 ns instead of the desired 324 ns. This data is shown in Table 20-17.

By utilizing the MEP, you can achieve an edge position much closer to the desired point of 324 ns. Table 20-17 shows that in addition to the CMPA value, 22 steps of the MEP (CMPAHR register) will position the edge at 323.96 ns, resulting in almost zero error. In this example, it is assumed that the MEP has a step resolution of 180 ps.

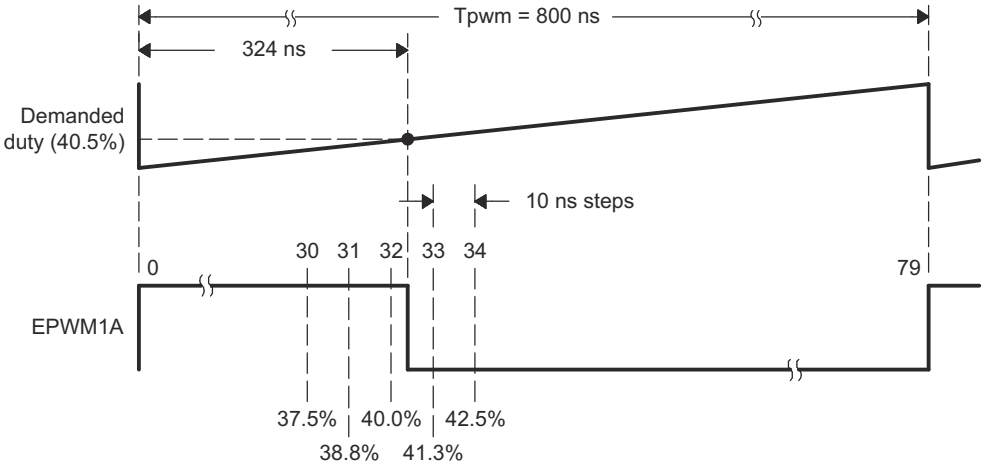


Figure 20-83. Required PWM Waveform for a Requested Duty = 40.5%

Table 20-17. CMPA vs Duty (left), and [CMPA:CMPAHR] vs Duty (right)

CMPA (count)(1) (2) (3)	Duty (%)	High Time (ns)	CMPA (count)	CMPAHR (count)	Duty (%)	High Time (ns)
28	***35.0***	280	32	18	***40.405***	323.24
29	***36.3***	290	32	19	***40.428***	323.42
30	***37.5***	300	32	20	***40.450***	323.60
31	***38.8***	310	32	21	***40.473***	323.78
32	***40.0***	320	32	22	***40.495***	323.96
33	***41.3***	330	32	23	***40.518***	324.14
34	***42.5***	340	32	24	***40.540***	324.32
			32	25	***40.563***	324.50
Required			32	26	***40.585***	324.68
32.40	***40.5***	324	32	27	***40.608***	324.86

(1) Assumed MEP step size for the above example = 180 ps. See the device-specific data manual for typical and maximum MEP values.
(2) TBCLK = 100 MHz, 10 ns
(3) For a PWM Period register value of 80 counts, PWM Period = 80 x 10 ns = 800 ns, PWM frequency = 1/800 ns = 1.25 MHz

20.15.1.5.2 Scaling Considerations

The mechanics of how to position an edge precisely in time has been demonstrated using the resources of the standard CMPA and MEP (CMPAHR) registers. In a practical application, however, it is necessary to seamlessly provide the CPU a mapping function from a per-unit (fractional) duty cycle to a final integer (non-fractional) representation that is written to the [CMPA:CMPAHR] register combination.

To do this, first examine the scaling or mapping steps involved. It is common in control software to express duty cycle in a per-unit or percentage basis. This has the advantage of performing all needed math calculations without concern for the final absolute duty cycle, expressed in clock counts or high time in ns. Furthermore, it makes the code more transportable across multiple converter types running different PWM frequencies.

To implement the mapping scheme, a two-step scaling procedure is required.

Assumptions for this example:

TBCLK	=	10 ns (100 MHz)
PWM frequency	=	1.25 MHz (1/800 ns)
Required PWM duty cycle, **PWMDuty**	=	0.405 (40.5%)
PWM period in terms of coarse steps, **PWMPeriod** (800 ns/10 ns)	=	80
Number of MEP steps per coarse step at 180 ps (10 n/180 ps), **MEP_ScaleFactor**	=	55
Value to keep CMPAHR within the range of 1-255 and fractional rounding constant (default value)	=	0.5 (0080h in Q8 format)

Step 1: Percentage Integer Duty value conversion for CMPA register

CMPA register value	=	int(**PWMDuty***PWMPeriod**); int means integer part
	=	int(0.405 * 80)
	=	int(32.4)
CMPA register value	=	32 (20h)

Step 2: Fractional value conversion for CMPAHR register

CMPAHR	=	(frac(**PWMDuty***PWMPeriod**)*MEP_ScaleFactor** +0.5) << 8); frac means fractional part
	=	(frac(32.4) * 55 + 0.5) << 8; Shifting is to move the value to the high byte of CMPAHR.
	=	(0.4 * 55 + 0.5) << 8
	=	(22 + 0.5) << 8
	=	22.5 * 256; Shifting left by 8 is the same as multiplying by 256.
	=	5760 (1680h)
CMPAHR	=	1680h CMPAHR value = 1600h (lower 8 bits will be ignored by hardware).

Note

If the AUTOCONV bit (HRCNFG.6) is set and the MEP_ScaleFactor is in the HRMSTEP register, then CMPAHR / CMPBHR register value = frac (PWMDuty*PWMperiod<<8). The rest of the conversion calculations are performed automatically in hardware, and the correct MEP-scaled signal edge appears on the ePWM channel output. If AUTOCONV is not set, the above calculations must be performed by software.

The MEP scale factor (MEP_ScaleFactor) varies with the system clock and DSP operating conditions. TI provides an MEP scale factor optimizing (SFO) software C function, which uses the built in diagnostics in each HRPWM and returns the best scale factor for a given operating point.

The scale factor varies slowly over a limited range so the optimizing C function can be run very slowly in a background loop.

The CMPA, CMPB, CMPAHR and CMPBHR registers are configured in memory so that the 32-bit data capability of the C28x CPU can write this as a single concatenated value, that is, [CMPA:CMPAHR], [CMPB:CMPBHR], and so on.

The mapping scheme has been implemented in both C and assembly, as shown in Section 20.15.1.8. The actual implementation takes advantage of the 32-bit CPU architecture of the C28x, and is somewhat different from the steps shown in Section 20.15.1.5.2.

For time-critical control loops where every cycle counts, the assembly version is recommended. This is a cycle optimized function (11 EPWMCLK cycles) that takes a Q15 duty value as input and writes a single [CMPA:CMPAHR] value.

20.15.1.5.3 Duty Cycle Range Limitation

In high resolution mode, the MEP is not active for 100% of the PWM period. It becomes operational:

- Three EPWMCLK cycles after the period starts when high-resolution period (TBPRDHR) control is not enabled.
- When high resolution period (TBPRDHR) control is enabled via the HRPCTL register:
  - In up-count mode: three EPWMCLK cycles after the period starts until three EPWMCLK cycles before the period ends.
  - In up-down count mode: when counting up, three cycles after CTR = 0 until three cycles before CTR = PRD, and when counting down, three cycles after CTR = PRD until three cycles before CTR = 0.
- When using DBREDHR or DBFEDHR, DBRED and/or DBFED (the register corresponding to the edge with hi-resolution displacement) must be greater than or equal to 7.

Duty cycle range limitations are illustrated in Figure 20-84 to Figure 20-87. This limitation imposes a duty cycle limit on the MEP. For example, precision edge control is not available all the way down to 0% duty cycle. When high-resolution period control is disabled, regular PWM duty control is fully operational down to 0% duty cycle despite the unavailability of HRPWM features in the first three cycles. In most applications this should not be an issue as the controller regulation point is usually not designed to be close to 0% duty cycle. To better understand the useable duty cycle range, see Table 20-18. When high-resolution period control is enabled (HRPCTL[HRPE]=1), the duty cycle must not fall within the restricted range. Otherwise, there may be undefined behavior on the ePWMxA output.

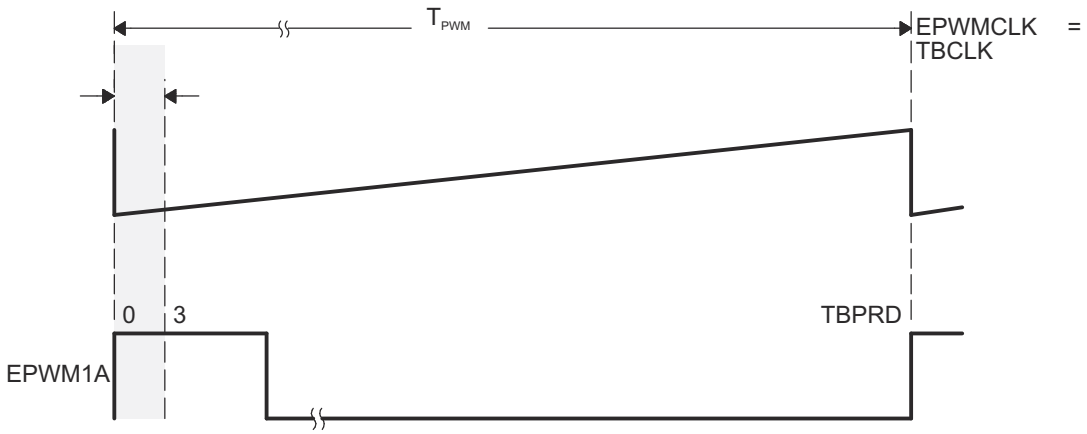


Figure 20-84. Low % Duty Cycle Range Limitation Example (HRPCTL[HRPE] = 0)

Table 20-18. Duty Cycle Range Limitation for Three EPWMCLK/TBCLK Cycles

PWM Frequency(1) (kHz)	3 Cycles Minimum Duty	3 Cycles Maximum Duty(2)
200	0.6%	99.4%
400	1.2%	98.8%
600	1.8%	98.2%
800	2.4%	97.6%
1000	3%	97%
1200	3.6%	96.4%
1400	4.2%	95.8%
1600	4.8%	95.2%
1800	5.4%	94.6%
2000	6%	94%

(1) EPWMCLK = TBCLK = 100 MHz
(2) This limitation applies only if high-resolution period (TBPRDHR) control is enabled.

If the application demands HRPWM operation below the minimum duty cycle limitation, then the HRPWM can be configured to operate in count-down mode with the rising edge position (REP) controlled by the MEP when high-resolution period is disabled (HRPCTL[HRPE] = 0). This is illustrated in Figure 20-85. In this configuration, the minimum duty cycle limitation is no longer an issue. However, there will be a maximum duty limitation with same percent numbers as given in Table 20-18.

CAUTION

If the application has enabled high-resolution period control (HRPCTL[HRPE]=1), the duty cycle must not fall within the restricted range. Otherwise, there will be undefined behavior on the ePWM output.

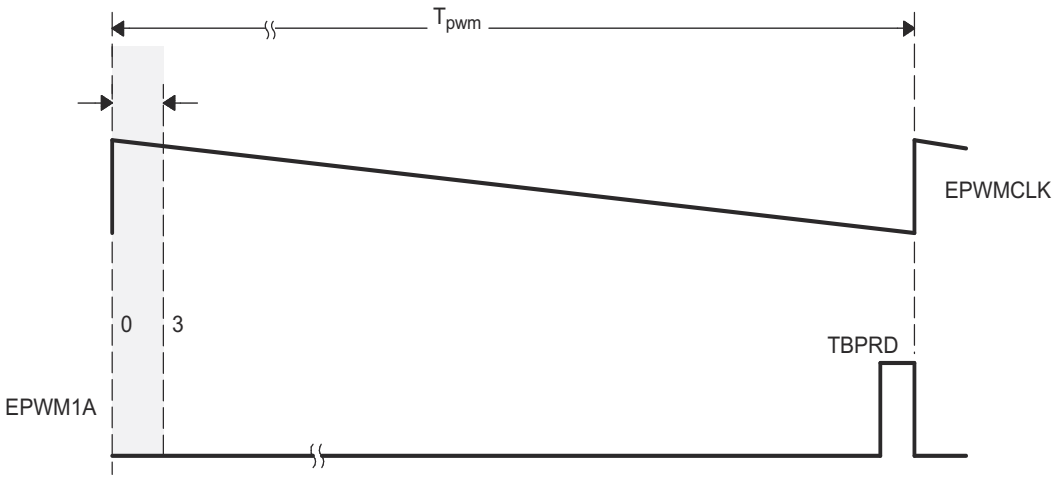


Figure 20-85. High % Duty Cycle Range Limitation Example (HRPCTL[HRPE] = 0)

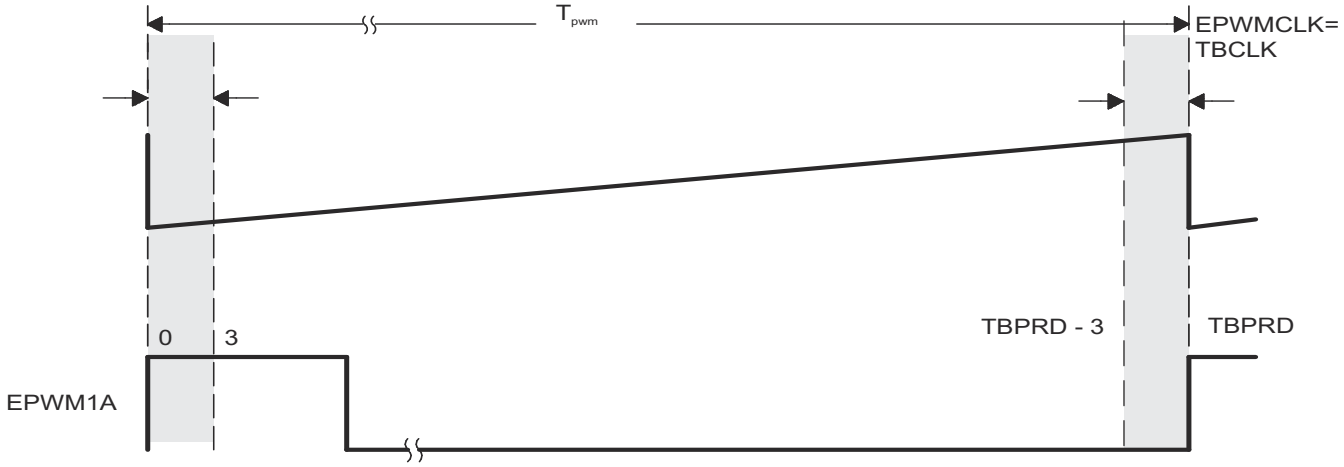


Figure 20-86. Up-Count Duty Cycle Range Limitation Example (HRPCTL[HRPE]=1)

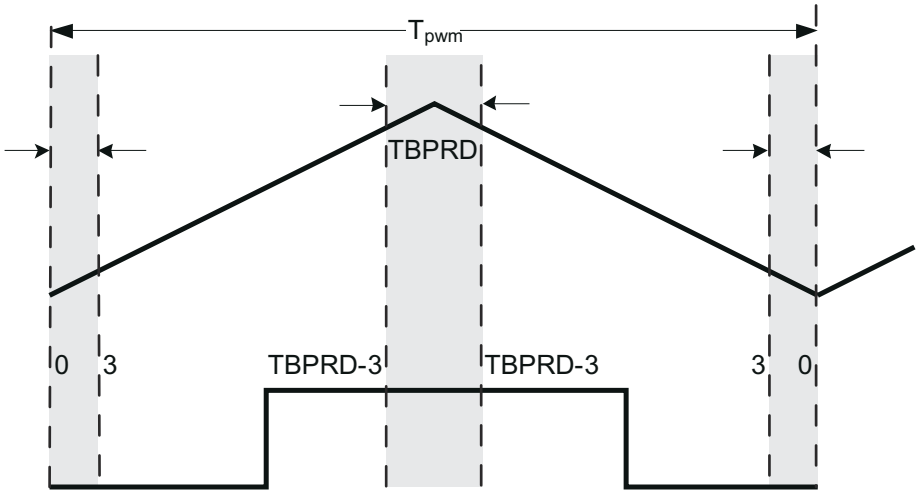


Figure 20-87. Up-Down Count Duty Cycle Range Limitation Example (HRPCTL[HRPE]=1)

20.15.1.5.4 High Resolution Period

High resolution period control using the MEP logic is supported on devices with a Type 1 ePWM module or greater.

Note

When high-resolution period control is enabled, on ePWMxA only, and not ePWMxB output and vice versa, the non hi-res output will have +/- 1 TBCLK cycle jitter in up-count mode and +/- 2 TBCLK cycle jitter in up-down count mode.

The scaling procedure described for duty cycle in Section 20.15.1.5.2 applies for high-resolution period as well:

Assumptions for this example:

TBCLK	= 10 ns (100 MHz)
Required PWM frequency	= 175 kHz (period of 571.428)
Number of MEP steps per coarse step at 180 ps (MEP_ScaleFactor)	= 55 (10 ns / 180 ps)
Value to keep TBPRDHR within range of 1-255 and fractional rounding constant (default value)	= 0.5 (0080h in Q8 format)

Problem:

In up-count mode:

If TBPRD = 571, then PWM frequency = 174.82 kHz (period = (571+1)* T_{TBCLK}).

If TBPRD = 570, then PWM frequency = 175.13 kHz (period = (570+1)* T_{TBCLK}).

In up-down count mode:

If TBPRD = 286, then PWM frequency = 174.82 kHz (period = (286*2)* T_{TBCLK}).

If TBPRD = 285, then PWM frequency = 175.44 kHz (period = (285*2)* T_{TBCLK}).

Solution:

With 55 MEP steps per coarse step at 180 ps each:

Step 1: Percentage Integer Period value conversion for TBPRD register

Integer period value	= 571 * T_{TBCLK}
	= int (571.428) * T_{TBCLK}
	= int (PWMperiod) * T_{TBCLK}
In up-count mode:	
TBPRD	= 570 (TBPRD = period value - 1)
	= 023Ah
In up-down count mode:	= 285 (TBPRD = period value / 2)
TBPRD	= 011Dh

Step 2: Fractional value conversion for TBPRDHR register

TBPRDHR register value	= (frac(PWMperiod) * MEP_ScaleFactor + 0.5)
If auto-conversion enabled and HRMSTEP =	
MEP_ScaleFactor value (55):	=frac (PWMperiod) << 8 (Shifting is to move the value to the high byte of TBPRDHR)
TBPRDHR register value	=frac (571.428) << 8
	=0.428 × 256
	=6D00h

The autoconversion will then automatically perform the calculation such that TBPRDHR MEP delay is scaled by hardware to:	=((TBPRDHR(15:0) >> 8) × HRMSTEP + 80h) << 8
	= (006Dh × 55 + 80h) >> 8
	=(17EBh) >> 8
Period MEP delay	=0017h MEP Steps

20.15.1.5.4.1 High-Resolution Period Configuration

To use High Resolution Period, the ePWMx module must be initialized in the exact order presented.

The steps below use CMPA with shadow registers and the corresponding HRCNFG bits for hi-resolution operation on EPWMxA. For hi-resolution operation on EPWMxB, make the appropriate substitutions with the B channel fields.

1. Enable ePWMx clock
2. Enable HRPWM clock
3. Disable TBCLKSYNC
4. Configure ePWMx registers - AQ, TBPRD, CC, and so on.
   - ePWMx may only be configured for up-count or up-down count modes. High-resolution period is not compatible with down-count mode.
   - TBPRD and CC registers must be configured for shadow loads.
   - CMPCTL[LOADAMODE]
     - In up-count mode: CMPCTL[LOADAMODE] = 1 (load on CTR = PRD)
     - In up-down count mode: CMPCTL[LOADAMODE] = 2 (load on CTR=0 or CTR=PRD)
5. Configure the HRCNFG register such that:
   - HRCNFG[HRLOAD] = 2 (load on either CTR = 0 or CTR = PRD)
   - HRCNFG[AUTOCONV] = 1 (Enable auto-conversion)
   - HRCNFG[EDGMODE] = 3 (MEP control on both edges)
6. For TBPHS:TBPHSHR synchronization with high-resolution period, set both HRPCTL[TBPSHRLOADE] = 1 and TBCTL[PHSEN] = 1. In up-down count mode these bits must be set to 1 regardless of the contents of TBPHSHR.
7. Enable high-resolution period control (HRPCTL[HRPE] = 1)
8. Enable TBCLKSYNC
9. TBCTL[SWFSYNC] = 1
10. HRMSTEP must contain an accurate MEP scale factor (# of MEP steps per EPWMCLK coarse step) because auto-conversion is enabled. The MEP scale factor can be acquired via the SFO() function described in Section 20.15.2.
11. To control high-resolution period, write to the TBPRDHR(M) registers.

Note

When high-resolution period mode is enabled, an EPWMxSYNC pulse will introduce +/- 1 - 2 cycle jitter to the PWM (+/- 1 cycle in up-count mode and +/- 2 cycle in up-down count mode). For this reason, EPWMxSYNCO source should not be set to CTR = 0 or CTR = CMPB. Otherwise, the jitter will occur on every PWM cycle with the synchronization pulse.

When EPWMxSYNCI is EPWMxSYNCO source, a software synchronization pulse should be issued only once during high-resolution period initialization. If a software sync pulse is applied while the PWM is running, the jitter will appear on the PWM output at the time of the sync pulse.

20.15.1.6 Deadband High Resolution Operation

Note

In up-count mode, the deadband module is **NOT** available when any high resolution mode is enabled.

Assumptions for this example:

System clock	= 10 ns (100 MHz)
Deadband Enabled in half cycle mode, TBCLK = EPWMCLK	
Required PWM frequency	1.33 MHz (1 / 750 ns)
Required PWM duty cycle	0.5 (50%)
Required Dead band Rising Edge Delay	5% over duty
Required Dead band Rising Edge Delay in ns	(0.05 * 375 ns) = 18.75 ns

Note

Just like the duty cycle restrictions when using HRPWM, the DBRED and DBFED values must be greater than 3 to use high resolution deadband.

Deadband Delay Values as a Function of DBFED and DBRED:

When half-cycle clocking is enabled, the formula to calculate the falling-edge-delay and rising-edge-delay becomes:

FED = DBFED * TBCLK / 2

RED = DBRED * TBCLK / 2

DBRED and DBFED Calculated Values:

Required Dead band Rising Edge Delay in ns = 18.75 ns

DBRED = RED / (TBCLK / 2)

DBRED = 18.75 ns / 5 ns

DBRED Required = 3.75 ns

With 55 MEP steps per coarse step at 180 ps each:

Step 1: Integer Dead band value conversion for DBREDM register

Integer DBRED value	= int (RED / (TBCLK / 2))
	= int (3.75)
DBRED	= 3

Step 2: Fractional value conversion for Dead band high resolution register DBREDHR

DBREDHR register value	= (frac(DBRED Required) * MEP_ScaleFactor + 0.5) << 8 (Shifting is to move the value to the high byte of DBREDHR)
	= (frac (3.75) * 55 + 0.5) << 8
	= (0.75 * 55 + 0.5) << 8
	= (41.75) * 256 Shifting left by 8 is the same as multiplying by 256.
DBREDHR value	= 29C0h MEP Steps
	Hardware will ignore lower 9 bits in the above calculated DBREDHR value

Note

If the AUTOCONV bit (HRCNFG.6) is set and the MEP_ScaleFactor is in the HRMSTEP register, then DBREDHR:DBRED = frac((required DB value) < <8). The rest of the conversion calculations are performed automatically in hardware, and the correct MEP-scaled signal edge appears on the ePWM channel output. If AUTOCONV is not set, the above calculations must be performed by software.

20.15.1.7 Scale Factor Optimizing Software (SFO)

The micro edge positioner (MEP) logic is capable of placing an edge in one of 255 discrete time steps. As previously mentioned, the size of these steps is on the order of 150 ps (see device-specific data sheet for typical MEP step size on your device). The MEP step size varies based on worst-case process parameters, operating temperature, and voltage. MEP step size increases with decreasing voltage and increasing temperature and decreases with increasing voltage and decreasing temperature. Applications that use the HRPWM feature should use the TI-supplied MEP scale factor optimization (SFO) software function. The SFO function helps to dynamically determine the number of MEP steps per EPWMCLK period while the HRPWM is in operation.

To utilize the MEP capabilities effectively, the correct value for the MEP scaling factor needs to be known by the software. To accomplish this, the HRPWM module has built in self-check and diagnostic capabilities that can be used to determine the optimum MEP scale factor value for any operating condition. TI provides a C-callable library containing one SFO function that utilizes this hardware and determines the optimum MEP scale factor. As such, MEP control and diagnostics registers are reserved for TI use.

A detailed description of the SFO library - SFO_TI_Build_V8.lib software can be found in Section 20.15.2.

20.15.1.8 HRPWM Examples Using Optimized Assembly Code

The best way to understand how to use the HRPWM capabilities is through two real examples:

1. Simple buck converter using asymmetrical PWM (count-up) with active high polarity.
2. DAC function using simple R+C reconstruction filter.

The following examples all have initialization and configuration code written in C. To make these easier to understand, the #defines shown below are used.

Example 20-2 assumes MEP step size of 150 ps and does not use the SFO library.

Example 20-2. #Defines for HRPWM Header Files

```
// HRPWM (High Resolution PWM) //
================================
// HRCNFG
#define HR_Disable 0x0
#define HR_REP 0x1          // Rising Edge position
#define HR_FEP 0x2          // Falling Edge position
#define HR_BEP 0x3          // Both Edge position #define HR_CMP 0x0 // CMPAHR controlled
#define HR_PHS 0x1          // TBPHSHR controlled #define HR_CTR_ZERO 0x0 // CTR = Zero event
#define HR_CTR_PRD 0x1      // CTR = Period event
#define HR_CTR_ZERO_PRD 0x2 // CTR = ZERO or Period event
#define HR_NORM_B  0x0      // Normal ePWMxB output
#define HR_INVERT_B 0x1     // ePWMxB is inverted ePWMxA output
```

20.15.1.8.1 Implementing a Simple Buck Converter

In this example, the PWM requirements are:

- PWM frequency = 1 MHz (that is, TBPRD = 100)
- PWM mode = asymmetrical, up-count
- Resolution = 12.7 bits (with a MEP step size of 150 ps)

Figure 20-88 and Figure 20-89 show the required PWM waveform. As explained previously, configuration for the ePWM1 module is almost identical to the normal case except that the appropriate MEP options need to be enabled/selected.

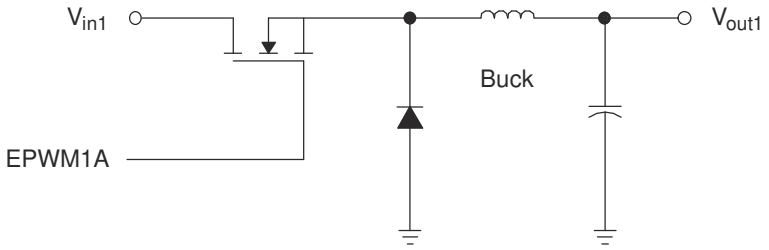


Figure 20-88. Simple Buck Controlled Converter Using a Single PWM

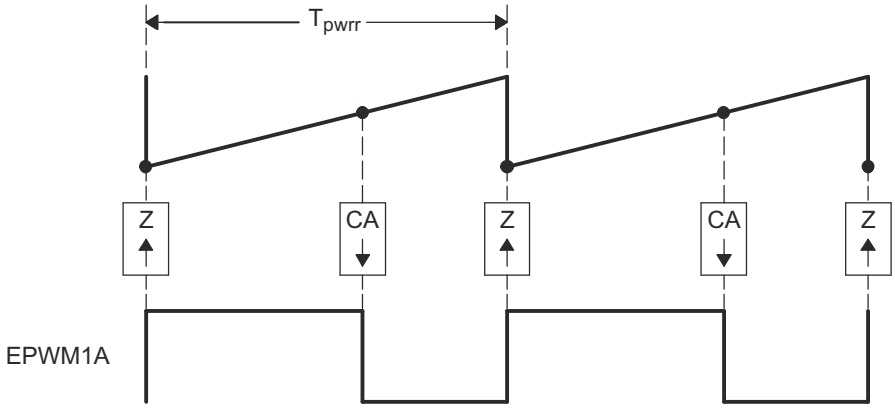


Figure 20-89. PWM Waveform Generated for Simple Buck Controlled Converter

The example code shown consists of two main parts:

- Initialization code (executed once)
- Run time code (typically executed within an ISR)

Example 20-3 shows the Initialization code. The first part is configured for conventional PWM. The second part sets up the HRPWM resources.

This example assumes MEP step size of 150 ps and does not use the SFO library.

Example 20-4 shows an assembly example of run-time code for the HRPWM buck converter.

Example 20-3. HRPWM Buck Converter Initialization Code

```
void HrBuckDrvCnf(void)
 {
// Config for conventional PWM first
EPwm1Regs.TBCTL.bit.PRDLD = TB_IMMEDIATE;         // set Immediate load
EPwm1Regs.TBPRD = 100;                            // Period set for 1000 kHz PWM
hrbuck_period = 200;                              // Used for Q15 to Q0 scaling
EPwm1Regs.TBCTL.bit.CTRMODE = TB_COUNT_UP;
EPwm1Regs.TBCTL.bit.PHSEN = TB_DISABLE;           // EPWM1 is the Master

EPwm1Regs.EPWMSYNCOUTEN.all = TB_SYNC_DISABLE;
EPwm1Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;
EPwm1Regs.TBCTL.bit.CLKDIV = TB_DIV1;
// Note: ChB is initialized here only for comparison purposes, it is not required

EPwm1Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;
EPwm1Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
EPwm1Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO;     // optional
EPwm1Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;       // optional
EPwm1Regs.AQCTLA.bit.ZRO = AQ_SET;
EPwm1Regs.AQCTLA.bit.CAU = AQ_CLEAR;
EPwm1Regs.AQCTLB.bit.ZRO = AQ_SET;                // optional
EPwm1Regs.AQCTLB.bit.CBU = AQ_CLEAR;              // optional
// Now configure the HRPWM resources
EALLOW;                                           // Note these registers are protected
                                                  // and act only on ChA
EPwm1Regs.HRCNFG.all = 0x0;                       // clear all bits first
EPwm1Regs.HRCNFG.bit.EDGMODE = HR_FEP;            // Control Falling Edge Position
EPwm1Regs.HRCNFG.bit.CTLMODE = HR_CMP;            // CMPAHR controls the MEP
EPwm1Regs.HRCNFG.bit.HRLOAD = HR_CTR_ZERO;        // Shadow load on CTR=Zero
EDIS;
MEP_ScaleFactor = 66*256;                         // Start with typical Scale Factor
                                                  // value for 100 MHz
                                                  // Note: Use SFO functions to update
                                                     MEP_ScaleFactor dynamically
}
```

Example 20-4. HRPWM Buck Converter Run-Time Code

```
EPWM1_BASE .set 0x6800
CMPAHR1 .set EPWM1_BASE+0x8
;================================================
HRBUCK_DRV; (can execute within an ISR or loop)
;================================================
     MOVW DP, #_HRBUCK_In
     MOVL XAR2,@_HRBUCK_In       ; Pointer to Input Q15 Duty (XAR2)
     MOVL XAR3,#CMPAHR1          ; Pointer to HRPWM CMPA reg (XAR3)

; Output for EPWM1A (HRPWM)
     MOV T,*XAR2 ; T <= Duty
     MPYU ACC,T,@_hrbuck_period  ; Q15 to Q0 scaling based on Period
     MOV T,@_MEP_ScaleFactor     ; MEP scale factor (from optimizer s/w)
     MPYU P,T,@AL                ; P <= T * AL, Optimizer scaling
     MOVH @AL,P                  ; AL <= P, move result back to ACC
     ADD ACC, #0x080             ; MEP range and rounding adjustment
     MOVL *XAR3,ACC              ; CMPA:CMPAHR(31:8) <= ACC

; Output for EPWM1B (Regular Res) Optional - for comparison purpose only
     MOV *+XAR3[2],AH            ; Store ACCH to regular CMPB
```

20.15.1.8.2 Implementing a DAC Function Using an R+C Reconstruction Filter

In this example, the PWM requirements are:

- PWM frequency = 400 kHz (that is, TBPRD = 250)
- PWM mode = Asymmetrical, Up-count
- Resolution = 14 bits (MEP step size = 150 ps)

Figure 20-90 and Figure 20-91 show the DAC function and the required PWM waveform. As explained previously, configuration for the ePWM1 module is almost identical to the normal case except that the appropriate MEP options need to be enabled/selected.

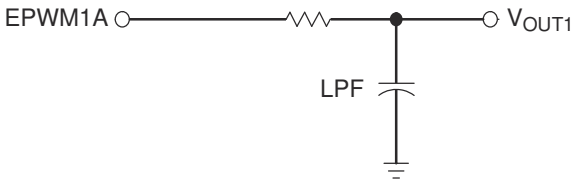


Figure 20-90. Simple Reconstruction Filter for a PWM-based DAC

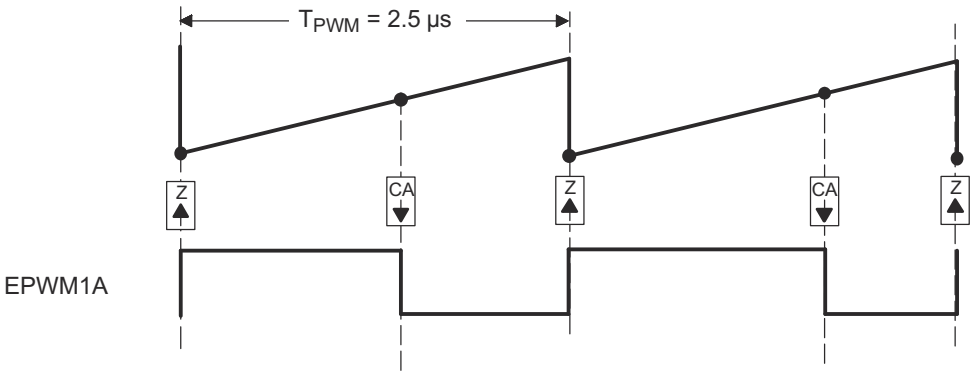


Figure 20-91. PWM Waveform Generated for the PWM DAC Function

The example code shown consists of two main parts:

- Initialization code (executed once)
- Run time code (typically executed within an ISR)

This example assumes a typical MEP_SP and does not use the SFO library.

Example 20-5 shows the Initialization code. The first part is configured for conventional PWM. The second part sets up the HRPWM resources.

Example 20-6 shows an assembly example of run-time code that can execute in a high-speed ISR loop.

Example 20-5. PWM DAC Function Initialization Code

```
void HrPwmDacDrvCnf(void)
{
// Config for conventional PWM first
EPwm1Regs.TBCTL.bit.PRDLD = TB_IMMEDIATE;      // Set Immediate load
EPwm1Regs.TBPRD = 250;                         // Period set for 400 kHz PWM
hrDAC_period = 250;                            // Used for Q15 to Q0 scaling
EPwm1Regs.TBCTL.bit.CTRMODE = TB_COUNT_UP;
EPwm1Regs.TBCTL.bit.PHSEN = TB_DISABLE;        // EPWM1 is the Master

EPwm1Regs.EPWMSYNCOUTEN.all = TB_SYNC_DISABLE;
EPwm1Regs.TBCTL.bit.HSPCLKDIV = TB_DIV1;
EPwm1Regs.TBCTL.bit.CLKDIV = TB_DIV1;
// Note: ChB is initialized here only for comparison purposes, it is not required

EPwm1Regs.CMPCTL.bit.LOADAMODE = CC_CTR_ZERO;
EPwm1Regs.CMPCTL.bit.SHDWAMODE = CC_SHADOW;
EPwm1Regs.CMPCTL.bit.LOADBMODE = CC_CTR_ZERO;  // optional
EPwm1Regs.CMPCTL.bit.SHDWBMODE = CC_SHADOW;    // optional

EPwm1Regs.AQCTLA.bit.ZRO = AQ_SET;
EPwm1Regs.AQCTLA.bit.CAU = AQ_CLEAR;
EPwm1Regs.AQCTLB.bit.ZRO = AQ_SET;             // optional
EPwm1Regs.AQCTLB.bit.CBU = AQ_CLEAR;           // optional
// Now configure the HRPWM resources
EALLOW;                                        // Note these registers are protected
                                               // and act only on ChA.
EPwm1Regs.HRCNFG.all = 0x0;                    // Clear all bits first
EPwm1Regs.HRCNFG.bit.EDGMODE = HR_FEP;         // Control falling edge position
EPwm1Regs.HRCNFG.bit.CTLMODE = HR_CMP;         // CMPAHR controls the MEP.
EPwm1Regs.HRCNFG.bit.HRLOAD = HR_CTR_ZERO;     // Shadow load on CTR=Zero.
EDIS;
MEP_ScaleFactor = 66*256;                      // Start with typical Scale Factor
                                               // value for 100 MHz.
                                               // Use SFO functions to update MEP_ScaleFactor
                                               // dynamically.
}
```

Example 20-6. PWM DAC Function Run-Time Code

```
EPWM1_BASE .set 0x6800
CMPAHR1 .set EPWM1_BASE+0x8
;================================================
HRPWM_DAC_DRV; (can execute within an ISR or loop)
;================================================
      MOVW DP, #_HRDAC_In
      MOVL XAR2,@_HRDAC_In             ; Pointer to input Q15 duty (XAR2)
      MOVL XAR3,#CMPAHR1               ; Pointer to HRPWM CMPA reg (XAR3)

; Output for EPWM1A (HRPWM
      MOV T,*XAR2                      ; T <= duty
      MPY ACC,T,@_hrDAC_period         ; Q15 to Q0 scaling based on period
      ADD ACC,@_HrDAC_period<<15       ; Offset for bipolar operation
      MOV T,@_MEP_ScaleFactor          ; MEP scale factor (from optimizer s/w)
      MPYU P,T,@AL                     ; P <= T * AL, optimizer scaling
      MOVH @AL,P                       ; AL <= P, move result back to ACC
      ADD ACC, #0x080                  ; MEP range and rounding adjustment
      MOVL *XAR3,ACC                   ; CMPA:CMPAHR(31:8) <= ACC

; Output for EPWM1B (Regular Res) Optional - for comparison purpose only
      MOV *+XAR3[2],AH                 ; Store ACCH to regular CMPB
```

20.15.2 SFO Library Software - SFO_TI_Build_V8.lib

Table 20-19 lists several features of the SFO_TI_Build_V8.lib library.

Table 20-19. SFO Library Features

	SFO_TI_Build_V8.lib	Unit
Completion-checking?	Yes	Function return value
Typical cycles required for SFO() to update MEP_ScaleFactor if called repetitively without interrupts	130,000	EPWMCLK cycles

20.15.2.1 Scale Factor Optimizer Function - int SFO()

This routine drives the micro-edge positioner (MEP) calibration module to run SFO diagnostics and determine the appropriate MEP scale factor (number of MEP steps per coarse EPWMCLK step) for a device at any given time.

If EPWMCLK = TBCLK = 100 MHz and assuming the MEP step size is 150 ps, the typical scale factor value at 100 MHz = 66 MEP steps per TBCLK unit (10 ns)

The function returns a MEP scale factor value:

MEP_ScaleFactor = Number of MEP steps per EPWMCLK.

Constraints when using this function:

- SFO() can be used with a minimum EPWMCLK = TBCLK = 50 MHz. MEP diagnostics logic uses EPWMCLK and not TBCLK, so the EPWMCLK restriction is an important constraint. Below 50 MHz, with device process variation, the MEP step size may decrease under cold temperature and high core voltage conditions to such a point, that 255 MEP steps will not span an entire EPWMCLK cycle.
- At any time, SFO() can be called to run SFO diagnostics on the MEP calibration module

Usage:

- SFO() can be called at any time in the background while the ePWM channels are running in HRPWM mode. The scale factor result obtained can be applied to all ePWM channels running in HRPWM mode because the function makes use of the diagnostics logic in the MEP calibration module (which runs independently of ePWM channels).
- This routine returns a 1 when calibration is finished and a new scale factor has been calculated, or a 0 if calibration is still running. The routine returns a 2 if there is an error, and the MEP_ScaleFactor is greater than the maximum 255 fine steps per coarse EPWMCLK cycle. In this case, the HRMSTEP register will maintain the last MEP scale factor value less than 256 for auto conversion.
- All ePWM modules operating in HRPWM incur only a 3-EPWMCLK cycle minimum duty cycle limitation when high-resolution period control is not used. If high-resolution period control is enabled, there is an additional duty cycle limitation 3-EPWMCLK cycles before the end of the PWM period (see Section 20.15.1.5.3).
- The SFO() function also updates the HRMSTEP register with the scale factor result. If the HRCNFG[AUTOCONV] bit is set, the application software is responsible only for setting CMPAHR = fraction(PWMduty*PWMperiod) << 8 or CMPBHR = fraction(PWMduty*PWMperiod) << 8 or TBPRDHR = fraction (PWMperiod) while running SFO() in the background. The MEP Calibration Module will then use the values in the HRMSTEP and CMPAHR/CMPBHR/TBPRDHR register to automatically calculate the appropriate number of MEP steps represented by the fractional duty cycle or period and move the high-resolution ePWM signal edge accordingly.
- If the HRCNFG[AUTOCONV] bit is clear, the HRMSTEP register is ignored. The application software will need to perform the necessary calculations manually so that:
  - CMPAHR = (fraction(PWMduty * PWMperiod) * MEP Scale Factor) << 8 + 0x080.
  - Similar behavior applies for TBPHSHR, CMPBHR, DBREDHR, DBFEDHR. Auto-conversion must be enabled when using TBPRDHR.

The following snippet shows how to use the HRPWM DUTY using driverlib functions.

```
float32_t dutyFine = 85.62;
float32_t count = (dutyFine * (float32_t)(EPWM_TIMER_TBPRD << 8))/100;
uint32_t compCount = (count);
HRPWM_setCounterCompareValue(EPWM1_BASE, HRPWM_COUNTER_COMPARE_A, compCount);
HRPWM_setCounterCompareValue(EPWM1_BASE, HRPWM_COUNTER_COMPARE_B, compCount);
```

The routine can be run as a background task in a slow loop requiring negligible CPU cycles. The repetition rate at which an SFO function needs to be executed depends on the application's operating environment. As with all digital CMOS devices, temperature and supply voltage variations have an effect on MEP operation. However, in most applications these parameters vary slowly and therefore it is often sufficient to execute the SFO function once every 5 to 10 seconds. If more rapid variations are expected, then execution may have to be performed more frequently to match the application. Note, there is no high limit restriction on the SFO function repetition rate, hence it can execute as quickly as the background loop is capable.

While using the HRPWM feature, HRPWM logic will not be active for the first three EPWMCLK cycles of the PWM period (and the last three EPWMCLK cycles of the PWM period if TBPRDHR is used). While running the application in this configuration, if high-resolution period control is disabled (HRPCTL[HRPE=0]) and the CMPA/CMPB register value is less than three cycles, then its CMPAHR/CMPBHR register must be cleared to zero. If high-resolution period control is enabled (HRPCTL[HRPE=1]), the CMPA register value must not fall below three or above TBPRD-3.This would avoid any unexpected transitions on the PWM signal.

20.15.2.2 Software Usage

The software library function SFO(), calculates the MEP scale factor for the HRPWM-supported ePWM modules. The scale factor is an integer value in the range 1-255, and represents the number of micro step edge positions available for a system clock period. The scale factor value is returned in an integer variable called MEP_ScaleFactor. For example, see Table 20-20.

Table 20-20. Factor Values

Software Function call	Functional Description	Updated Variables
SFO()	Returns MEP scale factor in the HRMSTEP register	MEP_ScaleFactor and HRMSTEP register.

To use the HRPWM feature of the ePWMs, it is recommended that the SFO function be used as described here.

Step 1. Add "Include" Files

The SFO_V8.h file needs to be included as follows. This include file is mandatory while using the SFO library function. For the SFO() to operate, the appropriate (Device)_Device.h and (Device)_Epwm_defines.h must be included in the project. These include files are optional if customized header files are used in the end applications.

Example 20-7. A Sample of How to Add "Include" Files

```
#include "F28x7x_Device.h"         // F28x7x Headerfile
#include "F28x7x_EPwm_defines.h"   // init defines
#include "SFO_V8.h"                // SFO lib functions (needed for HRPWM)
```

Step 2. Element Declaration

Declare an integer variable for the scale factor value as shown below.

Example 20-8. Declaring an Element

```
int MEP_ScaleFactor = 0;   //scale factor value
volatile struct EPWM_REGS *ePWM[] = {0, &EPwm1Regs, &EPwm2Regs, &EPwm3Regs,
&EPwm4Regs};
```

Step 3. MEP_ScaleFactor Initialization

The SFO() function does not require a starting scale factor value in MEP_ScaleFactor. Prior to using the MEP_ScaleFactor variable in application code, SFO() should be called to drive the MEP calibration module to calculate an MEP_ScaleFactor value.

As part of the one-time initialization code prior to using MEP_ScaleFactor, include the following:

Example 20-9. Initializing With a Scale Factor Value

```
MEP_ScaleFactor initialized using function SFO ()
while (SFO() == 0) {} // MEP_ScaleFactor calculated by MEP Cal Module
```

Step 4. Application Code

While the application is running, fluctuations in both device temperature and supply voltage may be expected. To be sure that optimal Scale Factors are used for each ePWM module, the SFO function should be re-run periodically as part of a slower back-ground loop. Some examples of this are shown here.

Note

See the HRPWM_SFO example in the device-specific C/C++ header files and peripheral examples available from the TI website.

Example 20-10. SFO Function Calls

```
main ()
 {
 int status;
    // User code
    // ePWM1, 2, 3, 4 are running in HRPWM mode
    // The status variable returns 1 once a new MEP_ScaleFactor has been
    // calculated by the MEP Calibration Module running SFO
    // diagnostics.
status = SFO();
if(status==2) {ESTOP0;}   // The function returns a 2 if MEP_ScaleFactor is greater
                          // than the maximum 255 allowed (error condition)
 }
```

20.16 Software

20.16.1 EPWM Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/epwm

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

20.16.1.1 ePWM Chopper

FILE: epwm_ex10_chopper.c

This example configures ePWM1, ePWM2, ePWM3 and ePWM4 as follows

- ePWM1 with Chopper disabled (Reference)
- ePWM2 with chopper enabled at 1/8 duty cycle
- ePWM3 with chopper enabled at 6/8 duty cycle
- ePWM4 with chopper enabled at 1/2 duty cycle with One-Shot Pulse enabled

External Connections

- GPIO0 EPWM1A
- GPIO1 EPWM1B
- GPIO2 EPWM2A
- GPIO3 EPWM2B
- GPIO4 EPWM3A
- GPIO5 EPWM3B
- GPIO6 EPWM4A
- GPIO7 EPWM4B

Watch Variables

- None.

20.16.1.2 EPWM Configure Signal

FILE: epwm_ex11_configure_signal.c

This example configures ePWM1, ePWM2, ePWM3 to produce signal of desired frequency and duty. It also configures phase between the configured modules.

Signal of 10kHz with duty of 0.5 is configured on ePWMxA & ePWMxB with ePWMxB inverted. Also, phase of 120 degree is configured between ePWM1 to ePWM3 signals.

During the test, monitor ePWM1, ePWM2, and/or ePWM3 outputs on an oscilloscope.

- ePWM1A is on GPIO0
- ePWM1B is on GPIO1
- ePWM2A is on GPIO2
- ePWM2B is on GPIO3
- ePWM3A is on GPIO4
- ePWM3B is on GPIO5

20.16.1.3 Realization of Monoshot Mode

FILE: epwm_ex12_monoshot_mode.c

This example showcases how to generate a monoshot PWM output based on an external trigger, that is, generating just a single pulse output on receipt of an external trigger. And the next pulse will be generated only when the next trigger comes. The example utilizes external synchronization and T1 action qualifier event features to achieve the desired output.

ePWM1 is used to generate the monoshot output and ePWM2 is used an external trigger for that. No external connections are required as ePWM2A is fed as the trigger using Input X-BAR automatically.

ePWM1 is configured to generated a single pulse of 0.5 µs when received an external trigger. This is achieved by enabling the phase synchronization feature and configuring EPWMxSYNCI as EXTSYNCIN1. And this EPWMxSYNCI is also configured as T1 event of action qualifier to set output HIGH while "CTR = PRD" action is used to set output LOW.

ePWM2 is configured to generate a 100 kHz signal with a duty of 1% (to simulate a rising edge trigger) which is routed to EXTSYNCIN1 using Input XBAR.

Observe GPIO0 (EPWM1A : Monoshot Output) and GPIO2(EPWM2 : External Trigger) on oscilloscope.

NOTE : In the following example, the ePWM timer is still running in a continuous mode rather than a one-shot mode thus for more reliable implementation, refer to CLB based one shot PWM implementation demonstrated in "clb_ex17_one_shot_pwm" example

20.16.1.4 EPWM Action Qualifier (epwm_up_aq)

FILE: epwm_ex13_up_aq.c

This example configures ePWM1, ePWM2, ePWM3 to produce an waveform with independent modulation on EPWMxA and EPWMxB.

The compare values CMPA and CMPB are modified within the ePWM's ISR.

The TB counter is in up count mode for this example.

View the EPWM1A/B(GPIO0 & GPIO1), EPWM2A/B(GPIO2 & GPIO3) and EPWM3A/B(GPIO4 & GPIO5) waveforms via an oscilloscope.

20.16.1.5 ePWM Trip Zone

FILE: epwm_ex1_trip_zone.c

This example configures ePWM1 and ePWM2 as follows
- ePWM1 has TZ1 as one shot trip source
- ePWM2 has TZ1 as cycle by cycle trip source

Initially tie TZ1 high. During the test, monitor ePWM1 or ePWM2 outputs on a scope. Pull TZ1 low to see the effect.

External Connections
- ePWM1A is on GPIO0
- ePWM2A is on GPIO2
- TZ1 is on GPIO12

This example also makes use of the Input X-BAR. GPIO12 (the external trigger) is routed to the input X-BAR, from which it is routed to TZ1.

The TZ-Event is defined such that ePWM1A will undergo a One-Shot Trip and ePWM2A will undergo a Cycle-By-Cycle Trip.

20.16.1.6 ePWM Up Down Count Action Qualifier

FILE: epwm_ex2_updown_aq.c

This example configures ePWM1, ePWM2, ePWM3 to produce a waveform with independent modulation on ePWMxA and ePWMxB.

The compare values CMPA and CMPB are modified within the ePWM's ISR.

The TB counter is in up/down count mode for this example.

View the ePWM1A/B(GPIO0 & GPIO1), ePWM2A/B(GPIO2 &GPIO3) and ePWM3A/B(GPIO4 & GPIO5) waveforms on oscilloscope.

20.16.1.7 ePWM Synchronization

FILE: epwm_ex3_synchronization.c

This example configures ePWM1, ePWM2, ePWM3 and ePWM4 as follows

- ePWM1 without phase shift as master
- ePWM2 with phase shift of 300 TBCLKs
- ePWM3 with phase shift of 600 TBCLKs
- ePWM4 with phase shift of 900 TBCLKs

External Connections

- GPIO0 EPWM1A
- GPIO1 EPWM1B
- GPIO2 EPWM2A
- GPIO3 EPWM2B
- GPIO4 EPWM3A
- GPIO5 EPWM3B
- GPIO6 EPWM4A
- GPIO7 EPWM4B

Watch Variables

- None.

20.16.1.8 ePWM Digital Compare

FILE: epwm_ex4_digital_compare.c

This example configures ePWM1 as follows

- ePWM1 with DCAEVT1 forcing the ePWM output LOW
- GPIO25 is used as the input to the INPUT XBAR INPUT1
- INPUT1 (from INPUT XBAR) is used as the source for DCAEVT1
- GPIO25's PULL-UP resistor is enabled, in order to test the trip, PULL this pin to GND

External Connections

- GPIO0 EPWM1A
- GPIO1 EPWM1B
- GPIO25 TZ1, pull this pin low to trip the ePWM

Watch Variables

- None.

20.16.1.9 ePWM Digital Compare Event Filter Blanking Window

FILE: epwm_ex5_digital_compare_event_filter.c

This example configures ePWM1 as follows

- ePWM1 with DCAEVT1 forcing the ePWM output LOW
- GPIO25 is used as the input to the INPUT XBAR INPUT1
- INPUT1 (from INPUT XBAR) is used as the source for DCAEVT1
- GPIO25's PULL-UP resistor is enabled, in order to test the trip, PULL this pin to GND
- ePWM1 with DCBEVT1 forcing the ePWM output LOW
- GPIO25 is used as the input to the INPUT XBAR INPUT1
- INPUT1 (from INPUT XBAR) is used as the source for DCAEVT1
- GPIO25's PULL-UP resistor is enabled, in order to test the trip, PULL this pin to GND
- DCBEVT1 uses the filtered version of DCBEVT1
- The DCFILT signal uses the blanking window to ignore the DCBEVT1 for the duration of DC Blanking window

External Connections

- GPIO0 EPWM1A
- GPIO1 EPWM1B
- GPIO25 TRIPIN1, pull this pin low to trip the ePWM

Watch Variables

- None.

20.16.1.10 ePWM Valley Switching

FILE: epwm_ex6_valley_switching.c

This example configures ePWM1 as follows

- ePWM1 with DCAEVT1 forcing the ePWM output LOW
- GPIO25 is used as the input to the INPUT XBAR INPUT1
- INPUT1 (from INPUT XBAR) is used as the source for DCAEVT1
- GPIO25 is set to output and toggled in the main loop to trip the PWM
- ePWM1 with DCBEVT1 forcing the ePWM output LOW
- GPIO25 is used as the input to the INPUT XBAR INPUT1
- INPUT1 (from INPUT XBAR) is used as the source for DCAEVT1
- GPIO25 is set to output and toggled in the main loop to trip the PWM
- DCBEVT1 uses the filtered version of DCBEVT1
- The DCFILT signal uses the valley switching module to delay the
- DCFILT signal by a software defined DELAY value.

External Connections

- GPIO0 EPWM1A
- GPIO1 EPWM1B
- GPIO25 TRIPIN1 (Output Pin, toggled through software)

Watch Variables

- None.

20.16.1.11 ePWM Digital Compare Edge Filter

FILE: epwm_ex7_edge_filter.c

This example configures ePWM1 as follows

- ePWM1 with DCBEVT2 forcing the ePWM output LOW as a CBC source
- GPIO25 is used as the input to the INPUT XBAR INPUT1
- INPUT1 (from INPUT XBAR) is used as the source for DCBEVT2
- GPIO25 is set to output and toggled in the main loop to trip the PWM
- The DCBEVT2 is the source for DCFILT
- The DCFILT will count edges of the DCBEVT2 and generate a signal to to trip the ePWM on the 4th edge of DCBEVT2

External Connections

- GPIO0 EPWM1A
- GPIO1 EPWM1B
- GPIO25 TRIPIN1 (Output Pin, toggled through software)

Watch Variables

- None.

20.16.1.12 ePWM Deadband

FILE: epwm_ex8_deadband.c

This example configures ePWM1 through ePWM6 as follows

- ePWM1 with Deadband disabled (Reference)
- ePWM2 with Deadband Active High
- ePWM3 with Deadband Active Low
- ePWM4 with Deadband Active High Complimentary
- ePWM5 with Deadband Active Low Complimentary
- ePWM6 with Deadband Output Swap (switch A and B outputs)

External Connections

- GPIO0 EPWM1A

- GPIO1 EPWM1B
- GPIO2 EPWM2A
- GPIO3 EPWM2B
- GPIO4 EPWM3A
- GPIO5 EPWM3B
- GPIO6 EPWM4A
- GPIO7 EPWM4B
- GPIO8 EPWM5A
- GPIO9 EPWM5B
- GPIO10 EPWM6A
- GPIO11 EPWM6B

Watch Variables
- None.

20.16.1.13 ePWM DMA

FILE: epwm_ex9_dma.c

This example configures ePWM1 and DMA as follows:
- ePWM1 is set up to generate PWM waveforms
- DMA5 is set up to update the CMPAHR, CMPA, CMPBHR and CMPB every period with the next value in the configuration array. This allows the user to create a DMA enabled fifo for all the CMPx and CMPxHR registers to generate unconventional PWM waveforms.
- DMA6 is set up to update the TBPHSHR, TBPHS, TBPRDHR and TBPRD every period with the next value in the configuration array.
- Other registers such as AQCTL can be controlled through the DMA as well by following the same procedure. (Not used in this example)

External Connections
- GPIO0 EPWM1A
- GPIO1 EPWM1B

Watch Variables
- None.

20.16.2 HRPWM Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/hrpwm

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

20.16.2.1 HRPWM Duty Control with SFO

FILE: hrpwm_ex1_duty_sfo.c

This example modifies the MEP control registers to show edge displacement for high-resolution period with ePWM in Up count mode due to the HRPWM control extension of the respective ePWM module.

This example calls the following TI's MEP Scale Factor Optimizer (SFO) software library V8 functions:

int SFO();
- updates MEP_ScaleFactor dynamically when HRPWM is in use
- updates HRMSTEP register (exists only in EPwm1Regs register space) with MEP_ScaleFactor value
- returns 2 if error: MEP_ScaleFactor is greater than maximum value of 255 (Auto-conversion may not function properly under this condition)
- returns 1 when complete for the specified channel
- returns 0 if not complete for the specified channel

This example is intended to explain the HRPWM capabilities. The code can be optimized for code efficiency. Refer to TI's Digital power application examples and TI Digital Power Supply software libraries for details.

External Connections

- Monitor ePWM1/2/3/4 A/B pins on an oscilloscope.

20.16.2.2 HRPWM Slider

FILE: hrpwm_ex2_slider.c

This example modifies the MEP control registers to show edge displacement due to HRPWM. Control blocks of the respective ePWM module channel A and B will have fine edge movement due to HRPWM logic.

Monitor ePWM1 A/B pins on an oscilloscope.

20.16.2.3 HRPWM Period Control

FILE: hrpwm_ex3_prd_updown_sfo.c

This example modifies the MEP control registers to show edge displacement for high-resolution period with ePWM in Up-Down count mode due to the HRPWM control extension of the respective ePWM module.

This example calls the following TI's MEP Scale Factor Optimizer (SFO) software library V8 functions:

int SFO();

- updates MEP_ScaleFactor dynamically when HRPWM is in use
- updates HRMSTEP register (exists only in EPwm1Regs register space) with MEP_ScaleFactor value
- returns 2 if error: MEP_ScaleFactor is greater than maximum value of 255 (Auto-conversion may not function properly under this condition)
- returns 1 when complete for the specified channel
- returns 0 if not complete for the specified channel

This example is intended to explain the HRPWM capabilities. The code can be optimized for code efficiency. Refer to TI's Digital power application examples and TI Digital Power Supply software libraries for details.

External Connections

- Monitor ePWM1/2/3/4 A/B pins on an oscilloscope.

20.16.2.4 HRPWM Duty Control with UPDOWN Mode

FILE: hrpwm_ex4_duty_updown_sfo.c

This example calls the following TI's MEP Scale Factor Optimizer (SFO) software library V8 functions:

int SFO();

- updates MEP_ScaleFactor dynamically when HRPWM is in use
- updates HRMSTEP register (exists only in EPwm1Regs register space) with MEP_ScaleFactor value
- returns 2 if error: MEP_ScaleFactor is greater than maximum value of 255 (Auto-conversion may not function properly under this condition)
- returns 1 when complete for the specified channel
- returns 0 if not complete for the specified channel

This example is intended to explain the HRPWM capabilities. The code can be optimized for code efficiency. Refer to TI's Digital power application examples and TI Digital Power Supply software libraries for details.

External Connections

- Monitor ePWM1/2/3/4 A/B pins on an oscilloscope.

20.16.2.5 HRPWM Slider Test

FILE: hrpwm_ex5_slider_qformat.c

This example modifies the MEP control registers to show edge displacement due to HRPWM. Control blocks of the respective ePWM module channel A and B will have fine edge movement due to HRPWM logic. Load the hrpwm_slider.gel file. Select the HRPWM_eval from the GEL menu. A FineDuty slider graphics will show up in CCS. Load the program and run. Use the Slider to and observe the EPWM edge displacement for each slider step change. This explains the MEP control on the EPwmxA channels.

Monitor ePWM1 and ePWM2 A/B pins on an oscilloscope.

20.16.2.6 HRPWM Duty Up Count

FILE: hrpwm_ex6_duty_sfo_qformat.c

This example modifies the MEP control registers to show edge displacement for high-resolution period with ePWM in Up count mode due to the HRPWM control extension of the respective ePWM module.

This example calls the following TI's MEP Scale Factor Optimizer (SFO) software library V8 functions:

int SFO();
- updates MEP_ScaleFactor dynamically when HRPWM is in use
- updates HRMSTEP register (exists only in EPwm1Regs register space) with MEP_ScaleFactor value
- returns 2 if error: MEP_ScaleFactor is greater than maximum value of 255 (Auto-conversion may not function properly under this condition)
- returns 1 when complete for the specified channel
- returns 0 if not complete for the specified channel

This example is intended to explain the HRPWM capabilities. The code can be optimized for code efficiency. Refer to TI's Digital power application examples and TI Digital Power Supply software libraries for details.

To run this example:
1. Run this example at maximum SYSCLKOUT
2. Activate Real time mode
3. Run the code

External Connections
- Monitor ePWM1/2 A/B pins on an oscilloscope.

Watch Variables
- status - Example run status
- updateFine - Set to 1 use HRPWM capabilities and observe in fine MEP steps(default) Set to 0 to disable HRPWM capabilities and observe in coarse SYSCLKOUT cycle steps

20.16.2.7 HRPWM Period Up-Down Count

FILE: hrpwm_ex7_prd_updown_sfo_qformat.c

This example modifies the MEP control registers to show edge displacement for high-resolution period with ePWM in Up-Down count mode due to the HRPWM control extension of the respective ePWM module.

This example calls the following TI's MEP Scale Factor Optimizer (SFO) software library V8 functions:

int SFO();
- updates MEP_ScaleFactor dynamically when HRPWM is in use
- updates HRMSTEP register (exists only in EPwm1Regs register space) with MEP_ScaleFactor value
- returns 2 if error: MEP_ScaleFactor is greater than maximum value of 255 (Auto-conversion may not function properly under this condition)
- returns 1 when complete for the specified channel
- returns 0 if not complete for the specified channel

This example is intended to explain the HRPWM capabilities. The code can be optimized for code efficiency. Refer to TI's Digital power application examples and TI Digital Power Supply software libraries for details.

To run this example:

1. Run this example at maximum SYSCLKOUT
2. Activate Real time mode
3. Run the code

External Connections

- Monitor ePWM1/2 A/B pins on an oscilloscope.

Watch Variables

- updateFine - Set to 1 use HRPWM capabilities and observe in fine MEP steps(default) Set to 0 to disable HRPWM capabilities and observe in coarse SYSCLKOUT cycle steps

20.17 ePWM Registers

This section describes the Enhanced Pulse Width Modulator registers.

20.17.1 EPWM Base Address Table

Table 20-21. EPWM Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
EPwm1Regs	EPWM_REGS	EPWM1_BASE	0x0000_4000	YES	YES	YES	YES	YES
EPwm2Regs	EPWM_REGS	EPWM2_BASE	0x0000_4100	YES	YES	YES	YES	YES
EPwm3Regs	EPWM_REGS	EPWM3_BASE	0x0000_4200	YES	YES	YES	YES	YES
EPwm4Regs	EPWM_REGS	EPWM4_BASE	0x0000_4300	YES	YES	YES	YES	YES
EPwm5Regs	EPWM_REGS	EPWM5_BASE	0x0000_4400	YES	YES	YES	YES	YES
EPwm6Regs	EPWM_REGS	EPWM6_BASE	0x0000_4500	YES	YES	YES	YES	YES
EPwm7Regs	EPWM_REGS	EPWM7_BASE	0x0000_4600	YES	YES	YES	YES	YES
EPwm8Regs	EPWM_REGS	EPWM8_BASE	0x0000_4700	YES	YES	YES	YES	YES

20.17.2 EPWM_REGS Registers

Table 20-22 lists the memory-mapped registers for the EPWM_REGS registers. All register offset addresses not listed in Table 20-22 should be considered as reserved locations and the register contents should not be modified.

Table 20-22. EPWM_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	TBCTL	Time Base Control Register		Go
1h	TBCTL2	Time Base Control Register 2		Go
3h	EPWMSYNCINSEL	EPWMxSYNCIN Source Select Register		Go
4h	TBCTR	Time Base Counter Register		Go
5h	TBSTS	Time Base Status Register		Go
6h	EPWMSYNCOUTEN	EPWMxSYNCOUT Source Enable Register		Go
7h	TBCTL3	Time Base Control Register 3		Go
8h	CMPCTL	Counter Compare Control Register		Go
9h	CMPCTL2	Counter Compare Control Register 2		Go
Ch	DBCTL	Dead-Band Generator Control Register		Go
Dh	DBCTL2	Dead-Band Generator Control Register 2		Go
10h	AQCTL	Action Qualifier Control Register		Go
11h	AQTSRCSEL	Action Qualifier Trigger Event Source Select Register		Go
14h	PCCTL	PWM Chopper Control Register		Go
18h	VCAPCTL	Valley Capture Control Register		Go
19h	VCNTCFG	Valley Counter Config Register		Go
20h	HRCNFG	HRPWM Configuration Register	EALLOW	Go
21h	HRPWR	HRPWM Power Register	EALLOW	Go
26h	HRMSTEP	HRPWM MEP Step Register	EALLOW	Go
27h	HRCNFG2	HRPWM Configuration 2 Register	EALLOW	Go
2Dh	HRPCTL	High Resolution Period Control Register	EALLOW	Go
2Eh	TRREM	HRPWM High Resolution Remainder Register	EALLOW	Go
34h	GLDCTL	Global PWM Load Control Register	EALLOW	Go
35h	GLDCFG	Global PWM Load Config Register	EALLOW	Go
38h	EPWMXLINK	EPWMx Link Register		Go
40h	AQCTLA	Action Qualifier Control Register For Output A		Go
41h	AQCTLA2	Additional Action Qualifier Control Register For Output A		Go
42h	AQCTLB	Action Qualifier Control Register For Output B		Go
43h	AQCTLB2	Additional Action Qualifier Control Register For Output B		Go
47h	AQSFRC	Action Qualifier Software Force Register		Go
49h	AQCSFRC	Action Qualifier Continuous S/W Force Register		Go
50h	DBREDHR	Dead-Band Generator Rising Edge Delay High Resolution Mirror Register		Go
51h	DBRED	Dead-Band Generator Rising Edge Delay High Resolution Mirror Register		Go
52h	DBFEDHR	Dead-Band Generator Falling Edge Delay High Resolution Register		Go
53h	DBFED	Dead-Band Generator Falling Edge Delay Count Register		Go
60h	TBPHS	Time Base Phase High		Go

Table 20-22. EPWM_REGS Registers (continued)

Offset	Acronym	Register Name	Write Protection	Section
62h	TBPRDHR	Time Base Period High Resolution Register		Go
63h	TBPRD	Time Base Period Register		Go
6Ah	CMPA	Counter Compare A Register		Go
6Ch	CMPB	Compare B Register		Go
6Fh	CMPC	Counter Compare C Register		Go
71h	CMPD	Counter Compare D Register		Go
74h	GLDCTL2	Global PWM Load Control Register 2		Go
77h	SWVDELVAL	Software Valley Mode Delay Register		Go
80h	TZSEL	Trip Zone Select Register	EALLOW	Go
82h	TZDCSEL	Trip Zone Digital Comparator Select Register	EALLOW	Go
84h	TZCTL	Trip Zone Control Register	EALLOW	Go
85h	TZCTL2	Additional Trip Zone Control Register	EALLOW	Go
86h	TZCTLDCA	Trip Zone Control Register Digital Compare A	EALLOW	Go
87h	TZCTLDCB	Trip Zone Control Register Digital Compare B	EALLOW	Go
8Dh	TZEINT	Trip Zone Enable Interrupt Register	EALLOW	Go
93h	TZFLG	Trip Zone Flag Register		Go
94h	TZCBCFLG	Trip Zone CBC Flag Register		Go
95h	TZOSTFLG	Trip Zone OST Flag Register		Go
97h	TZCLR	Trip Zone Clear Register	EALLOW	Go
98h	TZCBCCLR	Trip Zone CBC Clear Register	EALLOW	Go
99h	TZOSTCLR	Trip Zone OST Clear Register	EALLOW	Go
9Bh	TZFRC	Trip Zone Force Register	EALLOW	Go
A4h	ETSEL	Event Trigger Selection Register		Go
A6h	ETPS	Event Trigger Pre-Scale Register		Go
A8h	ETFLG	Event Trigger Flag Register		Go
AAh	ETCLR	Event Trigger Clear Register		Go
ACh	ETFRC	Event Trigger Force Register		Go
AEh	ETINTPS	Event-Trigger Interrupt Pre-Scale Register		Go
B0h	ETSOCPS	Event-Trigger SOC Pre-Scale Register		Go
B2h	ETCNTINITCTL	Event-Trigger Counter Initialization Control Register		Go
B4h	ETCNTINIT	Event-Trigger Counter Initialization Register		Go
C0h	DCTRIPSEL	Digital Compare Trip Select Register	EALLOW	Go
C3h	DCACTL	Digital Compare A Control Register	EALLOW	Go
C4h	DCBCTL	Digital Compare B Control Register	EALLOW	Go
C7h	DCFCTL	Digital Compare Filter Control Register	EALLOW	Go
C8h	DCCAPCTL	Digital Compare Capture Control Register	EALLOW	Go
C9h	DCFOFFSET	Digital Compare Filter Offset Register		Go
CAh	DCFOFFSETCNT	Digital Compare Filter Offset Counter Register		Go
CBh	DCFWINDOW	Digital Compare Filter Window Register		Go
CCh	DCFWINDOWCNT	Digital Compare Filter Window Counter Register		Go
CDh	BLANKPULSEMIXSEL	Blanking window trigger pulse select register	EALLOW	Go
CFh	DCCAP	Digital Compare Counter Capture Register		Go
D2h	DCAHTRIPSEL	Digital Compare AH Trip Select	EALLOW	Go
D3h	DCALTRIPSEL	Digital Compare AL Trip Select	EALLOW	Go

Table 20-22. EPWM_REGS Registers (continued)

Offset	Acronym	Register Name	Write Protection	Section
D4h	DCBHTRIPSEL	Digital Compare BH Trip Select	EALLOW	Go
D5h	DCBLTRIPSEL	Digital Compare BL Trip Select	EALLOW	Go
FAh	EPWMLOCK	EPWM Lock Register		Go
FDh	HWVDELVAL	Hardware Valley Mode Delay Register		Go
FEh	VCNTVAL	Hardware Valley Counter Register		Go

Complex bit access types are encoded to fit into small table cells. Table 20-23 shows the codes that are used for access types in this section.

Table 20-23. EPWM_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
WOnce	W Once	Write Write once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

20.17.2.1 TBCTL Register (Offset = 0h) [Reset = 83h]

TBCTL is shown in Figure 20-89 and described in Table 20-24.

Return to the Summary Table.

Time Base Control Register

Figure 20-89. TBCTL Register

15	14	13	12	11	10	9	8
FREE_SOFT		PHSDIR	CLKDIV			HSPCLKDIV	
R/W-0h		R/W-0h	R/W-0h			R/W-1h	
7	6	5	4	3	2	1	0
HSPCLKDIV	SWFSYNC	RESERVED		PRDLD	PHSEN	CTRMODE	
R/W-1h	R-0/W1S-0h	R-0h		R/W-0h	R/W-0h	R/W-3h	

Table 20-24. TBCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	FREE_SOFT	R/W	0h	Emulation Mode Bits. These bits select the behavior of the ePWM time-base counter during emulation events 00: Stop after the next time-base counter increment or decrement 01: Stop when counter completes a whole cycle: - Up-count mode: stop when the time-base counter = period (TBCTR = TBPRD) - Down-count mode: stop when the time-base counter = 0x00 (TBCTR = 0x00) - Up-down-count mode: stop when the time-base counter = 0x00 (TBCTR = 0x00) 1x: Free run Reset type: SYSRSn
13	PHSDIR	R/W	0h	Phase Direction Bit This bit is only used when the time-base counter is configured in the up-down-count mode. The PHSDIR bit indicates the direction the time-base counter (TBCTR) will count after a synchronization event occurs and a new phase value is loaded from the phase (TBPHS) register. This is irrespective of the direction of the counter before the synchronization event.. In the up-count and down-count modes this bit is ignored. 0: Count down after the synchronization event. 1: Count up after the synchronization event. Reset type: SYSRSn
12-10	CLKDIV	R/W	0h	Time Base Clock Pre-Scale Bits These bits select the time base clock pre-scale value (TBCLK = EPWMCLK/(HSPCLKDIV * CLKDIV): 000: /1 (default on reset) 001: /2 010: /4 011: /8 100: /16 101: /32 110: /64 111: /128 Reset type: SYSRSn

Table 20-24. TBCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9-7	HSPCLKDIV	R/W	1h	High Speed Time Base Clock Pre-Scale Bits These bits determine part of the time-base clock prescale value. TBCLK = EPWMCLK / (HSPCLKDIV x CLKDIV). This divisor emulates the HSPCLK in the TMS320x281x system as used on the Event Manager (EV) peripheral. 000: /1 001: /2 (default on reset) 010: /4 011: /6 100: /8 101: /10 110: /12 111: /14 Reset type: SYSRSn
6	SWFSYNC	R-0/W1S	0h	Software Forced Sync Pulse 0: Writing a 0 has no effect and reads always return a 0. 1: Writing a 1 forces a one-time synchronization pulse to be generated. SWFSYNC can be enabled to affect EPWMxSYNCO by setting the EPWMSYNCOUTEN.SWEN bit. Reset type: SYSRSn
5-4	RESERVED	R	0h	Reserved
3	PRDLD	R/W	0h	Active Period Reg Load from Shadow Select 0: The period register (TBPRD) is loaded from its shadow register when the time-base counter, TBCTR, is equal to zero and/or a sync event as determined by the TBCTL2[PRDLDSYNC] bit. A write/read to the TBPRD register accesses the shadow register. 1: Immediate Mode (Shadow register bypassed): A write or read to the TBPRD register accesses the active register. Reset type: SYSRSn
2	PHSEN	R/W	0h	Counter Reg Load from Phase Reg Enable 0: Do not load the time-base counter (TBCTR) from the time-base phase register (TBPHS). 1: Allow Counter to be loaded from the Phase register (TBPHS) and shadow to active load events when an EPWMxSYNCI input signal occurs or a software-forced sync signal, see bit 6. Reset type: SYSRSn
1-0	CTRMODE	R/W	3h	Counter Mode The time-base counter mode is normally configured once and not changed during normal operation. If you change the mode of the counter, the change will take effect at the next TBCLK edge and the current counter value shall increment or decrement from the value before the mode change. These bits set the time-base counter mode of operation as follows: 00: Up-count mode 01: Down-count mode 10: Up-down count mode 11: Freeze counter operation (default on reset) Reset type: SYSRSn

20.17.2.2 TBCTL2 Register (Offset = 1h) [Reset = 0h]

TBCTL2 is shown in Figure 20-90 and described in Table 20-25.

Return to the Summary Table.

Time Base Control Register 2

Figure 20-90. TBCTL2 Register

15	14	13	12	11	10	9	8
PRDLDSYNC		RESERVED		RESERVED			
R/W-0h		R-0h		R-0h			
7	6	5	4	3	2	1	0
OSHTSYNC	OSHTSYNCMODE	RESERVED	RESERVED				
R-0/W1S-0h	R/W-0h	R/W-0h	R-0h				

Table 20-25. TBCTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	PRDLDSYNC	R/W	0h	Shadow to Active Period Register Load on SYNC event 00: Shadow to Active Load of TBPRD occurs only when TBCTR = 0 (same as legacy). 01: Shadow to Active Load of TBPRD occurs both when TBCTR = 0 and when SYNC occurs. 10: Shadow to Active Load of TBPRD occurs only when a SYNC is received. 11: Reserved Note: This bit selection is valid only if TBCTL[PRDLD]=0. Reset type: SYSRSn
13-12	RESERVED	R	0h	Reserved
11-8	RESERVED	R	0h	Reserved
7	OSHTSYNC	R-0/W1S	0h	Oneshot sync bit 0: Writing a '0' has no effect. 1: Allow one sync pulse to propogate. Reset type: SYSRSn
6	OSHTSYNCMODE	R/W	0h	Oneshot sync enable bit 0: Oneshot sync mode disabled 1: Oneshot sync mode enabled Reset type: SYSRSn
5	RESERVED	R/W	0h	Reserved
4-0	RESERVED	R	0h	Reserved

20.17.2.3 EPWMSYNCINSEL Register (Offset = 3h) [Reset = 1h]

EPWMSYNCINSEL is shown in Figure 20-91 and described in Table 20-26.

Return to the Summary Table.

EPWMxSYNCIN Source Select Register

Figure 20-91. EPWMSYNCINSEL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			SEL				
R-0h			R/W-1h				

Table 20-26. EPWMSYNCINSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R	0h	Reserved
4-0	SEL	R/W	1h	These bits determine the source of the EPWMxSYNCI signal. 0x00 Disabled 0x01 EPWM1.SYNCOUT 0x02 EPWM2.SYNCOUT 0x03 EPWM3.SYNCOUT 0x04 EPWM4.SYNCOUT 0x05 EPWM5.SYNCOUT 0x06 EPWM6.SYNCOUT 0x07 EPWM7.SYNCOUT 0x08 EPWM8.SYNCOUT 0x11 ECAP1.SYNCOUT 0x12 ECAP2.SYNCOUT 0x13 ECAP3.SYNCOUT 0x18 INPUT-XBAR.INPUT5 0x19 INPUT-XBAR.INPUT6 Other Values Reserved The SYNCOUT from an EPWM or ECAP instance cannot be connected to its own SYNCIN (i.e. PWM1 SYNCOUT cannot be connected to PWM1 SYNCIN) Reset type: SYSRSn

20.17.2.4 TBCTR Register (Offset = 4h) [Reset = 0h]

TBCTR is shown in Figure 20-92 and described in Table 20-27.

Return to the Summary Table.

Time Base Counter Register

Figure 20-92. TBCTR Register

15	14	13	12	11	10	9	8
TBCTR							
R/W-0h							
7	6	5	4	3	2	1	0
TBCTR							
R/W-0h							

Table 20-27. TBCTR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	TBCTR	R/W	0h	Time Base Counter Register Reset type: SYSRSn

20.17.2.5 TBSTS Register (Offset = 5h) [Reset = 1h]

TBSTS is shown in Figure 20-93 and described in Table 20-28.

Return to the Summary Table.

Time Base Status Register

Figure 20-93. TBSTS Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					CTRMAX	SYNCI	CTRDIR
R-0h					R/W1C-0h	R/W1C-0h	R-1h

Table 20-28. TBSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-3	RESERVED	R	0h	Reserved
2	CTRMAX	R/W1C	0h	Time-Base Counter Max Latched Status Bit 0: Reading a 0 indicates the time-base counter never reached its maximum value. Writing a 0 will have no effect. 1: Reading a 1 on this bit indicates that the time-base counter reached the max value 0xFFFF. Writing a 1 to this bit will clear the latched event. Reset type: SYSRSn
1	SYNCI	R/W1C	0h	Input Synchronization Latched Status Bit 0: Writing a 0 will have no effect. Reading a 0 indicates no external synchronization event has occurred. 1: Reading a 1 on this bit indicates that an external synchronization event has occurred (EPWMxSYNCI). Writing a 1 to this bit will clear the latched event. Reset type: SYSRSn
0	CTRDIR	R	1h	Time Base Counter Direction Status Bit 0: Time-Base Counter is currently counting down. 1: Time-Base Counter is currently counting up. Note: This bit is only valid when the counter is not frozen. Reset type: SYSRSn

20.17.2.6 EPWMSYNCOUTEN Register (Offset = 6h) [Reset = 1h]

EPWMSYNCOUTEN is shown in Figure 20-94 and described in Table 20-29.

Return to the Summary Table.

EPWMxSYNCOUT Source Enable Register

Figure 20-94. EPWMSYNCOUTEN Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	DCBEVT1EN	DCAEVT1EN	CMPDEN	CMPCEN	CMPBEN	ZEROEN	SWEN
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-1h

Table 20-29. EPWMSYNCOUTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	RESERVED	R	0h	Reserved
6	DCBEVT1EN	R/W	0h	This bit enables the DCBEVT1.sync event to set the EPWMxSYNCO signal. 0 Disabled 1 The EPWMxSYNCO signal is pulsed for one PWM clock period upon a DCBEVT1.sync event Reset type: SYSRSn
5	DCAEVT1EN	R/W	0h	This bit enables the DCAEVT1.sync event to set the EPWMxSYNCOUT signal. 0 Disabled 1 The EPWMxSYNCOUT signal is pulsed for one PWM clock period upon a DCAEVT1.sync event Reset type: SYSRSn
4	CMPDEN	R/W	0h	This bit enables the TBCTR = CMPD event to set the EPWMxSYNCO signal. 0 Disabled 1 The EPWMxSYNCO signal is pulsed for one PWM clock period upon a time-base counter equal to counter compare D event (TBCTR = CMPD) Reset type: SYSRSn
3	CMPCEN	R/W	0h	This bit enables the TBCTR = CMPC event to set the EPWMxSYNCO signal. 0 Disabled 1 The EPWMxSYNCO signal is pulsed for one PWM clock period upon a time-base counter equal to counter compare C event (TBCTR = CMPC) Reset type: SYSRSn
2	CMPBEN	R/W	0h	This bit enables the TBCTR = CMPB event to set the EPWMxSYNCO signal. 0 Disabled 1 The EPWMxSYNCO signal is pulsed for one PWM clock period upon a time-base counter equal to counter compare B event (TBCTR = CMPB) Reset type: SYSRSn
1	ZEROEN	R/W	0h	This bit enables the TBCTR = 0x0000 event to set the EPWMxSYNCOUT signal. 0 Disabled 1 The EPWMxSYNCOUT signal is pulsed for one PWM clock period upon the value of TBCTR changing to 0x0000 Reset type: SYSRSn

Table 20-29. EPWMSYNCOUTEN Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	SWEN	R/W	1h	This bit enables the TBCTL.SWFSYNC bit to set the EPWMxSYNCO signal. 0 Disabled 1 The EPWMxSYNCO signal is pulsed for one PWM clock period when the TBCTL.SWFSYNC bit is set Reset type: SYSRSn

20.17.2.7 TBCTL3 Register (Offset = 7h) [Reset = 0h]

TBCTL3 is shown in Figure 20-95 and described in Table 20-30.

Return to the Summary Table.

Time Base Control Register 3

Figure 20-95. TBCTL3 Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							OSSFRCEN
R-0h							R/W-0h

Table 20-30. TBCTL3 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R	0h	Reserved
0	OSSFRCEN	R/W	0h	This bit determines which bit sets the EPWMxSYNCOUT One Shot Latch. 0 TBCTL2[OSHTSYNC] sets the One Shot Latch 1 GLDCTL2[OSHTLD] sets the One Shot Latch Reset type: SYSRSn

20.17.2.8 CMPCTL Register (Offset = 8h) [Reset = 0h]

CMPCTL is shown in Figure 20-96 and described in Table 20-31.

Return to the Summary Table.

Counter Compare Control Register

Figure 20-96. CMPCTL Register

15	14	13	12	11	10	9	8
RESERVED		LOADBSYNC		LOADASYNC		SHDWBFULL	SHDWAFULL
R-0h		R/W-0h		R/W-0h		R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	SHDWBMODE	RESERVED	SHDWAMODE	LOADBMODE		LOADAMODE	
R-0h	R/W-0h	R-0h	R/W-0h	R/W-0h		R/W-0h	

Table 20-31. CMPCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	RESERVED	R	0h	Reserved
13-12	LOADBSYNC	R/W	0h	Shadow to Active CMPB Register Load on SYNC event 00: Shadow to Active Load of CMPB:CMPBHR occurs according to LOADBMODE (bits 1,0) (same as legacy) 01: Shadow to Active Load of CMPB:CMPBHR occurs both according to LOADBMODE bits and when SYNC occurs 10: Shadow to Active Load of CMPB:CMPBHR occurs only when a SYNC is received 11: Reserved Note: This bit is valid only if CMPCTL[SHDWBMODE] = 0. Reset type: SYSRSn
11-10	LOADASYNC	R/W	0h	Shadow to Active CMPA Register Load on SYNC event 00: Shadow to Active Load of CMPA:CMPAHR occurs according to LOADAMODE (bits 1,0) (same as legacy) 01: Shadow to Active Load of CMPA:CMPAHR occurs both according to LOADAMODE bits and when SYNC occurs 10: Shadow to Active Load of CMPA:CMPAHR occurs only when a SYNC is received 11: Reserved Note: This bit is valid only if CMPCTL[SHDWAMODE] = 0. Reset type: SYSRSn
9	SHDWBFULL	R	0h	Counter-compare B (CMPB) Shadow Register Full Status Flag This bit self clears once a loadstrobe occurs. 0: CMPB shadow FIFO not full yet 1: Indicates the CMPB shadow FIFO is full a CPU write will overwrite current shadow value Reset type: SYSRSn
8	SHDWAFULL	R	0h	Counter-compare A (CMPA) Shadow Register Full Status Flag The flag bit is set when a 32-bit write to CMPA:CMPAHR register or a 16-bit write to CMPA register is made. A 16-bit write to CMPAHR register will not affect the flag. This bit self clears once a load-strobe occurs. 0: CMPA shadow FIFO not full yet 1: Indicates the CMPA shadow FIFO is full, a CPU write will overwrite the current shadow value Reset type: SYSRSn
7	RESERVED	R	0h	Reserved

Table 20-31. CMPCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	SHDWBMODE	R/W	0h	Counter-compare B (CMPB) Register Operating Mode 0: Shadow mode. Operates as a double buffer. All writes via the CPU access the shadow register 1: Immediate mode. Only the active compare B register is used. All writes and reads directly access the active register for immediate compare action Reset type: SYSRSn
5	RESERVED	R	0h	Reserved
4	SHDWAMODE	R/W	0h	Counter-compare A (CMPA) Register Operating Mode 0: Shadow mode. Operates as a double buffer. All writes via the CPU access the shadow register 1: Immediate mode. Only the active compare register is used. All writes and reads directly access the active register for immediate compare action Reset type: SYSRSn
3-2	LOADBMODE	R/W	0h	Active Counter-Compare B (CMPB) Load From Shadow Select Mode This bit has no effect in immediate mode (CMPCTL[SHDWBMODE] = 1). 00: Load on CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000) 01: Load on CTR = PRD: Time-base counter equal to period (TBCTR = TBPRD) 10: Load on either CTR = Zero or CTR = PRD 11: Freeze (no loads possible) Reset type: SYSRSn
1-0	LOADAMODE	R/W	0h	Active Counter-Compare A (CMPA) Load From Shadow Select Mode This bit has no effect in immediate mode (CMPCTL[SHDWAMODE] = 1). 00: Load on CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000) 01: Load on CTR = PRD: Time-base counter equal to period (TBCTR = TBPRD) 10: Load on either CTR = Zero or CTR = PRD 11: Freeze (no loads possible) Reset type: SYSRSn

20.17.2.9 CMPCTL2 Register (Offset = 9h) [Reset = 0h]

CMPCTL2 is shown in Figure 20-97 and described in Table 20-32.

Return to the Summary Table.

Counter Compare Control Register 2

Figure 20-97. CMPCTL2 Register

15	14	13	12	11	10	9	8
RESERVED		LOADDSYNC		LOADCSYNC		RESERVED	
R-0h		R/W-0h		R/W-0h		R-0h	
7	6	5	4	3	2	1	0
RESERVED	SHDWDMODE	RESERVED	SHDWCMODE	LOADDMODE		LOADCMODE	
R-0h	R/W-0h	R-0h	R/W-0h	R/W-0h		R/W-0h	

Table 20-32. CMPCTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	RESERVED	R	0h	Reserved
13-12	LOADDSYNC	R/W	0h	Shadow to Active CMPD Register Load on SYNC event 00: Shadow to Active Load of CMPD occurs according to LOADDMODE 01: Shadow to Active Load of CMPD occurs both according to LOADDMODE bits and when SYNC occurs 10: Shadow to Active Load of CMPD occurs only when a SYNC is received 11: Reserved Note: This bit is valid only if CMPCTL2[SHDWDMODE] = 0. Reset type: SYSRSn
11-10	LOADCSYNC	R/W	0h	Shadow to Active CMPC Register Load on SYNC event 00: Shadow to Active Load of CMPC occurs according to LOADCMODE 01: Shadow to Active Load of CMPC occurs both according to LOADCMODE bits and when SYNC occurs 10: Shadow to Active Load of CMPC occurs only when a SYNC is received 11: Reserved Note: This bit is valid only if CMPCTL2[SHDWCMODE] = 0. Reset type: SYSRSn
9-7	RESERVED	R	0h	Reserved
6	SHDWDMODE	R/W	0h	Counter-Compare D Register Operating Mode 0: Shadow mode - operates as a double buffer. All writes via the CPU access Shadow register. 1: Immediate mode - only the Active compare register is used. All writes/reads via the CPU directly access the Active register for immediate Compare action. Reset type: SYSRSn
5	RESERVED	R	0h	Reserved
4	SHDWCMODE	R/W	0h	Counter-Compare C Register Operating Mode 0: Shadow mode - operates as a double buffer. All writes via the CPU access Shadow register. 1: Immediate mode - only the Active compare register is used. All writes/reads via the CPU directly access the Active register for immediate Compare action. Reset type: SYSRSn

Table 20-32. CMPCTL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	LOADDMODE	R/W	0h	Active Counter-Compare D (CMPD) Load from Shadow Select Mode 00: Load on CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000) 01: Load on CTR = PRD: Time-base counter equal to period (TBCTR = TBPRD) 10: Load on either CTR = Zero or CTR = PRD 11: Freeze (no loads possible) Note: Has no effect in Immediate mode. Reset type: SYSRSn
1-0	LOADCMODE	R/W	0h	Active Counter-Compare C (CMPC) Load from Shadow Select Mode 00: Load on CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000) 01: Load on CTR = PRD: Time-base counter equal to period (TBCTR = TBPRD) 10: Load on either CTR = Zero or CTR = PRD 11: Freeze (no loads possible) Note: Has no effect in Immediate mode. Reset type: SYSRSn

20.17.2.10 DBCTL Register (Offset = Ch) [Reset = 0h]

DBCTL is shown in Figure 20-98 and described in Table 20-33.

Return to the Summary Table.

Dead-Band Generator Control Register

Figure 20-98. DBCTL Register

15	14	13	12	11	10	9	8
HALFCYCLE	DEDB_MODE	OUTSWAP		SHDWDBFED MODE	SHDWDBRED MODE	LOADFEDMODE	
R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h	R/W-0h	
7	6	5	4	3	2	1	0
LOADREDMODE		IN_MODE		POLSEL		OUT_MODE	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 20-33. DBCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	HALFCYCLE	R/W	0h	Half Cycle Clocking Enable Bit 0: Full cycle clocking enabled. The dead-band counters are clocked at the TBCLK rate. 1: Half cycle clocking enabled. The dead-band counters are clocked at TBCLK*2. Reset type: SYSRSn
14	DEDB_MODE	R/W	0h	Dead Band Dual-Edge B Mode Control (S8 switch) 0: Rising edge delay applied to InA/InB as selected by S4 switch (IN-MODE bits) on A signal path only. Falling edge delay applied to InA/InB as selected by S5 switch (INMODE bits) on B signal path only. 1: Rising edge delay and falling edge delay applied to source selected by S4 switch (INMODE bits) and output to B signal path only. Note: When this bit is set to 1, user should always either set OUT_MODE bits such that Apath = InA OR OUTSWAP bits such that OutA=Bpath otherwise, OutA will be invalid. Reset type: SYSRSn
13-12	OUTSWAP	R/W	0h	Dead Band Output Swap Control Bit 13 controls the S6 switch and bit 12 controls the S7 switch. 00: OutA and OutB signals are as defined by OUT-MODE bits. 01: OutA = A-path as defined by OUT-MODE bits. OutB = A-path as defined by OUT-MODE bits (rising edge delay or delay-bypassed A signal path). 10: OutA = B-path as defined by OUT-MODE bits (falling edge delay or delay-bypassed B signal path). OutB = B-path as defined by OUT-MODE bits. 11: OutA = B-path as defined by OUT-MODE bits (falling edge delay or delay-bypassed B signal path). OutB = A-path as defined by OUT-MODE bits (rising edge delay or delay-bypassed A signal path). Reset type: SYSRSn
11	SHDWDBFEDMODE	R/W	0h	FED Dead-Band Load Mode 0: Immediate mode. Only the active DBFED register is used. All writes/reads via the CPU directly access the active register for immediate "FED dead-band action." 1: Shadow mode. Operates as a double buffer. All writes via the CPU access Shadow register. Default at Reset is Immediate mode (for compatibility with legacy). Reset type: SYSRSn

Table 20-33. DBCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	SHDWDBREDMODE	R/W	0h	RED Dead-Band Load Mode 0: Immediate mode. Only the active DBRED register is used. All writes/reads via the CPU directly access the active register for immediate "RED dead-band action." 1: Shadow mode. Operates as a double buffer. All writes via the CPU access Shadow register. Default at Reset is Immediate mode (for compatibility with legacy). Reset type: SYSRSn
9-8	LOADFEDMODE	R/W	0h	Active DBFED Load from Shadow Select Mode 00: Load on Counter = 0 (CNT_eq) 01: Load on Counter = Period (PRD_eq) 10: Load on either Counter = 0, or Counter = Period 11: Freeze (no loads possible) Note: has no effect in Immediate mode. Reset type: SYSRSn
7-6	LOADREDMODE	R/W	0h	Active DBRED Load from Shadow Select Mode 00: Load on Counter = 0 (CNT_eq) 01: Load on Counter = Period (PRD_eq) 10: Load on either Counter = 0, or Counter = Period 11: Freeze (no loads possible) Note: has no effect in Immediate mode. Reset type: SYSRSn
5-4	IN_MODE	R/W	0h	Dead-Band Input Mode Control Bit 5 controls the S5 switch and bit 4 controls the S4 switch shown. This allows you to select the input source to the falling-edge and rising-edge delay. To produce classical dead-band waveforms the default is EPWMxA In is the source for both falling and rising-edge delays. 00: EPWMxA In (from the action-qualifier) is the source for both falling-edge and rising-edge delay. 01: EPWMxB In (from the action-qualifier) is the source for rising-edge delayed signal. EPWMxA In (from the action-qualifier) is the source for falling-edge delayed signal. 10: EPWMxA In (from the action-qualifier) is the source for rising-edge delayed signal. EPWMxB In (from the action-qualifier) is the source for falling-edge delayed signal. 11: EPWMxB In (from the action-qualifier) is the source for both rising-edge delay and falling-edge delayed signal. Reset type: SYSRSn
3-2	POLSEL	R/W	0h	Polarity Select Control Bit 3 controls the S3 switch and bit 2 controls the S2 switch. This allows you to selectively invert one of the delayed signals before it is sent out of the dead-band submodule. The following descriptions correspond to classical upper/lower switch control as found in one leg of a digital motor control inverter. These assume that DBCTL[OUT_MODE] = 1,1 and DBCTL[IN_MODE] = 0x0. Other enhanced modes are also possible, but not regarded as typical usage modes. 00: Active high (AH) mode. Neither EPWMxA nor EPWMxB is inverted (default). 01: Active low complementary (ALC) mode. EPWMxA is inverted. 10: Active high complementary (AHC). EPWMxB is inverted. 11: Active low (AL) mode. Both EPWMxA and EPWMxB are inverted. Reset type: SYSRSn

Table 20-33. DBCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	OUT_MODE	R/W	0h	Dead-Band Output Mode Control Bit 1 controls the S1 switch and bit 0 controls the S0 switch. 00: DBM is fully disabled or by-passed. In this mode the POLSEL and IN-MODE bits have no effect. 01: Apath = InA (delay is by-passed for A signal path) Bpath = FED (Falling Edge Delay in B signal path) 10: Apath = RED (Rising Edge Delay in A signal path) Bpath = InB (delay is by-passed for B signal path) 11: DBM is fully enabled (i.e. both RED and FED active) Reset type: SYSRSn

20.17.2.11 DBCTL2 Register (Offset = Dh) [Reset = 0h]

DBCTL2 is shown in Figure 20-99 and described in Table 20-34.

Return to the Summary Table.

Dead-Band Generator Control Register 2

Figure 20-99. DBCTL2 Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					SHDWDBCTLMODE	LOADDBCTLMODE	
R-0h					R/W-0h	R/W-0h	

Table 20-34. DBCTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-3	RESERVED	R	0h	Reserved
2	SHDWDBCTLMODE	R/W	0h	DBCTL Load Mode 0: Immediate mode - only the Active DBCTL register is used. All writes/reads via the CPU directly access the Active register. 1: Shadow mode - All writes and reads to bits [5:0] of the DBCTL register are shadowed. All other bits still access the active register. Reset type: SYSRSn
1-0	LOADDBCTLMODE	R/W	0h	Active DBCTL Load from Shadow Select Mode 00: Load on Counter = 0 (CNT_eq) 01: Load on Counter = Period (PRD_eq) 10: Load on either Counter = 0, or Counter = Period 11: Freeze (no loads possible) Note: has no effect in Immediate mode Reset type: SYSRSn

20.17.2.12 AQCTL Register (Offset = 10h) [Reset = 0h]

AQCTL is shown in Figure 20-100 and described in Table 20-35.

Return to the Summary Table.

Action Qualifier Control Register

Figure 20-100. AQCTL Register

15	14	13	12	11	10	9	8
RESERVED				LDAQBSYNC		LDAQASYNC	
R-0h				R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED	SHDWAQBMODE	RESERVED	SHDWAQAMODE	LDAQBMODE		LDAQAMODE	
R-0h	R/W-0h	R-0h	R/W-0h	R/W-0h		R/W-0h	

Table 20-35. AQCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-10	LDAQBSYNC	R/W	0h	Shadow to Active AQCTLB Register Load on SYNC event 00: Shadow to Active Load of AQCTLB occurs according to LDAQBMODE 01: Shadow to Active Load of AQCTLB occurs both according to LDAQBMODE bits and when SYNC occurs. 10: Shadow to Active Load of AQCTLB occurs only when a SYNC is received. 11: Reserved Note: This bit is valid only if AQCTL[SHDWAQBMODE] = 1. Reset type: SYSRSn
9-8	LDAQASYNC	R/W	0h	Shadow to Active AQCTLA Register Load on SYNC event 00: Shadow to Active Load of AQCTLA occurs according to LDAQAMODE 01: Shadow to Active Load of AQCTLA occurs both according to LDAQAMODE bits and when SYNC occurs. 10: Shadow to Active Load of AQCTLA occurs only when a SYNC is received. 11: Reserved Note: This bit is valid only if AQCTL[SHDWAQAMODE] = 1. Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6	SHDWAQBMODE	R/W	0h	Action Qualifier B Register operating mode 1: Shadow mode - operates as a double buffer. All writes via the CPU access Shadow register. 0: Immediate mode - only the Active action qualifier register is used. All writes/reads via the CPU directly access the Active register. Reset type: SYSRSn
5	RESERVED	R	0h	Reserved
4	SHDWAQAMODE	R/W	0h	Action Qualifier A Register operating mode 1: Shadow mode - operates as a double buffer. All writes via the CPU access Shadow register. 0: Immediate mode - only the Active action qualifier register is used. All writes/reads via the CPU directly access the Active register. Reset type: SYSRSn

Table 20-35. AQCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	LDAQBMODE	R/W	0h	Active Action Qualifier B Load from Shadow Select Mode 00: Load on CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000) 01: Load on CTR = PRD: Time-base counter equal to period (TBCTR = TBPRD) 10: Load on either CTR = Zero or CTR = PRD 11: Freeze (no loads possible) Note: has no effect in Immediate mode. Reset type: SYSRSn
1-0	LDAQAMODE	R/W	0h	Active Action Qualifier A Load from Shadow Select Mode 00: Load on CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000) 01: Load on CTR = PRD: Time-base counter equal to period (TBCTR = TBPRD) 10: Load on either CTR = Zero or CTR = PRD 11: Freeze (no loads possible) Note: has no effect in Immediate mode. Reset type: SYSRSn

20.17.2.13 AQTSRCSEL Register (Offset = 11h) [Reset = 0h]

AQTSRCSEL is shown in Figure 20-101 and described in Table 20-36.

Return to the Summary Table.

Action Qualifier Trigger Event Source Select Register

Figure 20-101. AQTSRCSEL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
T2SEL				T1SEL			
R/W-0h				R/W-0h			

Table 20-36. AQTSRCSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-4	T2SEL	R/W	0h	T2 Event Source Select Bits 0000: DCAEVT1 0001: DCAEVT2 0010: DCBEVT1 0011: DCBEVT2 0100: TZ1 0101: TZ2 0110: TZ3 0111: EPWMxSYNCI 1000: DCEVTFILT Others: Reserved Reset type: SYSRSn
3-0	T1SEL	R/W	0h	T1 Event Source Select Bits 0000: DCAEVT1 0001: DCAEVT2 0010: DCBEVT1 0011: DCBEVT2 0100: TZ1 0101: TZ2 0110: TZ3 0111: EPWMxSYNCI 1000: DCEVTFILT Others: Reserved Reset type: SYSRSn

20.17.2.14 PCCTL Register (Offset = 14h) [Reset = 0h]

PCCTL is shown in Figure 20-102 and described in Table 20-37.

Return to the Summary Table.

PWM Chopper Control Register

Figure 20-102. PCCTL Register

15	14	13	12	11	10	9	8
RESERVED						CHPDUTY	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
CHPFREQ			OSHTWTH				CHPEN
R/W-0h			R/W-0h				R/W-0h

Table 20-37. PCCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R	0h	Reserved
10-8	CHPDUTY	R/W	0h	Chopping Clock Duty Cycle 000: Duty = 1/8 (12.5%) 001: Duty = 2/8 (25.0%) 010: Duty = 3/8 (37.5%) 011: Duty = 4/8 (50.0%) 100: Duty = 5/8 (62.5%) 101: Duty = 6/8 (75.0%) 110: Duty = 7/8 (87.5%) 111: Reserved Reset type: SYSRSn
7-5	CHPFREQ	R/W	0h	Chopping Clock Frequency 000: Divide by 1 (no prescale, = 12.5 MHz at 100 MHz TBCLK) 001: Divide by 2 (6.25 MHz at 100 MHz TBCLK) 010: Divide by 3 (4.16 MHz at 100 MHz TBCLK) 011: Divide by 4 (3.12 MHz at 100 MHz TBCLK) 100: Divide by 5 (2.50 MHz at 100 MHz TBCLK) 101: Divide by 6 (2.08 MHz at 100 MHz TBCLK) 110: Divide by 7 (1.78 MHz at 100 MHz TBCLK) 111: Divide by 8 (1.56 MHz at 100 MHz TBCLK) Reset type: SYSRSn
4-1	OSHTWTH	R/W	0h	One-Shot Pulse Width 0000: 1 x EPWMCLK / 8 wide (= 80 ns at 100 MHz EPWMCLK) 0001: 2 x EPWMCLK / 8 wide (= 160 ns at 100 MHz EPWMCLK) 0010: 3 x EPWMCLK / 8 wide (= 240 ns at 100 MHz EPWMCLK) 0011: 4 x EPWMCLK / 8 wide (= 320 ns at 100 MHz EPWMCLK) 0100: 5 x EPWMCLK / 8 wide (= 400 ns at 100 MHz EPWMCLK) 0101: 6 x EPWMCLK / 8 wide (= 480 ns at 100 MHz EPWMCLK) 0110: 7 x EPWMCLK / 8 wide (= 560 ns at 100 MHz EPWMCLK) 0111: 8 x EPWMCLK / 8 wide (= 640 ns at 100 MHz EPWMCLK) 1000: 9 x EPWMCLK / 8 wide (= 720 ns at 100 MHz EPWMCLK) 1001: 10 x EPWMCLK / 8 wide (= 800 ns at 100 MHz EPWMCLK) 1010: 11 x EPWMCLK / 8 wide (= 880 ns at 100 MHz EPWMCLK) 1011: 12 x EPWMCLK / 8 wide (= 960 ns at 100 MHz EPWMCLK) 1100: 13 x EPWMCLK / 8 wide (= 1040 ns at 100 MHz EPWMCLK) 1101: 14 x EPWMCLK / 8 wide (= 1120 ns at 100 MHz EPWMCLK) 1110: 15 x EPWMCLK / 8 wide (= 1200 ns at 100 MHz EPWMCLK) 1111: 16 x EPWMCLK / 8 wide (= 1280 ns at 100 MHz EPWMCLK) Reset type: SYSRSn
0	CHPEN	R/W	0h	PWM-Chopping Enable 0: Disable (bypass) PWM chopping function 1: Enable chopping function Reset type: SYSRSn

20.17.2.15 VCAPCTL Register (Offset = 18h) [Reset = 0h]

VCAPCTL is shown in Figure 20-103 and described in Table 20-38.

Return to the Summary Table.

Valley Capture Control Register

Figure 20-103. VCAPCTL Register

15	14	13	12	11	10	9	8
RESERVED					EDGEFILTDLYSEL	VDELAYDIV	
R-0h					R/W-0h	R/W-0h	
7	6	5	4	3	2	1	0
VDELAYDIV	RESERVED		TRIGSEL			VCAPSTART	VCAPE
R/W-0h	R-0h		R/W-0h			R-0/W1S-0h	R/W-0h

Table 20-38. VCAPCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R	0h	Reserved
10	EDGEFILTDLYSEL	R/W	0h	Valley Switching Mode Delay Selection 0: No delay applied to the edge filter output 1: HWDELAYVAL delay applied to the edge filter output Reset type: SYSRSn
9-7	VDELAYDIV	R/W	0h	Valley Delay Mode Divide Enable 000: HWVDELVAL = SWVDELVAL 001: HWVDELVAL = VCNTVAL+SWVDELVAL 010: HWVDELVAL = VCNTVAL>>1+SWVDELVAL 011: HWVDELVAL = VCNTVAL>>2+SWVDELVAL 100: HWVDELVAL = VCNTVAL>>4+SWVDELVAL Note: Delay value between the consecutive edge captures can optionally be divided by using these bits. Reset type: SYSRSn
6-5	RESERVED	R	0h	Reserved
4-2	TRIGSEL	R/W	0h	Status of Numbered of Captured Events 000: Capture sequence is triggered by software via writes to VCAPCTL[VCAPSTART]. 001: Capture sequence is triggered by CNT_zero event. 010: Capture sequence is triggered by PRD_eq event. 011: Capture sequence is triggered by CNT_zero or PRD_eq event. 100: Capture sequence is triggered by DCAEVT1 event. 101: Capture sequence is triggered by DCAEVT2 event. 110: Capture sequence is triggered by DCBEVT1 event. 111: Capture sequence is triggered by DCBEVT2 event. Note: Valley capture sequence triggered by the selected event in this register field. Once the chosen event occurs the capture sequence is armed. Event captures occur based of the event chosen in DCFCTL[SRCSEL] register. Note: Same event may not be chosen in both DCFCTL[SRCSEL] and VCAPCTL[TRIGSEL] registers. Note: Once the chosen event in VCAPCTL[TRIGSEL] occurs, irrespective of the current capture status, capture sequence is retriggered. Reset type: SYSRSn
1	VCAPSTART	R-0/W1S	0h	Valley Capture Start 0: Writing a 0 has no effect 1: Trigger the capture sequence once if VCAPCTL[TRIGSEL]=0x0 Note: This bit is used to start valley capture sequence through software. VCAPCTL[TRIGSEL] has to be chosen for software trigger for this bit to have any effect. Writing of 1 will result in one capture sequence trigger. Reset type: SYSRSn

Table 20-38. VCAPCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	VCAPE	R/W	0h	Valley Capture Enable/Disable 0: Disabled 1: Enabled Reset type: SYSRSn

20.17.2.16 VCNTCFG Register (Offset = 19h) [Reset = 0h]

VCNTCFG is shown in Figure 20-104 and described in Table 20-39.

Return to the Summary Table.

Valley Counter Config Register

Figure 20-104. VCNTCFG Register

15	14	13	12	11	10	9	8
STOPEDGESTS	RESERVED			STOPEDGE			
R-0h	R-0h			R/W-0h			
7	6	5	4	3	2	1	0
STARTEDGESTS	RESERVED			STARTEDGE			
R-0h	R-0h			R/W-0h			

Table 20-39. VCNTCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
15	STOPEDGESTS	R	0h	Stop Edge Status Bit 0: Stop edge has not occurred 1: Stop edge occurred Note: This bit is set only after the trigger sequence is armed (upon occurrence of trigger pulse selected through VCAPCTL[TRIGSEL]) and STOPEDGE occurs. Note:This bit is reset by the occurrence of the trigger pulse selected through VCAPCTL[TRIGSEL] Reset type: SYSRSn
14-12	RESERVED	R	0h	Reserved
11-8	STOPEDGE	R/W	0h	Counter Stop Edge Selection Once the counter operation is armed, upon occurrence of trigger pulse selected through VCAPCTL[TRIGSEL] pulse - valley counter would stop counting upon the occurrence of chosen number of events thorough this bit field. Stop counting on occurrence of: 0000: Do not stop 0001 1st edge 0010: 2nd edge 0011: 3rd edge ... 1,1,1,1: 15th edge Reset type: SYSRSn
7	STARTEDGESTS	R	0h	Start Edge Status Bit 0: Start edge has not occurred 1: Start edge occurred Note: This bit is set only after the trigger sequence is armed (upon occurrence of trigger pulse selected through VCAPCTL[TRIGSEL]) and STARTEDGE occurs. Note:This bit is reset by the occurrence of the trigger pulse selected through VCAPCTL[TRIGSEL] Reset type: SYSRSn
6-4	RESERVED	R	0h	Reserved

Table 20-39. VCNTCFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	STARTEDGE	R/W	0h	Counter Start Edge Selection Once the counter operation is armed, upon occurrence of trigger pulse selected through VCAPCTL[TRIGSEL] pulse - valley counter would start counting upon the occurrence of chosen number of events thorough this bit field. Start counting on occurrence of 0000: Do not start 0001: 1st edge 0010: 2nd edge 0011: 3rd edge ... 1111: 15th edge Reset type: SYSRSn

20.17.2.17 HRCNFG Register (Offset = 20h) [Reset = 0h]

HRCNFG is shown in Figure 20-105 and described in Table 20-40.

Return to the Summary Table.

HRPWM Configuration Register
This register is only accessible on EPWM modules with HRPWM capabilities.

Figure 20-105. HRCNFG Register

15	14	13	12	11	10	9	8
RESERVED		RESERVED	HRLOADB		CTLMODEB	EDGMODEB	
R/W-0h		R-0h	R/W-0h		R/W-0h	R/W-0h	
7	6	5	4	3	2	1	0
SWAPAB	AUTOCONV	SELOUTB	HRLOAD		CTLMODE	EDGMODE	
R/W-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h	

Table 20-40. HRCNFG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	RESERVED	R/W	0h	Reserved
13	RESERVED	R	0h	Reserved
12-11	HRLOADB	R/W	0h	Shadow Mode Bit Selects the time event that loads the CMPBHR shadow value into the active register. 00: Load on CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000) 01: Load on CTR = PRD: Time-base counter equal to period (TBCTR = TBPRD) 10: Load on either CTR = Zero or CTR = PRD 11: Reserved Reset type: SYSRSn
10	CTLMODEB	R/W	0h	Control Mode Bits Selects the register (CMP/TBPRD or TBPHS) that controls the MEP: 0: CMPBHR(8) or TBPRDHR(8) Register controls the edge position (i.e., this is duty or period control mode). (Default on Reset) 1: TBPHSHR(8) Register controls the edge position (i.e., this is phase control mode). Reset type: SYSRSn
9-8	EDGMODEB	R/W	0h	Edge Mode Bits Selects the edge of the PWM that is controlled by the micro-edge position (MEP) logic: 00: HRPWM capability is disabled (default on reset) 01: MEP control of rising edge (CMPBHR) 10: MEP control of falling edge (CMPBHR) 11: MEP control of both edges (TBPHSHR or TBPRDHR) Reset type: SYSRSn
7	SWAPAB	R/W	0h	Swap ePWM A & B Output Signals This bit enables the swapping of the A & B signal outputs. The selection is as follows: 0: ePWMxA and ePWMxB outputs are unchanged. 1: ePWMxA signal appears on ePWMxB output and ePWMxB signal appears on ePWMxA output. Reset type: SYSRSn

Table 20-40. HRCNFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	AUTOCONV	R/W	0h	Auto Convert Delay Line Value Selects whether the fractional duty cycle/period/phase in the CMPAHR/TBPRDHR/TBPHSHR register is automatically scaled by the MEP scale factor in the HRMSTEP register or manually scaled by calculations in application software. The SFO library function automatically updates the HRMSTEP register with the appropriate MEP scale factor. 0: Automatic HRMSTEP scaling is disabled. 1: Automatic HRMSTEP scaling is enabled. If application software is manually scaling the fractional duty cycle, or phase (i.e. software sets CMPAHR = (fraction(PWMduty * PWMperiod) * MEP Scale Factor)<<8 + 0x080 for duty cycle), then this mode must be disabled. Reset type: SYSRSn
5	SELOUTB	R/W	0h	EPWMxB Output Select Bit This bit selects which signal is output on the ePWMxB channel output. The inversion will take the high resolution mode into account and the inverted signal will contain any high resolution modification. The inversion takes place as the last step in modifying the ePWMxB signal. 0: ePWMxB output is normal. 1: ePWMxB output is inverted version of ePWMxA signal. Reset type: SYSRSn
4-3	HRLOAD	R/W	0h	Shadow Mode Bit Selects the time event that loads the CMPAHR shadow value into the active register. 00: Load on CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000) 01: Load on CTR = PRD: Time-base counter equal to period (TBCTR = TBPRD) 10: Load on either CTR = Zero or CTR = PRD 11: Reserved Reset type: SYSRSn
2	CTLMODE	R/W	0h	Control Mode Bits Selects the register (CMP/TBPRD or TBPHS) that controls the MEP: 0: CMPAHR(8) or TBPRDHR(8) Register controls the edge position (i.e., this is duty or period control mode). (Default on Reset) 1: TBPHSHR(8) Register controls the edge position (i.e., this is phase control mode). Reset type: SYSRSn
1-0	EDGMODE	R/W	0h	Edge Mode Bits Selects the edge of the PWM that is controlled by the micro-edge position (MEP) logic: 00: HRPWM capability is disabled (default on reset) 01: MEP control of rising edge (CMPAHR) 10: MEP control of falling edge (CMPAHR) 11: MEP control of both edges (TBPHSHR or TBPRDHR) Reset type: SYSRSn

20.17.2.18 HRPWR Register (Offset = 21h) [Reset = 0h]

HRPWR is shown in Figure 20-106 and described in Table 20-41.

Return to the Summary Table.

HRPWM Power Register
This register is only accessible on EPWM modules with HRPWM capabilities.

Figure 20-106. HRPWR Register

15	14	13	12	11	10	9	8
CALPWRON	RESERVED					RESERVED	
R/W-0h	R-0h					R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		RESERVED	RESERVED	RESERVED	RESERVED	RESERVED	
R/W-0h		R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	

Table 20-41. HRPWR Register Field Descriptions

Bit	Field	Type	Reset	Description
15	CALPWRON	R/W	0h	MEP Calibration Power Bits (only available on ePWM1) 0: Disables MEP calibration logic in the HRPWM and reduces power consumption. 1: Enables MEP calibration logic Reset type: SYSRSn
14-10	RESERVED	R	0h	Reserved
9-6	RESERVED	R/W	0h	Reserved
5	RESERVED	R/W	0h	Reserved
4	RESERVED	R	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	RESERVED	R/W	0h	Reserved
1-0	RESERVED	R/W	0h	Reserved

20.17.2.19 HRMSTEP Register (Offset = 26h) [Reset = 0h]

HRMSTEP is shown in Figure 20-107 and described in Table 20-42.

Return to the Summary Table.

HRPWM MEP Step Register
This register is only accessible on EPWM modules with HRPWM capabilities. Only 16 bit accesses are allowed for this register. Debugger access in 32 bit mode may display incorrect values.

Figure 20-107. HRMSTEP Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
HRMSTEP							
R/W-0h							

Table 20-42. HRMSTEP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-0	HRMSTEP	R/W	0h	High Resolution MEP Step When auto-conversion is enabled (HRCNFG[AUTOCONV] = 1), This 8-bit field contains the MEP_ScaleFactor (number of MEP steps per coarse steps) used by the hardware to automatically convert the value in the CMPAHR, CMPBHR, DBFEDHR, DBREDHR , TBPHSHR, or TBPRDHR register to a scaled micro-edge delay on the high-resolution ePWM output. The value in this register is written by the SFO calibration software at the end of each calibration run. Reset type: SYSRSn

20.17.2.20 HRCNFG2 Register (Offset = 27h) [Reset = 0h]

HRCNFG2 is shown in Figure 20-108 and described in Table 20-43.

Return to the Summary Table.

HRPWM Configuration 2 Register
This register is only accessible on EPWM modules with HRPWM capabilities. Only 16 bit accesses are allowed for this register. Debugger access in 32 bit mode may display incorrect values.

Figure 20-108. HRCNFG2 Register

15	14	13	12	11	10	9	8
RESERVED	RESERVED	RESERVED					
R/W-0h	R-0/W1S-0h	R-0h					
7	6	5	4	3	2	1	0
RESERVED		CTLMODEDBFED		CTLMODEDBRED		EDGMODEDB	
R-0h		R/W-0h		R/W-0h		R/W-0h	

Table 20-43. HRCNFG2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R/W	0h	Reserved
14	RESERVED	R-0/W1S	0h	Reserved
13-6	RESERVED	R	0h	Reserved
5-4	CTLMODEDBFED	R/W	0h	Shadow Mode Bit - selection should match DBCTL[LOADFEDMODE] Selects the time event that loads the DBFEDHR shadow value into the active register. 00 Load on CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000) 01 Load on CTR = PRD: Time-base counter equal to period (TBCTR = TBPRD) 10 Load on either CTR = Zero or CTR = PRD 11 Reserved Reset type: SYSRSn
3-2	CTLMODEDBRED	R/W	0h	Shadow Mode Bit - selection should match DBCTL[LOADREDMODE] Selects the time event that loads the DBREDHR shadow value into the active register. 00 Load on CTR = Zero: Time-base counter equal to zero (TBCTR = 0x0000) 01 Load on CTR = PRD: Time-base counter equal to period (TBCTR = TBPRD) 10 Load on either CTR = Zero or CTR = PRD 11 Reserved Reset type: SYSRSn
1-0	EDGMODEDB	R/W	0h	Edge Mode Bits Selects the edge of the PWM that is controlled by the micro-edge position (MEP) logic: 00 HRPWM capability is disabled (default on reset) 01 MEP control of rising edge (DBREDHR) 10 MEP control of falling edge (DBFEDHR) 11 MEP control of both edges (rising edge of DBREDHR or falling edge of DBFEDHR) Reset type: SYSRSn

20.17.2.21 HRPCTL Register (Offset = 2Dh) [Reset = 0h]

HRPCTL is shown in Figure 20-109 and described in Table 20-44.

Return to the Summary Table.

High Resolution Period Control Register
This register is only accessible on EPWM modules with HRPWM capabilities.

Figure 20-109. HRPCTL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	PWMSYNCSELX			RESERVED	TBPHSHRLOADE	PWMSYNCSEL	HRPE
R-0h	R/W-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 20-44. HRPCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R	0h	Reserved
6-4	PWMSYNCSELX	R/W	0h	Extended selection bits for EPWMSYNCPER 000: EPWMSYNCPER is defined by PWMSYNCSEL - > default condition (compatible with previous EPWM versions) 001: Reserved 010: Reserved 011: Reserved 100: CTR = CMPC, Count direction Up 101: CTR = CMPC, Count direction Down 110: CTR = CMPD, Count direction Up 111: CTR = CMPD, Count direction Down Reset type: SYSRSn
3	RESERVED	R/W	0h	Reserved
2	TBPHSHRLOADE	R/W	0h	TBPHSHR Load Enable This bit allows you to synchronize ePWM modules with a high-resolution phase on a SYNCIN, TBCTL[SWFSYNC] or digital compare event. This allows for multiple ePWM modules operating at the same frequency to be phase aligned with high-resolution. 0: Disables synchronization of high-resolution phase on a SYNCIN, TBCTL[SWFSYNC] or digital compare event: 1: Synchronize the high-resolution phase on a SYNCIN, TBCTL[SWFSYNC] or digital comparator synchronization event. The phase is synchronized using the contents of the high-resolution phase TBPHSHR register. The TBCTL[PHSEN] bit which enables the loading of the TBCTR register with TBPHS register value on a SYNCIN or TBCTL[SWFSYNC] event works independently. However, users need to enable this bit also if they want to control phase in conjunction with the high-resolution period feature. This bit and the TBCTL[PHSEN] bit must be set to 1 when high-resolution period is enabled for up-down count mode even if TBPHSHR = 0x0000. This bit does not need to be set when only high-resolution duty is enabled. Reset type: SYSRSn
1	PWMSYNCSEL	R/W	0h	PWMSYNC Source Select Bit: This bit selects the source for the EPWMSYNCPER signal that goes to the CMPSS and GPDAC: 0 CTR = PRD: Time-base counter equal to period (TBCTR = TBPRD) 1 CTR = zero: Time-base counter equal to zero (TBCTR = 0x00) Reset type: SYSRSn

Table 20-44. HRPCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	HRPE	R/W	0h	High Resolution Period Enable Bit 0: High resolution period feature disabled. In this mode the ePWM behaves as a Type 0 ePWM. 1: High resolution period enabled. In this mode the HRPWM module can control high-resolution of both the duty and frequency. When high-resolution period is enabled, TBCTL[CTRMODE] = 0,1 (down-count mode) is not supported. Reset type: SYSRSn

20.17.2.22 TRREM Register (Offset = 2Eh) [Reset = 0h]

TRREM is shown in Figure 20-110 and described in Table 20-45.

Return to the Summary Table.

HRPWM High Resolution Remainder Register
This register is only accessible on EPWM modules with HRPWM capabilities.

Figure 20-110. TRREM Register

15	14	13	12	11	10	9	8
RESERVED					TRREM		
R-0h					R/W-0h		
7	6	5	4	3	2	1	0
TRREM							
R/W-0h							

Table 20-45. TRREM Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R	0h	Reserved
10-0	TRREM	R/W	0h	HRPWM Remainder Bits: This 11-bit value keeps track of the remainder portion of the HRPWM algorithm calculations. This value keeps track of the remainder portion of the HRPWM hardware calculations. Notes: 1. The lower 8-bits of the TRREM register can be automatically initialized with the TBPHSHR value on a SYNCIN or TBCTL[SWFSYNC] event or DC event (if enabled). The user can also write a value with the CPU. 2. Priority of TRREM register updates: Sync (software or hardware) TBPHSHR copied to TRREM : Highest Priority HRPWM Hardware (updates TRREM register): Next priority CPU Write To TRREM Register: Lowest Priority 3. Bit 10 of TRREM register is not used in asymmetrical mode. This bit can be forced to zero. TRREM will be initialized to 0x0 and 0x100 in Up and Up-down modes respectively. Asymmetrical Mode: TRREM[7:0] = TBPHSHR[15:8] TRREM[10,9,8] = 0,0,0 Symmetrical Mode: TRREM[7:0] = TBPHSHR[15:8] TRREM[10,9,8] = 0,0,1 Reset type: SYSRSn

20.17.2.23 GLDCTL Register (Offset = 34h) [Reset = 0h]

GLDCTL is shown in Figure 20-111 and described in Table 20-46.

Return to the Summary Table.

Global PWM Load Control Register

Figure 20-111. GLDCTL Register

15	14	13	12	11	10	9	8
RESERVED			GLDCNT			GLDPRD	
R-0h			R-0h			R/W-0h	
7	6	5	4	3	2	1	0
GLDPRD	RESERVED	OSHTMODE	GLDMODE				GLD
R/W-0h	R-0h	R/W-0h	R/W-0h				R/W-0h

Table 20-46. GLDCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12-10	GLDCNT	R	0h	Global Load Strobe Counter Register These bits indicate how many selected events have occurred: 000: No events 001: 1 event 010: 2 events 011: 3 events 100: 4 events 101: 5 events 110: 6 events 111: 7 events Reset type: SYSRSn
9-7	GLDPRD	R/W	0h	Global Load Strobe Period Select Register These bits select how many selected events need to occur before a load strobe is generated 000: Disable counter 001: Generate strobe on GLDCNT = 001 (1st event) 010: Generate strobe on GLDCNT = 010 (2nd event) 011: Generate strobe on GLDCNT = 011 (3rd event) 100: Generate strobe on GLDCNT = 011 (4th event) 101: Generate strobe on GLDCNT = 001 (5th event) 110: Generate strobe on GLDCNT = 010 (6th event) 111: Generate strobe on GLDCNT = 011 (7th event) Reset type: SYSRSn
6	RESERVED	R	0h	Reserved
5	OSHTMODE	R/W	0h	One Shot Load Mode Control Bit 0: One shot load mode is disabled and shadow to active loading happens continuously on all the chosen load strobes. 1: One shot mode is active. All load strobes are blocked until GLDCTL2[OSHTLD] is written with 1. Note: One Shot mode can only be used with global shadow to active load mode enabled (GLDCTL[GLD]=1) Reset type: SYSRSn

Table 20-46. GLDCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-1	GLDMODE	R/W	0h	Global Load Pulse selection for Shadow to Active Mode Reloads 0000: Load on Counter = 0 (CNT_ZRO) 0001: Load on Counter = Period (PRD_EQ) 0010: Load on either Counter = 0, or Counter = Period 0011: Load on SYNCEVT - this is logical OR of DCAEVT1.sync, DCBEVT1.sync, EPWMxSYNCI and TBCTL[SWFSYNC] 0100: Load on SYNCEVT or CNT_ZRO 0101: Load on SYNCEVT or PRD_EQ 0110: Load on SYNCEVT or CNT_ZRO or PRD_EQ 1000: Reserved ... 1110: Reserved 1111: Load on GLDCTL2[GFRCLD] write Reset type: SYSRSn
0	GLD	R/W	0h	Global Shadow to Active Load Event Control 0: Shadow to active reload for all shadowed registers happens as per the individual reload control bits specified (Compatible with previous EPWM versions). 1: When set, all the shadow to active reload events are defined by GLDMODE bits in GLDCTL register. All the shadow registers use same reload pulse from shadow to active reloading. Individual LOADMODE bits are ignored. Reset type: SYSRSn

20.17.2.24 GLDCFG Register (Offset = 35h) [Reset = 0h]

GLDCFG is shown in Figure 20-112 and described in Table 20-47.

Return to the Summary Table.

Global PWM Load Config Register

Figure 20-112. GLDCFG Register

15	14	13	12	11	10	9	8
RESERVED					AQCSFRC	AQCTLB_AQCTLB2	AQCTLA_AQCTLA2
R-0h					R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DBCTL	DBFED_DBFEDHR	DBRED_DBREDHR	CMPD	CMPC	CMPB_CMPBHR	CMPA_CMPAHR	TBPRD_TBPRDHR
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 20-47. GLDCFG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R	0h	Reserved
10	AQCSFRC	R/W	0h	Global load event configuration for AQCSFRC 0: Registers use local reload configuration even if GLDCTL(GLD)=1 (reload is compatible with previous EPWMs) 1: Registers use global load configuration if this bit is set and GLDCTL(GLD)=1 Reset type: SYSRSn
9	AQCTLB_AQCTLB2	R/W	0h	Global load event configuration for AQCTLB_AQCTLB2 0: Registers use local reload configuration even if GLDCTL(GLD)=1 (reload is compatible with previous EPWMs) 1: Registers use global load configuration if this bit is set and GLDCTL(GLD)=1 Reset type: SYSRSn
8	AQCTLA_AQCTLA2	R/W	0h	Global load event configuration for AQCTLA_AQCTLA2 0: Registers use local reload configuration even if GLDCTL(GLD)=1 (reload is compatible with previous EPWMs) 1: Registers use global load configuration if this bit is set and GLDCTL(GLD)=1 Reset type: SYSRSn
7	DBCTL	R/W	0h	Global load event configuration for DBCTL 0: Registers use local reload configuration even if GLDCTL(GLD)=1 (reload is compatible with previous EPWMs) 1: Registers use global load configuration if this bit is set and GLDCTL(GLD)=1 Reset type: SYSRSn
6	DBFED_DBFEDHR	R/W	0h	Global load event configuration for DBFED_DBFEDHR 0: Registers use local reload configuration even if GLDCTL(GLD)=1 (reload is compatible with previous EPWMs) 1: Registers use global load configuration if this bit is set and GLDCTL(GLD)=1 Reset type: SYSRSn
5	DBRED_DBREDHR	R/W	0h	Global load event configuration for DBRED_DBREDHR 0: Registers use local reload configuration even if GLDCTL(GLD)=1 (reload is compatible with previous EPWMs) 1: Registers use global load configuration if this bit is set and GLDCTL(GLD)=1 Reset type: SYSRSn

Table 20-47. GLDCFG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	CMPD	R/W	0h	Global load event configuration for CMPD 0: Registers use local reload configuration even if GLDCTL(GLD)=1 (reload is compatible with previous EPWMs) 1: Registers use global load configuration if this bit is set and GLDCTL(GLD)=1 Reset type: SYSRSn
3	CMPC	R/W	0h	Global load event configuration for CMPC 0: Registers use local reload configuration even if GLDCTL(GLD)=1 (reload is compatible with previous EPWMs) 1: Registers use global load configuration if this bit is set and GLDCTL(GLD)=1 Reset type: SYSRSn
2	CMPB_CMPBHR	R/W	0h	Global load event configuration for CMPB_CMPBHR 0: Registers use local reload configuration even if GLDCTL(GLD)=1 (reload is compatible with previous EPWMs) 1: Registers use global load configuration if this bit is set and GLDCTL(GLD)=1 Reset type: SYSRSn
1	CMPA_CMPAHR	R/W	0h	Global load event configuration for CMPA_CMPAHR 0: Registers use local reload configuration even if GLDCTL(GLD)=1 (reload is compatible with previous EPWMs) 1: Registers use global load configuration if this bit is set and GLDCTL(GLD)=1 Reset type: SYSRSn
0	TBPRD_TBPRDHR	R/W	0h	Global load event configuration for TBPRD_TBPRDHR 0: Registers use local reload configuration even if GLDCTL(GLD)=1 (reload is compatible with previous EPWMs) 1: Registers use global load configuration if this bit is set and GLDCTL(GLD)=1 Reset type: SYSRSn

20.17.2.25 EPWMXLINK Register (Offset = 38h) [Reset = X]

EPWMXLINK is shown in Figure 20-113 and described in Table 20-48.

Return to the Summary Table.

EPWMx Link Register
This register controls which EPWMs are linked to other EPWM modules. The default reset value will vary for each module. The reset value will link each EPWM module to itself to prevent unintentional linking of modules.

Figure 20-113. EPWMXLINK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
GLDCTL2LINK				RESERVED								CMPDLINK			
R/W-X				R-0h								R/W-X			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMPCLINK				CMPBLINK				CMPALINK				TBPRDLINK			
R/W-X				R/W-X				R/W-X				R/W-X			

Table 20-48. EPWMXLINK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	GLDCTL2LINK	R/W	X	GLDCTL2 Link Bits Writes to the GLDCTL2 registers in the ePWM module selected by the following bit selections results in a simultaneous write to the current ePWM module's GLDCTL2 registers. 0000: ePWM1 0001: ePWM2 0010: ePWM3 0011: ePWM4 0100: ePWM5 0101: ePWM6 0110: ePWM7 0111: ePWM8 1000: ePWM9 1001: ePWM10 1010: ePWM11 1011: ePWM12 1100: Reserved ... 1111: Reserved Reset type: SYSRSn
27-20	RESERVED	R	0h	Reserved
19-16	CMPDLINK	R/W	X	CMPD Link Bits Writes to the CMPD registers in the ePWM module selected by the following bit selections results in a simultaneous write to the current ePWM module's CMPD registers. 0000: ePWM1 0001: ePWM2 0010: ePWM3 0011: ePWM4 0100: ePWM5 0101: ePWM6 0110: ePWM7 0111: ePWM8 1000: ePWM9 1001: ePWM10 1010: ePWM11 1011: ePWM12 1100: Reserved ... 1111: Reserved Reset type: SYSRSn

Table 20-48. EPWMXLINK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-12	CMPCLINK	R/W	X	CMPC Link Bits Writes to the CMPC registers in the ePWM module selected by the following bit selections results in a simultaneous write to the current ePWM module's CMPC registers. 0000: ePWM1 0001: ePWM2 0010: ePWM3 0011: ePWM4 0100: ePWM5 0101: ePWM6 0110: ePWM7 0111: ePWM8 1000: ePWM9 1001: ePWM10 1010: ePWM11 1011: ePWM12 1100: Reserved ... 1111: Reserved Reset type: SYSRSn
11-8	CMPBLINK	R/W	X	CMPB_CMPBHR Link Bits Writes to the CMPB_CMPBHR registers in the ePWM module selected by the following bit selections results in a simultaneous write to the current ePWM module's CMPB_CMPBHR registers. 0000: ePWM1 0001: ePWM2 0010: ePWM3 0011: ePWM4 0100: ePWM5 0101: ePWM6 0110: ePWM7 0111: ePWM8 1000: ePWM9 1001: ePWM10 1010: ePWM11 1011: ePWM12 1100: Reserved ... 1111: Reserved Reset type: SYSRSn
7-4	CMPALINK	R/W	X	CMPA_CMPAHR Link Bits Writes to the CMPA_CMPAHR registers in the ePWM module selected by the following bit selections results in a simultaneous write to the current ePWM module's CMPA_CMPAHR registers. 0000: ePWM1 0001: ePWM2 0010: ePWM3 0011: ePWM4 0100: ePWM5 0101: ePWM6 0110: ePWM7 0111: ePWM8 1000: ePWM9 1001: ePWM10 1010: ePWM11 1011: ePWM12 1100: Reserved ... 1111: Reserved Reset type: SYSRSn

Table 20-48. EPWMXLINK Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	TBPRDLINK	R/W	X	TBPRD_TBPRDHR Link Bits Writes to the TBPRD:TBPRDHR registers in the ePWM module selected by the following bit selections results in a simultaneous write to the current ePWM module's TBPRD_TBPRDHR registers. 0000: ePWM1 0001: ePWM2 0010: ePWM3 0011: ePWM4 0100: ePWM5 0101: ePWM6 0110: ePWM7 0111: ePWM8 1000: ePWM9 1001: ePWM10 1010: ePWM11 1011: ePWM12 1100: Reserved ... 1111: Reserved Reset type: SYSRSn

20.17.2.26 AQCTLA Register (Offset = 40h) [Reset = 0h]

AQCTLA is shown in Figure 20-114 and described in Table 20-49.

Return to the Summary Table.

Action Qualifier Control Register For Output A

Figure 20-114. AQCTLA Register

15	14	13	12	11	10	9	8
RESERVED				CBD		CBU	
R-0h				R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
CAD		CAU		PRD		ZRO	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 20-49. AQCTLA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-10	CBD	R/W	0h	Action When TBCTR = CMPB on Down Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxA output low. 10: Set: force EPWMxA output high. 11: Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
9-8	CBU	R/W	0h	Action When TBCTR = CMPB on Up Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxA output low. 10: Set: force EPWMxA output high. 11: Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
7-6	CAD	R/W	0h	Action When TBCTR = CMPA on Down Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxA output low. 10: Set: force EPWMxA output high. 11: Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
5-4	CAU	R/W	0h	Action When TBCTR = CMPA on Up Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxA output low. 10: Set: force EPWMxA output high. 11: Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn

Table 20-49. AQCTLA Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	PRD	R/W	0h	Action When TBCTR = TBPRD Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxA output low. 10: Set: force EPWMxA output high. 11: Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
1-0	ZRO	R/W	0h	Action When TBCTR = 0 Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxA output low. 10: Set: force EPWMxA output high. 11: Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn

20.17.2.27 AQCTLA2 Register (Offset = 41h) [Reset = 0h]

AQCTLA2 is shown in Figure 20-115 and described in Table 20-50.

Return to the Summary Table.

Additional Action Qualifier Control Register For Output A

Figure 20-115. AQCTLA2 Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
T2D		T2U		T1D		T1U	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 20-50. AQCTLA2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-6	T2D	R/W	0h	Action when event occurs on T2 in DOWN-Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxA output low. 10: Set: force EPWMxA output high. 11: Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
5-4	T2U	R/W	0h	Action when event occurs on T2 in UP-Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxA output low. 10: Set: force EPWMxA output high. 11: Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
3-2	T1D	R/W	0h	Action when event occurs on T1 in DOWN-Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxA output low. 10: Set: force EPWMxA output high. 11: Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
1-0	T1U	R/W	0h	Action when event occurs on T1 in UP-Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxA output low. 10: Set: force EPWMxA output high. 11: Toggle EPWMxA output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn

20.17.2.28 AQCTLB Register (Offset = 42h) [Reset = 0h]

AQCTLB is shown in Figure 20-116 and described in Table 20-51.

Return to the Summary Table.

Action Qualifier Control Register For Output B

Figure 20-116. AQCTLB Register

15	14	13	12	11	10	9	8
RESERVED				CBD		CBU	
R-0h				R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
CAD		CAU		PRD		ZRO	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 20-51. AQCTLB Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-10	CBD	R/W	0h	Action When TBCTR = CMPB on Down Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxB output low. 10: Set: force EPWMxB output high. 11: Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
9-8	CBU	R/W	0h	Action When TBCTR = CMPB on Up Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxB output low. 10: Set: force EPWMxB output high. 11: Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
7-6	CAD	R/W	0h	Action When TBCTR = CMPA on Down Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxB output low. 10: Set: force EPWMxB output high. 11: Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
5-4	CAU	R/W	0h	Action When TBCTR = CMPA on Up Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxB output low. 10: Set: force EPWMxB output high. 11: Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn

Table 20-51. AQCTLB Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-2	PRD	R/W	0h	Action When TBCTR = TBPRD Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxB output low. 10: Set: force EPWMxB output high. 11: Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
1-0	ZRO	R/W	0h	Action When TBCTR = 0 Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxB output low. 10: Set: force EPWMxB output high. 11: Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn

20.17.2.29 AQCTLB2 Register (Offset = 43h) [Reset = 0h]

AQCTLB2 is shown in Figure 20-117 and described in Table 20-52.

Return to the Summary Table.

Additional Action Qualifier Control Register For Output B

Figure 20-117. AQCTLB2 Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
T2D		T2U		T1D		T1U	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 20-52. AQCTLB2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-6	T2D	R/W	0h	Action when event occurs on T2 in DOWN-Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxB output low. 10: Set: force EPWMxB output high. 11: Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
5-4	T2U	R/W	0h	Action when event occurs on T2 in UP-Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxB output low. 10: Set: force EPWMxB output high. 11: Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
3-2	T1D	R/W	0h	Action when event occurs on T1 in DOWN-Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxB output low. 10: Set: force EPWMxB output high. 11: Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn
1-0	T1U	R/W	0h	Action when event occurs on T1 in UP-Count Note: By definition, in count up-down mode when the counter equals 0 the direction is defined as 1 or counting up. 00: Do nothing (action disabled) 01: Clear: force EPWMxB output low. 10: Set: force EPWMxB output high. 11: Toggle EPWMxB output: low output signal will be forced high, and a high signal will be forced low. Reset type: SYSRSn

20.17.2.30 AQSFRC Register (Offset = 47h) [Reset = 0h]

AQSFRC is shown in Figure 20-118 and described in Table 20-53.

Return to the Summary Table.

Action Qualifier Software Force Register

Figure 20-118. AQSFRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RLDCSF		OTSFB	ACTSFB		OTSFA	ACTSFA	
R/W-0h		R-0/W1S-0h	R/W-0h		R-0/W1S-0h	R/W-0h	

Table 20-53. AQSFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-6	RLDCSF	R/W	0h	AQCSFRC Active Register Reload From Shadow Options 00: Load on time-base counter equals zero 01: Load on time-base counter equals period 10: Load on time-base counter equals zero or counter equals period 11: Load immediately (the active register is directly accessed by the CPU and is not loaded from the shadow register). Reset type: SYSRSn
5	OTSFB	R-0/W1S	0h	One-Time Software Forced Event on Output B 0: Writing a 0 (zero) has no effect. Always reads back a 0. This bit is auto cleared once a write to this register is complete (i.e., a forced event is initiated.). This is a one-shot forced event. It can be overridden by another subsequent event on output B. 1: Initiates a single software forced event Reset type: SYSRSn
4-3	ACTSFB	R/W	0h	Action When One-Time Software Force B is Invoked 00: Does nothing (action disabled) 01: Clear (low) 10: Set (high) 11: Toggle (Low -> High, High -> Low) Note: This action is not qualified by counter direction (CNT_dir) Reset type: SYSRSn
2	OTSFA	R-0/W1S	0h	One-Time Software Forced Event on Output A 0: Writing a 0 (zero) has no effect. Always reads back a 0. This bit is auto cleared once a write to this register is complete (i.e., a forced event is initiated). This is a one-shot forced event. It can be overridden by another subsequent event on output A. 1: Initiates a single software forced event Reset type: SYSRSn
1-0	ACTSFA	R/W	0h	Action When One-Time Software Force A Is Invoked 00: Does nothing (action disabled) 01: Clear (low) 10: Set (high) 11: Toggle (Low -> High, High -> Low) Note: This action is not qualified by counter direction (CNT_dir) Reset type: SYSRSn

20.17.2.31 AQCSFRC Register (Offset = 49h) [Reset = 0h]

AQCSFRC is shown in Figure 20-119 and described in Table 20-54.

Return to the Summary Table.

Action Qualifier Continuous S/W Force Register

Figure 20-119. AQCSFRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				CSFB		CSFA	
R-0h				R/W-0h		R/W-0h	

Table 20-54. AQCSFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3-2	CSFB	R/W	0h	Continuous Software Force on Output B In immediate mode, a continuous force takes effect on the next TBCLK edge. In shadow mode, a continuous force takes effect on the next TBCLK edge after a shadow load into the active register. To configure shadow mode, use AQSFRC[RLDCSF]. 00: Software forcing is disabled and has no effect 01: Forces a continuous low on output B 10: Forces a continuous high on output B 11: Software forcing is disabled and has no effect Reset type: SYSRSn
1-0	CSFA	R/W	0h	Continuous Software Force on Output A In immediate mode, a continuous force takes effect on the next TBCLK edge. In shadow mode, a continuous force takes effect on the next TBCLK edge after a shadow load into the active register. 00: Software forcing is disabled and has no effect 01: Forces a continuous low on output A 10: Forces a continuous high on output A 11: Software forcing is disabled and has no effect Reset type: SYSRSn

20.17.2.32 DBREDHR Register (Offset = 50h) [Reset = 0h]

DBREDHR is shown in Figure 20-120 and described in Table 20-55.

Return to the Summary Table.

Dead-Band Generator Rising Edge Delay High Resolution Mirror Register

Figure 20-120. DBREDHR Register

15	14	13	12	11	10	9	8
DBREDHR							RESERVED
R/W-0h							R-0h
7	6	5	4	3	2	1	0
RESERVED							RESERVED
R-0h							R-0h

Table 20-55. DBREDHR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-9	DBREDHR	R/W	0h	Dead Band Rising Edge Delay High Resolution Bits Reset type: SYSRSn
8	RESERVED	R	0h	Reserved
7-1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

20.17.2.33 DBRED Register (Offset = 51h) [Reset = 0h]

DBRED is shown in Figure 20-121 and described in Table 20-56.

Return to the Summary Table.

Dead-Band Generator Rising Edge Delay High Resolution Mirror Register

Figure 20-121. DBRED Register

15	14	13	12	11	10	9	8
RESERVED		DBRED					
R-0h		R/W-0h					
7	6	5	4	3	2	1	0
DBRED							
R/W-0h							

Table 20-56. DBRED Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	RESERVED	R	0h	Reserved
13-0	DBRED	R/W	0h	Rising edge delay value Reset type: SYSRSn

20.17.2.34 DBFEDHR Register (Offset = 52h) [Reset = 0h]

DBFEDHR is shown in Figure 20-122 and described in Table 20-57.

Return to the Summary Table.

Dead-Band Generator Falling Edge Delay High Resolution Register

Figure 20-122. DBFEDHR Register

15	14	13	12	11	10	9	8
DBFEDHR							RESERVED
R/W-0h							R-0h
7	6	5	4	3	2	1	0
RESERVED							RESERVED
R-0h							R-0h

Table 20-57. DBFEDHR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-9	DBFEDHR	R/W	0h	Dead Band Falling Edge Delay High Resolution Bits Reset type: SYSRSn
8	RESERVED	R	0h	Reserved
7-1	RESERVED	R	0h	Reserved
0	RESERVED	R	0h	Reserved

20.17.2.35 DBFED Register (Offset = 53h) [Reset = 0h]

DBFED is shown in Figure 20-123 and described in Table 20-58.

Return to the Summary Table.

Dead-Band Generator Falling Edge Delay Count Register

Figure 20-123. DBFED Register

15	14	13	12	11	10	9	8
RESERVED			DBFED				
R-0h			R/W-0h				
7	6	5	4	3	2	1	0
DBFED							
R/W-0h							

Table 20-58. DBFED Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	RESERVED	R	0h	Reserved
13-0	DBFED	R/W	0h	Falling Edge Delay Count 14-bit counter Reset type: SYSRSn

20.17.2.36 TBPHS Register (Offset = 60h) [Reset = 0h]

TBPHS is shown in Figure 20-124 and described in Table 20-59.

Return to the Summary Table.

Time Base Phase High

Figure 20-124. TBPHS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBPHS																TBPHSHR															
R/W-0h																R/W-0h															

Table 20-59. TBPHS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	TBPHS	R/W	0h	Phase Offset Register These bits set time-base counter phase of the selected ePWM relative to the time-base that is supplying the synchronization input signal. - If TBCTL[PHSEN] = 0, then the synchronization event is ignored and the time-base counter is not loaded with the phase. - If TBCTL[PHSEN] = 1, then the time-base counter (TBCTR) will be loaded with the phase (TBPHS) when a synchronization event occurs. The synchronization event can be initiated by the input synchronization signal (EPWMxSYNCI) or by a software forced synchronization. Reset type: SYSRSn
15-0	TBPHSHR	R/W	0h	Phase Offset (High Resolution) Register. TBPHSHR must not be used. Instead TRREM (HRPWM remainder register) must be used to mimic the functionality of TBPHSHR. The lower 8 bits in this register are ignored - writes are ignored and reads return zero Reset type: SYSRSn

20.17.2.37 TBPRDHR Register (Offset = 62h) [Reset = 0h]

TBPRDHR is shown in Figure 20-125 and described in Table 20-60.

Return to the Summary Table.

Time Base Period High Resolution Register

Figure 20-125. TBPRDHR Register

15	14	13	12	11	10	9	8
TBPRDHR							
R/W-0h							
7	6	5	4	3	2	1	0
TBPRDHR							
R/W-0h							

Table 20-60. TBPRDHR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	TBPRDHR	R/W	0h	Period High Resolution Bits The upper 8-bits contain the high-resolution portion of the period value. The TBPRDHR register is not affected by the TBCTL[PRDLD] bit. Reads from this register always reflect the shadow register. Likewise writes are also to the shadow register. The TBPRDHR register is only used when the high resolution period feature is enabled. This register is only available with ePWM modules which support high-resolution period control. The lower 8 bits in this register are ignored - writes are ignored and reads return zero Reset type: SYSRSn

20.17.2.38 TBPRD Register (Offset = 63h) [Reset = 0h]

TBPRD is shown in Figure 20-126 and described in Table 20-61.

Return to the Summary Table.

Time Base Period Register

Figure 20-126. TBPRD Register

15	14	13	12	11	10	9	8
TBPRD							
R/W-0h							
7	6	5	4	3	2	1	0
TBPRD							
R/W-0h							

Table 20-61. TBPRD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	TBPRD	R/W	0h	Time Base Period Register These bits determine the period of the time-base counter. This sets the PWM frequency. Shadowing of this register is enabled and disabled by the TBCTL[PRDLD] bit. By default this register is shadowed. - If TBCTL[PRDLD] = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the active register will be loaded from the shadow register when the time-base counter equals zero. - If TBCTL[PRDLD] = 1, then the shadow is disabled and any write or read will go directly to the active register, that is the register actively controlling the hardware. - The active and shadow registers share the same memory map address. Reset type: SYSRSn

20.17.2.39 CMPA Register (Offset = 6Ah) [Reset = 0h]

CMPA is shown in Figure 20-127 and described in Table 20-62.

Return to the Summary Table.

Counter Compare A Register

Figure 20-127. CMPA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMPA																CMPAHR															
R/W-0h																R/W-0h															

Table 20-62. CMPA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	CMPA	R/W	0h	Compare A Register The value in the active CMPA register is continuously compared to the time-base counter (TBCTR). When the values are equal, the counter-compare module generates a "time-base counter equal to counter compare A" event. This event is sent to the action-qualifier where it is qualified and converted it into one or more actions. These actions can be applied to either the EPWMxA or the EPWMxB output depending on the configuration of the AQCTLA and AQCTLB registers. The actions that can be defined in the AQCTLA and AQCTLB registers include: - Do nothing the event is ignored. - Clear: Pull the EPWMxA and/or EPWMxB signal low - Set: Pull the EPWMxA and/or EPWMxB signal high - Toggle the EPWMxA and/or EPWMxB signal Shadowing of this register is enabled and disabled by the CMPCTL[SHDWAMODE] bit. By default this register is shadowed. - If CMPCTL[SHDWAMODE] = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the CMPCTL[LOADAMODE] bit field determines which event will load the active register from the shadow register. - Before a write, the CMPCTL[SHDWAFULL] bit can be read to determine if the shadow register is currently full. - If CMPCTL[SHDWAMODE] = 1, then the shadow register is disabled and any write or read will go directly to the active register, that is the register actively controlling the hardware. - In either mode, the active and shadow registers share the same memory map address. Reset type: SYSRSn
15-0	CMPAHR	R/W	0h	Compare A HRPWM Extension Register The UPPER 8-bits contain the high-resolution portion (most significant 8-bits) of the counter-compare A value. CMPA:CMPAHR can be accessed in a single 32-bit read/write. Shadowing is enabled and disabled by the CMPCTL[SHDWAMODE] bit as described for the CMPA register. The lower 8 bits in this register are ignored Reset type: SYSRSn

20.17.2.40 CMPB Register (Offset = 6Ch) [Reset = 0h]

CMPB is shown in Figure 20-128 and described in Table 20-63.

Return to the Summary Table.

Compare B Register

Figure 20-128. CMPB Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CMPB																CMPBHR															
R/W-0h																R/W-0h															

Table 20-63. CMPB Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	CMPB	R/W	0h	Compare B Register The value in the active CMPB register is continuously compared to the time-base counter (TBCTR). When the values are equal, the counter-compare module generates a "time-base counter equal to counter compare B" event. This event is sent to the action-qualifier where it is qualified and converted it into one or more actions. These actions can be applied to either the EPWMxA or the EPWMxB output depending on the configuration of the AQCTLA and AQCTLB registers. The actions that can be defined in the AQCTLA and AQCTLB registers include: - Do nothing the event is ignored. - Clear: Pull the EPWMxA and/or EPWMxB signal low - Set: Pull the EPWMxA and/or EPWMxB signal high - Toggle the EPWMxA and/or EPWMxB signal Shadowing of this register is enabled and disabled by the CMPCTL[SHDWBMODE] bit. By default this register is shadowed. - If CMPCTL[SHDWBMODE] = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the CMPCTL[LOADBMODE] bit field determines which event will load the active register from the shadow register. - Before a write, the CMPCTL[SHDWBFULL] bit can be read to determine if the shadow register is currently full. - If CMPCTL[SHDWBMODE] = 1, then the shadow register is disabled and any write or read will go directly to the active register, that is the register actively controlling the hardware. - In either mode, the active and shadow registers share the same memory map address. Reset type: SYSRSn
15-0	CMPBHR	R/W	0h	Compare B High Resolution Bits The lower 8 bits in this register are ignored Reset type: SYSRSn

20.17.2.41 CMPC Register (Offset = 6Fh) [Reset = 0h]

CMPC is shown in Figure 20-129 and described in Table 20-64.

Return to the Summary Table.

Counter Compare C Register
LINK feature access should always be 16-bit

Figure 20-129. CMPC Register

15	14	13	12	11	10	9	8
CMPC							
R/W-0h							
7	6	5	4	3	2	1	0
CMPC							
R/W-0h							

Table 20-64. CMPC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	CMPC	R/W	0h	Compare C Register The value in the active CMPC register is continuously compared to the time-base counter (TBCTR). When the values are equal, the counter-compare module generates a "time-base counter equal to counter compare C" event. Shadowing of this register is enabled and disabled by the CMPCTL2[SHDWCMODE] bit. By default this register is shadowed. - If CMPCTL2[SHDWCMODE] = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the CMPCTL2[LOADCMODE] bit field determines which event will load the active register from the shadow register: - If CMPCTL2[SHDWCMODE] = 1, then the shadow register is disabled and any write or read will go directly to the active register that is, the register actively controlling the hardware. - In either mode, the active and shadow registers share the same memory map address. Reset type: SYSRSn

20.17.2.42 CMPD Register (Offset = 71h) [Reset = 0h]

CMPD is shown in Figure 20-130 and described in Table 20-65.

Return to the Summary Table.

Counter Compare D Register
LINK feature access should always be 16-bit

Figure 20-130. CMPD Register

15	14	13	12	11	10	9	8
CMPD							
R/W-0h							
7	6	5	4	3	2	1	0
CMPD							
R/W-0h							

Table 20-65. CMPD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	CMPD	R/W	0h	Compare D Register The value in the active CMPD register is continuously compared to the time-base counter (TBCTR). When the values are equal, the counter-compare module generates a "time-base counter equal to counter compare D" event. Shadowing of this register is enabled and disabled by the CMPCTL2[SHDWDMODE] bit. By default this register is shadowed. - If CMPCTL2[SHDWDMODE] = 0, then the shadow is enabled and any write or read will automatically go to the shadow register. In this case, the CMPCTL2[LOADDMODE] bit field determines which event will load the active register from the shadow register: - If CMPCTL2[SHDWDMODE] = 1, then the shadow register is disabled and any write or read will go directly to the active register that is, the register actively controlling the hardware. - In either mode, the active and shadow registers share the same memory map address. Reset type: SYSRSn

20.17.2.43 GLDCTL2 Register (Offset = 74h) [Reset = 0h]

GLDCTL2 is shown in Figure 20-131 and described in Table 20-66.

Return to the Summary Table.

Global PWM Load Control Register 2

Figure 20-131. GLDCTL2 Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						GFRCLD	OSHTLD
R-0h						R-0/W1S-0h	R-0/W1S-0h

Table 20-66. GLDCTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-2	RESERVED	R	0h	Reserved
1	GFRCLD	R-0/W1S	0h	Force Load Event in One Shot Mode 0: Writing of 0 will be ignored. Always reads back a 0. 1: Force one load event at the input of the event pre-scale counter as shown in the diagram below. This bit is intended to be used for testing and/or software force loading of the events in global load mode. Reset type: SYSRSn
0	OSHTLD	R-0/W1S	0h	Enable Reload Event in One Shot Mode 0: Writing of 0 will be ignored. Always reads back a 0. 1: Turns the one shot latch condition ON. Upon occurrence of a chosen load strobe, one shadow to active reload occurs and the latch will be cleared. Hence writing 1 to this bit would allow one load strobe event to pass through and block further strobe events. Reset type: SYSRSn

20.17.2.44 SWVDELVAL Register (Offset = 77h) [Reset = 0h]

SWVDELVAL is shown in Figure 20-132 and described in Table 20-67.

Return to the Summary Table.

Software Valley Mode Delay Register

Figure 20-132. SWVDELVAL Register

15	14	13	12	11	10	9	8
SWVDELVAL							
R/W-0h							
7	6	5	4	3	2	1	0
SWVDELVAL							
R/W-0h							

Table 20-67. SWVDELVAL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SWVDELVAL	R/W	0h	Software Valley Delay Value Register This register can be optionally used define offset value for the hardware calculated delay HWDELAYVAL as defined in VCAPCTL[VDELAYDIV] bits. Reset type: SYSRSn

20.17.2.45 TZSEL Register (Offset = 80h) [Reset = 0h]

TZSEL is shown in Figure 20-133 and described in Table 20-68.

Return to the Summary Table.

Trip Zone Select Register

Figure 20-133. TZSEL Register

15	14	13	12	11	10	9	8
DCBEVT1	DCAEVT1	OSHT6	OSHT5	OSHT4	OSHT3	OSHT2	OSHT1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DCBEVT2	DCAEVT2	CBC6	CBC5	CBC4	CBC3	CBC2	CBC1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 20-68. TZSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	DCBEVT1	R/W	0h	Digital Compare Output B Event 1 Select 0: Disable DCBEVT1 as one-shot-trip source for this ePWM module. 1: Enable DCBEVT1 as one-shot-trip source for this ePWM module. Reset type: SYSRSn
14	DCAEVT1	R/W	0h	Digital Compare Output A Event 1 Select 0: Disable DCAEVT1 as one-shot-trip source for this ePWM module. 1: Enable DCAEVT1 as one-shot-trip source for this ePWM module. Reset type: SYSRSn
13	OSHT6	R/W	0h	Trip-zone 6 (TZ6) Select 0: Disable TZ6 as a one-shot trip source for this ePWM module 1: Enable TZ6 as a one-shot trip source for this ePWM module Reset type: SYSRSn
12	OSHT5	R/W	0h	Trip-zone 5 (TZ5) Select 0: Disable TZ5 as a one-shot trip source for this ePWM module 1: Enable TZ5 as a one-shot trip source for this ePWM module Reset type: SYSRSn
11	OSHT4	R/W	0h	Trip-zone 4 (TZ4) Select 0: Disable TZ4 as a one-shot trip source for this ePWM module 1: Enable TZ4 as a one-shot trip source for this ePWM module Reset type: SYSRSn
10	OSHT3	R/W	0h	Trip-zone 3 (TZ3) Select 0: Disable TZ3 as a one-shot trip source for this ePWM module 1: Enable TZ3 as a one-shot trip source for this ePWM module Reset type: SYSRSn
9	OSHT2	R/W	0h	Trip-zone 2 (TZ2) Select 0: Disable TZ2 as a one-shot trip source for this ePWM module 1: Enable TZ2 as a one-shot trip source for this ePWM module Reset type: SYSRSn
8	OSHT1	R/W	0h	Trip-zone 1 (TZ1) Select 0: Disable TZ1 as a one-shot trip source for this ePWM module 1: Enable TZ1 as a one-shot trip source for this ePWM module Reset type: SYSRSn
7	DCBEVT2	R/W	0h	Digital Compare Output B Event 2 Select 0: Disable DCBEVT2 as a CBC trip source for this ePWM module 1: Enable DCBEVT2 as a CBC trip source for this ePWM module Reset type: SYSRSn
6	DCAEVT2	R/W	0h	Digital Compare Output A Event 2 Select 0: Disable DCAEVT2 as a CBC trip source for this ePWM module 1: Enable DCAEVT2 as a CBC trip source for this ePWM module Reset type: SYSRSn

Table 20-68. TZSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	CBC6	R/W	0h	Trip-zone 6 (TZ6) Select 0: Disable TZ6 as a CBC trip source for this ePWM module 1: Enable TZ6 as a CBC trip source for this ePWM module Reset type: SYSRSn
4	CBC5	R/W	0h	Trip-zone 5 (TZ5) Select 0: Disable TZ5 as a CBC trip source for this ePWM module 1: Enable TZ5 as a CBC trip source for this ePWM module Reset type: SYSRSn
3	CBC4	R/W	0h	Trip-zone 4 (TZ4) Select 0: Disable TZ4 as a CBC trip source for this ePWM module 1: Enable TZ4 as a CBC trip source for this ePWM module Reset type: SYSRSn
2	CBC3	R/W	0h	Trip-zone 3 (TZ3) Select 0: Disable TZ3 as a CBC trip source for this ePWM module 1: Enable TZ3 as a CBC trip source for this ePWM module Reset type: SYSRSn
1	CBC2	R/W	0h	Trip-zone 2 (TZ2) Select 0: Disable TZ2 as a CBC trip source for this ePWM module 1: Enable TZ2 as a CBC trip source for this ePWM module Reset type: SYSRSn
0	CBC1	R/W	0h	Trip-zone 1 (TZ1) Select 0: Disable TZ1 as a CBC trip source for this ePWM module 1: Enable TZ1 as a CBC trip source for this ePWM module Reset type: SYSRSn

20.17.2.46 TZDCSEL Register (Offset = 82h) [Reset = 0h]

TZDCSEL is shown in Figure 20-134 and described in Table 20-69.

Return to the Summary Table.

Trip Zone Digital Comparator Select Register

Figure 20-134. TZDCSEL Register

15	14	13	12	11	10	9	8
RESERVED				DCBEVT2			DCBEVT1
R-0h				R/W-0h			R/W-0h
7	6	5	4	3	2	1	0
DCBEVT1		DCAEVT2			DCAEVT1		
R/W-0h		R/W-0h			R/W-0h		

Table 20-69. TZDCSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-9	DCBEVT2	R/W	0h	Digital Compare Output B Event 2 Selection 000: Event disabled 001: DCBH = low, DCBL = don't care 010: DCBH = high, DCBL = don't care 011: DCBL = low, DCBH = don't care 100: DCBL = high, DCBH = don't care 101: DCBL = high, DCBH = low 110: Reserved 111: Reserved Reset type: SYSRSn
8-6	DCBEVT1	R/W	0h	Digital Compare Output B Event 1 Selection 000: Event disabled 001: DCBH = low, DCBL = don't care 010: DCBH = high, DCBL = don't care 011: DCBL = low, DCBH = don't care 100: DCBL = high, DCBH = don't care 101: DCBL = high, DCBH = low 110: Reserved 111: Reserved Reset type: SYSRSn
5-3	DCAEVT2	R/W	0h	Digital Compare Output A Event 2 Selection 000: Event disabled 001: DCAH = low, DCAL = don't care 010: DCAH = high, DCAL = don't care 011: DCAL = low, DCAH = don't care 100: DCAL = high, DCAH = don't care 101: DCAL = high, DCAH = low 110: Reserved 111: Reserved Reset type: SYSRSn
2-0	DCAEVT1	R/W	0h	Digital Compare Output A Event 1 Selection 000: Event disabled 001: DCAH = low, DCAL = don't care 010: DCAH = high, DCAL = don't care 011: DCAL = low, DCAH = don't care 100: DCAL = high, DCAH = don't care 101: DCAL = high, DCAH = low 110: Reserved 111: Reserved Reset type: SYSRSn

20.17.2.47 TZCTL Register (Offset = 84h) [Reset = 0h]

TZCTL is shown in Figure 20-135 and described in Table 20-70.

Return to the Summary Table.

Trip Zone Control Register

Figure 20-135. TZCTL Register

15	14	13	12	11	10	9	8
RESERVED				DCBEVT2		DCBEVT1	
R-0h				R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
DCAEVT2		DCAEVT1		TZB		TZA	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 20-70. TZCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-10	DCBEVT2	R/W	0h	Digital Compare Output B Event 2 Action On EPWMxB 00: High-impedance (EPWMxB = High-impedance state) 01: Force EPWMxB to a high state. 10: Force EPWMxB to a low state. 11: Do Nothing, trip action is disabled Reset type: SYSRSn
9-8	DCBEVT1	R/W	0h	Digital Compare Output B Event 1 Action On EPWMxB 00: High-impedance (EPWMxB = High-impedance state) 01: Force EPWMxB to a high state. 10: Force EPWMxB to a low state. 11: Do Nothing, trip action is disabled Reset type: SYSRSn
7-6	DCAEVT2	R/W	0h	Digital Compare Output A Event 2 Action On EPWMxA 00: High-impedance (EPWMxA = High-impedance state) 01: Force EPWMxA to a high state. 10: Force EPWMxA to a low state. 11: Do Nothing, trip action is disabled Reset type: SYSRSn
5-4	DCAEVT1	R/W	0h	Digital Compare Output A Event 1 Action On EPWMxA 00: High-impedance (EPWMxA = High-impedance state) 01: Force EPWMxA to a high state. 10: Force EPWMxA to a low state. 11: Do Nothing, trip action is disabled Reset type: SYSRSn
3-2	TZB	R/W	0h	TZ1 to TZ6, DCAEVT1/2, DCBEVT1/2Trip Action On EPWMxB When a trip event occurs the following action is taken on output EPWMxB. Which trip-zone pins can cause an event is defined in the TZSEL register. 00: High-impedance (EPWMxB = High-impedance state) 01: Force EPWMxB to a high state 10: Force EPWMxB to a low state 11: Do nothing, no action is taken on EPWMxB. Reset type: SYSRSn
1-0	TZA	R/W	0h	TZ1 to TZ6, DCAEVT1/2, DCBEVT1/2 Trip Action On EPWMxA When a trip event occurs the following action is taken on output EPWMxA. Which trip-zone pins can cause an event is defined in the TZSEL register. 00: High-impedance (EPWMxA = High-impedance state) 01: Force EPWMxA to a high state 10: Force EPWMxA to a low state 11: Do nothing, no action is taken on EPWMxA. Reset type: SYSRSn

20.17.2.48 TZCTL2 Register (Offset = 85h) [Reset = 0h]

TZCTL2 is shown in Figure 20-136 and described in Table 20-71.

Return to the Summary Table.

Additional Trip Zone Control Register

Figure 20-136. TZCTL2 Register

15	14	13	12	11	10	9	8
ETZE	RESERVED			TZBD			TZBU
R/W-0h	R-0h			R/W-0h			R/W-0h
7	6	5	4	3	2	1	0
TZBU		TZAD			TZAU		
R/W-0h		R/W-0h			R/W-0h		

Table 20-71. TZCTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15	ETZE	R/W	0h	TZCTL2 Enable 0: Use trip action from TZCTL (legacy EPWM compatibility) 1: Use trip action defined in TZCTL2, TZCTLDCA and TZCTLDCB. Settings in TZCTL are ignored Reset type: SYSRSn
14-12	RESERVED	R	0h	Reserved
11-9	TZBD	R/W	0h	TZ1 to TZ6 Trip Action On EPWMxB while Count direction is DOWN 000: HiZ (EPWMxB = HiZ state) 001: Forced Hi (EPWMxB = High state) 010: Forced Lo (EPWMxB = Lo state) 011: Toggle (Low -> High, High -> Low) 100: Reserved 101: Reserved 110: Reserved 111: Do Nothing, trip action is disabled Reset type: SYSRSn
8-6	TZBU	R/W	0h	TZ1 to TZ6, DCAEVT1/2, DCBEVT1/2 Trip Action On EPWMxB while Count direction is UP 000: HiZ (EPWMxB = HiZ state) 001: Forced Hi (EPWMxB = High state) 010: Forced Lo (EPWMxB = Lo state) 011: Toggle (Low -> High, High -> Low) 100: Reserved 101: Reserved 110: Reserved 111: Do Nothing, trip action is disabled Reset type: SYSRSn
5-3	TZAD	R/W	0h	TZ1 to TZ6, DCAEVT1/2, DCBEVT1/2 Trip Action On EPWMxA while Count direction is DOWN 000: HiZ (EPWMxA = HiZ state) 001: Forced Hi (EPWMxA = High state) 010: Forced Lo (EPWMxA = Lo state) 011: Toggle (Low -> High, High -> Low) 100: Reserved 101: Reserved 110: Reserved 111: Do Nothing, trip action is disabled Reset type: SYSRSn

Table 20-71. TZCTL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-0	TZAU	R/W	0h	TZ1 to TZ6, DCAEVT1/2, DCBEVT1/2 Trip Action On EPWMxA while Count direction is UP 000: HiZ (EPWMxA = HiZ state) 001: Forced Hi (EPWMxA = High state) 010: Forced Lo (EPWMxA = Lo state) 011: Toggle (Low -> High, High -> Low) 100: Reserved 101: Reserved 110: Reserved 111: Do Nothing, trip action is disabled Reset type: SYSRSn

20.17.2.49 TZCTLDCA Register (Offset = 86h) [Reset = 0h]

TZCTLDCA is shown in Figure 20-137 and described in Table 20-72.

Return to the Summary Table.

Trip Zone Control Register Digital Compare A

Figure 20-137. TZCTLDCA Register

15	14	13	12	11	10	9	8
RESERVED				DCAEVT2D			DCAEVT2U
R-0h				R/W-0h			R/W-0h
7	6	5	4	3	2	1	0
DCAEVT2U		DCAEVT1D			DCAEVT1U		
R/W-0h		R/W-0h			R/W-0h		

Table 20-72. TZCTLDCA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-9	DCAEVT2D	R/W	0h	Digital Compare Output A Event 2 Action On EPWMxA while Count direction is DOWN 000: HiZ (EPWMxA = HiZ state) 001: Forced Hi (EPWMxA = High state) 010: Forced Lo (EPWMxA = Lo state) 011: Toggle (Low -> High, High -> Low) 100: Reserved 101: Reserved 110: Reserved 111: Do Nothing, trip action is disabled Reset type: SYSRSn
8-6	DCAEVT2U	R/W	0h	Digital Compare Output A Event 2 Action On EPWMxA while Count direction is UP 000: HiZ (EPWMxA = HiZ state) 001: Forced Hi (EPWMxA = High state) 010: Forced Lo (EPWMxA = Lo state) 011: Toggle (Low -> High, High -> Low) 100: Reserved 101: Reserved 110: Reserved 111: Do Nothing, trip action is disabled Reset type: SYSRSn
5-3	DCAEVT1D	R/W	0h	Digital Compare Output A Event 1 Action On EPWMxA while Count direction is DOWN 000: HiZ (EPWMxA = HiZ state) 001: Forced Hi (EPWMxA = High state) 010: Forced Lo (EPWMxA = Lo state) 011: Toggle (Low -> High, High -> Low) 100: Reserved 101: Reserved 110: Reserved 111: Do Nothing, trip action is disabled Reset type: SYSRSn

Table 20-72. TZCTLDCA Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-0	DCAEVT1U	R/W	0h	Digital Compare Output A Event 1 Action On EPWMxA while Count direction is UP 000: HiZ (EPWMxA = HiZ state) 001: Forced Hi (EPWMxA = High state) 010: Forced Lo (EPWMxA = Lo state) 011: Toggle (Low -> High, High -> Low) 100: Reserved 101: Reserved 110: Reserved 111: Do Nothing, trip action is disabled Reset type: SYSRSn

20.17.2.50 TZCTLDCB Register (Offset = 87h) [Reset = 0h]

TZCTLDCB is shown in Figure 20-138 and described in Table 20-73.

Return to the Summary Table.

Trip Zone Control Register Digital Compare B

Figure 20-138. TZCTLDCB Register

15	14	13	12	11	10	9	8
RESERVED				DCBEVT2D			DCBEVT2U
R-0h				R/W-0h			R/W-0h
7	6	5	4	3	2	1	0
DCBEVT2U		DCBEVT1D			DCBEVT1U		
R/W-0h		R/W-0h			R/W-0h		

Table 20-73. TZCTLDCB Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-9	DCBEVT2D	R/W	0h	Digital Compare Output B Event 2 Action On EPWMxB while Count direction is DOWN 000: HiZ (EPWMxB = HiZ state) 001: Forced Hi (EPWMxB = High state) 010: Forced Lo (EPWMxB = Lo state) 011: Toggle (Low -> High, High -> Low) 100: Reserved 101: Reserved 110: Reserved 111: Do Nothing, trip action is disabled Reset type: SYSRSn
8-6	DCBEVT2U	R/W	0h	Digital Compare Output B Event 2 Action On EPWMxB while Count direction is UP 000: HiZ (EPWMxB = HiZ state) 001: Forced Hi (EPWMxB = High state) 010: Forced Lo (EPWMxB = Lo state) 011: Toggle (Low -> High, High -> Low) 100: Reserved 101: Reserved 110: Reserved 111: Do Nothing, trip action is disabled Reset type: SYSRSn
5-3	DCBEVT1D	R/W	0h	Digital Compare Output B Event 1 Action On EPWMxB while Count direction is DOWN 000: HiZ (EPWMxB = HiZ state) 001: Forced Hi (EPWMxB = High state) 010: Forced Lo (EPWMxB = Lo state) 011: Toggle (Low -> High, High -> Low) 100: Reserved 101: Reserved 110: Reserved 111: Do Nothing, trip action is disabled Reset type: SYSRSn

Table 20-73. TZCTLDCB Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-0	DCBEVT1U	R/W	0h	Digital Compare Output B Event 1 Action On EPWMxB while Count direction is UP 000: HiZ (EPWMxB = HiZ state) 001: Forced Hi (EPWMxB = High state) 010: Forced Lo (EPWMxB = Lo state) 011: Toggle (Low -> High, High -> Low) 100: Reserved 101: Reserved 110: Reserved 111: Do Nothing, trip action is disabled Reset type: SYSRSn

20.17.2.51 TZEINT Register (Offset = 8Dh) [Reset = 0h]

TZEINT is shown in Figure 20-139 and described in Table 20-74.

Return to the Summary Table.

Trip Zone Enable Interrupt Register

Figure 20-139. TZEINT Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	DCBEVT2	DCBEVT1	DCAEVT2	DCAEVT1	OST	CBC	RESERVED
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h

Table 20-74. TZEINT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R	0h	Reserved
6	DCBEVT2	R/W	0h	Digital Compare Output B Event 2 Interrupt Enable 0: Disabled 1: Enabled Reset type: SYSRSn
5	DCBEVT1	R/W	0h	Digital Compare Output B Event 1 Interrupt Enable 0: Disabled 1: Enabled Reset type: SYSRSn
4	DCAEVT2	R/W	0h	Digital Compare Output A Event 2 Interrupt Enable 0: Disabled 1: Enabled Reset type: SYSRSn
3	DCAEVT1	R/W	0h	Digital Compare Output A Event 1 Interrupt Enable 0: Disabled 1: Enabled Reset type: SYSRSn
2	OST	R/W	0h	Trip-zone One-Shot Interrupt Enable 0: Disable one-shot interrupt generation 1: Enable Interrupt generation a one-shot trip event will cause a EPWMx_TZINT PIE interrupt. Reset type: SYSRSn
1	CBC	R/W	0h	Trip-zone Cycle-by-Cycle Interrupt Enable 0: Disable cycle-by-cycle interrupt generation. 1: Enable interrupt generation a cycle-by-cycle trip event will cause an EPWMx_TZINT PIE interrupt. Reset type: SYSRSn
0	RESERVED	R	0h	Reserved

20.17.2.52 TZFLG Register (Offset = 93h) [Reset = 0h]

TZFLG is shown in Figure 20-140 and described in Table 20-75.

Return to the Summary Table.

Trip Zone Flag Register

Figure 20-140. TZFLG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	DCBEVT2	DCBEVT1	DCAEVT2	DCAEVT1	OST	CBC	INT
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 20-75. TZFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R	0h	Reserved
6	DCBEVT2	R	0h	Latched Status Flag for Digital Compare Output B Event 2 0: Indicates no trip event has occurred on DCBEVT2 1: Indicates a trip event has occurred for the event defined for DCBEVT2 Reset type: SYSRSn
5	DCBEVT1	R	0h	Latched Status Flag for Digital Compare Output B Event 1 0: Indicates no trip event has occurred on DCBEVT1 1: Indicates a trip event has occurred for the event defined for DCBEVT1 Reset type: SYSRSn
4	DCAEVT2	R	0h	Latched Status Flag for Digital Compare Output A Event 2 0: Indicates no trip event has occurred on DCAEVT2 1: Indicates a trip event has occurred for the event defined for DCAEVT2 Reset type: SYSRSn
3	DCAEVT1	R	0h	Latched Status Flag for Digital Compare Output A Event 1 0: Indicates no trip event has occurred on DCAEVT1 1: Indicates a trip event has occurred for the event defined for DCAEVT1 Reset type: SYSRSn
2	OST	R	0h	Latched Status Flag for A One-Shot Trip Event 0: No one-shot trip event has occurred. 1: Indicates a trip event has occurred on a pin selected as a one-shot trip source. This bit is cleared by writing the appropriate value to the TZCLR register. Reset type: SYSRSn
1	CBC	R	0h	Latched Status Flag for Cycle-By-Cycle Trip Event 0: No cycle-by-cycle trip event has occurred. 1: Indicates a trip event has occurred on a signal selected as a cycle-by-cycle trip source. The TZFLG[CBC] bit will remain set until it is manually cleared by the user. If the cycle-by-cycle trip event is still present when the CBC bit is cleared, then CBC will be immediately set again. The specified condition on the signal is automatically cleared when the ePWM time-base counter reaches zero (TBCTR = 0x00) if the trip condition is no longer present. The condition on the signal is only cleared when the TBCTR = 0x00 no matter where in the cycle the CBC flag is cleared. This bit is cleared by writing the appropriate value to the TZCLR register. Reset type: SYSRSn

Table 20-75. TZFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INT	R	0h	Latched Trip Interrupt Status Flag 0: Indicates no interrupt has been generated. 1: Indicates an EPWMx_TZINT PIE interrupt was generated because of a trip condition. No further EPWMx_TZINT PIE interrupts will be generated until this flag is cleared. If the interrupt flag is cleared when either CBC or OST is set, then another interrupt pulse will be generated. Clearing all flag bits will prevent further interrupts. This bit is cleared by writing the appropriate value to the TZCLR register. Reset type: SYSRSn

20.17.2.53 TZCBCFLG Register (Offset = 94h) [Reset = 0h]

TZCBCFLG is shown in Figure 20-141 and described in Table 20-76.

Return to the Summary Table.

Trip Zone CBC Flag Register

Figure 20-141. TZCBCFLG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
DCBEVT2	DCAEVT2	CBC6	CBC5	CBC4	CBC3	CBC2	CBC1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 20-76. TZCBCFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	DCBEVT2	R	0h	Latched Status Flag for Digital Compare B Output Event 2 Trip Latch 0: Reading a 0 indicates that no trip has occurred on DCBEVT2. 1: Reading a 1 indicates a trip has occured on the DCBEVT2 selected event. Reset type: SYSRSn
6	DCAEVT2	R	0h	Latched Status Flag for Digital Compare A Output Event 2 Trip Latch 0: Reading a 0 indicates that no trip has occurred on DCAEVT2. 1: Reading a 1 indicates a trip has occured on the DCAEVT2 selected event. Reset type: SYSRSn
5	CBC6	R	0h	Latched Status Flag for CBC6 Trip Latch 0: Reading a 0 indicates that no trip has occurred on CBC6. 1: Reading a 1 indicates a trip has occured on the CBC6 selected event. Reset type: SYSRSn
4	CBC5	R	0h	Latched Status Flag for CBC5 Trip Latch 0: Reading a 0 indicates that no trip has occurred on CBC5. 1: Reading a 1 indicates a trip has occured on the CBC5 selected event. Reset type: SYSRSn
3	CBC4	R	0h	Latched Status Flag for CBC4 Trip Latch 0: Reading a 0 indicates that no trip has occurred on CBC4. 1: Reading a 1 indicates a trip has occured on the CBC4 selected event. Reset type: SYSRSn
2	CBC3	R	0h	Latched Status Flag for CBC3 Trip Latch 0: Reading a 0 indicates that no trip has occurred on CBC3. 1: Reading a 1 indicates a trip has occured on the CBC3 selected event. Reset type: SYSRSn
1	CBC2	R	0h	Latched Status Flag for CBC2 Trip Latch 0: Reading a 0 indicates that no trip has occurred on CBC2. 1: Reading a 1 indicates a trip has occured on the CBC2 selected event. Reset type: SYSRSn
0	CBC1	R	0h	Latched Status Flag for CBC1 Trip Latch 0: Reading a 0 indicates that no trip has occurred on CBC1. 1: Reading a 1 indicates a trip has occured on the CBC1 selected event. Reset type: SYSRSn

20.17.2.54 TZOSTFLG Register (Offset = 95h) [Reset = 0h]

TZOSTFLG is shown in Figure 20-142 and described in Table 20-77.

Return to the Summary Table.

Trip Zone OST Flag Register

Figure 20-142. TZOSTFLG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
DCBEVT1	DCAEVT1	OST6	OST5	OST4	OST3	OST2	OST1
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 20-77. TZOSTFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	DCBEVT1	R	0h	Latched Status Flag for Digital Compare B Output Event 1 Trip Latch 0: Reading a 0 indicates that no trip has occurred on DCBEVT1. 1: Reading a 1 indicates a trip has occured on the DCBEVT1 selected event. Reset type: SYSRSn
6	DCAEVT1	R	0h	Latched Status Flag for Digital Compare A Output Event 1 Trip Latch 0: Reading a 0 indicates that no trip has occurred on DCAEVT1. 1: Reading a 1 indicates a trip has occured on the DCAEVT1 selected event. Reset type: SYSRSn
5	OST6	R	0h	Latched Status Flag for OST6 Trip Latch 0: Reading a 0 indicates that no trip has occurred on OST6. 1: Reading a 1 indicates a trip has occured on the OST6 selected event. Reset type: SYSRSn
4	OST5	R	0h	Latched Status Flag for OST5 Trip Latch 0: Reading a 0 indicates that no trip has occurred on OST5. 1: Reading a 1 indicates a trip has occured on the OST5 selected event. Reset type: SYSRSn
3	OST4	R	0h	Latched Status Flag for OST4 Trip Latch 0: Reading a 0 indicates that no trip has occurred on OST4. 1: Reading a 1 indicates a trip has occured on the OST4 selected event. Reset type: SYSRSn
2	OST3	R	0h	Latched Status Flag for OST3 Trip Latch 0: Reading a 0 indicates that no trip has occurred on OST3. 1: Reading a 1 indicates a trip has occured on the OST3 selected event. Reset type: SYSRSn
1	OST2	R	0h	Latched Status Flag for OST2 Trip Latch 0: Reading a 0 indicates that no trip has occurred on OST2. 1: Reading a 1 indicates a trip has occured on the OST2 selected event. Reset type: SYSRSn
0	OST1	R	0h	Latched Status Flag for OST1 Trip Latch 0: Reading a 0 indicates that no trip has occurred on OST1. 1: Reading a 1 indicates a trip has occured on the OST1 selected event. Reset type: SYSRSn

20.17.2.55 TZCLR Register (Offset = 97h) [Reset = 0h]

TZCLR is shown in Figure 20-143 and described in Table 20-78.

Return to the Summary Table.

Trip Zone Clear Register

Figure 20-143. TZCLR Register

15	14	13	12	11	10	9	8
CBCPULSE		RESERVED					
R/W-0h		R-0h					
7	6	5	4	3	2	1	0
RESERVED	DCBEVT2	DCBEVT1	DCAEVT2	DCAEVT1	OST	CBC	INT
R-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 20-78. TZCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	CBCPULSE	R/W	0h	Clear Pulse for Cycle-By-Cycle (CBC) Trip Latch This bit field determines which pulse clears the CBC trip latch. 00: CTR = zero pulse clears CBC trip latch. (Same as legacy designs.) 01: CTR = PRD pulse clears CBC trip latch. 10: CTR = zero or CTR = PRD pulse clears CBC trip latch. 11: CBC trip latch is not cleared Reset type: SYSRSn
13-7	RESERVED	R	0h	Reserved
6	DCBEVT2	R-0/W1S	0h	Clear Flag for Digital Compare Output B Event 2 0: Writing 0 has no effect. This bit always reads back 0. 1: Writing 1 clears the DCBEVT2 event trip condition. Reset type: SYSRSn
5	DCBEVT1	R-0/W1S	0h	Clear Flag for Digital Compare Output B Event 1 0: Writing 0 has no effect. This bit always reads back 0. 1: Writing 1 clears the DCBEVT1 event trip condition. Reset type: SYSRSn
4	DCAEVT2	R-0/W1S	0h	Clear Flag for Digital Compare Output A Event 2 0: Writing 0 has no effect. This bit always reads back 0. 1: Writing 1 clears the DCAEVT2 event trip condition. Reset type: SYSRSn
3	DCAEVT1	R-0/W1S	0h	Clear Flag for Digital Compare Output A Event 1 0: Writing 0 has no effect. This bit always reads back 0. 1: Writing 1 clears the DCAEVT1 event trip condition. Reset type: SYSRSn
2	OST	R-0/W1S	0h	Clear Flag for One-Shot Trip (OST) Latch 0: Has no effect. Always reads back a 0. 1: Clears this Trip (set) condition. Reset type: SYSRSn
1	CBC	R-0/W1S	0h	Clear Flag for Cycle-By-Cycle (CBC) Trip Latch 0: Has no effect. Always reads back a 0. 1: Clears this Trip (set) condition. Reset type: SYSRSn
0	INT	R-0/W1S	0h	Global Interrupt Clear Flag 0: Has no effect. Always reads back a 0. 1: Clears the trip-interrupt flag for this ePWM module (TZFLG[INT]). NOTE: No further EPWMx_TZINT PIE interrupts will be generated until the flag is cleared. If the TZFLG[INT] bit is cleared and any of the other flag bits are set, then another interrupt pulse will be generated. Clearing all flag bits will prevent further interrupts. Reset type: SYSRSn

20.17.2.56 TZCBCCLR Register (Offset = 98h) [Reset = 0h]

TZCBCCLR is shown in Figure 20-144 and described in Table 20-79.

Return to the Summary Table.

Trip Zone CBC Clear Register

Figure 20-144. TZCBCCLR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
DCBEVT2	DCAEVT2	CBC6	CBC5	CBC4	CBC3	CBC2	CBC1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 20-79. TZCBCCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	DCBEVT2	R-0/W1S	0h	Clear Flag for Digital Compare Output B Event 2 selected for CBC 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZCBCFLG[DCBEVT2] bit. Reset type: SYSRSn
6	DCAEVT2	R-0/W1S	0h	Clear Flag for Digital Compare Output A Event 2 selected for CBC 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZCBCFLG[DCAEVT2] bit. Reset type: SYSRSn
5	CBC6	R-0/W1S	0h	Clear Flag for Cycle-By-Cycle (CBC6) Trip Latch 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZCBCFLG[CBC6] bit. Reset type: SYSRSn
4	CBC5	R-0/W1S	0h	Clear Flag for Cycle-By-Cycle (CBC5) Trip Latch 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZCBCFLG[CBC5] bit. Reset type: SYSRSn
3	CBC4	R-0/W1S	0h	Clear Flag for Cycle-By-Cycle (CBC4) Trip Latch 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZCBCFLG[CBC4] bit. Reset type: SYSRSn
2	CBC3	R-0/W1S	0h	Clear Flag for Cycle-By-Cycle (CBC3) Trip Latch 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZCBCFLG[CBC3] bit. Reset type: SYSRSn
1	CBC2	R-0/W1S	0h	Clear Flag for Cycle-By-Cycle (CBC2) Trip Latch 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZCBCFLG[CBC2] bit. Reset type: SYSRSn
0	CBC1	R-0/W1S	0h	Clear Flag for Cycle-By-Cycle (CBC1) Trip Latch 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZCBCFLG[CBC1] bit. Reset type: SYSRSn

20.17.2.57 TZOSTCLR Register (Offset = 99h) [Reset = 0h]

TZOSTCLR is shown in Figure 20-145 and described in Table 20-80.

Return to the Summary Table.

Trip Zone OST Clear Register

Figure 20-145. TZOSTCLR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
DCBEVT1	DCAEVT1	OST6	OST5	OST4	OST3	OST2	OST1
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 20-80. TZOSTCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	DCBEVT1	R-0/W1S	0h	Clear Flag for Digital Compare Output B Event 1 selected for OST 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZOSTFLG[DCBEVT1] bit. Reset type: SYSRSn
6	DCAEVT1	R-0/W1S	0h	Clear Flag for Digital Compare Output A Event 1 selected for OST 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZOSTFLG[DCAEVT1] bit. Reset type: SYSRSn
5	OST6	R-0/W1S	0h	Clear Flag for Oneshot (OST6) Trip Latch 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZOSTFLG[OST6] bit. Reset type: SYSRSn
4	OST5	R-0/W1S	0h	Clear Flag for Oneshot (OST5) Trip Latch 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZOSTFLG[OST5] bit. Reset type: SYSRSn
3	OST4	R-0/W1S	0h	Clear Flag for Oneshot (OST4) Trip Latch 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZOSTFLG[OST4] bit. Reset type: SYSRSn
2	OST3	R-0/W1S	0h	Clear Flag for Oneshot (OST3) Trip Latch 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZOSTFLG[OST3] bit. Reset type: SYSRSn
1	OST2	R-0/W1S	0h	Clear Flag for Oneshot (OST2) Trip Latch 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZOSTFLG[OST2] bit. Reset type: SYSRSn
0	OST1	R-0/W1S	0h	Clear Flag for Oneshot (OST1) Trip Latch 0: Writing a 0 has no effect. 1: Writing a 1 will clear the TZOSTFLG[OST1] bit. Reset type: SYSRSn

20.17.2.58 TZFRC Register (Offset = 9Bh) [Reset = 0h]

TZFRC is shown in Figure 20-146 and described in Table 20-81.

Return to the Summary Table.

Trip Zone Force Register

Figure 20-146. TZFRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	DCBEVT2	DCBEVT1	DCAEVT2	DCAEVT1	OST	CBC	RESERVED
R-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0h

Table 20-81. TZFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R	0h	Reserved
6	DCBEVT2	R-0/W1S	0h	Force Flag for Digital Compare Output B Event 2 0: Writing 0 has no effect. This bit always reads back 0. 1: Writing 1 forces the DCBEVT2 event trip condition and sets the TZFLG[DCBEVT2] bit. Reset type: SYSRSn
5	DCBEVT1	R-0/W1S	0h	Force Flag for Digital Compare Output B Event 1 0: Writing 0 has no effect. This bit always reads back 0. 1: Writing 1 forces the DCBEVT1 event trip condition and sets the TZFLG[DCBEVT1] bit. Reset type: SYSRSn
4	DCAEVT2	R-0/W1S	0h	Force Flag for Digital Compare Output A Event 2 0: Writing 0 has no effect. This bit always reads back 0. 1: Writing 1 forces the DCAEVT2 event trip condition and sets the TZFLG[DCAEVT2] bit. Reset type: SYSRSn
3	DCAEVT1	R-0/W1S	0h	Force Flag for Digital Compare Output A Event 1 0: Writing 0 has no effect. This bit always reads back 0 1: Writing 1 forces the DCAEVT1 event trip condition and sets the TZFLG[DCAEVT1] bit. Reset type: SYSRSn
2	OST	R-0/W1S	0h	Force a One-Shot Trip Event via Software 0: Writing of 0 is ignored. Always reads back a 0. 1: Forces a one-shot trip event and sets the TZFLG[OST] bit. Reset type: SYSRSn
1	CBC	R-0/W1S	0h	Force a Cycle-by-Cycle Trip Event via Software 0: Writing of 0 is ignored. Always reads back a 0. 1: Forces a cycle-by-cycle trip event and sets the TZFLG[CBC] bit. Reset type: SYSRSn
0	RESERVED	R	0h	Reserved

20.17.2.59 ETSEL Register (Offset = A4h) [Reset = 0h]

ETSEL is shown in Figure 20-147 and described in Table 20-82.

Return to the Summary Table.

Event Trigger Selection Register

Figure 20-147. ETSEL Register

15	14	13	12	11	10	9	8
SOCBEN	SOCBSEL			SOCAEN	SOCASEL		
R/W-0h	R/W-0h			R/W-0h	R/W-0h		
7	6	5	4	3	2	1	0
RESERVED	INTSELCMP	SOCBSELCMP	SOCASELCMP	INTEN	INTSEL		
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h		

Table 20-82. ETSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	SOCBEN	R/W	0h	Enable the ADC Start of Conversion B (EPWMxSOCB) Pulse 0: Disable EPWMxSOCB. 1: Enable EPWMxSOCB pulse. Reset type: SYSRSn
14-12	SOCBSEL	R/W	0h	EPWMxSOCB Selection Options These bits determine when a EPWMxSOCB pulse will be generated. 000: Enable DCBEVT1.soc event 001: Enable event time-base counter equal to zero. (TBCTR = 0x00) 010: Enable event time-base counter equal to period (TBCTR = TBPRD) 011: Enable event time-base counter equal to zero or period (TBCTR = 0x00 or TBCTR = TBPRD). This mode is useful in up-down count mode. 100: Enable event time-base counter equal to CMPA when the timer is incrementing or CMPC when the timer is incrementing 101: Enable event time-base counter equal to CMPA when the timer is decrementing or CMPC when the timer is decrementing 110: Enable event: time-base counter equal to CMPB when the timer is incrementing or CMPD when the timer is incrementing 111: Enable event: time-base counter equal to CMPB when the timer is decrementing or CMPD when the timer is decrementing (*) Event selected is determined by SOCBSELCMP bit. Reset type: SYSRSn
11	SOCAEN	R/W	0h	Enable the ADC Start of Conversion A (EPWMxSOCA) Pulse 0: Disable EPWMxSOCA. 1: Enable EPWMxSOCA pulse. Reset type: SYSRSn

Table 20-82. ETSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10-8	SOCASEL	R/W	0h	EPWMxSOCA Selection Options These bits determine when a EPWMxSOCA pulse will be generated. 000: Enable DCAEVT1.soc event 001: Enable event time-base counter equal to zero. (TBCTR = 0x00) 010: Enable event time-base counter equal to period (TBCTR = TBPRD) 011: Enable event time-base counter equal to zero or period (TBCTR = 0x00 or TBCTR = TBPRD). This mode is useful in up-down count mode. 100: Enable event time-base counter equal to CMPA when the timer is incrementing or CMPC when the timer is incrementing 101: Enable event time-base counter equal to CMPA when the timer is decrementing or CMPC when the timer is decrementing 110: Enable event: time-base counter equal to CMPB when the timer is incrementing or CMPD when the timer is incrementing 111: Enable event: time-base counter equal to CMPB when the timer is decrementing or CMPD when the timer is decrementing (*) Event selected is determined by SOCASELCMP bit. Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6	INTSELCMP	R/W	0h	EPWMxINT Compare Register Selection Options 0: Enable event time-base counter equal to CMPA when the timer is incrementing / Enable event time-base counter equal to CMPA when the timer is decrementing / Enable event: time-base counter equal to CMPB when the timer is incrementing / Enable event: time-base counter equal to CMPB when the timer is decrementing to INTSEL selection mux. 1: Enable event time-base counter equal to CMPC when the timer is incrementing / Enable event time-base counter equal to CMPC when the timer is decrementing / Enable event: time-base counter equal to CMPD when the timer is incrementing / Enable event: time-base counter equal to CMPD when the timer is decrementing to INTSEL selection mux. Reset type: SYSRSn
5	SOCBSELCMP	R/W	0h	EPWMxSOCB Compare Register Selection Options 0: Enable event time-base counter equal to CMPA when the timer is incrementing / Enable event time-base counter equal to CMPA when the timer is decrementing / Enable event: time-base counter equal to CMPB when the timer is incrementing / Enable event: time-base counter equal to CMPB when the timer is decrementing to SOCBSEL selection mux. 1: Enable event time-base counter equal to CMPC when the timer is incrementing / Enable event time-base counter equal to CMPC when the timer is decrementing / Enable event: time-base counter equal to CMPD when the timer is incrementing / Enable event: time-base counter equal to CMPD when the timer is decrementing to SOCBSEL selection mux. Reset type: SYSRSn
4	SOCASELCMP	R/W	0h	EPWMxSOCA Compare Register Selection Options 0: Enable event time-base counter equal to CMPA when the timer is incrementing / Enable event time-base counter equal to CMPA when the timer is decrementing / Enable event: time-base counter equal to CMPB when the timer is incrementing / Enable event: time-base counter equal to CMPB when the timer is decrementing to SOCASEL selection mux. 1: Enable event time-base counter equal to CMPC when the timer is incrementing / Enable event time-base counter equal to CMPC when the timer is decrementing / Enable event: time-base counter equal to CMPD when the timer is incrementing / Enable event: time-base counter equal to CMPD when the timer is decrementing to SOCASEL selection mux. Reset type: SYSRSn

Table 20-82. ETSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	INTEN	R/W	0h	Enable ePWM Interrupt (EPWMx_INT) Generation 0: Disable EPWMx_INT generation 1: Enable EPWMx_INT generation Reset type: SYSRSn
2-0	INTSEL	R/W	0h	ePWM Interrupt (EPWMx_INT) Selection Options 000: Reserved 001: Enable event time-base counter equal to zero. (TBCTR = 0x00) 010: Enable event time-base counter equal to period (TBCTR = TBPRD) 011: Enable event time-base counter equal to zero or period (TBCTR = 0x00 or TBCTR = TBPRD). This mode is useful in up-down count mode. 100: Enable event time-base counter equal to CMPA when the timer is incrementing or CMPC when the timer is incrementing 101: Enable event time-base counter equal to CMPA when the timer is decrementing or CMPC when the timer is decrementing 110: Enable event: time-base counter equal to CMPB when the timer is incrementing or CMPD when the timer is incrementing 111: Enable event: time-base counter equal to CMPB when the timer is decrementing or CMPD when the timer is decrementing (*) Event selected is determined by INTSELCMP bit. Reset type: SYSRSn

20.17.2.60 ETPS Register (Offset = A6h) [Reset = 0h]

ETPS is shown in Figure 20-148 and described in Table 20-83.

Return to the Summary Table.

Event Trigger Pre-Scale Register

Figure 20-148. ETPS Register

15	14	13	12	11	10	9	8
SOCBCNT		SOCBPRD		SOCACNT		SOCAPRD	
R-0h		R/W-0h		R-0h		R/W-0h	
7	6	5	4	3	2	1	0
RESERVED		SOCPSSEL	INTPSSEL	INTCNT		INTPRD	
R-0h		R/W-0h	R/W-0h	R-0h		R/W-0h	

Table 20-83. ETPS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	SOCBCNT	R	0h	ePWM ADC Start-of-Conversion B Event (EPWMxSOCB) Counter Register These bits indicate how many selected ETSEL[SOCBSEL] events have occurred: 00: No events have occurred. 01: 1 event has occurred. 10: 2 events have occurred. 11: 3 events have occurred. Reset type: SYSRSn
13-12	SOCBPRD	R/W	0h	ePWM ADC Start-of-Conversion B Event (EPWMxSOCB) Period Select These bits determine how many selected ETSEL[SOCBSEL] events need to occur before an EPWMxSOCB pulse is generated. To be generated, the pulse must be enabled (ETSEL[SOCBEN] = 1). The SOCB pulse will be generated even if the status flag is set from a previous start of conversion (ETFLG[SOCB] = 1). Once the SOCB pulse is generated, the ETPS[SOCBCNT] bits will automatically be cleared. 00: Disable the SOCB event counter. No EPWMxSOCB pulse will be generated 01: Generate the EPWMxSOCB pulse on the first event: ETPS[SOCBCNT] = 0,1 10: Generate the EPWMxSOCB pulse on the second event: ETPS[SOCBCNT] = 1,0 11: Generate the EPWMxSOCB pulse on the third event: ETPS[SOCBCNT] = 1,1 Reset type: SYSRSn
11-10	SOCACNT	R	0h	ePWM ADC Start-of-Conversion A Event (EPWMxSOCA) Counter Register These bits indicate how many selected ETSEL[SOCASEL] events have occurred: 00: No events have occurred. 01: 1 event has occurred. 10: 2 events have occurred. 11: 3 events have occurred. Reset type: SYSRSn

Table 20-83. ETPS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9-8	SOCAPRD	R/W	0h	ePWM ADC Start-of-Conversion A Event (EPWMxSOCA) Period Select These bits determine how many selected ETSEL[SOCASEL] events need to occur before an EPWMxSOCA pulse is generated. To be generated, the pulse must be enabled (ETSEL[SOCAEN] = 1). The SOCA pulse will be generated even if the status flag is set from a previous start of conversion (ETFLG[SOCA] = 1). Once the SOCA pulse is generated, the ETPS[SOCACNT] bits will automatically be cleared. 00: Disable the SOCA event counter. No EPWMxSOCA pulse will be generated 01: Generate the EPWMxSOCA pulse on the first event: ETPS[SOCACNT] = 0,1 10: Generate the EPWMxSOCA pulse on the second event: ETPS[SOCACNT] = 1,0 11: Generate the EPWMxSOCA pulse on the third event: ETPS[SOCACNT] = 1,1 Reset type: SYSRSn
7-6	RESERVED	R	0h	Reserved
5	SOCPSSEL	R/W	0h	EPWMxSOC A/B Pre-Scale Selection Bits 0: Selects ETPS [SOCACNT/SOCBCNT] and [SOCAPRD/ SOCBPRD] registers to determine frequency of events (interrupt once every 0-3 events). 1: Selects ETSOCPS [SOCACNT2/SOCBCNT2] and [SOCAPRD2/ SOCBPRD2] registers to determine frequency of events (interrupt once every 0-15 events). Reset type: SYSRSn
4	INTPSSEL	R/W	0h	EPWMxINTn Pre-Scale Selection Bits 0: Selects ETPS [INTCNT, and INTPRD] registers to determine frequency of events (interrupt once every 0-3 events). 1: Selects ETINTPS [INTCNT2, and INTPRD2] registers to determine frequency of events (interrupt once every 0-15 events). Reset type: SYSRSn
3-2	INTCNT	R	0h	ePWM Interrupt Event (EPWMx_INT) Counter Register These bits indicate how many selected ETSEL[INTSEL] events have occurred. These bits are automatically cleared when an interrupt pulse is generated. If interrupts are disabled, ETSEL[INT] = 0 or the interrupt flag is set, ETFLG[INT] = 1, the counter will stop counting events when it reaches the period value ETPS[INTCNT] = ETPS[INTPRD]. 00: No events have occurred. 01: 1 event has occurred. 10: 2 events have occurred. 11: 3 events have occurred. Reset type: SYSRSn

Table 20-83. ETPS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1-0	INTPRD	R/W	0h	ePWM Interrupt (EPWMx_INT) Period Select These bits determine how many selected ETSEL[INTSEL] events need to occur before an interrupt is generated. To be generated, the interrupt must be enabled (ETSEL[INT] = 1). If the interrupt status flag is set from a previous interrupt (ETFLG[INT] = 1) then no interrupt will be generated until the flag is cleared via the ETCLR[INT] bit. This allows for one interrupt to be pending while another is still being serviced. Once the interrupt is generated, the ETPS[INTCNT] bits will automatically be cleared. Writing a INTPRD value that is the same as the current counter value will trigger an interrupt if it is enabled and the status flag is clear. Writing a INTPRD value that is less than the current counter value will result in an undefined state. If a counter event occurs at the same instant as a new zero or non-zero INTPRD value is written, the counter is incremented. 00: Disable the interrupt event counter. No interrupt will be generated and ETFRC[INT] is ignored. 01: Generate an interrupt on the first event INTCNT = 01 (first event) 10: Generate interrupt on ETPS[INTCNT] = 1,0 (second event) 11: Generate interrupt on ETPS[INTCNT] = 1,1 (third event) Reset type: SYSRSn

20.17.2.61 ETFLG Register (Offset = A8h) [Reset = 0h]

ETFLG is shown in Figure 20-149 and described in Table 20-84.

Return to the Summary Table.

Event Trigger Flag Register

Figure 20-149. ETFLG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				SOCB	SOCA	RESERVED	INT
R-0h				R-0h	R-0h	R-0h	R-0h

Table 20-84. ETFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	SOCB	R	0h	Latched ePWM ADC Start-of-Conversion A (EPWMxSOCB) Status Flag Unlike the ETFLG[INT] flag, the EPWMxSOCB output will continue to pulse even if the flag bit is set. 0: Indicates no event occurred 1: Indicates that a start of conversion pulse was generated on EPWMxSOCB. The EPWMxSOCB output will continue to be generated even if the flag bit is set. Reset type: SYSRSn
2	SOCA	R	0h	Latched ePWM ADC Start-of-Conversion A (EPWMxSOCA) Status Flag Unlike the ETFLG[INT] flag, the EPWMxSOCA output will continue to pulse even if the flag bit is set. 0: Indicates no event occurred 1: Indicates that a start of conversion pulse was generated on EPWMxSOCA. The EPWMxSOCA output will continue to be generated even if the flag bit is set. Reset type: SYSRSn
1	RESERVED	R	0h	Reserved
0	INT	R	0h	Latched ePWM Interrupt (EPWMx_INT) Status Flag 0: Indicates no event occurred 1: Indicates that an ePWMx interrupt (EPWMx_INT) was generated. No further interrupts will be generated until the flag bit is cleared. Up to one interrupt can be pending while the ETFLG[INT] bit is still set. If an interrupt is pending, it will not be generated until after the ETFLG[INT] bit is cleared. Reset type: SYSRSn

20.17.2.62 ETCLR Register (Offset = AAh) [Reset = 0h]

ETCLR is shown in Figure 20-150 and described in Table 20-85.

Return to the Summary Table.

Event Trigger Clear Register

Figure 20-150. ETCLR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				SOCB	SOCA	RESERVED	INT
R-0h				R-0/W1S-0h	R-0/W1S-0h	R-0h	R-0/W1S-0h

Table 20-85. ETCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	SOCB	R-0/W1S	0h	ePWM ADC Start-of-Conversion A (EPWMxSOCB) Flag Clear Bit 0: Writing a 0 has no effect. Always reads back a 0 1: Clears the ETFLG[SOCB] flag bit Reset type: SYSRSn
2	SOCA	R-0/W1S	0h	ePWM ADC Start-of-Conversion A (EPWMxSOCA) Flag Clear Bit 0: Writing a 0 has no effect. Always reads back a 0 1: Clears the ETFLG[SOCA] flag bit Reset type: SYSRSn
1	RESERVED	R	0h	Reserved
0	INT	R-0/W1S	0h	ePWM Interrupt (EPWMx_INT) Flag Clear Bit 0: Writing a 0 has no effect. Always reads back a 0 1: Clears the ETFLG[INT] flag bit and enable further interrupts pulses to be generated Reset type: SYSRSn

20.17.2.63 ETFRC Register (Offset = ACh) [Reset = 0h]

ETFRC is shown in Figure 20-151 and described in Table 20-86.

Return to the Summary Table.

Event Trigger Force Register

Figure 20-151. ETFRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				SOCB	SOCA	RESERVED	INT
R-0h				R-0/W1S-0h	R-0/W1S-0h	R-0h	R-0/W1S-0h

Table 20-86. ETFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	SOCB	R-0/W1S	0h	SOCB Force Bit The SOCB pulse will only be generated if the event is enabled in the ETSEL register. The ETFLG[SOCB] flag bit will be set regardless. 0: Writing 0 to this bit will be ignored. Always reads back a 0. 1: Generates a pulse on EPWMxSOCB and set the SOCBFLG bit. This bit is used for test purposes. Reset type: SYSRSn
2	SOCA	R-0/W1S	0h	SOCA Force Bit The SOCA pulse will only be generated if the event is enabled in the ETSEL register. The ETFLG[SOCA] flag bit will be set regardless. 0: Writing 0 to this bit will be ignored. Always reads back a 0. 1: Generates a pulse on EPWMxSOCA and set the SOCAFLG bit. This bit is used for test purposes. Reset type: SYSRSn
1	RESERVED	R	0h	Reserved
0	INT	R-0/W1S	0h	INT Force Bit The interrupt will only be generated if the event is enabled in the ETSEL register. The INT flag bit will be set regardless. 0: Writing 0 to this bit will be ignored. Always reads back a 0. 1: Generates an interrupt on EPWMxINT and set the INT flag bit. This bit is used for test purposes. Reset type: SYSRSn

20.17.2.64 ETINTPS Register (Offset = AEh) [Reset = 0h]

ETINTPS is shown in Figure 20-152 and described in Table 20-87.

Return to the Summary Table.

Event-Trigger Interrupt Pre-Scale Register

Figure 20-152. ETINTPS Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
INTCNT2				INTPRD2			
R-0h				R/W-0h			

Table 20-87. ETINTPS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-4	INTCNT2	R	0h	EPWMxINT Counter 2 When ETPS[INTPSSEL]=1, these bits indicate how many selected events have occurred: 0000: No events 0001: 1 event 0010: 2 events 0011: 3 events 0100: 4 events ... 1111: 15 events Reset type: SYSRSn
3-0	INTPRD2	R/W	0h	EPWMxINT Period 2 Select When ETPS[INTPSSEL] = 1, these bits select how many selected events need to occur before an interrupt is generated: 0000: Disable counter 0001: Generate interrupt on INTCNT = 1 (first event) 0010: Generate interrupt on INTCNT = 2 (second event) 0011: Generate interrupt on INTCNT = 3 (third event) 0100: Generate interrupt on INTCNT = 4 (fourth event) ... 1111: Generate interrupt on INTCNT = 15 (fifteenth event) Reset type: SYSRSn

20.17.2.65 ETSOCPS Register (Offset = B0h) [Reset = 0h]

ETSOCPS is shown in Figure 20-153 and described in Table 20-88.

Return to the Summary Table.

Event-Trigger SOC Pre-Scale Register

Figure 20-153. ETSOCPS Register

15	14	13	12	11	10	9	8
SOCBCNT2				SOCBPRD2			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
SOCACNT2				SOCAPRD2			
R-0h				R/W-0h			

Table 20-88. ETSOCPS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	SOCBCNT2	R	0h	EPWMxSOCB Counter 2 When ETPS[SOCPSSEL] = 1, these bits indicate how many selected events have occurred: 0000: No events 0001: 1 event 0010: 2 events 0011: 3 events 0100: 4 events ... 1111: 15 events Reset type: SYSRSn
11-8	SOCBPRD2	R/W	0h	EPWMxSOCB Period 2 Select When ETPS[SOCPSSEL] = 1, these bits select how many selected event need to occur before an SOCB pulse is generated: 0000: Disable counter 0001: Generate interrupt on SOCBCNT2 = 1 (first event) 0010: Generate interrupt on SOCBCNT2 = 2 (second event) 0011: Generate interrupt on SOCBCNT2 = 3 (third event) 0100: Generate interrupt on SOCBCNT2 = 4 (fourth event) ... 1111: Generate interrupt on SOCBCNT2 = 15 (fifteenth event) Reset type: SYSRSn
7-4	SOCACNT2	R	0h	EPWMxSOCA Counter 2 When ETPS[SOCPSSEL] = 1, these bits indicate how many selected events have occurred: 0000: No events 0001: 1 event 0010: 2 events 0011: 3 events 0100: 4 events ... 1111: 15 events Reset type: SYSRSn
3-0	SOCAPRD2	R/W	0h	EPWMxSOCA Period 2 Select When ETPS[SOCPSSEL] = 1, these bits select how many selected event need to occur before an SOCA pulse is generated: 0000: Disable counter 0001: Generate interrupt on SOCACNT2 = 1 (first event) 0010: Generate interrupt on SOCACNT2 = 2 (second event) 0011: Generate interrupt on SOCACNT2 = 3 (third event) 0100: Generate interrupt on SOCACNT2 = 4 (fourth event) ... 1111: Generate interrupt on SOCACNT2 = 15 (fifteenth event) Reset type: SYSRSn

20.17.2.66 ETCNTINITCTL Register (Offset = B2h) [Reset = 0h]

ETCNTINITCTL is shown in Figure 20-154 and described in Table 20-89.

Return to the Summary Table.

Event-Trigger Counter Initialization Control Register

Figure 20-154. ETCNTINITCTL Register

15	14	13	12	11	10	9	8
SOCBINITEN	SOCAINITEN	INTINITEN	SOCBINITFRC	SOCAINITFRC	INTINITFRC	RESERVED	
R/W-0h	R/W-0h	R/W-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0h	
7	6	5	4	3	2	1	0
RESERVED							
R-0h							

Table 20-89. ETCNTINITCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	SOCBINITEN	R/W	0h	EPWMxSOCB Counter 2 Initialization Enable 0: Has no effect. 1: Enable initialization of EPWMxSOCB counter with contents of ETCNTINIT[SOCBINIT] on a SYNC event or software force. Reset type: SYSRSn
14	SOCAINITEN	R/W	0h	EPWMxSOCA Counter 2 Initialization Enable 0: Has no effect. 1: Enable initialization of EPWMxSOCA counter with contents of ETCNTINIT[SOCAINIT] on a SYNC event or software force. Reset type: SYSRSn
13	INTINITEN	R/W	0h	EPWMxINT Counter 2 Initialization Enable 0: Has no effect. 1: Enable initialization of EPWMxINT counter 2 with contents of ETCNTINIT[INTINIT] on a SYNC event or software force. Reset type: SYSRSn
12	SOCBINITFRC	R-0/W1S	0h	EPWMxSOCB Counter 2 Initialization Force 0: Has no effect. 1: This bit forces the ET EPWMxSOCB counter to be initialized with the contents of ETCNTINIT[SOCBINIT]. Reset type: SYSRSn
11	SOCAINITFRC	R-0/W1S	0h	EPWMxSOCA Counter 2 Initialization Force 0: Has no effect. 1: This bit forces the ET EPWMxSOCA counter to be initialized with the contents of ETCNTINIT[SOCAINIT]. Reset type: SYSRSn
10	INTINITFRC	R-0/W1S	0h	EPWMxINT Counter 2 Initialization Force 0: Has no effect. 1: This bit forces the ET EPWMxINT counter to be initialized with the contents of ETCNTINIT[INTINIT]. Reset type: SYSRSn
9-0	RESERVED	R	0h	Reserved

20.17.2.67 ETCNTINIT Register (Offset = B4h) [Reset = 0h]

ETCNTINIT is shown in Figure 20-155 and described in Table 20-90.

Return to the Summary Table.

Event-Trigger Counter Initialization Register

Figure 20-155. ETCNTINIT Register

15	14	13	12	11	10	9	8
RESERVED				SOCBINIT			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
SOCAINIT				INTINIT			
R/W-0h				R/W-0h			

Table 20-90. ETCNTINIT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-8	SOCBINIT	R/W	0h	EPWMxSOCB Counter 2 Initialization Bits The ET EPWMxSOCB counter is initialized with the contents of this register on an ePWM SYNC event or a software force. Reset type: SYSRSn
7-4	SOCAINIT	R/W	0h	EPWMxSOCA Counter 2 Initialization Bits The ET EPWMxSOCA counter is initialized with the contents of this register on an ePWM SYNC event or a software force. Reset type: SYSRSn
3-0	INTINIT	R/W	0h	EPWMxINT Counter 2 Initialization Bits The ET EPWMxINT counter is initialized with the contents of this register on an ePWM SYNC event or a software force. Reset type: SYSRSn

20.17.2.68 DCTRIPSEL Register (Offset = C0h) [Reset = 0h]

DCTRIPSEL is shown in Figure 20-156 and described in Table 20-91.

Return to the Summary Table.

Digital Compare Trip Select Register

Figure 20-156. DCTRIPSEL Register

15	14	13	12	11	10	9	8
DCBLCOMPSEL				DCBHCOMPSEL			
R/W-0h				R/W-0h			
7	6	5	4	3	2	1	0
DCALCOMPSEL				DCAHCOMPSEL			
R/W-0h				R/W-0h			

Table 20-91. DCTRIPSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	DCBLCOMPSEL	R/W	0h	Digital Compare B Low Input Select Bits 0000: TRIPIN1 0001: TRIPIN2 0010: TRIPIN3 0011: TRIPIN4 ... 1011: TRIPIN12 1100: Reserved 1101: TRIPIN14 1110: TRIPIN15 1111: Trip combination input (all trip inputs selected by DCBLTRIPSEL register ORed together) Reset type: SYSRSn
11-8	DCBHCOMPSEL	R/W	0h	Digital Compare B High Input Select Bits 0000: TRIPIN1 0001: TRIPIN2 0010: TRIPIN3 0011: TRIPIN4 ... 1011: TRIPIN12 1100: Reserved 1101: TRIPIN14 1110: TRIPIN15 1111: Trip combination input (all trip inputs selected by DCBHTRIPSEL register ORed together) Reset type: SYSRSn
7-4	DCALCOMPSEL	R/W	0h	Digital Compare A Low Input Select Bits 0000: TRIPIN1 0001: TRIPIN2 0010: TRIPIN3 0011: TRIPIN4 ... 1011: TRIPIN12 1100: Reserved 1101: TRIPIN14 1110: TRIPIN15 1111: Trip combination input (all trip inputs selected by DCALTRIPSEL register ORed together) Reset type: SYSRSn

Table 20-91. DCTRIPSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	DCAHCOMPSEL	R/W	0h	Digital Compare A High Input Select Bits 0000: TRIPIN1 0001: TRIPIN2 0010: TRIPIN3 0011: TRIPIN4 ... 1011: TRIPIN12 1100: Reserved 1101: TRIPIN14 1110: TRIPIN15 1111: Trip combination input (all trip inputs selected by DCAHTRIPSEL register ORed together) Reset type: SYSRSn

20.17.2.69 DCACTL Register (Offset = C3h) [Reset = 0h]

DCACTL is shown in Figure 20-157 and described in Table 20-92.

Return to the Summary Table.

Digital Compare A Control Register

Figure 20-157. DCACTL Register

15	14	13	12	11	10	9	8
EVT2LAT	EVT2LATCLRSEL		EVT2LATSEL	RESERVED		EVT2FRCSYNCSEL	EVT2SRCSEL
R-0h	R/W-0h		R/W-0h	R-0h		R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
EVT1LAT	EVT1LATCLRSEL		EVT1LATSEL	EVT1SYNCE	EVT1SOCE	EVT1FRCSYNCSEL	EVT1SRCSEL
R-0h	R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 20-92. DCACTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	EVT2LAT	R	0h	Indicates the status of DCAEVT2LAT signal. 0 The DCAEVT2LAT latch is cleared. 1 The DCAEVT2LAT latch is set. Reset type: SYSRSn
14-13	EVT2LATCLRSEL	R/W	0h	DCAEVT2 Latched clear source select: 00 CNT_ZERO event clears DCAEVT2 latch. 01 PRD_EQ event clears DCAEVT2 latch. 10 CNT_ZERO event or PRD_EQ event clears DCAEVT2 latch. 11 Reserved. Reset type: SYSRSn
12	EVT2LATSEL	R/W	0h	DCAEVT2 Latched signal select: 0 Does not select the DCAEVT2 latched signal (Refer figure "Modifications to DCAEVT1.force/DCAEVT2.force generation.") as source of DCAEVT2.force. 1 Selects the DCAEVT2 latched signal as source of DCAEVT2.force. Reset type: SYSRSn
11-10	RESERVED	R	0h	Reserved
9	EVT2FRCSYNCSEL	R/W	0h	DCAEVT2 Force Synchronization Signal Select 0: Source is synchronized with EPWMCLK 1: Source is passed through asynchronously Reset type: SYSRSn
8	EVT2SRCSEL	R/W	0h	DCAEVT2 Source Signal Select 0: Source Is DCAEVT2 Signal 1: Source Is DCEVTFILT Signal Reset type: SYSRSn
7	EVT1LAT	R	0h	Indicates the status of DCAEVT1LAT signal. 0 The DCAEVT1LAT latch is cleared. 1 The DCAEVT1LAT latch is set. Reset type: SYSRSn
6-5	EVT1LATCLRSEL	R/W	0h	DCAEVT1 Latched clear source select: 00 CNT_ZERO event clears DCAEVT1 latch. 01 PRD_EQ event clears DCAEVT1 latch. 10 CNT_ZERO event or PRD_EQ event clears DCAEVT1 latch. 11 Reserved. Reset type: SYSRSn
4	EVT1LATSEL	R/W	0h	DCAEVT1 Latched signal select: 0 Does not select the DCAEVT1 latched signal (Refer figure "Modifications to DCAEVT1.force/DCAEVT2.force generation.") as source of DCAEVT1.force. 1 Selects the DCAEVT1 latched signal as source of DCAEVT1.force. Reset type: SYSRSn

Table 20-92. DCACTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	EVT1SYNCE	R/W	0h	DCAEVT1 SYNC, Enable/Disable 0: SYNC Generation Disabled 1: SYNC Generation Enabled Reset type: SYSRSn
2	EVT1SOCE	R/W	0h	DCAEVT1 SOC, Enable/Disable 0: SOC Generation Disabled 1: SOC Generation Enabled Reset type: SYSRSn
1	EVT1FRCSYNCSEL	R/W	0h	DCAEVT1 Force Synchronization Signal Select 0: Source is synchronized with EPWMCLK 1: Source is passed through asynchronously Reset type: SYSRSn
0	EVT1SRCSEL	R/W	0h	DCAEVT1 Source Signal Select 0: Source Is DCAEVT1 Signal 1: Source Is DCEVTFILT Signal Reset type: SYSRSn

20.17.2.70 DCBCTL Register (Offset = C4h) [Reset = 0h]

DCBCTL is shown in Figure 20-158 and described in Table 20-93.

Return to the Summary Table.

Digital Compare B Control Register

Figure 20-158. DCBCTL Register

15	14	13	12	11	10	9	8
EVT2LAT	EVT2LATCLRSEL		EVT2LATSEL	RESERVED		EVT2FRCSYNCSEL	EVT2SRCSEL
R-0h	R/W-0h		R/W-0h	R-0h		R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
EVT1LAT	EVT1LATCLRSEL		EVT1LATSEL	EVT1SYNCE	EVT1SOCE	EVT1FRCSYNCSEL	EVT1SRCSEL
R-0h	R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 20-93. DCBCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	EVT2LAT	R	0h	Indicates the status of DCBEVT2LAT signal. 0 The DCBEVT2LAT latch is cleared. 1 The DCBEVT2LAT latch is set. Reset type: SYSRSn
14-13	EVT2LATCLRSEL	R/W	0h	DCBEVT2 Latched clear source select: 00 CNT_ZERO event clears DCBEVT2 latch. 01 PRD_EQ event clears DCBEVT2 latch. 10 CNT_ZERO event or PRD_EQ event clears DCBEVT2 latch. 11 Reserved. Reset type: SYSRSn
12	EVT2LATSEL	R/W	0h	DCBEVT2 Latched signal select: 0 Does not select the DCBEVT2 latched signal (Refer figure "Modifications to DCBEVT1.force/DCBEVT2.force generation.") as source of DCBEVT2.force. 1 Selects the DCBEVT2 latched signal as source of DCBEVT2.force. Reset type: SYSRSn
11-10	RESERVED	R	0h	Reserved
9	EVT2FRCSYNCSEL	R/W	0h	DCBEVT2 Force Synchronization Signal Select 0: Source is synchronized with EPWMCLK 1: Source is passed through asynchronously Reset type: SYSRSn
8	EVT2SRCSEL	R/W	0h	DCBEVT2 Source Signal Select 0: Source Is DCBEVT2 Signal 1: Source Is DCEVTFILT Signal Reset type: SYSRSn
7	EVT1LAT	R	0h	Indicates the status of DCBEVT1LAT signal. 0 The DCBEVT1LAT latch is cleared. 1 The DCBEVT1LAT latch is set. Reset type: SYSRSn
6-5	EVT1LATCLRSEL	R/W	0h	DCBEVT1 Latched clear source select: 00 CNT_ZERO event clears DCBEVT1 latch. 01 PRD_EQ event clears DCBEVT1 latch. 10 CNT_ZERO event or PRD_EQ event clears DCBEVT1 latch. 11 Reserved. Reset type: SYSRSn
4	EVT1LATSEL	R/W	0h	DCBEVT1 Latched signal select: 0 Does not select the DCBEVT1 latched signal (Refer figure "Modifications to DCBEVT1.force/DCBEVT2.force generation.") as source of DCBEVT1.force. 1 Selects the DCBEVT1 latched signal as source of DCBEVT1.force. Reset type: SYSRSn

Table 20-93. DCBCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	EVT1SYNCE	R/W	0h	DCBEVT1 SYNC, Enable/Disable 0: SYNC Generation Disabled 1: SYNC Generation Enabled Reset type: SYSRSn
2	EVT1SOCE	R/W	0h	DCBEVT1 SOC, Enable/Disable 0: SOC Generation Disabled 1: SOC Generation Enabled Reset type: SYSRSn
1	EVT1FRCSYNCSEL	R/W	0h	DCBEVT1 Force Synchronization Signal Select 0: Source is synchronized with EPWMCLK 1: Source is passed through asynchronously Reset type: SYSRSn
0	EVT1SRCSEL	R/W	0h	DCBEVT1 Source Signal Select 0: Source Is DCBEVT1 Signal 1: Source Is DCEVTFILT Signal Reset type: SYSRSn

20.17.2.71 DCFCTL Register (Offset = C7h) [Reset = 0h]

DCFCTL is shown in Figure 20-159 and described in Table 20-94.

Return to the Summary Table.

Digital Compare Filter Control Register

Figure 20-159. DCFCTL Register

15	14	13	12	11	10	9	8
EDGESTATUS			EDGECOUNT			EDGEMODE	
R-0h			R/W-0h			R/W-0h	
7	6	5	4	3	2	1	0
RESERVED	EDGEFILTSEL	PULSESEL		BLANKINV	BLANKE	SRCSEL	
R-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h	R/W-0h	

Table 20-94. DCFCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	EDGESTATUS	R	0h	Edge Status: These bits reflect the total number of edges currently captured. When the value matches the EDGECOUNT, the status bits are set to zero. and a TBCLK wide pulse is generated which can then be output on the DCEVTFILT signal. The edge counter can be reset by writing 000 to the EDGECOUNT value: Reset type: SYSRSn
12-10	EDGECOUNT	R/W	0h	Edge Count: These bits select how many edges to count before generating a TBCLK wide pulse on the DCEVTFILT signal: 000: no edges, reset current EDGESTATUS bits to 0,0,0 001: 1 edge 010: 2 edges 011: 3 edges 100: 4 edges 101: 5 edges 110: 6 edges 111: 7 edges Reset type: SYSRSn
9-8	EDGEMODE	R/W	0h	Edge Mode Select: 00: Low To High Edge 01: High To Low Edge 10: Both Edges 11: Reserved Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6	EDGEFILTSEL	R/W	0h	Edge Filter Select: 0: Edge Filter Not Selected 1: Edge Filter Selected Reset type: SYSRSn
5-4	PULSESEL	R/W	0h	Pulse Select For Blanking & Capture Alignment 00: Time-base counter equal to period (TBCTR = TBPRD) 01: Time-base counter equal to zero (TBCTR = 0x00) 10: Time-base counter equal to zero (TBCTR = 0x00) or period (TBCTR = TBPRD) 11: BLANKPULSEMIX Reset type: SYSRSn
3	BLANKINV	R/W	0h	Blanking Window Inversion 0: Blanking window not inverted 1: Blanking window inverted Reset type: SYSRSn

Table 20-94. DCFCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	BLANKE	R/W	0h	Blanking Window Enable/Disable 0: Blanking window is disabled 1: Blanking window is enabled Reset type: SYSRSn
1-0	SRCSEL	R/W	0h	Filter Block Signal Source Select 00: Source Is DCAEVT1 Signal 01: Source Is DCAEVT2 Signal 10: Source Is DCBEVT1 Signal 11: Source Is DCBEVT2 Signal Reset type: SYSRSn

20.17.2.72 DCCAPCTL Register (Offset = C8h) [Reset = 0h]

DCCAPCTL is shown in Figure 20-160 and described in Table 20-95.

Return to the Summary Table.

Digital Compare Capture Control Register

Figure 20-160. DCCAPCTL Register

15	14	13	12	11	10	9	8
CAPMODE	CAPCLR	CAPSTS	RESERVED				
R/W-0h	R-0/W1S-0h	R-0h	R-0h				
7	6	5	4	3	2	1	0
RESERVED						SHDWMODE	CAPE
R-0h						R/W-0h	R/W-0h

Table 20-95. DCCAPCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	CAPMODE	R/W	0h	Counter Capture Mode 0: When a DCEVTFILT occurs and the counter capture is enabled, then the current TBCNT value is captured in the active register. When the respective trip event occurs, further trip (capture) events are ignored until the next PRD_eq or CNT_zero event (as selected by the PULSESEL bit in the DCFCTL register) re-triggers the capture mechanism. If active mode is enabled, via SHDWMODE bit in DCCAPCTL register, CPU reads of this register will return the active register value. If shadow mode is enabled, via SHDWMODE bit in DCCAPCTL register, the active register is copied to the shadow register on the PRD_eq or CNT_zero event (whichever is selected by PULSESEL bit in DCFCTL register). CPU reads of this register will return the shadow register value. 1: When a DCEVTFILT occurs and the counter capture is enabled, then the current TBCNT value is captured in the active register. When the respective trip event occurs - it will set the CAPSTS flag and further trip (capture) events are ignored until this bit is cleared. CAPSTS can be cleared by writing to CAPCLR bit in DCCAPCTL register and it re-triggers the capture mechanism. If active mode is enabled, via SHDWMODE bit in DCCAPCTL register, CPU reads of this register will return the active register value. If shadow mode is enabled, via SHDWMODE bit in DCCAPCTL register, the active register is copied to the shadow register on the PRD_eq or CNT_zero event (whichever is selected by PULSESEL bit in DCFCTL register). CPU reads of this register will return the shadow register value. Reset type: SYSRSn
14	CAPCLR	R-0/W1S	0h	DC Capture Latched Status Clear Flag 0: Writing a 0 has no effect. 1: Writing a 1 will clear this CAPSTS (set) condition. Reset type: SYSRSn
13	CAPSTS	R	0h	Latched Status Flag for Capture Event 0: No DC capture event occurred. 1: A DC capture event has occurred. Reset type: SYSRSn
12-2	RESERVED	R	0h	Reserved

Table 20-95. DCCAPCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	SHDWMODE	R/W	0h	TBCTR Counter Capture Shadow Select Mode 0: Enable shadow mode. The DCCAP active register is copied to shadow register on a TBCTR = TBPRD or TBCTR = zero event as defined by the DCFCTL[PULSESEL] bit. CPU reads of the DCCAP register will return the shadow register contents. 1: Active Mode. In this mode the shadow register is disabled. CPU reads from the DCCAP register will always return the active register contents. Reset type: SYSRSn
0	CAPE	R/W	0h	TBCTR Counter Capture Enable/Disable 0: Disable the time-base counter capture. 1: Enable the time-base counter capture. Reset type: SYSRSn

20.17.2.73 DCFOFFSET Register (Offset = C9h) [Reset = 0h]

DCFOFFSET is shown in Figure 20-161 and described in Table 20-96.

Return to the Summary Table.

Digital Compare Filter Offset Register

Figure 20-161. DCFOFFSET Register

15	14	13	12	11	10	9	8
DCFOFFSET							
R/W-0h							
7	6	5	4	3	2	1	0
DCFOFFSET							
R/W-0h							

Table 20-96. DCFOFFSET Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	DCFOFFSET	R/W	0h	Blanking Window Offset These 16-bits specify the number of TBCLK cycles from the blanking window reference to the point when the blanking window is applied. The blanking window reference is either period or zero as defined by the DCFCTL[PULSESEL] bit. This offset register is shadowed and the active register is loaded at the reference point defined by DCFCTL[PULSESEL]. The offset counter is also initialized and begins to count down when the active register is loaded. When the counter expires, the blanking window is applied. If the blanking window is currently active, then the blanking window counter is restarted. Reset type: SYSRSn

20.17.2.74 DCFOFFSETCNT Register (Offset = CAh) [Reset = 0h]

DCFOFFSETCNT is shown in Figure 20-162 and described in Table 20-97.

Return to the Summary Table.

Digital Compare Filter Offset Counter Register

Figure 20-162. DCFOFFSETCNT Register

15	14	13	12	11	10	9	8
DCFOFFSETCNT							
R-0h							
7	6	5	4	3	2	1	0
DCFOFFSETCNT							
R-0h							

Table 20-97. DCFOFFSETCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	DCFOFFSETCNT	R	0h	Blanking Offset Counter These 16-bits are read only and indicate the current value of the offset counter. The counter counts down to zero and then stops until it is re-loaded on the next period or zero event as defined by the DCFCTL[PULSESEL] bit. The offset counter is not affected by the free/soft emulation bits. That is, it will always continue to count down if the device is halted by a emulation stop. Reset type: SYSRSn

20.17.2.75 DCFWINDOW Register (Offset = CBh) [Reset = 0h]

DCFWINDOW is shown in Figure 20-163 and described in Table 20-98.

Return to the Summary Table.

Digital Compare Filter Window Register

Figure 20-163. DCFWINDOW Register

15	14	13	12	11	10	9	8
DCFWINDOW							
R/W-0h							
7	6	5	4	3	2	1	0
DCFWINDOW							
R/W-0h							

Table 20-98. DCFWINDOW Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	DCFWINDOW	R/W	0h	Blanking Window Width 00h: No blanking window is generated. 01-FFFFh: Specifies the width of the blanking window in TBCLK cycles. The blanking window begins when the offset counter expires. When this occurs, the window counter is loaded and begins to count down. If the blanking window is currently active and the offset counter expires, the blanking window counter is not restarted and the blanking window is cut short prematurely. Care should be taken to avoid this situation. The blanking window can cross a PWM period boundary. Reset type: SYSRSn

20.17.2.76 DCFWINDOWCNT Register (Offset = CCh) [Reset = 0h]

DCFWINDOWCNT is shown in Figure 20-164 and described in Table 20-99.

Return to the Summary Table.

Digital Compare Filter Window Counter Register

Figure 20-164. DCFWINDOWCNT Register

15	14	13	12	11	10	9	8
DCFWINDOWCNT							
R-0h							
7	6	5	4	3	2	1	0
DCFWINDOWCNT							
R-0h							

Table 20-99. DCFWINDOWCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	DCFWINDOWCNT	R	0h	Blanking Window Counter These 16 bits are read only and indicate the current value of the window counter. The counter counts down to zero and then stops until it is re-loaded when the offset counter reaches zero again. Reset type: SYSRSn

20.17.2.77 BLANKPULSEMIXSEL Register (Offset = CDh) [Reset = 0h]

BLANKPULSEMIXSEL is shown in Figure 20-165 and described in Table 20-100.

Return to the Summary Table.

Blanking window trigger pulse select register

Figure 20-165. BLANKPULSEMIXSEL Register

15	14	13	12	11	10	9	8
RESERVED						CDD	CDU
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CCD	CCU	CBD	CBU	CAD	CAU	PRD	ZRO
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 20-100. BLANKPULSEMIXSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9	CDD	R/W	0h	Enable event time-base counter equal to CMPD when the timer is decrementing to the blanking window trigger (BLANKPULSEMIX). 0: CMPD down-count match enable event is not enabled 1: Enable CMPD down-count match enable event Reset type: SYSRSn
8	CDU	R/W	0h	Enable event time-base counter equal to CMPD when the timer is incrementing to the blanking window trigger (BLANKPULSEMIX). 0: CMPD up-count match enable event is not enabled 1: Enable CMPD up-count match enable event Reset type: SYSRSn
7	CCD	R/W	0h	Enable event time-base counter equal to CMPC when the timer is decrementing to the blanking window trigger (BLANKPULSEMIX). 0: CMPC down-count match enable event is not enabled 1: Enable CMPC down-count match enable event Reset type: SYSRSn
6	CCU	R/W	0h	Enable event time-base counter equal to CMPC when the timer is incrementing to the blanking window trigger (BLANKPULSEMIX). 0: CMPC up-count match enable event is not enabled 1: Enable CMPC up-count match enable event Reset type: SYSRSn
5	CBD	R/W	0h	Enable event time-base counter equal to CMPB when the timer is decrementing to the blanking window trigger (BLANKPULSEMIX). 0: CMPB down-count match enable event is not enabled 1: Enable CMPB down-count match enable event Reset type: SYSRSn
4	CBU	R/W	0h	Enable event time-base counter equal to CMPB when the timer is incrementing to the mixed ET interrupt trigger signal (BLANKPULSEMIX). 0: CMPB up-count match enable event is not enabled 1: Enable CMPB up-count match enable event Reset type: SYSRSn
3	CAD	R/W	0h	Enable event time-base counter equal to CMPA when the timer is decrementing to the blanking window trigger (BLANKPULSEMIX). 0: CMPA down-count match enable event is not enabled 1: Enable CMPA down-count match enable event Reset type: SYSRSn
2	CAU	R/W	0h	Enable event time-base counter equal to CMPA when the timer is incrementing to the blanking window trigger (BLANKPULSEMIX). 0: CMPA up-count match enable event is not enabled 1: Enable CMPA up-count match enable event Reset type: SYSRSn

Table 20-100. BLANKPULSEMIXSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	PRD	R/W	0h	Enable event time-base counter equal to period (TBCTR = TBPRD) to the blanking window trigger (BLANKPULSEMIX). 0: Period match event is not enabled 1: Enable period match event Reset type: SYSRSn
0	ZRO	R/W	0h	Enable event time-base counter equal to zero (TBCTR = 0x00) to the blanking window trigger (BLANKPULSEMIX). 0: Zero match event is not enabled 1: Enable zero match event Reset type: SYSRSn

20.17.2.78 DCCAP Register (Offset = CFh) [Reset = 0h]

DCCAP is shown in Figure 20-166 and described in Table 20-101.

Return to the Summary Table.

Digital Compare Counter Capture Register

Figure 20-166. DCCAP Register

15	14	13	12	11	10	9	8
DCCAP							
R-0h							
7	6	5	4	3	2	1	0
DCCAP							
R-0h							

Table 20-101. DCCAP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	DCCAP	R	0h	Digital Compare Time-Base Counter Capture To enable time-base counter capture, set the DCCAPCLT[CAPE] bit to 1. If enabled, reflects the value of the time-base counter (TBCTR) on the low to high edge transition of a filtered (DCEVTFLT) event. Further capture events are ignored until the next period or zero as selected by the DCFCTL[PULSESEL] bit. Shadowing of DCCAP is enabled and disabled by the DCCAPCTL[SHDWMODE] bit. By default this register is shadowed. - If DCCAPCTL[SHDWMODE] = 0, then the shadow is enabled. In this mode, the active register is copied to the shadow register on the TBCTR = TBPRD or TBCTR = zero as defined by the DCFCTL[PULSESEL] bit. CPU reads of this register will return the shadow register value. - If DCCAPCTL[SHDWMODE] = 1, then the shadow register is disabled. In this mode, CPU reads will return the active register value. The active and shadow registers share the same memory map address. Reset type: SYSRSn

20.17.2.79 DCAHTRIPSEL Register (Offset = D2h) [Reset = 0h]

DCAHTRIPSEL is shown in Figure 20-167 and described in Table 20-102.

Return to the Summary Table.

Digital Compare AH Trip Select

Figure 20-167. DCAHTRIPSEL Register

15	14	13	12	11	10	9	8
RESERVED	TRIPINPUT15	TRIPINPUT14	RESERVED	TRIPINPUT12	TRIPINPUT11	TRIPINPUT10	TRIPINPUT9
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
TRIPINPUT8	TRIPINPUT7	TRIPINPUT6	TRIPINPUT5	TRIPINPUT4	TRIPINPUT3	TRIPINPUT2	TRIPINPUT1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 20-102. DCAHTRIPSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	TRIPINPUT15	R/W	0h	TRIP Input 15 0: Trip Input 15 not selected as combinational ORed input 1: Trip Input 15 selected as combinational ORed input to DCAH mux Reset type: SYSRSn
13	TRIPINPUT14	R/W	0h	TRIP Input 14 0: Trip Input 14 not selected as combinational ORed input 1: Trip Input 14 selected as combinational ORed input to DCAH mux Reset type: SYSRSn
12	RESERVED	R/W	0h	Reserved
11	TRIPINPUT12	R/W	0h	TRIP Input 12 0: Trip Input 12 not selected as combinational ORed input 1: Trip Input 12 selected as combinational ORed input to DCAH mux Reset type: SYSRSn
10	TRIPINPUT11	R/W	0h	TRIP Input 11 0: Trip Input 11 not selected as combinational ORed input 1: Trip Input 11 selected as combinational ORed input to DCAH mux Reset type: SYSRSn
9	TRIPINPUT10	R/W	0h	TRIP Input 10 0: Trip Input 10 not selected as combinational ORed input 1: Trip Input 10 selected as combinational ORed input to DCAH mux Reset type: SYSRSn
8	TRIPINPUT9	R/W	0h	TRIP Input 9 0: Trip Input 9 not selected as combinational ORed input 1: Trip Input 9 selected as combinational ORed input to DCAH mux Reset type: SYSRSn
7	TRIPINPUT8	R/W	0h	TRIP Input 8 0: Trip Input 8 not selected as combinational ORed input 1: Trip Input 8 selected as combinational ORed input to DCAH mux Reset type: SYSRSn
6	TRIPINPUT7	R/W	0h	TRIP Input 7 0: Trip Input 7 not selected as combinational ORed input 1: Trip Input 7 selected as combinational ORed input to DCAH mux Reset type: SYSRSn
5	TRIPINPUT6	R/W	0h	TRIP Input 6 0: Trip Input 6 not selected as combinational ORed input 1: Trip Input 6 selected as combinational ORed input to DCAH mux Reset type: SYSRSn

Table 20-102. DCAHTRIPSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	TRIPINPUT5	R/W	0h	TRIP Input 5 0: Trip Input 5 not selected as combinational ORed input 1: Trip Input 5 selected as combinational ORed input to DCAH mux Reset type: SYSRSn
3	TRIPINPUT4	R/W	0h	TRIP Input 4 0: Trip Input 4 not selected as combinational ORed input 1: Trip Input 4 selected as combinational ORed input to DCAH mux Reset type: SYSRSn
2	TRIPINPUT3	R/W	0h	TRIP Input 3 0: Trip Input 3 not selected as combinational ORed input 1: Trip Input 3 selected as combinational ORed input to DCAH mux Reset type: SYSRSn
1	TRIPINPUT2	R/W	0h	TRIP Input 2 0: Trip Input 2 not selected as combinational ORed input 1: Trip Input 2 selected as combinational ORed input to DCAH mux Reset type: SYSRSn
0	TRIPINPUT1	R/W	0h	TRIP Input 1 0: Trip Input 1 not selected as combinational ORed input 1: Trip Input 1 selected as combinational ORed input to DCAH mux Reset type: SYSRSn

20.17.2.80 DCALTRIPSEL Register (Offset = D3h) [Reset = 0h]

DCALTRIPSEL is shown in Figure 20-168 and described in Table 20-103.

Return to the Summary Table.

Digital Compare AL Trip Select

Figure 20-168. DCALTRIPSEL Register

15	14	13	12	11	10	9	8
RESERVED	TRIPINPUT15	TRIPINPUT14	RESERVED	TRIPINPUT12	TRIPINPUT11	TRIPINPUT10	TRIPINPUT9
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
TRIPINPUT8	TRIPINPUT7	TRIPINPUT6	TRIPINPUT5	TRIPINPUT4	TRIPINPUT3	TRIPINPUT2	TRIPINPUT1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 20-103. DCALTRIPSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	TRIPINPUT15	R/W	0h	TRIP Input 15 0: Trip Input 15 not selected as combinational ORed input 1: Trip Input 15 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
13	TRIPINPUT14	R/W	0h	TRIP Input 14 0: Trip Input 14 not selected as combinational ORed input 1: Trip Input 14 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
12	RESERVED	R/W	0h	Reserved
11	TRIPINPUT12	R/W	0h	TRIP Input 12 0: Trip Input 12 not selected as combinational ORed input 1: Trip Input 12 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
10	TRIPINPUT11	R/W	0h	TRIP Input 11 0: Trip Input 11 not selected as combinational ORed input 1: Trip Input 11 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
9	TRIPINPUT10	R/W	0h	TRIP Input 10 0: Trip Input 10 not selected as combinational ORed input 1: Trip Input 10 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
8	TRIPINPUT9	R/W	0h	TRIP Input 9 0: Trip Input 9 not selected as combinational ORed input 1: Trip Input 9 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
7	TRIPINPUT8	R/W	0h	TRIP Input 8 0: Trip Input 8 not selected as combinational ORed input 1: Trip Input 8 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
6	TRIPINPUT7	R/W	0h	TRIP Input 7 0: Trip Input 7 not selected as combinational ORed input 1: Trip Input 7 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
5	TRIPINPUT6	R/W	0h	TRIP Input 6 0: Trip Input 6 not selected as combinational ORed input 1: Trip Input 6 selected as combinational ORed input to DCAL mux Reset type: SYSRSn

Table 20-103. DCALTRIPSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	TRIPINPUT5	R/W	0h	TRIP Input 5 0: Trip Input 5 not selected as combinational ORed input 1: Trip Input 5 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
3	TRIPINPUT4	R/W	0h	TRIP Input 4 0: Trip Input 4 not selected as combinational ORed input 1: Trip Input 4 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
2	TRIPINPUT3	R/W	0h	TRIP Input 3 0: Trip Input 3 not selected as combinational ORed input 1: Trip Input 3 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
1	TRIPINPUT2	R/W	0h	TRIP Input 2 0: Trip Input 2 not selected as combinational ORed input 1: Trip Input 2 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
0	TRIPINPUT1	R/W	0h	TRIP Input 1 0: Trip Input 1 not selected as combinational ORed input 1: Trip Input 1 selected as combinational ORed input to DCAL mux Reset type: SYSRSn

20.17.2.81 DCBHTRIPSEL Register (Offset = D4h) [Reset = 0h]

DCBHTRIPSEL is shown in Figure 20-169 and described in Table 20-104.

Return to the Summary Table.

Digital Compare BH Trip Select

Figure 20-169. DCBHTRIPSEL Register

15	14	13	12	11	10	9	8
RESERVED	TRIPINPUT15	TRIPINPUT14	RESERVED	TRIPINPUT12	TRIPINPUT11	TRIPINPUT10	TRIPINPUT9
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
TRIPINPUT8	TRIPINPUT7	TRIPINPUT6	TRIPINPUT5	TRIPINPUT4	TRIPINPUT3	TRIPINPUT2	TRIPINPUT1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 20-104. DCBHTRIPSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	TRIPINPUT15	R/W	0h	TRIP Input 15 0: Trip Input 15 not selected as combinational ORed input 1: Trip Input 15 selected as combinational ORed input to DCBH mux Reset type: SYSRSn
13	TRIPINPUT14	R/W	0h	TRIP Input 14 0: Trip Input 14 not selected as combinational ORed input 1: Trip Input 14 selected as combinational ORed input to DCBH mux Reset type: SYSRSn
12	RESERVED	R/W	0h	Reserved
11	TRIPINPUT12	R/W	0h	TRIP Input 12 0: Trip Input 12 not selected as combinational ORed input 1: Trip Input 12 selected as combinational ORed input to DCBH mux Reset type: SYSRSn
10	TRIPINPUT11	R/W	0h	TRIP Input 11 0: Trip Input 11 not selected as combinational ORed input 1: Trip Input 11 selected as combinational ORed input to DCBH mux Reset type: SYSRSn
9	TRIPINPUT10	R/W	0h	TRIP Input 10 0: Trip Input 10 not selected as combinational ORed input 1: Trip Input 10 selected as combinational ORed input to DCBH mux Reset type: SYSRSn
8	TRIPINPUT9	R/W	0h	TRIP Input 9 0: Trip Input 9 not selected as combinational ORed input 1: Trip Input 9 selected as combinational ORed input to DCBH mux Reset type: SYSRSn
7	TRIPINPUT8	R/W	0h	TRIP Input 8 0: Trip Input 8 not selected as combinational ORed input 1: Trip Input 8 selected as combinational ORed input to DCBH mux Reset type: SYSRSn
6	TRIPINPUT7	R/W	0h	TRIP Input 7 0: Trip Input 7 not selected as combinational ORed input 1: Trip Input 7 selected as combinational ORed input to DCBH mux Reset type: SYSRSn
5	TRIPINPUT6	R/W	0h	TRIP Input 6 0: Trip Input 6 not selected as combinational ORed input 1: Trip Input 6 selected as combinational ORed input to DCBH mux Reset type: SYSRSn

Table 20-104. DCBHTRIPSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	TRIPINPUT5	R/W	0h	TRIP Input 5 0: Trip Input 5 not selected as combinational ORed input 1: Trip Input 5 selected as combinational ORed input to DCBH mux Reset type: SYSRSn
3	TRIPINPUT4	R/W	0h	TRIP Input 4 0: Trip Input 4 not selected as combinational ORed input 1: Trip Input 4 selected as combinational ORed input to DCBH mux Reset type: SYSRSn
2	TRIPINPUT3	R/W	0h	TRIP Input 3 0: Trip Input 3 not selected as combinational ORed input 1: Trip Input 3 selected as combinational ORed input to DCBH mux Reset type: SYSRSn
1	TRIPINPUT2	R/W	0h	TRIP Input 2 0: Trip Input 2 not selected as combinational ORed input 1: Trip Input 2 selected as combinational ORed input to DCBH mux Reset type: SYSRSn
0	TRIPINPUT1	R/W	0h	TRIP Input 1 0: Trip Input 1 not selected as combinational ORed input 1: Trip Input 1 selected as combinational ORed input to DCBH mux Reset type: SYSRSn

20.17.2.82 DCBLTRIPSEL Register (Offset = D5h) [Reset = 0h]

DCBLTRIPSEL is shown in Figure 20-170 and described in Table 20-105.

Return to the Summary Table.

Digital Compare BL Trip Select

Figure 20-170. DCBLTRIPSEL Register

15	14	13	12	11	10	9	8
RESERVED	TRIPINPUT15	TRIPINPUT14	RESERVED	TRIPINPUT12	TRIPINPUT11	TRIPINPUT10	TRIPINPUT9
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
TRIPINPUT8	TRIPINPUT7	TRIPINPUT6	TRIPINPUT5	TRIPINPUT4	TRIPINPUT3	TRIPINPUT2	TRIPINPUT1
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 20-105. DCBLTRIPSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	TRIPINPUT15	R/W	0h	TRIP Input 15 0: Trip Input 15 not selected as combinational ORed input 1: Trip Input 15 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
13	TRIPINPUT14	R/W	0h	TRIP Input 14 0: Trip Input 14 not selected as combinational ORed input 1: Trip Input 14 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
12	RESERVED	R/W	0h	Reserved
11	TRIPINPUT12	R/W	0h	TRIP Input 12 0: Trip Input 12 not selected as combinational ORed input 1: Trip Input 12 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
10	TRIPINPUT11	R/W	0h	TRIP Input 11 0: Trip Input 11 not selected as combinational ORed input 1: Trip Input 11 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
9	TRIPINPUT10	R/W	0h	TRIP Input 10 0: Trip Input 10 not selected as combinational ORed input 1: Trip Input 10 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
8	TRIPINPUT9	R/W	0h	TRIP Input 9 0: Trip Input 9 not selected as combinational ORed input 1: Trip Input 9 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
7	TRIPINPUT8	R/W	0h	TRIP Input 8 0: Trip Input 8 not selected as combinational ORed input 1: Trip Input 8 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
6	TRIPINPUT7	R/W	0h	TRIP Input 7 0: Trip Input 7 not selected as combinational ORed input 1: Trip Input 7 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
5	TRIPINPUT6	R/W	0h	TRIP Input 6 0: Trip Input 6 not selected as combinational ORed input 1: Trip Input 6 selected as combinational ORed input to DCAL mux Reset type: SYSRSn

Table 20-105. DCBLTRIPSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	TRIPINPUT5	R/W	0h	TRIP Input 5 0: Trip Input 5 not selected as combinational ORed input 1: Trip Input 5 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
3	TRIPINPUT4	R/W	0h	TRIP Input 4 0: Trip Input 4 not selected as combinational ORed input 1: Trip Input 4 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
2	TRIPINPUT3	R/W	0h	TRIP Input 3 0: Trip Input 3 not selected as combinational ORed input 1: Trip Input 3 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
1	TRIPINPUT2	R/W	0h	TRIP Input 2 0: Trip Input 2 not selected as combinational ORed input 1: Trip Input 2 selected as combinational ORed input to DCAL mux Reset type: SYSRSn
0	TRIPINPUT1	R/W	0h	TRIP Input 1 0: Trip Input 1 not selected as combinational ORed input 1: Trip Input 1 selected as combinational ORed input to DCAL mux Reset type: SYSRSn

20.17.2.83 EPWMLOCK Register (Offset = FAh) [Reset = 0h]

EPWMLOCK is shown in Figure 20-171 and described in Table 20-106.

Return to the Summary Table.

EPWM Lock Register

Figure 20-171. EPWMLOCK Register

31	30	29	28	27	26	25	24
KEY							
R-0/W-0h							
23	22	21	20	19	18	17	16
KEY							
R-0/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			DCLOCK	TZCLRLOCK	TZCFGLOCK	GLLOCK	HRLOCK
R-0h			R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h	R/WOnce-0h

Table 20-106. EPWMLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	R-0/W	0h	Write to this register succeeds only if this field is written with a value of 0xa5a5 Note: [1] Due to this KEY, only 32-bit writes will succeed (provided the KEY matches). 16-bit writes to the upper or lower half of this register will be ignored Reset type: SYSRSn
15-5	RESERVED	R	0h	Reserved
4	DCLOCK	R/WOnce	0h	0:Digital Compare registers from 0xC0 to 0xD9 offsets are protected by EALLOW. 1: Digital Compare registers from 0xC0 and 0xD9 offsets are locked and not writable. Reset type: SYSRSn
3	TZCLRLOCK	R/WOnce	0h	0:Digital Compare registers from 0x97 to 0x9B offsets are protected by EALLOW. 1: Digital Compare registers from 0x97 and 0x9B offsets are locked and not writable. Reset type: SYSRSn
2	TZCFGLOCK	R/WOnce	0h	0:TripZone registers from 0x80 to 0x8D and TZTRIPOUTSEL at 0x9D offsets are protected by EALLOW. 1: TripZone registers from 0x80 and 0x8D and TZTRIPOUTSEL at 0x9D offsets are locked and not writable. Reset type: SYSRSn
1	GLLOCK	R/WOnce	0h	0:TripZone registers from 0x34 to 0x35 offsets are protected by EALLOW. 1: TripZone registers from 0x34 to 0x35 offsets are locked and not writable Reset type: SYSRSn
0	HRLOCK	R/WOnce	0h	0: HRPWM registers from 0x20 to 0x2D offsets are protected by EALLOW 1: HRPWM registers from 0x20 and 0x2D offsets are locked and not writable. Reset type: SYSRSn

20.17.2.84 HWVDELVAL Register (Offset = FDh) [Reset = 0h]

HWVDELVAL is shown in Figure 20-172 and described in Table 20-107.

Return to the Summary Table.

Hardware Valley Mode Delay Register

Figure 20-172. HWVDELVAL Register

15	14	13	12	11	10	9	8
HWVDELVAL							
R-0h							
7	6	5	4	3	2	1	0
HWVDELVAL							
R-0h							

Table 20-107. HWVDELVAL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	HWVDELVAL	R	0h	Hardware Valley Delay Value Register This read only register reflects the hardware delay value calculated by the equations defined in VCAPCTL[VDELAYDIV]. This reflects the latest value from the hardware calculations and can change every time valley capture sequence is triggered and VCAP1 and VCAP2 values are updated. Reset type: SYSRSn

20.17.2.85 VCNTVAL Register (Offset = FEh) [Reset = 0h]

VCNTVAL is shown in Figure 20-173 and described in Table 20-108.

Return to the Summary Table.

Hardware Valley Counter Register

Figure 20-173. VCNTVAL Register

15	14	13	12	11	10	9	8
VCNTVAL							
R-0h							
7	6	5	4	3	2	1	0
VCNTVAL							
R-0h							

Table 20-108. VCNTVAL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	VCNTVAL	R	0h	Valley Time Base Counter Register This register reflects the captured VCNT value upon occurrence of STOPEDGE selected in VCNTCFG register. Reset type: SYSRSn

20.17.3 Register to Driverlib Function Mapping

20.17.3.1 EPWM Registers to Driverlib Functions

Table 20-109. EPWM Registers to Driverlib Functions

File	Driverlib Function
TBCTL	
epwm.c	EPWM_setEmulationMode
epwm.h	EPWM_setCountModeAfterSync
epwm.h	EPWM_setClockPrescaler
epwm.h	EPWM_forceSyncPulse
epwm.h	EPWM_setOneShotSyncOutTrigger
epwm.h	EPWM_setPeriodLoadMode
epwm.h	EPWM_enablePhaseShiftLoad
epwm.h	EPWM_disablePhaseShiftLoad
epwm.h	EPWM_setTimeBaseCounterMode
epwm.h	EPWM_selectPeriodLoadEvent
epwm.h	EPWM_enableOneShotSync
epwm.h	EPWM_disableOneShotSync
epwm.h	EPWM_startOneShotSync
TBCTL2	
epwm.h	EPWM_selectPeriodLoadEvent
epwm.h	EPWM_enableOneShotSync
epwm.h	EPWM_disableOneShotSync
epwm.h	EPWM_startOneShotSync
SYNCINSEL	
epwm.h	EPWM_setSyncInPulseSource
TBCTR	
epwm.h	EPWM_setTimeBaseCounter
epwm.h	EPWM_getTimeBaseCounterValue
TBSTS	
epwm.h	EPWM_getTimeBaseCounterOverflowStatus
epwm.h	EPWM_clearTimeBaseCounterOverflowEvent
epwm.h	EPWM_getSyncStatus
epwm.h	EPWM_clearSyncEvent
epwm.h	EPWM_getTimeBaseCounterDirection
SYNCOUTEN	
epwm.h	EPWM_enableSyncOutPulseSource
epwm.h	EPWM_disableSyncOutPulseSource
TBCTL3	
epwm.h	EPWM_setOneShotSyncOutTrigger
CMPCTL	
epwm.h	EPWM_setCounterCompareShadowLoadMode
epwm.h	EPWM_disableCounterCompareShadowLoadMode
epwm.h	EPWM_getCounterCompareShadowStatus
CMPCTL2	
epwm.h	EPWM_setCounterCompareShadowLoadMode
epwm.h	EPWM_disableCounterCompareShadowLoadMode

Table 20-109. EPWM Registers to Driverlib Functions (continued)

File	Driverlib Function
DBCTL	
epwm.h	EPWM_setDeadBandOutputSwapMode
epwm.h	EPWM_setDeadBandDelayMode
epwm.h	EPWM_setDeadBandDelayPolarity
epwm.h	EPWM_setRisingEdgeDeadBandDelayInput
epwm.h	EPWM_setFallingEdgeDeadBandDelayInput
epwm.h	EPWM_setDeadBandControlShadowLoadMode
epwm.h	EPWM_disableDeadBandControlShadowLoadMode
epwm.h	EPWM_setRisingEdgeDelayCountShadowLoadMode
epwm.h	EPWM_disableRisingEdgeDelayCountShadowLoadMode
epwm.h	EPWM_setFallingEdgeDelayCountShadowLoadMode
epwm.h	EPWM_disableFallingEdgeDelayCountShadowLoadMode
epwm.h	EPWM_setDeadBandCounterClock
DBCTL2	
epwm.h	EPWM_setDeadBandControlShadowLoadMode
epwm.h	EPWM_disableDeadBandControlShadowLoadMode
AQCTL	
epwm.h	EPWM_setActionQualifierShadowLoadMode
epwm.h	EPWM_disableActionQualifierShadowLoadMode
epwm.h	EPWM_setActionQualifierAction
epwm.h	EPWM_setActionQualifierActionComplete
epwm.h	EPWM_setAdditionalActionQualifierActionComplete
AQTSRCSEL	
epwm.h	EPWM_setActionQualifierT1TriggerSource
epwm.h	EPWM_setActionQualifierT2TriggerSource
PCCTL	
epwm.h	EPWM_enableChopper
epwm.h	EPWM_disableChopper
epwm.h	EPWM_setChopperDutyCycle
epwm.h	EPWM_setChopperFreq
epwm.h	EPWM_setChopperFirstPulseWidth
VCAPCTL	
epwm.h	EPWM_enableValleyCapture
epwm.h	EPWM_disableValleyCapture
epwm.h	EPWM_startValleyCapture
epwm.h	EPWM_setValleyTriggerSource
epwm.h	EPWM_enableValleyHWDelay
epwm.h	EPWM_disableValleyHWDelay
epwm.h	EPWM_setValleyDelayDivider
VCNTCFG	
epwm.h	EPWM_setValleyTriggerEdgeCounts
epwm.h	EPWM_getValleyEdgeStatus
HRCNFG	
-	
HRPWR	

Table 20-109. EPWM Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
HRMSTEP	
-	
HRCNFG2	
-	
HRPCTL	
-	
TRREM	
-	
GLDCTL	
epwm.h	EPWM_enableGlobalLoad
epwm.h	EPWM_disableGlobalLoad
epwm.h	EPWM_setGlobalLoadTrigger
epwm.h	EPWM_setGlobalLoadEventPrescale
epwm.h	EPWM_getGlobalLoadEventCount
epwm.h	EPWM_disableGlobalLoadOneShotMode
epwm.h	EPWM_enableGlobalLoadOneShotMode
epwm.h	EPWM_setGlobalLoadOneShotLatch
epwm.h	EPWM_forceGlobalLoadOneShotEvent
GLDCFG	
epwm.h	EPWM_enableGlobalLoadRegisters
epwm.h	EPWM_disableGlobalLoadRegisters
XLINK	
epwm.h	EPWM_setupEPWMLinks
AQCTLA	
epwm.h	EPWM_setActionQualifierAction
epwm.h	EPWM_setActionQualifierActionComplete
epwm.h	EPWM_setAdditionalActionQualifierActionComplete
AQCTLA2	
epwm.h	EPWM_setActionQualifierAction
epwm.h	EPWM_setAdditionalActionQualifierActionComplete
AQCTLB	
-	See AQCTLA
AQCTLB2	
-	See AQCTLA2
AQSFRC	
epwm.h	EPWM_setActionQualifierContSWForceShadowMode
epwm.h	EPWM_setActionQualifierSWAction
epwm.h	EPWM_forceActionQualifierSWAction
AQCSFRC	
epwm.h	EPWM_setActionQualifierContSWForceAction
DBREDHR	
-	
DBRED	
epwm.h	EPWM_setRisingEdgeDelayCount

Table 20-109. EPWM Registers to Driverlib Functions (continued)

File	Driverlib Function
DBFEDHR	
-	
DBFED	
epwm.h	EPWM_setFallingEdgeDelayCount
TBPHS	
epwm.h	EPWM_setPhaseShift
TBPRDHR	
-	
TBPRD	
epwm.h	EPWM_setTimeBasePeriod
epwm.h	EPWM_getTimeBasePeriod
CMPA	
epwm.h	EPWM_setCounterCompareValue
epwm.h	EPWM_getCounterCompareValue
CMPB	
-	See CMPA
CMPC	
epwm.h	EPWM_setCounterCompareShadowLoadMode
epwm.h	EPWM_disableCounterCompareShadowLoadMode
epwm.h	EPWM_getCounterCompareShadowStatus
CMPD	
-	See CMPC
GLDCTL2	
epwm.h	EPWM_setGlobalLoadOneShotLatch
epwm.h	EPWM_forceGlobalLoadOneShotEvent
SWVDELVAL	
epwm.h	EPWM_setValleySWDelayValue
TZSEL	
epwm.h	EPWM_enableTripZoneSignals
epwm.h	EPWM_disableTripZoneSignals
TZDCSEL	
epwm.h	EPWM_setTripZoneDigitalCompareEventCondition
TZCTL	
epwm.h	EPWM_enableTripZoneAdvAction
epwm.h	EPWM_disableTripZoneAdvAction
epwm.h	EPWM_setTripZoneAction
epwm.h	EPWM_setTripZoneAdvAction
epwm.h	EPWM_setTripZoneAdvDigitalCompareActionA
epwm.h	EPWM_setTripZoneAdvDigitalCompareActionB
TZCTL2	
epwm.h	EPWM_enableTripZoneAdvAction
epwm.h	EPWM_disableTripZoneAdvAction
epwm.h	EPWM_setTripZoneAdvAction
epwm.h	EPWM_setTripZoneAdvDigitalCompareActionA
epwm.h	EPWM_setTripZoneAdvDigitalCompareActionB

Table 20-109. EPWM Registers to Driverlib Functions (continued)

File	Driverlib Function
TZCTLDCA	
epwm.h	EPWM_setTripZoneAdvDigitalCompareActionA
TZCTLDCB	
epwm.h	EPWM_setTripZoneAdvDigitalCompareActionB
TZEINT	
epwm.h	EPWM_enableTripZoneInterrupt
epwm.h	EPWM_disableTripZoneInterrupt
TZFLG	
epwm.h	EPWM_getTripZoneFlagStatus
TZCBCFLG	
epwm.h	EPWM_getCycleByCycleTripZoneFlagStatus
TZOSTFLG	
epwm.h	EPWM_getOneShotTripZoneFlagStatus
TZCLR	
epwm.h	EPWM_selectCycleByCycleTripZoneClearEvent
epwm.h	EPWM_clearTripZoneFlag
TZCBCCLR	
epwm.h	EPWM_clearCycleByCycleTripZoneFlag
TZOSTCLR	
epwm.h	EPWM_clearOneShotTripZoneFlag
TZFRC	
epwm.h	EPWM_forceTripZoneEvent
ETSEL	
epwm.h	EPWM_enableInterrupt
epwm.h	EPWM_disableInterrupt
epwm.h	EPWM_setInterruptSource
epwm.h	EPWM_enableADCTrigger
epwm.h	EPWM_disableADCTrigger
epwm.h	EPWM_setADCTriggerSource
ETPS	
epwm.h	EPWM_setInterruptEventCount
epwm.h	EPWM_setADCTriggerEventPrescale
ETFLG	
epwm.h	EPWM_getEventTriggerInterruptStatus
epwm.h	EPWM_getADCTriggerFlagStatus
ETCLR	
epwm.h	EPWM_clearEventTriggerInterruptFlag
epwm.h	EPWM_clearADCTriggerFlag
ETFRC	
epwm.h	EPWM_forceEventTriggerInterrupt
epwm.h	EPWM_forceADCTrigger
ETINTPS	
epwm.h	EPWM_setInterruptEventCount
epwm.h	EPWM_getInterruptEventCount
ETSOCPS	

Table 20-109. EPWM Registers to Driverlib Functions (continued)

File	Driverlib Function
epwm.h	EPWM_setADCTriggerEventPrescale
epwm.h	EPWM_getADCTriggerEventCount
ETCNTINITCTL	
epwm.h	EPWM_enableInterruptEventCountInit
epwm.h	EPWM_disableInterruptEventCountInit
epwm.h	EPWM_forceInterruptEventCountInit
epwm.h	EPWM_enableADCTriggerEventCountInit
epwm.h	EPWM_disableADCTriggerEventCountInit
epwm.h	EPWM_forceADCTriggerEventCountInit
ETCNTINIT	
epwm.h	EPWM_enableInterruptEventCountInit
epwm.h	EPWM_disableInterruptEventCountInit
epwm.h	EPWM_forceInterruptEventCountInit
epwm.h	EPWM_setInterruptEventCountInitValue
epwm.h	EPWM_enableADCTriggerEventCountInit
epwm.h	EPWM_disableADCTriggerEventCountInit
epwm.h	EPWM_forceADCTriggerEventCountInit
epwm.h	EPWM_setADCTriggerEventCountInitValue
DCTRIPSEL	
epwm.h	EPWM_selectDigitalCompareTripInput
epwm.h	EPWM_enableDigitalCompareTripCombinationInput
DCACTL	
epwm.h	EPWM_setDigitalCompareEventSource
epwm.h	EPWM_setDigitalCompareEventSyncMode
epwm.h	EPWM_enableDigitalCompareADCTrigger
epwm.h	EPWM_disableDigitalCompareADCTrigger
epwm.h	EPWM_enableDigitalCompareSyncEvent
epwm.h	EPWM_disableDigitalCompareSyncEvent
epwm.h	EPWM_setDigitalCompareCBCLatchMode
epwm.h	EPWM_selectDigitalCompareCBCLatchClearEvent
epwm.h	EPWM_getDigitalCompareCBCLatchStatus
DCBCTL	
-	See DCACTL
DCFCTL	
epwm.h	EPWM_enableDigitalCompareBlankingWindow
epwm.h	EPWM_disableDigitalCompareBlankingWindow
epwm.h	EPWM_enableDigitalCompareWindowInverseMode
epwm.h	EPWM_disableDigitalCompareWindowInverseMode
epwm.h	EPWM_setDigitalCompareBlankingEvent
epwm.h	EPWM_setDigitalCompareFilterInput
epwm.h	EPWM_enableDigitalCompareEdgeFilter
epwm.h	EPWM_disableDigitalCompareEdgeFilter
epwm.h	EPWM_setDigitalCompareEdgeFilterMode
epwm.h	EPWM_setDigitalCompareEdgeFilterEdgeCount
epwm.h	EPWM_getDigitalCompareEdgeFilterEdgeCount

Table 20-109. EPWM Registers to Driverlib Functions (continued)

File	Driverlib Function
epwm.h	EPWM_getDigitalCompareEdgeFilterEdgeStatus
DCCAPCTL	
epwm.h	EPWM_enableDigitalCompareCounterCapture
epwm.h	EPWM_disableDigitalCompareCounterCapture
epwm.h	EPWM_setDigitalCompareCounterShadowMode
epwm.h	EPWM_getDigitalCompareCaptureStatus
DCFOFFSET	
epwm.h	EPWM_setDigitalCompareWindowOffset
epwm.h	EPWM_getDigitalCompareBlankingWindowOffsetCount
DCFOFFSETCNT	
epwm.h	EPWM_getDigitalCompareBlankingWindowOffsetCount
DCFWINDOW	
epwm.h	EPWM_setDigitalCompareWindowLength
epwm.h	EPWM_getDigitalCompareBlankingWindowLengthCount
DCFWINDOWCNT	
epwm.h	EPWM_getDigitalCompareBlankingWindowLengthCount
BLANKPULSEMIXSEL	
-	
DCCAP	
epwm.h	EPWM_enableDigitalCompareCounterCapture
epwm.h	EPWM_disableDigitalCompareCounterCapture
epwm.h	EPWM_setDigitalCompareCounterShadowMode
epwm.h	EPWM_getDigitalCompareCaptureStatus
epwm.h	EPWM_getDigitalCompareCaptureCount
DCAHTRIPSEL	
epwm.h	EPWM_enableDigitalCompareTripCombinationInput
epwm.h	EPWM_disableDigitalCompareTripCombinationInput
DCALTRIPSEL	
-	See DCAHTRIPSEL
DCBHTRIPSEL	
-	See DCAHTRIPSEL
DCBLTRIPSEL	
-	See DCAHTRIPSEL
LOCK	
epwm.h	EPWM_lockRegisters
HWVDELVAL	
epwm.h	EPWM_getValleyHWDelay
VCNTVAL	
epwm.h	EPWM_getValleyCount

20.17.3.2 HRPWM Registers to Driverlib Functions

Table 20-110. HRPWM Registers to Driverlib Functions

File	Driverlib Function
TBCTL	
-	

Table 20-110. HRPWM Registers to Driverlib Functions (continued)

File	Driverlib Function
TBCTL2	
-	
EPWMSYNCINSEL	
-	
TBCTR	
-	
TBSTS	
-	
EPWMSYNCOUTEN	
-	
TBCTL3	
-	
CMPCTL	
-	
CMPCTL2	
-	
DBCTL	
-	
DBCTL2	
-	
AQCTL	
-	
AQTSRCSEL	
-	
PCCTL	
-	
VCAPCTL	
-	
VCNTCFG	
-	
HRCNFG	
hrpwm.h	HRPWM_setMEPEdgeSelect
hrpwm.h	HRPWM_setMEPControlMode
hrpwm.h	HRPWM_setCounterCompareShadowLoadEvent
hrpwm.h	HRPWM_setOutputSwapMode
hrpwm.h	HRPWM_setChannelBOutputPath
hrpwm.h	HRPWM_enableAutoConversion
hrpwm.h	HRPWM_disableAutoConversion
hrpwm.h	HRPWM_setDeadbandMEPEdgeSelect
hrpwm.h	HRPWM_setRisingEdgeDelayLoadMode
hrpwm.h	HRPWM_setFallingEdgeDelayLoadMode
HRPWR	
-	
HRMSTEP	
hrpwm.h	HRPWM_setMEPStep

Table 20-110. HRPWM Registers to Driverlib Functions (continued)

File	Driverlib Function
HRCNFG2	
hrpwm.h	HRPWM_setDeadbandMEPEdgeSelect
hrpwm.h	HRPWM_setRisingEdgeDelayLoadMode
hrpwm.h	HRPWM_setFallingEdgeDelayLoadMode
HRPCTL	
hrpwm.h	HRPWM_enablePeriodControl
hrpwm.h	HRPWM_disablePeriodControl
hrpwm.h	HRPWM_enablePhaseShiftLoad
hrpwm.h	HRPWM_disablePhaseShiftLoad
hrpwm.h	HRPWM_setSyncPulseSource
TRREM	
hrpwm.h	HRPWM_setTranslatorRemainder
GLDCTL	
-	
GLDCFG	
-	
EPWMXLINK	
-	
AQCTLA	
-	
AQCTLA2	
-	
AQCTLB	
-	
AQCTLB2	
-	
AQSFRC	
-	
AQCSFRC	
-	
DBREDHR	
hrpwm.h	HRPWM_setRisingEdgeDelay
hrpwm.h	HRPWM_setHiResRisingEdgeDelayOnly
DBRED	
hrpwm.h	HRPWM_setRisingEdgeDelay
hrpwm.h	HRPWM_setHiResRisingEdgeDelayOnly
DBFEDHR	
hrpwm.h	HRPWM_setFallingEdgeDelay
hrpwm.h	HRPWM_setHiResFallingEdgeDelayOnly
DBFED	
hrpwm.h	HRPWM_setFallingEdgeDelay
hrpwm.h	HRPWM_setHiResFallingEdgeDelayOnly
TBPHS	
hrpwm.h	HRPWM_setPhaseShift
hrpwm.h	HRPWM_setHiResPhaseShiftOnly

Table 20-110. HRPWM Registers to Driverlib Functions (continued)

File	Driverlib Function
TBPRDHR	
hrpwm.h	HRPWM_setTimeBasePeriod
hrpwm.h	HRPWM_setHiResTimeBasePeriodOnly
hrpwm.h	HRPWM_getTimeBasePeriod
hrpwm.h	HRPWM_getHiResTimeBasePeriodOnly
TBPRD	
hrpwm.h	HRPWM_setTimeBasePeriod
hrpwm.h	HRPWM_setHiResTimeBasePeriodOnly
hrpwm.h	HRPWM_getTimeBasePeriod
hrpwm.h	HRPWM_getHiResTimeBasePeriodOnly
CMPA	
hrpwm.h	HRPWM_setCounterCompareValue
hrpwm.h	HRPWM_setHiResCounterCompareValueOnly
hrpwm.h	HRPWM_getCounterCompareValue
hrpwm.h	HRPWM_getHiResCounterCompareValueOnly
CMPB	
hrpwm.h	HRPWM_setCounterCompareValue
hrpwm.h	HRPWM_setHiResCounterCompareValueOnly
hrpwm.h	HRPWM_getCounterCompareValue
hrpwm.h	HRPWM_getHiResCounterCompareValueOnly
CMPC	
-	
CMPD	
-	
GLDCTL2	
-	
SWVDELVAL	
-	
TZSEL	
-	
TZDCSEL	
-	
TZCTL	
-	
TZCTL2	
-	
TZCTLDCA	
-	
TZCTLDCB	
-	
TZEINT	
-	
TZFLG	
-	
TZCBCFLG	

Table 20-110. HRPWM Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
TZOSTFLG	
-	
TZCLR	
-	
TZCBCCLR	
-	
TZOSTCLR	
-	
TZFRC	
-	
ETSEL	
-	
ETPS	
-	
ETFLG	
-	
ETCLR	
-	
ETFRC	
-	
ETINTPS	
-	
ETSOCPS	
-	
ETCNTINITCTL	
-	
ETCNTINIT	
-	
DCTRIPSEL	
-	
DCACTL	
-	
DCBCTL	
-	
DCFCTL	
-	
DCCAPCTL	
-	
DCFOFFSET	
-	
DCFOFFSETCNT	
-	
DCFWINDOW	
-	

Table 20-110. HRPWM Registers to Driverlib Functions (continued)

File	Driverlib Function
DCFWINDOWCNT	
-	
BLANKPULSEMIXSEL	
-	
DCCAP	
-	
DCAHTRIPSEL	
-	
DCALTRIPSEL	
-	
DCBHTRIPSEL	
-	
DCBLTRIPSEL	
-	
EPWMLOCK	
hrpwm.h	HRPWM_lockRegisters
HWVDELVAL	
-	
VCNTVAL	
-	

Chapter 21

Enhanced Capture (eCAP)

This chapter describes the enhanced capture (eCAP) module, which is used in systems where accurate timing of external events is important.

The enhanced capture (eCAP) module is a Type 2 eCAP. See the *C2000 Real-Time Control Peripheral Reference Guide* for a list of all devices with an eCAP module of the same type, to determine the differences between the types, and for a list of device-specific differences within a type.

21.1 Introduction

21.1.1 Features

The features of the eCAP module include:

- Speed measurements of rotating machinery (for example, toothed sprockets sensed by way of Hall sensors)
- Elapsed time measurements between position sensor pulses
- Period and duty cycle measurements of pulse train signals
- Decoding current or voltage amplitude derived from duty cycle encoded current/voltage sensors

The eCAP module features described in this chapter include:

- 4-event time-stamp registers (each 32 bits)
- Edge polarity selection for up to four sequenced time-stamp capture events
- Interrupt on either of the four events
- Single-shot capture of up to four event time-stamps
- Continuous mode capture of time stamps in a four-deep circular buffer
- Absolute time-stamp capture
- Difference (Delta) mode time-stamp capture
- When not used in capture mode, the eCAP module can be configured as a single-channel PWM output

The capture functionality of the Type 1 eCAP is enhanced from the Type 0 eCAP with the following added features:

- Event filter reset bit
  - Writing a 1 to ECCTL2[CTRFILTRESET] will clear the event filter, the modulo counter, and any pending interrupts flags. Resetting the bit is useful for initialization and debug.
- Modulo counter status bits
  - The modulo counter (ECCTL2 [MODCNTRSTS]) indicates which capture register will be loaded next. In the Type 0 eCAP, it was not possible to know current state of modulo counter.
- DMA trigger source
  - eCAPxDMA was added as a DMA trigger. CEVT[1-4] can be configured as the source for eCAPxDMA.
- Input multiplexer
  - ECCTL0 [INPUTSEL] selects one of 128 input signals, which are detailed in Section 21.3.
- EALLOW protection
  - EALLOW protection was added to critical registers. To maintain software compatibility with Type-0, configure DEV_CFG_REGS.ECAPTYPE to make these registers unprotected.

The capture functionality of the Type 2 eCAP is enhanced from the Type 1 eCAP with the following added features:

- Added ECAPxSYNCINSEL register
  - ECAPxSYNCINSEL register is added for each eCAP to select an external SYNCIN. Every eCAP can have a separate SYNCIN signal.

21.1.2 ECAP Related Collateral

Foundational Materials

- C2000 Academy - Control Peripherals

Getting Started Materials

- *Leveraging High Resolution Capture (HRCAP) for Single Wire Data Transfer Application Report*

21.2 Description

The eCAP module represents one complete capture channel that can be instantiated multiple times, depending on the target device. In the context of this guide, one eCAP channel has the following independent key resources:

- Capture inputs can be connected using the Input X-BAR
- 128:1 input multiplexer
- Output X-BAR is used to configure output in APWM mode
- 32-bit time base (counter)
- 4 x 32-bit time-stamp capture registers (CAP1-CAP4)
- Four-stage sequencer (modulo4 counter) that is synchronized to external events, eCAP pin rising/falling edges.
- Modulo counter status register (MODCNTRSTS) to indicate sequencer state
- Independent edge polarity (rising/falling edge) selection for all four events
- Input capture signal prescaling (from 2-62 or bypass)
- One-shot compare register (two bits) to freeze captures after 1-4 time-stamp events
- Control for continuous time-stamp captures using a four-deep circular buffer (CAP1-CAP4) scheme
- Ability to reset event filter, modulo counter, and interrupt flags
- Interrupt capabilities on any of the four capture events
- Separate DMA trigger
- EALLOW protection to control registers

21.3 Configuring Device Pins for the eCAP

The Input X-BAR connects the device pins to the module as input. Any GPIO on the device can be configured as an input. The GPIO input qualification can be set to synchronous or asynchronous mode by setting the GPxQSELn register bits. Using synchronized inputs can help with noise immunity but will affect the eCAP's accuracy by ±2 cycles. The internal pull-ups can be configured in the GPyPUD register. Since the GPIO mode is used, the GPyINV register can invert the signals.

New to the Type 1 eCAP module, a 128:1 input multiplexer must also be configured (see Figure 21-3). This multiplexer can select a variety of inputs detailed in Table 21-1 by configuring ECCTL0.INPUTSEL.

Table 21-1. eCAP Input Selection

Selection of eCAP Input	Select Index
eCAP1-GPTRIP7; eCAP2-GPTRIP8; eCAP3-GPTRIP9;	127
Reserved	126:124
CMPSS.CTRIPH_OR_CTRIPL[4:1]	123:120
Reserved	119
GPIO23 (Direct Connect)	118
GPIO22 (Direct Connect)	117
GPIO9 (Direct Connect)	116
GPIO8 (Direct Connect)	115
Reserved	114:112
CMPSS.CTRIPH[4:1]	111:108
Reserved	107:100
CMPSS.CTRIPL[4:1]	99:96
Reserved	95:92
SDFM1.Filter[4:1].COMPH_OR_COMPL	91:88
SDFM2.Filter[4:1].COMPH_OR_COMPL	87:84
SDFM1.Filter[4:1].COMPH	83:80
SDFM2.Filter[4:1].COMPH	79:76
SDFM1.Filter[4:1].COMPZ	75:72

Table 21-1. eCAP Input Selection (continued)

Selection of eCAP Input	Select Index
SDFM2.Filter[4:1].COMPZ	71:68
SDFM1.Filter[4:1].COMPL	67:64
SDFM2.Filter[4:1].COMPL	63:60
Reserved	59:51
FSI_RX_MEASURE_FALL	50
FSI_RX_MEASURE_RISE	49
FSI_RX_MEASURE	48
ADCAEVENT[4:1]	47:44
ADCBEVENT[4:1]	43:40
ADCCEVENT[4:1]	39:36
CLB4.CLBOUT[15:14]	35:34
CLB3.CLBOUT[15:14]	33:32
XTRIPOUT-XBAR[7:0]	31:24
Reserved	23:21
DCANA_INT0	20
CLB2.CLBOUT[15:14]	19:18
CLB1.CLBOUT[15:14]	17:16
GPIOTRIP-XBAR[16:1]	15:0

The Output X-BAR must be used to connect output signals to the OUTPUTXBARx output locations. The GPIO mux must then be configured to connect the OUTPUTXBARx lines to any of several IO pins with the GPIO mux. To avoid glitches on the pins, the GPyGMUX bits must be configured first (while keeping the corresponding GPyMUX bits at the default of zero), followed by writing the GPyMUX register to the desired value.

See the *General-Purpose Input/Output (GPIO)* chapter for more details on GPIO mux, GPIO settings, and XBAR configuration.

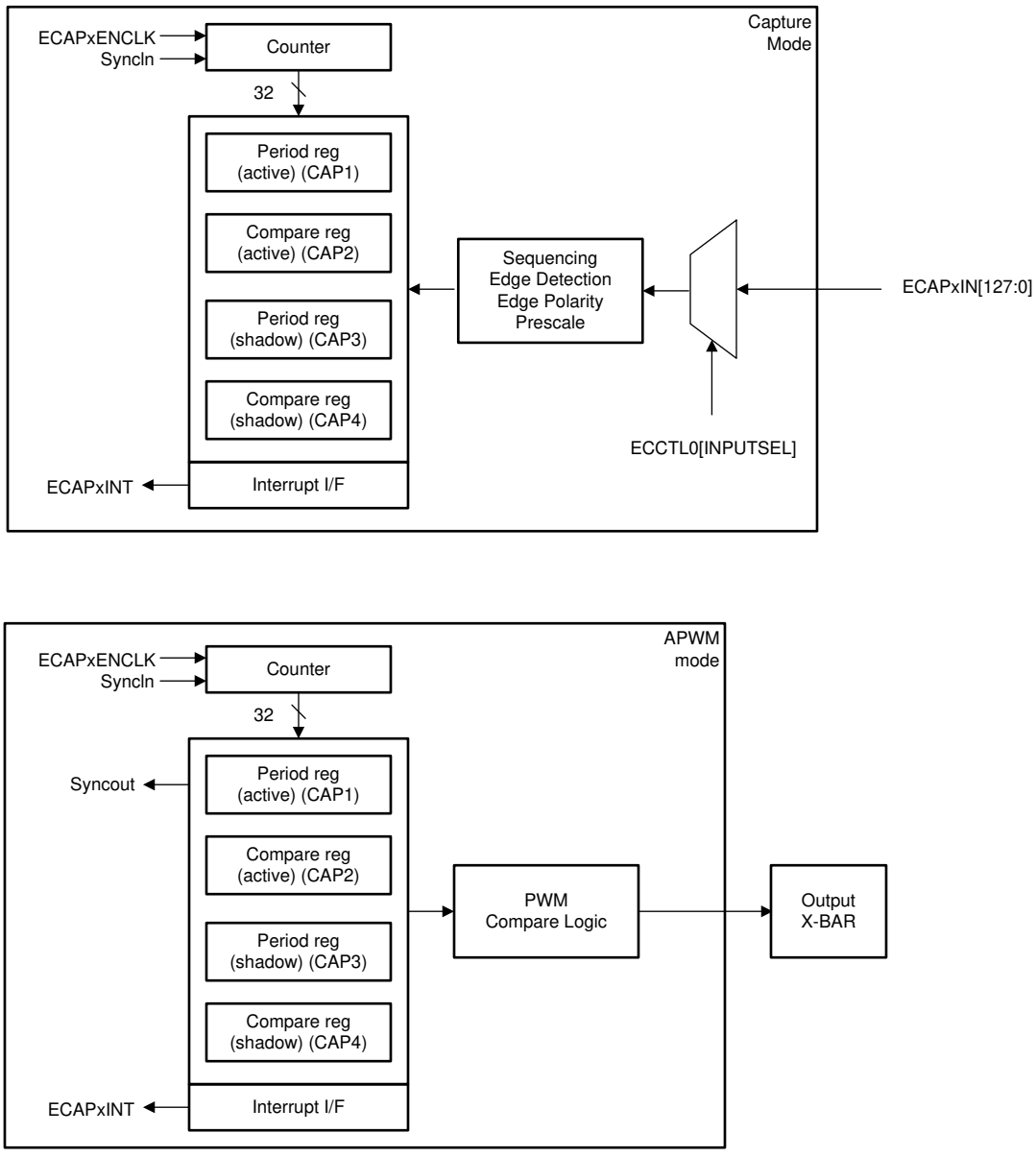
Note

ECAPxIN has to be at least 2*SYSCLK cycles wide to be properly captured by the ECAP module. Otherwise the input pulse might get missed from sampling by the SYSCLK.

21.4 Capture and APWM Operating Mode

You can use the eCAP module resources to implement a single-channel PWM generator (with 32-bit capabilities) when it is not being used for input captures. The counter operates in count-up mode, providing a time-base for asymmetrical pulse width modulation (PWM) waveforms. The CAP1 and CAP2 registers become the active period and compare registers, respectively, while CAP3 and CAP4 registers become the period and compare shadow registers, respectively. Figure 21-1 is a high-level view of both the capture and auxiliary pulse-width modulator (APWM) modes of operation.

Figure 21-2 further descries the output of the eCAP in APWM mode based on the CMP and PRD values.



A. A single pin is shared between CAP and APWM functions. In capture mode, it is an input; in APWM mode, it is an output.

B. In APWM mode, writing any value to CAP1/CAP2 active registers also writes the same value to the corresponding shadow registers CAP3/CAP4. This emulates immediate mode. Writing to the shadow registers CAP3/CAP4 invokes the shadow mode.

Figure 21-1. Capture and APWM Modes of Operation

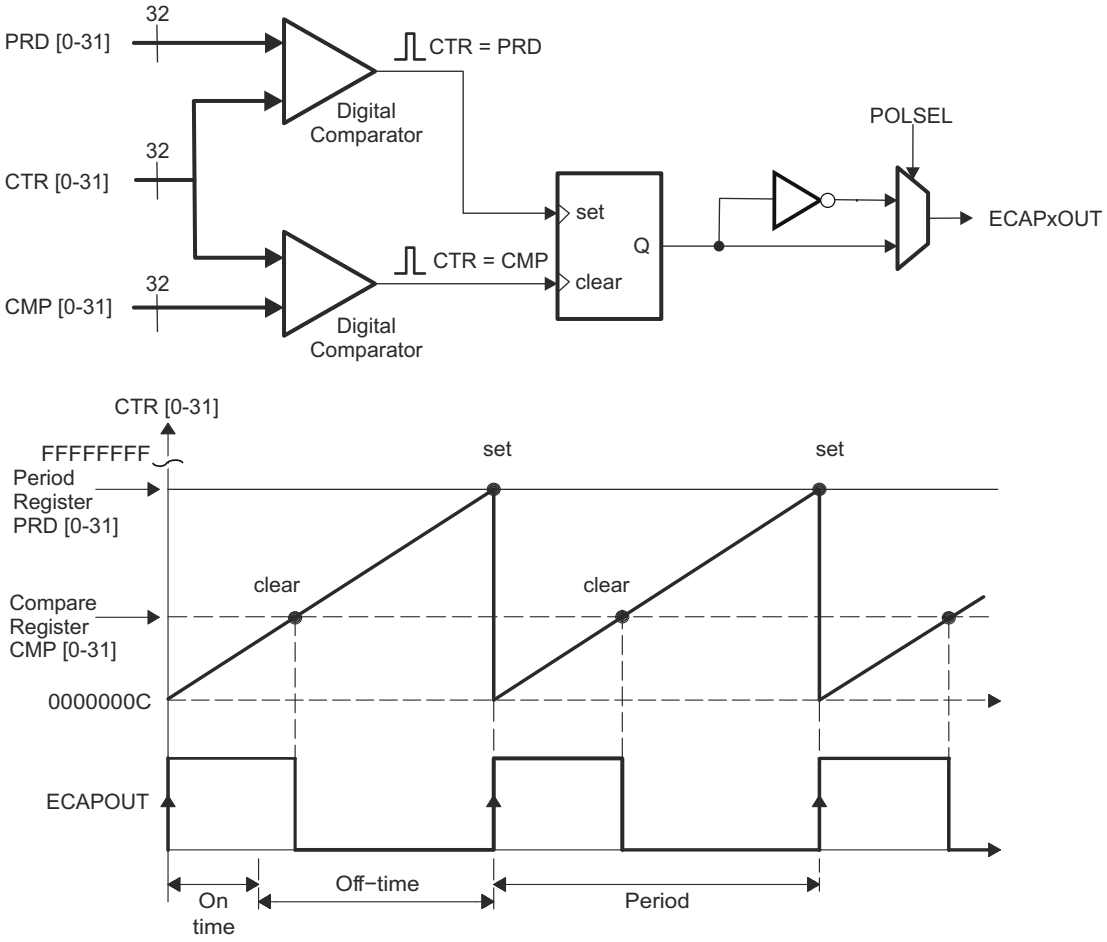
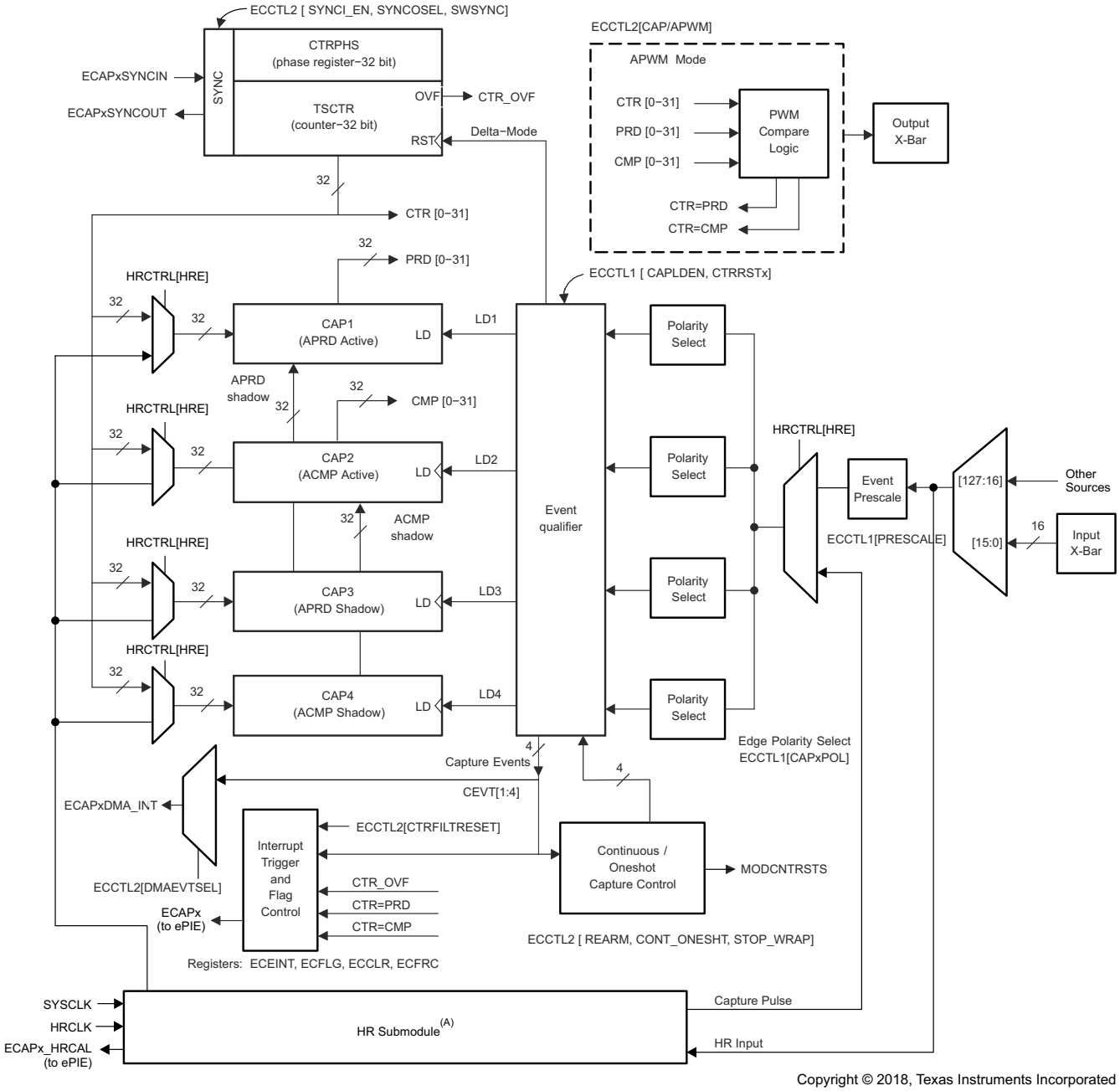


Figure 21-2. Counter Compare and PRD Effects on the eCAP Output in APWM Mode

21.5 Capture Mode Description

Figure 21-3 shows the various components that implement the capture function.

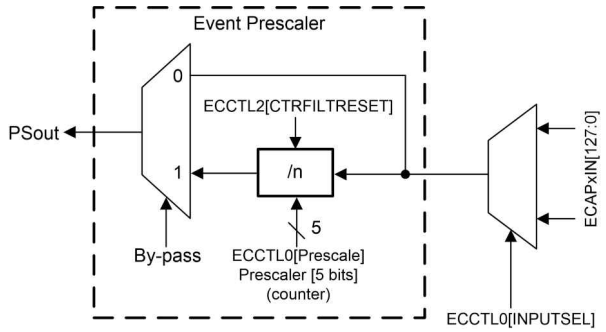


A. The HRCAP submodule is not available on all eCAP modules; in this case, the high-resolution muxes and hardware are not implemented.

Figure 21-3. eCAP Block Diagram

21.5.1 Event Prescaler

An input capture signal (pulse train) can be prescaled by N = 2-62 (in multiples of 2) or can bypass the prescaler. This is useful when very high frequency signals are used as inputs. Figure 21-4 shows a functional diagram and Figure 21-5 shows the operation of the prescale function. The event prescaler can be reset by setting the ECCTL2.CTRFILTRESET register bit.



A. When a prescale value of 1 is chosen (ECCTL1[13:9] = 0,0,0,0,0), the input capture signal bypasses the prescale logic completely.
B. The first Rise edge after Prescale configuration change is not passed to Capture logic, prescaler value takes into effect on the second rising edge after the configuration.

Figure 21-4. Event Prescale Control

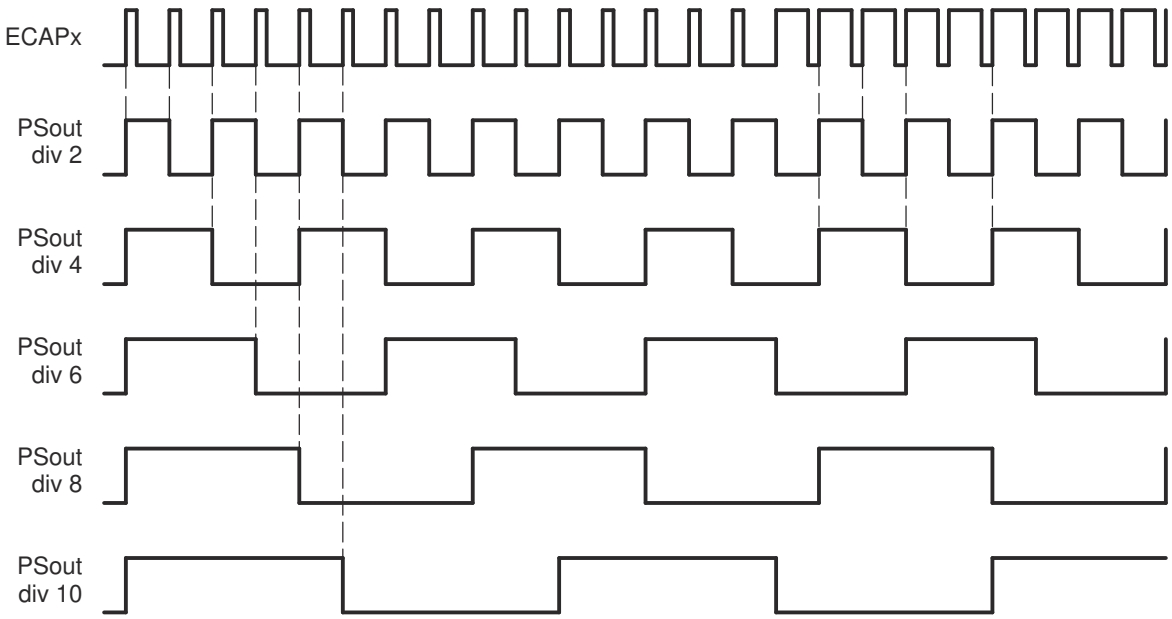


Figure 21-5. Prescale Function Waveforms

21.5.2 Edge Polarity Select and Qualifier

Functionality and features include:

- Four independent edge polarity (rising edge/falling edge) selection muxes are used, one for each capture event.
- Each edge (up to 4) is event qualified by the Modulo4 sequencer.
- The edge event is gated to its respective CAPx register by the Mod4 counter. The CAPx register is loaded on the falling edge.

21.5.3 Continuous/One-Shot Control

Operation of eCAP in Continuous/One-Shot mode:

- The Mod4 (2-bit) counter is incremented via edge qualified events (CEVT1-CEVT4).
- The Mod4 counter continues counting (0->1->2->3->0) and wraps around unless stopped.
- During one-shot operation, a 2-bit stop register (STOP_WRAP) is used to compare the Mod4 counter output, and when equal, stops the Mod4 counter and inhibits further loads of the CAP1-CAP4 registers. In this mode if TSCCTR counter is configured to reset on capture event (CEVTx) by configuring ECCTL1.CTRRSTx bit, it will still keep resetting the TSCCTR counter on capture event (CEVTx) after the STOP_WRAP value is reached and re-arm (REARM) has not occurred.

The continuous/one-shot block controls the start, stop and reset (zero) functions of the Mod4 counter, via a mono-shot type of action that can be triggered by the stop-value comparator and re-armed via software control.

Once armed, the eCAP module waits for 1-4 (defined by stop-value) capture events before freezing both the Mod4 counter and contents of CAP1-4 registers (time stamps).

Re-arming prepares the eCAP module for another capture sequence. Also, re-arming clears (to zero) the Mod4 counter and permits loading of CAP1-4 registers again, providing the CAPLDEN bit is set.

In continuous mode, the Mod4 counter continues to run (0->1->2->3->0, the one-shot action is ignored, and capture values continue to be written to CAP1-4 in a circular buffer sequence.

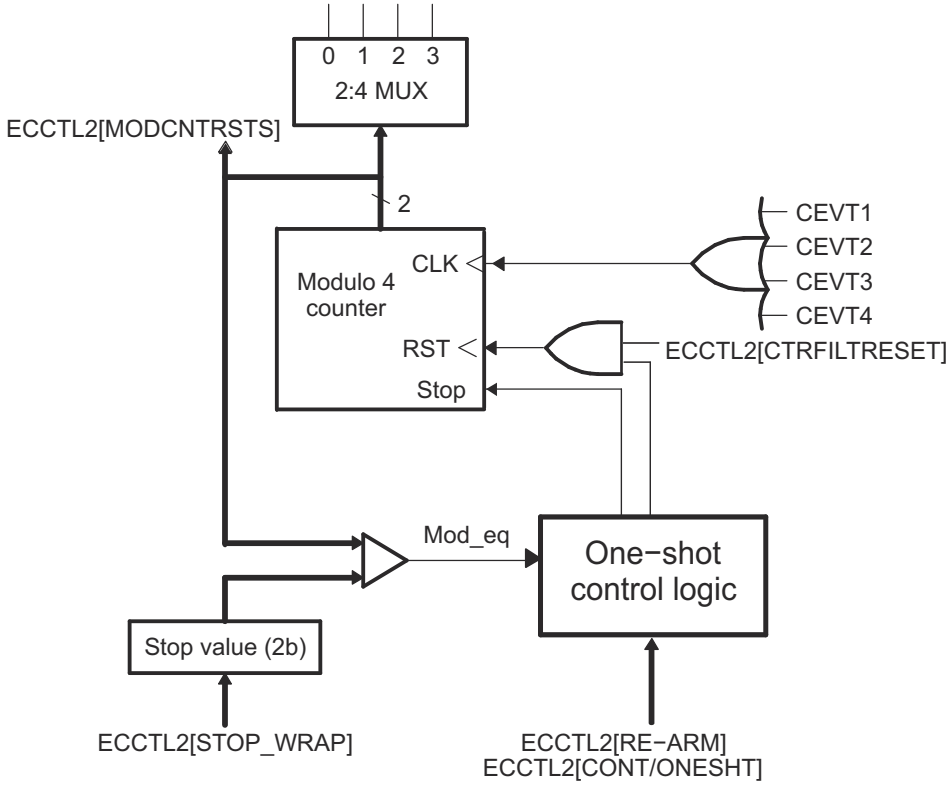


Figure 21-6. Details of the Continuous/One-shot Block

21.5.4 32-Bit Counter and Phase Control

This counter provides the time-base for event captures, and is clocked via the system clock.

A phase register is provided to achieve synchronization with other counters, via a hardware and software forced sync. This is useful in APWM mode when a phase offset between modules is needed.

On any of the four event loads, an option to reset the 32-bit counter is given. This is useful for time difference capture. The 32-bit counter value is captured first, then it is reset to 0 by any of the LD1-LD4 signals.

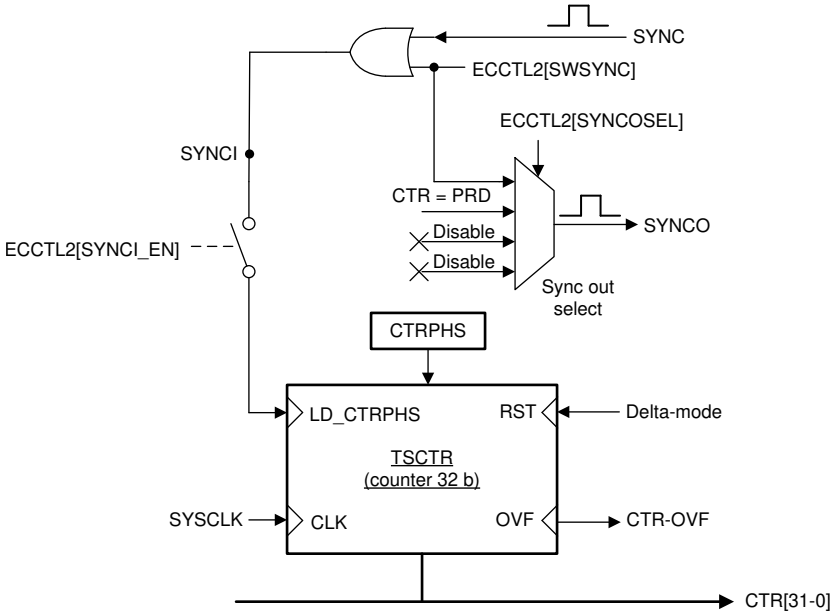


Figure 21-7. Details of the Counter and Synchronization Block

21.5.5 CAP1-CAP4 Registers

These 32-bit registers are fed by the 32-bit counter timer bus, CTR[0-31] and are loaded (capture a time-stamp) when their respective LD inputs are strobed.

Control bit CAPLDEN can inhibit loading of the capture registers. During one-shot operation, this bit is cleared (loading is inhibited) automatically when a stop condition occurs, StopValue = Mod4.

CAP1 and CAP2 registers become the active period and compare registers, respectively, in APWM mode.

CAP3 and CAP4 registers become the respective shadow registers (APRD and ACMP) for CAP1 and CAP2 during APWM operation.

21.5.6 eCAP Synchronization

eCAP modules can be synchronized with each other by selecting a common SYNCIN source. SYNCIN source for eCAP can be either software sync-in or external sync-in. The external sync-in signal can come from eCAP, X-Bar or EPWM. The SWSYNC of the eCAP module is logical ORed with the SYNC signal as shown in Figure 21-7. The SYNC signal is defined by the selection of ECAPxSYNCINSEL[SEL] as shown in Figure 21-8.

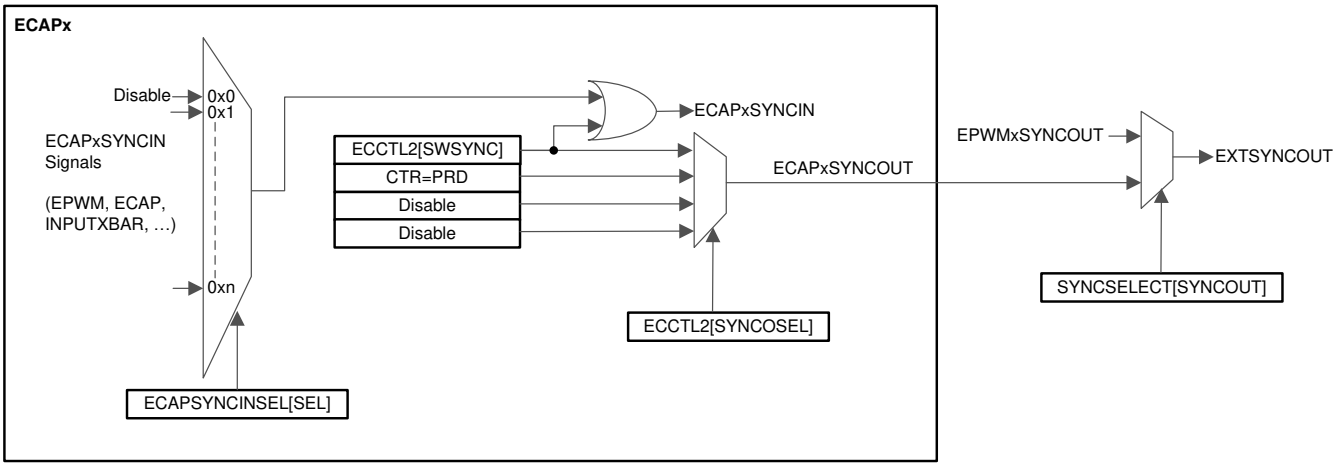


Figure 21-8. eCAP Synchronization Scheme

Figure 21-9. eCAP Synchronization Scheme

21.5.6.1 Example 1 - Using SWSYNC with ECAP Module

Implement the following steps to use SWSYNC with ECAP1 and ECAP2.

- Configure ECAP[1..2].ECAPSYNCINSEL.SEL = 0x0 to disable external SYNCIN coming to eCAP1.
- Configure ECAP[1..2].ECCTL2.SWSYNC = 0x1, to force Software Synchronization of the TSCTR counter.

To use SWSYNC with other eCAP modules, ensure that the previous eCAP chain is not generating a SYNCOUT signal which will interfere with the software synchronization.

21.5.7 Interrupt Control

Operation and features of the eCAP interrupt control include (see Figure 21-10):

- An interrupt can be generated on capture events (CEVT1-CEVT4, CTROVF) or APWM events (CTR = PRD, CTR = CMP).
- A counter overflow event (FFFFFFFF->00000000) is also provided as an interrupt source (CTROVF).
- The capture events are edge and sequencer-qualified (ordered in time) by the polarity select and Mod4 gating, respectively.
- One of these events can be selected as the interrupt source (from the eCAPx module) going to the PIE and CLA.
- Seven interrupt events (CEVT1, CEVT2, CEVT3, CEVT4, CNTOVF, CTR=PRD, CTR=CMP) can be generated.
- The interrupt enable register (ECEINT) is used to enable/disable individual interrupt event sources. The interrupt flag register (ECFLG) indicates if any interrupt event has been latched and contains the global interrupt flag bit (INT). An interrupt pulse is generated to the PIE only if any of the interrupt events are enabled, the flag bit is 1, and the INT flag bit is 0. The interrupt service routine must clear the global interrupt flag bit and the serviced event via the interrupt clear register (ECCLR) before any other interrupt pulses are generated. All interrupt flags will be cleared upon an event filter reset by writing a "1" to ECCTL2[CLRFILTRESET]. You can force an interrupt event via the interrupt force register (ECFRC). This is useful for test purposes.

Note

The CEVT1, CEVT2, CEVT3, CEVT4 flags are only active in capture mode (ECCTL2[CAP/APWM == 0]). The CTR=PRD, CTR=CMP flags are only valid in APWM mode (ECCTL2[CAP/APWM == 1]). CNTOVF flag is valid in both modes.

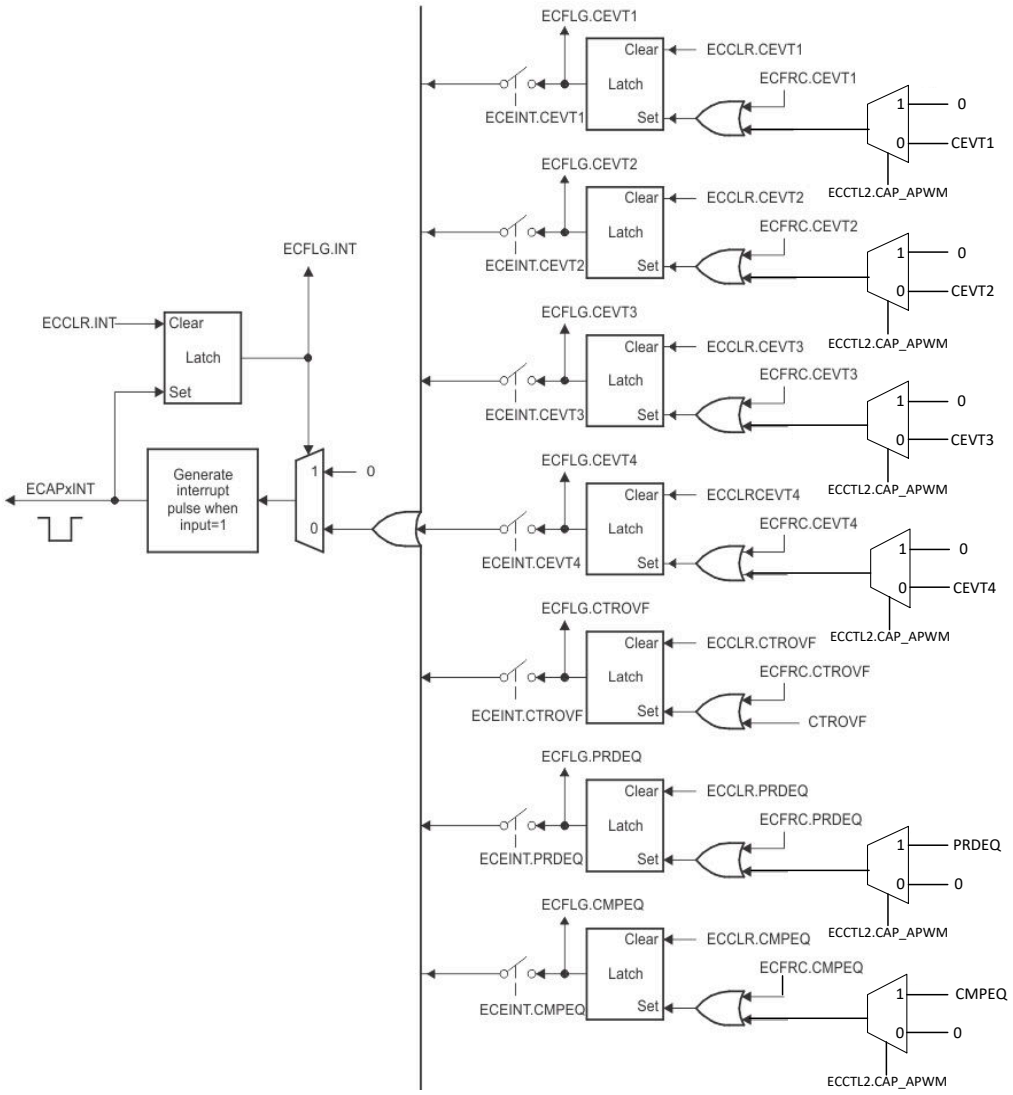


Figure 21-10. Interrupts in eCAP Module

21.5.8 DMA Interrupt

On Type 0 eCAP modules, the CPU was required to begin data transfers using DMA. New to the Type 1 eCAP, a separate DMA Trigger (ECAP_DMA_INT) enables continuous transfer of capture data from eCAP registers to on-chip memory using DMA. Any one of the four available interrupt events (CEVT1, CEVT2, CEVT3 and CEVT4) can be selected as the trigger source for ECAP_DMA_INT using ECCTL2 [DMAEVTSEL].

21.5.9 Shadow Load and Lockout Control

In capture mode, this logic inhibits (locks out) any shadow loading of CAP1 or CAP2 from APRD and ACMP registers, respectively.

In APWM mode, shadow loading is active and two choices are permitted:

- Immediate - APRD or ACMP are transferred to CAP1 or CAP2 immediately upon writing a new value.
- On period equal, CTR[31:0] = PRD[31:0].

21.5.10 APWM Mode Operation

Main operating highlights of the APWM section:

- The time-stamp counter bus is made available for comparison by way of 2 digital (32-bit) comparators.
- When CAP1/2 registers are not used in capture mode, their contents can be used as Period and Compare values in APWM mode.
- Double buffering is achieved via shadow registers APRD and ACMP (CAP3/4). The shadow register contents are transferred over to CAP1/2 registers, either immediately upon a write, or on a CTR = PRD trigger.
- In APWM mode, writing to CAP1/CAP2 active registers will also write the same value to the corresponding shadow registers CAP3/CAP4. This emulates immediate mode. Writing to the shadow registers CAP3/CAP4 will invoke the shadow mode.
- During initialization, you must write to the active registers for both period and compare. This automatically copies the initial values into the shadow values. For subsequent compare updates, during run-time, you only need to use the shadow registers.

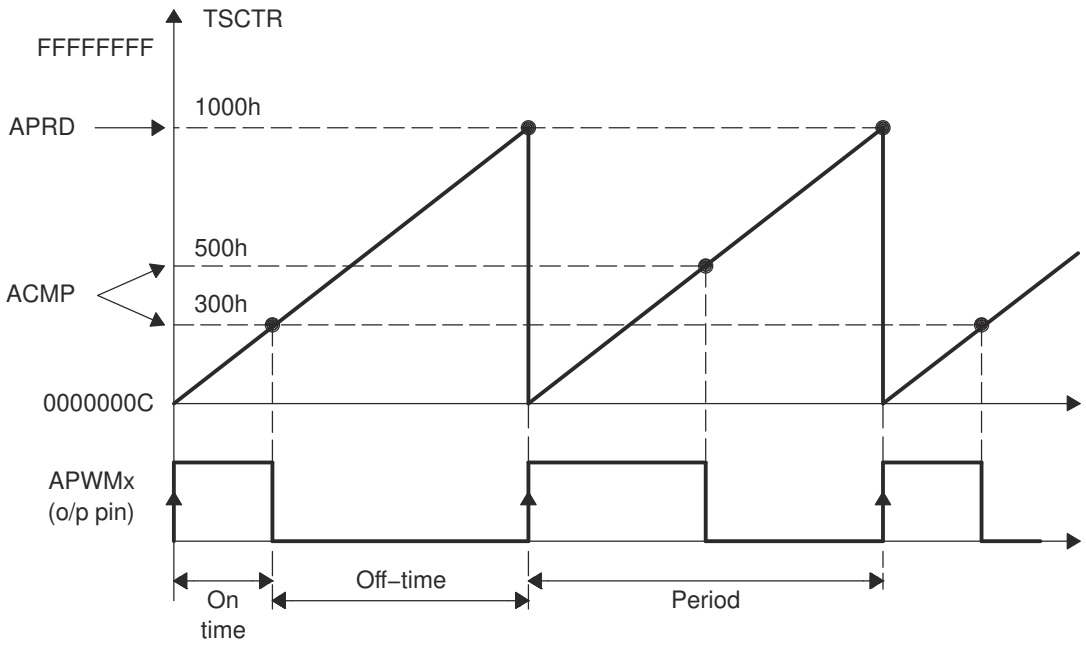


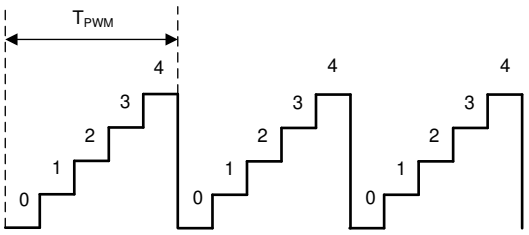
Figure 21-11. PWM Waveform Details Of APWM Mode Operation

The behavior of APWM active high mode (APWMPOL == 0) is as follows:

```
CMP = 0x00000000, output low for duration of period (0% duty)

CMP = 0x00000001, output high 1 cycle

CMP = 0x00000002, output high 2 cycles

CMP = PERIOD, output high except for 1 cycle (<100% duty)

CMP = PERIOD+1, output high for complete period (100% duty)

CMP > PERIOD+1, output high for complete period
```

The behavior of APWM active low mode (APWMPOL == 1) is as follows:

```
CMP = 0x00000000, output high for duration of period (0% duty)

CMP = 0x00000001, output low 1 cycle

CMP = 0x00000002, output low 2 cycles

CMP = PERIOD, output low except for 1 cycle (<100% duty)

CMP = PERIOD+1, output low for complete period (100% duty)

CMP > PERIOD+1, output low for complete period
```



$$T_{PWM} = (CAP1 + 1) \times T_{TSCTR}$$

$$F_{PWM} = \frac{1}{T_{PWM}}$$

Figure 21-12. Time-Base Frequency and Period Calculation

21.6 Application of the eCAP Module

The following sections will provide applications examples to show how to operate the eCAP module.

21.6.1 Example 1 - Absolute Time-Stamp Operation Rising Edge Trigger

Figure 21-13 shows an example of continuous capture operation (Mod4 counter wraps around). In this figure, TSCTR counts-up without resetting and capture events are qualified on the rising edge only, this gives period (and frequency) information.

On an event, the TSCTR contents (time-stamp) is first captured, then Mod4 counter is incremented to the next state. When the TSCTR reaches FFFFFFFF (maximum value), it wraps around to 00000000 (not shown in Figure 21-13), if this occurs, the CTROVF (counter overflow) flag is set, and an interrupt (if enabled) occurs, CTROVF (counter overflow) Flag is set, and an Interrupt (if enabled) occurs. Captured Time-stamps are valid at the point indicated by the diagram (after the 4th event), hence event CEVT4 can conveniently be used to trigger an interrupt and the CPU can read data from the CAPx registers.

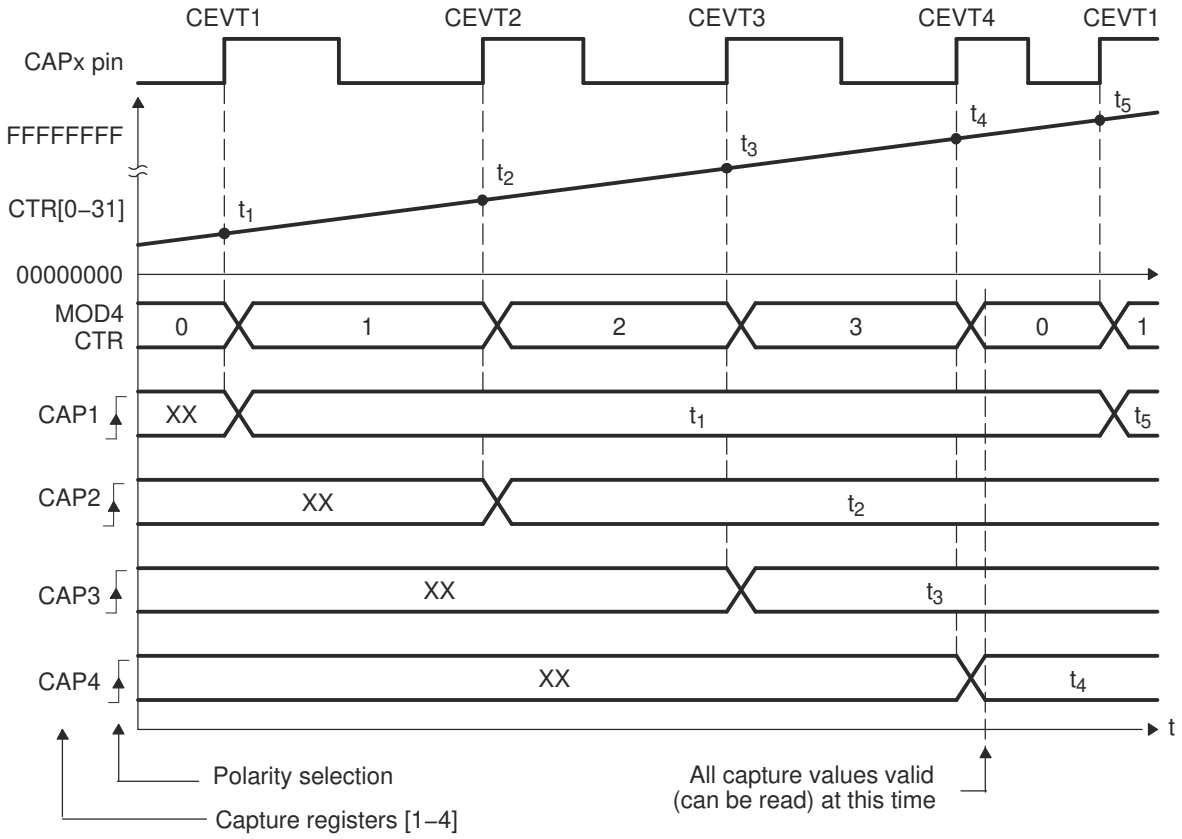


Figure 21-13. Capture Sequence for Absolute Time-stamp and Rising Edge Detect

21.6.2 Example 2 - Absolute Time-Stamp Operation Rising and Falling Edge Trigger

In Figure 21-14, the eCAP operating mode is almost the same as in the previous section except capture events are qualified as either rising or falling edge, this now gives both period and duty cycle information, that is: Period1 = $t_3 - t_1$, Period2 = $t_5 - t_3$, …and so on. Duty Cycle1 (on-time %) = $(t_2 - t_1)$ / Period1 x 100%, and so on. Duty Cycle1 (off-time %) = $(t_3 - t_2)$ / Period1 x 100%, and so on.

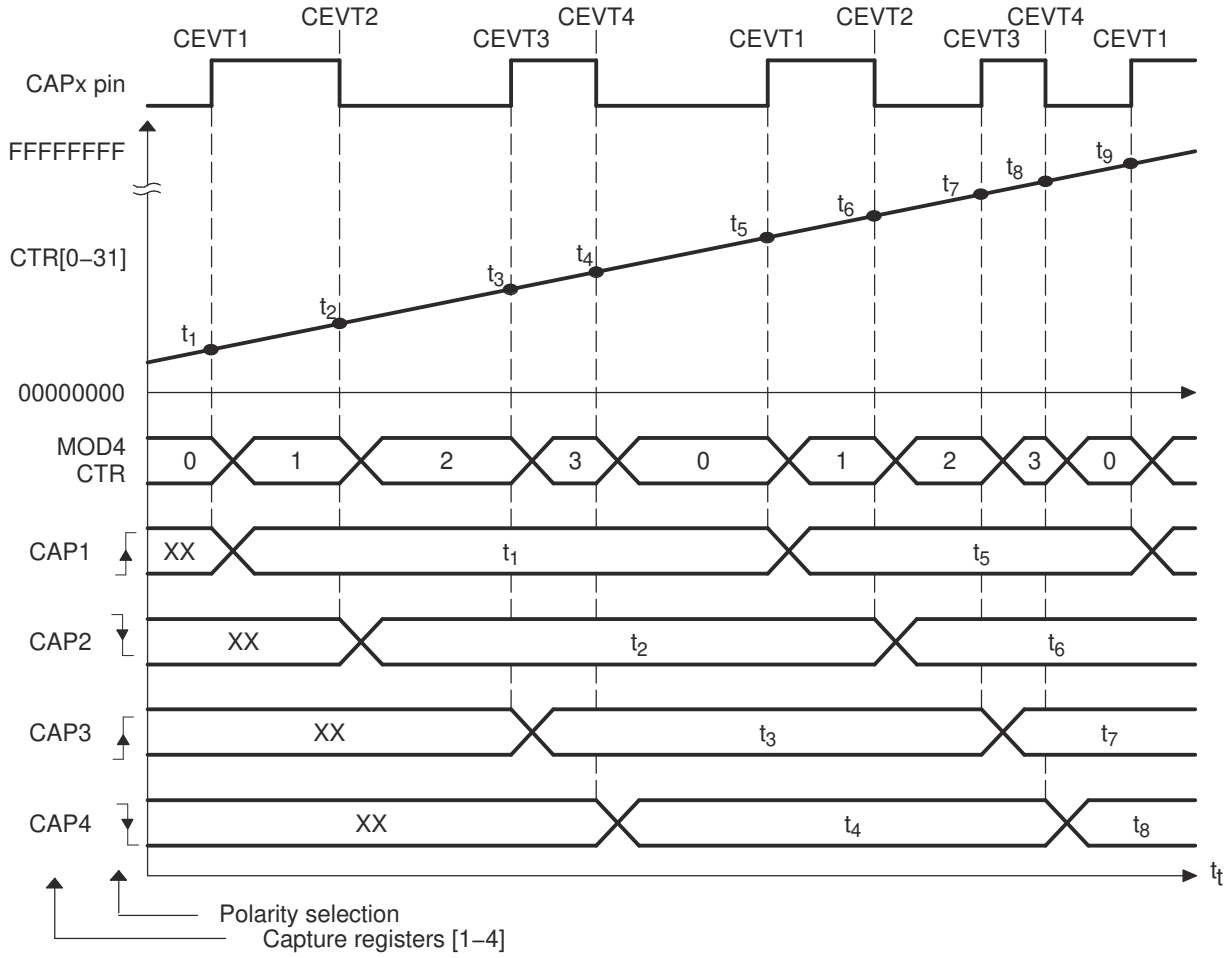


Figure 21-14. Capture Sequence for Absolute Time-stamp With Rising and Falling Edge Detect

21.6.3 Example 3 - Time Difference (Delta) Operation Rising Edge Trigger

Figure 21-15 shows how the eCAP module can be used to collect Delta timing data from pulse train waveforms. Here Continuous Capture mode (TSCTR counts-up without resetting, and Mod4 counter wraps around) is used. In Delta-time mode, TSCTR is Reset back to Zero on every valid event. Here Capture events are qualified as Rising edge only. On an event, TSCTR contents (Time-Stamp) is captured first, and then TSCTR is reset to Zero. The Mod4 counter then increments to the next state. If TSCTR reaches FFFFFFFF (Max value), before the next event, it wraps around to 00000000 and continues, a CNTOVF (counter overflow) Flag is set, and an Interrupt (if enabled) occurs. The advantage of Delta-time Mode is that the CAPx contents directly give timing data without the need for CPU calculations, that is, Period1 = T_1, Period2 = T_2, and so on. As shown in Figure 21-15, the CEVT1 event is a good trigger point to read the timing data, T_1, T_2, T_3, T_4 are all valid here.

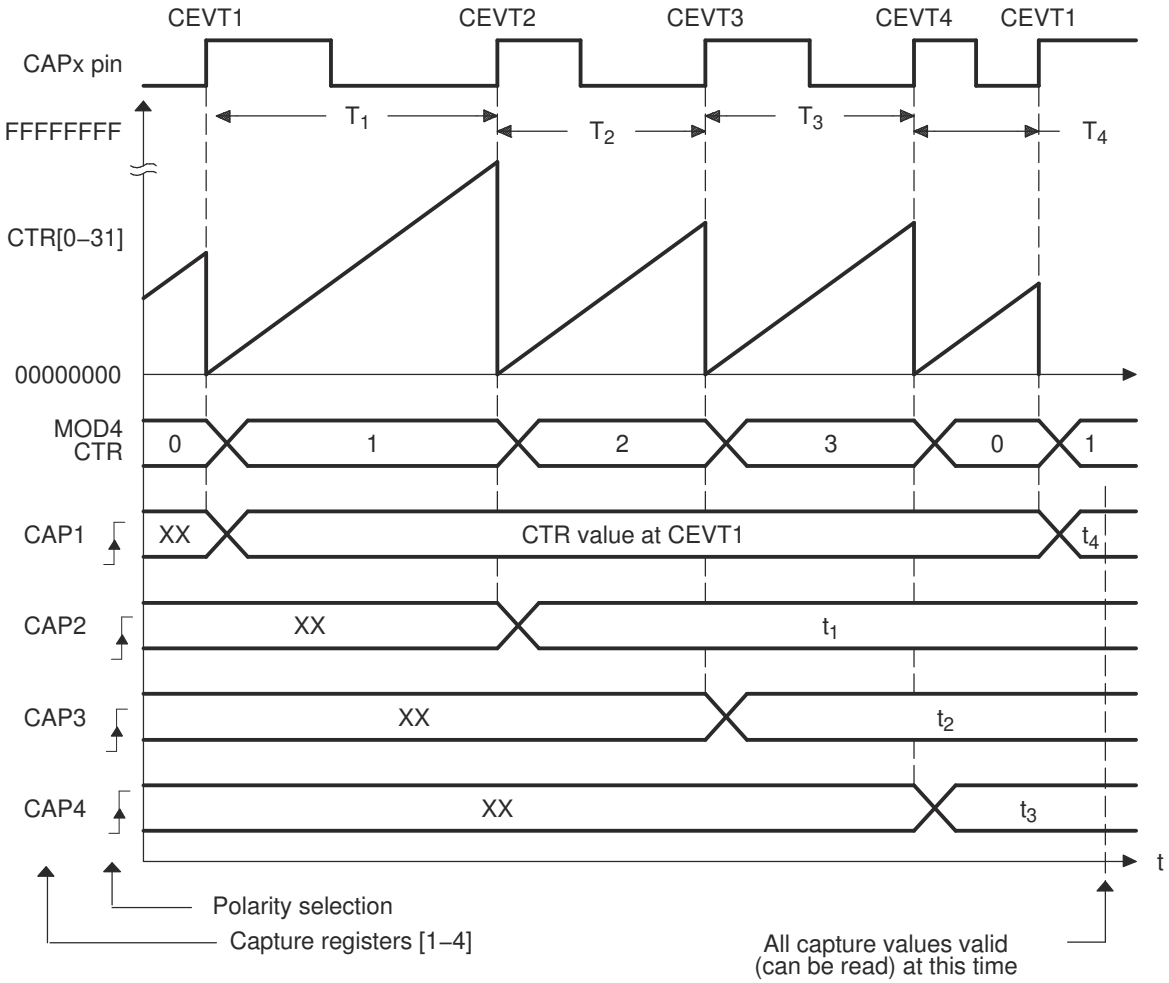


Figure 21-15. Capture Sequence for Delta Mode Time-stamp and Rising Edge Detect

21.6.4 Example 4 - Time Difference (Delta) Operation Rising and Falling Edge Trigger

In Figure 21-16, the eCAP operating mode is almost the same as in previous section except Capture events are qualified as either Rising or Falling edge, this now gives both Period and Duty cycle information, that is: Period1 = T_1+T_2, Period2 = T_3+T_4, and so on. Duty Cycle1 (on-time %) = T_1 / Period1 x 100%, Duty Cycle1 (off-time %) = T_2 / Period1 x 100%, and so on.

During initialization, you must write to the active registers for both period and compare. This action will automatically copy the init values into the shadow values. For subsequent compare updates during run-time, the shadow registers must be used.

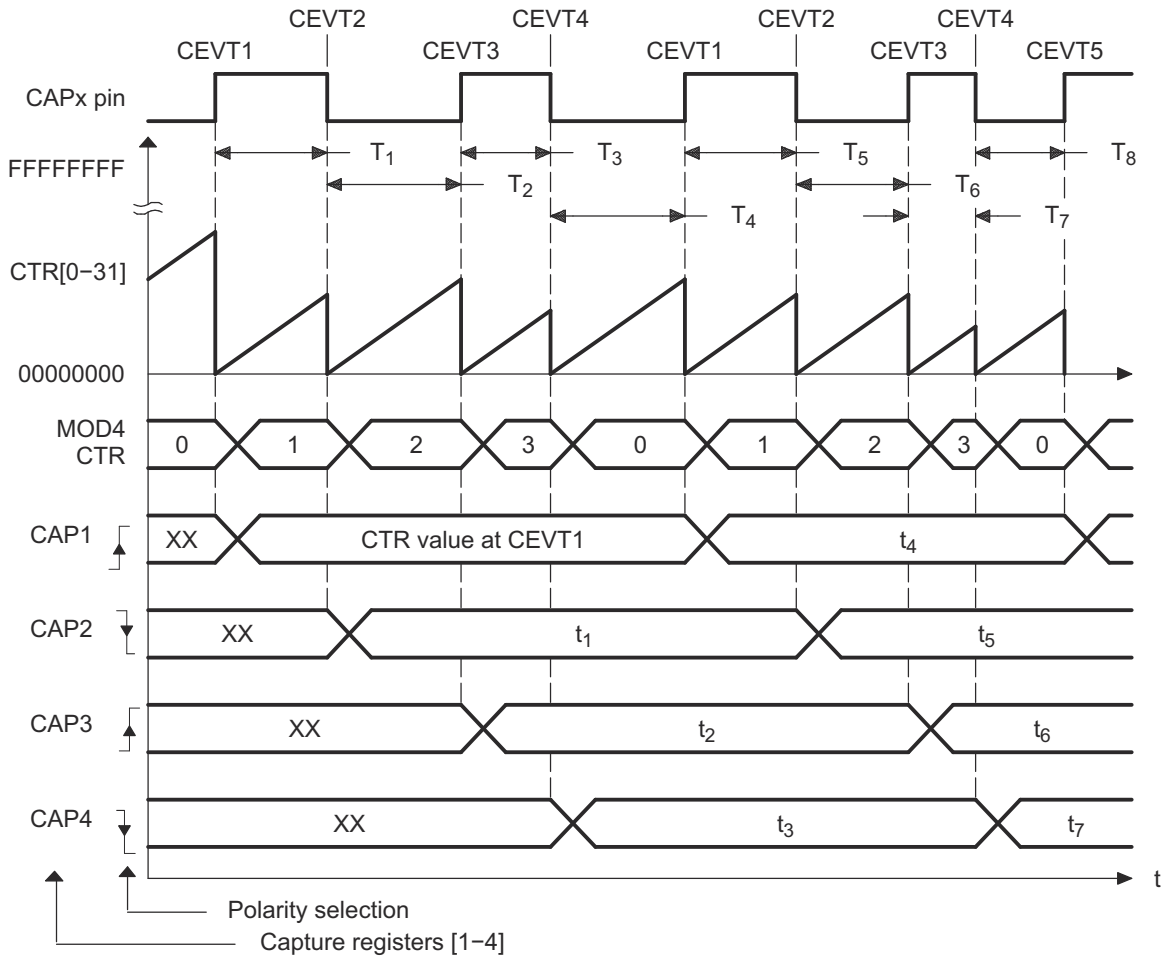


Figure 21-16. Capture Sequence for Delta Mode Time-stamp With Rising and Falling Edge Detect

21.7 Application of the APWM Mode

In this example, the eCAP module is configured to operate as a PWM generator. Here, a very simple single-channel PWM waveform is generated from the APWMx output pin. The PWM polarity is active high, which means that the compare value (CAP2 reg is now a compare register) represents the on-time (high level) of the period. Alternatively, if the APWMPOL bit is configured for active low, then the compare value represents the off-time.

21.7.1 Example 1 - Simple PWM Generation (Independent Channel/s)

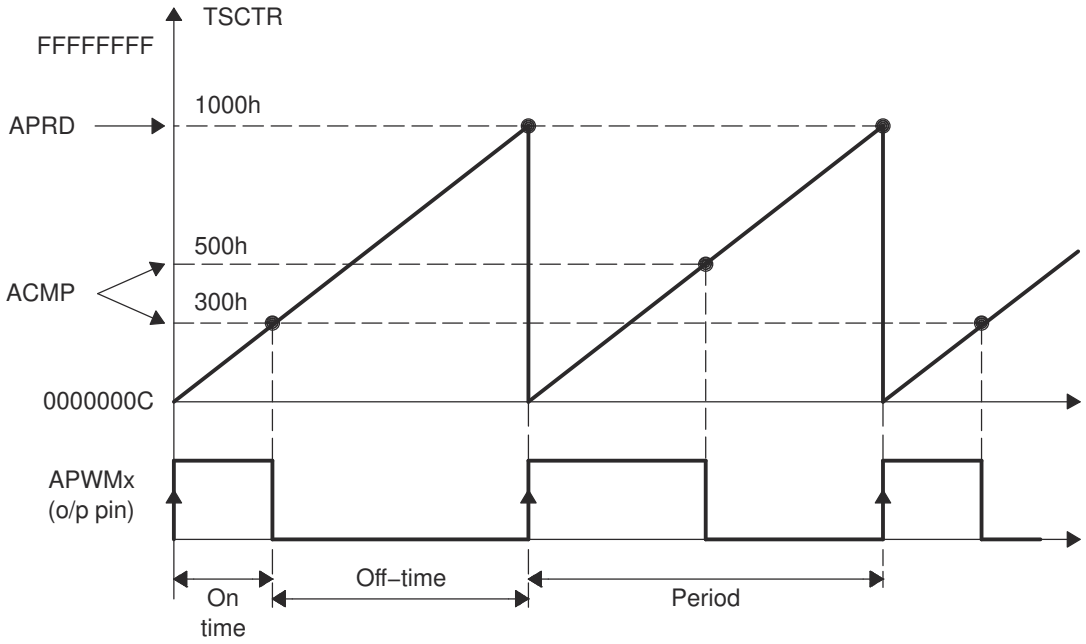


Figure 21-17. PWM Waveform Details of APWM Mode Operation

21.8 Software

21.8.1 ECAP Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/ecap

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

21.8.1.1 eCAP APWM Example

FILE: ecap_ex1_apwm.c

This program sets up the eCAP module in APWM mode. The PWM waveform will come out on GPIO5. The frequency of PWM is configured to vary between 5Hz and 10Hz using the shadow registers to load the next period/compare values.

21.8.1.2 eCAP Capture PWM Example

FILE: ecap_ex2_capture_pwm.c

This example configures ePWM3A for:
- Up count mode
- Period starts at 500 and goes up to 8000
- Toggle output on PRD

eCAP1 is configured to capture the time between rising and falling edge of the ePWM3A output.

External Connections
- eCAP1 is on GPIO16
- ePWM3A is on GPIO4
- Connect GPIO4 to GPIO16.

Watch Variables
- *ecap1PassCount* - Successful captures.
- *ecap1IntCount* - Interrupt counts.

21.8.1.3 eCAP APWM Phase-shift Example

FILE: ecap_ex3_apwm_phase_shift.c

This program sets up the eCAP1 and eCAP2 modules in APWM mode to generate the two phase-shifted PWM outputs of same duty and frequency value The frequency, duty and phase values can be programmed of choice by updating the defined macros. By default 10 Khz frequency, 50% duty and 30% phase shift values are used. eCAP2 output leads the eCAP1 output by 30% GPIO5 and GPIO6 are used as eCAP1/2 outputs and can be probed using analyzer/CRO to observe the waveforms.

21.8.1.4 eCAP Software Sync Example

FILE: ecap_ex4_sw_sync.c

This example configures ePWM3A for:

- Up count mode
- Period starts at 500 and goes up to 8000
- Toggle output on PRD

eCAP1, eCAP2, and eCAP3 are configured to capture the time between rising and falling edge of the ePWM3A output.

- eCAP1, eCAP2, eCAP3 are on GPIO16
- ePWM3A is on GPIO4
- Connect GPIO4 to GPIO16.

Watch Variables

- *ecapPassCount* - Successful captures.
- *ecap3IntCount* - Interrupt counts.

21.9 eCAP Registers

This section describes the Enhanced Capture Registers.

21.9.1 ECAP Base Address Table

Table 21-2. ECAP Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
ECap1Regs	ECAP_REGS	ECAP1_BASE	0x0000_5200	YES	YES	YES	YES	YES
ECap2Regs	ECAP_REGS	ECAP2_BASE	0x0000_5240	YES	YES	YES	YES	YES
ECap3Regs	ECAP_REGS	ECAP3_BASE	0x0000_5280	YES	YES	YES	YES	YES

21.9.2 ECAP_REGS Registers

Table 21-3 lists the memory-mapped registers for the ECAP_REGS registers. All register offset addresses not listed in Table 21-3 should be considered as reserved locations and the register contents should not be modified.

Table 21-3. ECAP_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	TSCTR	Time-Stamp Counter		Go
2h	CTRPHS	Counter Phase Offset Value Register		Go
4h	CAP1	Capture 1 Register		Go
6h	CAP2	Capture 2 Register		Go
8h	CAP3	Capture 3 Register		Go
Ah	CAP4	Capture 4 Register		Go
12h	ECCTL0	Capture Control Register 0	EALLOW	Go
14h	ECCTL1	Capture Control Register 1	EALLOW	Go
15h	ECCTL2	Capture Control Register 2	EALLOW	Go
16h	ECEINT	Capture Interrupt Enable Register	EALLOW	Go
17h	ECFLG	Capture Interrupt Flag Register		Go
18h	ECCLR	Capture Interrupt Clear Register		Go
19h	ECFRC	Capture Interrupt Force Register	EALLOW	Go
1Eh	ECAPSYNCINSEL	SYNC source select register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 21-4 shows the codes that are used for access types in this section.

Table 21-4. ECAP_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.

Table 21-4. ECAP_REGS Access Type Codes (continued)

Access Type	Code	Description
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

21.9.2.1 TSCTR Register (Offset = 0h) [Reset = 0h]

TSCTR is shown in Figure 21-18 and described in Table 21-5.

Return to the Summary Table.

Time-Stamp Counter

Figure 21-18. TSCTR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TSCTR																															
R/W-0h																															

Table 21-5. TSCTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TSCTR	R/W	0h	Active 32-bit counter register that is used as the capture time-base HR mode : 1) This register reads HRCOUNTER value and is not writable 2) can be reset using CTRFILTRESET 3) Its not synchronized to SYSCLK domain so reads may not be accurate Reset type: SYSRSn

21.9.2.2 CTRPHS Register (Offset = 2h) [Reset = 0h]

CTRPHS is shown in Figure 21-19 and described in Table 21-6.

Return to the Summary Table.

Counter Phase Offset Value Register

Figure 21-19. CTRPHS Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CTRPHS																															
R/W-0h																															

Table 21-6. CTRPHS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CTRPHS	R/W	0h	Counter phase value register that can be programmed for phase lag/lead. This register CTRPHS is loaded into TSCTR upon either a SYNCI event or S/W force via a control bit. Used to achieve phase control synchronization with respect to other eCAP and EPWM time-bases. This register is not applicable in HR mode. Reset type: SYSRSn

21.9.2.3 CAP1 Register (Offset = 4h) [Reset = 0h]

CAP1 is shown in Figure 21-20 and described in Table 21-7.

Return to the Summary Table.

Capture 1 Register

Figure 21-20. CAP1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAP1																															
R/W-0h																															

Table 21-7. CAP1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAP1	R/W	0h	This register can be loaded (written) by: - Time-Stamp counter value (TSCTR) during a capture event - Software - may be useful for test purposes or initialization - ARPD shadow register (CAP3) when used in APWM mode Reset type: SYSRSn

21.9.2.4 CAP2 Register (Offset = 6h) [Reset = 0h]

CAP2 is shown in Figure 21-21 and described in Table 21-8.

Return to the Summary Table.

Capture 2 Register

Figure 21-21. CAP2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAP2																															
R/W-0h																															

Table 21-8. CAP2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAP2	R/W	0h	This register can be loaded (written) by: - Time-Stamp (counter value) during a capture event - Software - may be useful for test purposes - ACMP shadow register (CAP4) when used in APWM mode Reset type: SYSRSn

21.9.2.5 CAP3 Register (Offset = 8h) [Reset = 0h]

CAP3 is shown in Figure 21-22 and described in Table 21-9.

Return to the Summary Table.

Capture 3 Register

Figure 21-22. CAP3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAP3																															
R/W-0h																															

Table 21-9. CAP3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAP3	R/W	0h	In CMP mode, this is a time-stamp capture register. In APWM mode, this is the period shadow (APRD) register. You can update the PWM period value through this register. CAP3 (APRD) shadows CAP1 in this mode. Reset type: SYSRSn

21.9.2.6 CAP4 Register (Offset = Ah) [Reset = 0h]

CAP4 is shown in Figure 21-23 and described in Table 21-10.

Return to the Summary Table.

Capture 4 Register

Figure 21-23. CAP4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CAP4																															
R/W-0h																															

Table 21-10. CAP4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	CAP4	R/W	0h	In CMP mode, this is a time-stamp capture register. In APWM mode, this is the compare shadow (ACMP) register. You can update the PWM compare value via this register. CAP4 (ACMP) shadows CAP2 in this mode. Reset type: SYSRSn

21.9.2.7 ECCTL0 Register (Offset = 12h) [Reset = 7Fh]

ECCTL0 is shown in Figure 21-24 and described in Table 21-11.

Return to the Summary Table.

Capture Control Register 0

Figure 21-24. ECCTL0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED									INPUTSEL						
R-0h									R/W-7Fh						

Table 21-11. ECCTL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reserved
6-0	INPUTSEL	R/W	7Fh	Capture input source select bits 0000000 capture input is ECAPxINPUT[0] 0000001 capture input is ECAPxINPUT[1] 0000010 capture input is ECAPxINPUT[2] ... 1111111 capture input is ECAPxINPUT[127] Reset type: CPU1.SYSRSn

21.9.2.8 ECCTL1 Register (Offset = 14h) [Reset = 0h]

ECCTL1 is shown in Figure 21-25 and described in Table 21-12.

Return to the Summary Table.

Capture Control Register 1

Figure 21-25. ECCTL1 Register

15	14	13	12	11	10	9	8
FREE_SOFT		PRESCALE					CAPLDEN
R/W-0h		R/W-0h					R/W-0h
7	6	5	4	3	2	1	0
CTRRST4	CAP4POL	CTRRST3	CAP3POL	CTRRST2	CAP2POL	CTRRST1	CAP1POL
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 21-12. ECCTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	FREE_SOFT	R/W	0h	Emulation Control Reset type: SYSRSn 0h (R/W) = TSCTR counter stops immediately on emulation suspend 1h (R/W) = TSCTR counter runs until = 0 2h (R/W) = TSCTR counter is unaffected by emulation suspend (Run Free) 3h (R/W) = TSCTR counter is unaffected by emulation suspend (Run Free)
13-9	PRESCALE	R/W	0h	Event Filter prescale select Reset type: SYSRSn 0h (R/W) = Divide by 1 (i.e,. no prescale, by-pass the prescaler) 1h (R/W) = Divide by 2 2h (R/W) = Divide by 4 3h (R/W) = Divide by 6 4h (R/W) = Divide by 8 5h (R/W) = Divide by 10 1Eh (R/W) = Divide by 60 1Fh (R/W) = Divide by 62
8	CAPLDEN	R/W	0h	Enable Loading of CAP1-4 registers on a capture event. Note that this bit does not disable CEVTn events from being generated. Reset type: SYSRSn 0h (R/W) = Disable CAP1-4 register loads at capture event time. 1h (R/W) = Enable CAP1-4 register loads at capture event time.
7	CTRRST4	R/W	0h	Counter Reset on Capture Event 4 Reset type: SYSRSn 0h (R/W) = Do not reset counter on Capture Event 4 (absolute time stamp operation) 1h (R/W) = Reset counter after Capture Event 4 time-stamp has been captured (used in difference mode operation)
6	CAP4POL	R/W	0h	Capture Event 4 Polarity select Reset type: SYSRSn 0h (R/W) = Capture Event 4 triggered on a rising edge (RE) 1h (R/W) = Capture Event 4 triggered on a falling edge (FE)
5	CTRRST3	R/W	0h	Counter Reset on Capture Event 3 Reset type: SYSRSn 0h (R/W) = Do not reset counter on Capture Event 3 (absolute time stamp) 1h (R/W) = Reset counter after Event 3 time-stamp has been captured (used in difference mode operation)
4	CAP3POL	R/W	0h	Capture Event 3 Polarity select Reset type: SYSRSn 0h (R/W) = Capture Event 3 triggered on a rising edge (RE) 1h (R/W) = Capture Event 3 triggered on a falling edge (FE)

Table 21-12. ECCTL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	CTRRST2	R/W	0h	Counter Reset on Capture Event 2 Reset type: SYSRSn 0h (R/W) = Do not reset counter on Capture Event 2 (absolute time stamp) 1h (R/W) = Reset counter after Event 2 time-stamp has been captured (used in difference mode operation)
2	CAP2POL	R/W	0h	Capture Event 2 Polarity select Reset type: SYSRSn 0h (R/W) = Capture Event 2 triggered on a rising edge (RE) 1h (R/W) = Capture Event 2 triggered on a falling edge (FE)
1	CTRRST1	R/W	0h	Counter Reset on Capture Event 1 Reset type: SYSRSn 0h (R/W) = Do not reset counter on Capture Event 1 (absolute time stamp) 1h (R/W) = Reset counter after Event 1 time-stamp has been captured (used in difference mode operation)
0	CAP1POL	R/W	0h	Capture Event 1 Polarity select Reset type: SYSRSn 0h (R/W) = Capture Event 1 triggered on a rising edge (RE) 1h (R/W) = Capture Event 1 triggered on a falling edge (FE)

21.9.2.9 ECCTL2 Register (Offset = 15h) [Reset = 6h]

ECCTL2 is shown in Figure 21-26 and described in Table 21-13.

Return to the Summary Table.

Capture Control Register 2

Figure 21-26. ECCTL2 Register

15	14	13	12	11	10	9	8
MODCNTRSTS		DMAEVTSEL		CTRFILTRESET	APWMPOL	CAP_APWM	SWSYNC
R-0h		R/W-0h		R-0/W1C-0h	R/W-0h	R/W-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
SYNCO_SEL		SYNCI_EN	TSCTRSTOP	REARM	STOP_WRAP		CONT_ONESHT
R/W-0h		R/W-0h	R/W-0h	R-0/W1S-0h	R/W-3h		R/W-0h

Table 21-13. ECCTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	MODCNTRSTS	R	0h	This bit field reads current status on modulo counter 00b (R) = CAP1 register gets loaded on next capture event. 01b (R) = CAP2 register gets loaded on next capture event. 10b (R) = CAP3 register gets loaded on next capture event. 11b (R) = CAP4 register gets loaded on next capture event. Reset type: CPU1.SYSRSn
13-12	DMAEVTSEL	R/W	0h	DMA event select 00b (R/W) = DMA interrupt source is CEVT1 01b (R/W) = DMA interrupt source is CEVT2 10b (R/W) = DMA interrupt source is CEVT3 11b (R/W) = DMA interrupt source is CEVT4 Note: ECCTL1.CAPLDEN also needs to be set to "1" for ECAPxDMA_INT to be generated Reset type: CPU1.SYSRSn
11	CTRFILTRESET	R-0/W1C	0h	Reset Bit 0h (R) = No effect 1h (W) = Resets event filter, counter, modulo counter and CEVT[1,2,3,4] and CNTOVF , HRERROR flags Note: This provides an ability start capture module from known state in case spurious inputs are captured while ECAP is configured. Reset type: CPU1.SYSRSn
10	APWMPOL	R/W	0h	APWM output polarity select. This is applicable only in APWM operating mode. Reset type: SYSRSn 0h (R/W) = Output is active high (Compare value defines high time) 1h (R/W) = Output is active low (Compare value defines low time)
9	CAP_APWM	R/W	0h	CAP/APWM operating mode select Reset type: SYSRSn 0h (R/W) = ECAP module operates in capture mode. This mode forces the following configuration: - Inhibits TSCTR resets via CTR = PRD event - Inhibits shadow loads on CAP1 and 2 registers - Permits user to enable CAP1-4 register load - CAPx/APWMx pin operates as a capture input 1h (R/W) = ECAP module operates in APWM mode. This mode forces the following configuration: - Resets TSCTR on CTR = PRD event (period boundary - Permits shadow loading on CAP1 and 2 registers - Disables loading of time-stamps into CAP1-4 registers - CAPx/APWMx pin operates as a APWM output

Table 21-13. ECCTL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	SWSYNC	R-0/W1S	0h	Software-forced Counter (TSCTR) Synchronizer. This provides the user a method to generate a synchronization pulse through software. In APWM mode, the synchronization pulse can also be sourced from the CTR = PRD event. Reset type: SYSRSn 0h (R/W) = Writing a zero has no effect. Reading always returns a zero 1h (R/W) = Writing a one forces a TSCTR shadow load of current ECAP module and any ECAP modules down-stream providing the SYNCO_SEL bits are 0,0. After writing a 1, this bit returns to a zero. Note: Selection CTR = PRD is meaningful only in APWM mode however, you can choose it in CAP mode if you find doing so useful.
7-6	SYNCO_SEL	R/W	0h	Sync-Out Select Reset type: SYSRSn 0h (R/W) = sync out signal is SWSYNC 1h (R/W) = Select CTR = PRD event to be the sync-out signal 2h (R/W) = Disable sync out signal 3h (R/W) = Disable sync out signal
5	SYNCI_EN	R/W	0h	Counter (TSCTR) Sync-In select mode Reset type: SYSRSn 0h (R/W) = Disable sync-in option 1h (R/W) = Enable counter (TSCTR) to be loaded from CTRPHS register upon either a SYNCI signal or a S/W force event.
4	TSCTRSTOP	R/W	0h	Time Stamp (TSCTR) Counter Stop (freeze) Control Reset type: SYSRSn 0h (R/W) = TSCTR stopped 1h (R/W) = TSCTR free-running
3	REARM	R-0/W1S	0h	Re-Arming Control. Note: The re-arm function is valid in one shot or continuous mode Reset type: SYSRSn 0h (R/W) = Has no effect (reading always returns a 0) 1h (R/W) = Arms the one-shot sequence as follows: 1) Resets the Mod4 counter to zero 2) Unfreezes the Mod4 counter 3) Enables capture register loads
2-1	STOP_WRAP	R/W	3h	Stop value for one-shot mode. This is the number (between 1-4) of captures allowed to occur before the CAP(1-4) registers are frozen, that is, capture sequence is stopped. Wrap value for continuous mode. This is the number (between 1-4) of the capture register in which the circular buffer wraps around and starts again. Notes: STOP_WRAP is compared to Mod4 counter and, when equal, 2 actions occur: - Mod4 counter is stopped (frozen) - Capture register loads are inhibited In one-shot mode, further interrupt events are blocked until re-armed. Reset type: SYSRSn 0h (R/W) = Stop after Capture Event 1 in one-shot mode Wrap after Capture Event 1 in continuous mode. 1h (R/W) = Stop after Capture Event 2 in one-shot mode Wrap after Capture Event 2 in continuous mode. 2h (R/W) = Stop after Capture Event 3 in one-shot mode Wrap after Capture Event 3 in continuous mode. 3h (R/W) = Stop after Capture Event 4 in one-shot mode Wrap after Capture Event 4 in continuous mode.
0	CONT_ONESHT	R/W	0h	Continuous or one-shot mode control (applicable only in capture mode) Reset type: SYSRSn 0h (R/W) = Operate in continuous mode 1h (R/W) = Operate in one-Shot mode

21.9.2.10 ECEINT Register (Offset = 16h) [Reset = 0h]

ECEINT is shown in Figure 21-27 and described in Table 21-14.

Return to the Summary Table.

The interrupt enable bits (CEVT1, ...) block any of the selected events from generating an interrupt. Events will still be latched into the flag bit (ECFLG register) and can be forced/cleared via the ECFRC/ECCLR registers.
The proper procedure for configuring peripheral modes and interrupts is as follows:
- Disable global interrupts
- Stop eCAP counter
- Disable eCAP interrupts
- Configure peripheral registers
- Clear spurious eCAP interrupt flags
- Enable eCAP interrupts
- Start eCAP counter
- Enable global interrupts

Figure 21-27. ECEINT Register

15	14	13	12	11	10	9	8
RESERVED							RESERVED
R-0h							R/W-0h
7	6	5	4	3	2	1	0
CTR_EQ_CMP	CTR_EQ_PRD	CTROVF	CEVT4	CEVT3	CEVT2	CEVT1	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h

Table 21-14. ECEINT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-9	RESERVED	R	0h	Reserved
8	RESERVED	R/W	0h	Reserved
7	CTR_EQ_CMP	R/W	0h	Counter Equal Compare Interrupt Enable Reset type: SYSRSn 0h (R/W) = Disable Compare Equal as an Interrupt source 1h (R/W) = Enable Compare Equal as an Interrupt source
6	CTR_EQ_PRD	R/W	0h	Counter Equal Period Interrupt Enable Reset type: SYSRSn 0h (R/W) = Disable Period Equal as an Interrupt source 1h (R/W) = Enable Period Equal as an Interrupt source
5	CTROVF	R/W	0h	Counter Overflow Interrupt Enable Reset type: SYSRSn 0h (R/W) = Disabled counter Overflow as an Interrupt source 1h (R/W) = Enable counter Overflow as an Interrupt source
4	CEVT4	R/W	0h	Capture Event 4 Interrupt Enable Reset type: SYSRSn 0h (R/W) = Disable Capture Event 4 as an Interrupt source 1h (R/W) = Capture Event 4 Interrupt Enable
3	CEVT3	R/W	0h	Capture Event 3 Interrupt Enable Reset type: SYSRSn 0h (R/W) = Disable Capture Event 3 as an Interrupt source 1h (R/W) = Enable Capture Event 3 as an Interrupt source
2	CEVT2	R/W	0h	Capture Event 2 Interrupt Enable Reset type: SYSRSn 0h (R/W) = Disable Capture Event 2 as an Interrupt source 1h (R/W) = Enable Capture Event 2 as an Interrupt source
1	CEVT1	R/W	0h	Capture Event 1 Interrupt Enable Reset type: SYSRSn 0h (R/W) = Disable Capture Event 1 as an Interrupt source 1h (R/W) = Enable Capture Event 1 as an Interrupt source

Table 21-14. ECEINT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	RESERVED	R	0h	Reserved

21.9.2.11 ECFLG Register (Offset = 17h) [Reset = 0h]

ECFLG is shown in Figure 21-28 and described in Table 21-15.

Return to the Summary Table.

Capture Interrupt Flag Register

Figure 21-28. ECFLG Register

15	14	13	12	11	10	9	8
RESERVED							RESERVED
R-0h							R-0h
7	6	5	4	3	2	1	0
CTR_CMP	CTR_PRD	CTROVF	CEVT4	CEVT3	CEVT2	CEVT1	INT
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 21-15. ECFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-9	RESERVED	R	0h	Reserved
8	RESERVED	R	0h	Reserved
7	CTR_CMP	R	0h	Compare Equal Compare Status Flag. This flag is active only in APWM mode. Reset type: SYSRSn 0h (R/W) = Indicates no event occurred 1h (R/W) = Indicates the counter (TSCTR) reached the compare register value (ACMP)
6	CTR_PRD	R	0h	Counter Equal Period Status Flag. This flag is only active in APWM mode. Reset type: SYSRSn 0h (R/W) = Indicates no event occurred 1h (R/W) = Indicates the counter (TSCTR) reached the period register value (APRD) and was reset.
5	CTROVF	R	0h	Counter Overflow Status Flag. This flag is active in CAP and APWM mode. Reset type: SYSRSn 0h (R/W) = Indicates no event occurred 1h (R/W) = Indicates the counter (TSCTR) has made the transition from FFFFFFFF to 00000000
4	CEVT4	R	0h	Capture Event 4 Status Flag This flag is only active in CAP mode. Reset type: SYSRSn 0h (R/W) = Indicates no event occurred 1h (R/W) = Indicates the fourth event occurred at ECAPx pin
3	CEVT3	R	0h	Capture Event 3 Status Flag. This flag is active only in CAP mode. Reset type: SYSRSn 0h (R/W) = Indicates no event occurred 1h (R/W) = Indicates the third event occurred at ECAPx pin.
2	CEVT2	R	0h	Capture Event 2 Status Flag. This flag is only active in CAP mode. Reset type: SYSRSn 0h (R/W) = Indicates no event occurred 1h (R/W) = Indicates the second event occurred at ECAPx pin.
1	CEVT1	R	0h	Capture Event 1 Status Flag. This flag is only active in CAP mode. Reset type: SYSRSn 0h (R/W) = Indicates no event occurred 1h (R/W) = Indicates the first event occurred at ECAPx pin.
0	INT	R	0h	Global Interrupt Status Flag Reset type: SYSRSn 0h (R/W) = Indicates no event occurred 1h (R/W) = Indicates that an interrupt was generated.

21.9.2.12 ECCLR Register (Offset = 18h) [Reset = 0h]

ECCLR is shown in Figure 21-29 and described in Table 21-16.

Return to the Summary Table.

Capture Interrupt Clear Register

Figure 21-29. ECCLR Register

15	14	13	12	11	10	9	8
RESERVED							RESERVED
R-0h							R-0/W1C-0h
7	6	5	4	3	2	1	0
CTR_CMP	CTR_PRD	CTROVF	CEVT4	CEVT3	CEVT2	CEVT1	INT
R-0/W1C-0h	R-0/W1C-0h	R-0/W1C-0h	R-0/W1C-0h	R-0/W1C-0h	R-0/W1C-0h	R-0/W1C-0h	R-0/W1C-0h

Table 21-16. ECCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-9	RESERVED	R	0h	Reserved
8	RESERVED	R-0/W1C	0h	Reserved
7	CTR_CMP	R-0/W1C	0h	Counter Equal Compare Status Clear Reset type: SYSRSn 0h (R/W) = Writing a 0 has no effect. Always reads back a 0 1h (R/W) = Writing a 1 clears the CTR=CMP flag.
6	CTR_PRD	R-0/W1C	0h	Counter Equal Period Status Clear Reset type: SYSRSn 0h (R/W) = Writing a 0 has no effect. Always reads back a 0 1h (R/W) = Writing a 1 clears the CTR=PRD flag.
5	CTROVF	R-0/W1C	0h	Counter Overflow Status Clear Reset type: SYSRSn 0h (R/W) = Writing a 0 has no effect. Always reads back a 0 1h (R/W) = Writing a 1 clears the CTROVF flag.
4	CEVT4	R-0/W1C	0h	Capture Event 4 Status Clear Reset type: SYSRSn 0h (R/W) = Writing a 0 has no effect. Always reads back a 0 1h (R/W) = Writing a 1 clears the CEVT4 flag.
3	CEVT3	R-0/W1C	0h	Capture Event 3 Status Clear Reset type: SYSRSn 0h (R/W) = Writing a 0 has no effect. Always reads back a 0 1h (R/W) = Writing a 1 clears the CEVT3 flag.
2	CEVT2	R-0/W1C	0h	Capture Event 2 Status Clear Reset type: SYSRSn 0h (R/W) = Writing a 0 has no effect. Always reads back a 0 1h (R/W) = Writing a 1 clears the CEVT2 flag.
1	CEVT1	R-0/W1C	0h	Capture Event 1 Status Clear Reset type: SYSRSn 0h (R/W) = Writing a 0 has no effect. Always reads back a 0 1h (R/W) = Writing a 1 clears the CEVT1 flag.
0	INT	R-0/W1C	0h	ECAP Global Interrupt Status Clear Reset type: SYSRSn 0h (R/W) = Writing a 0 has no effect. Always reads back a 0 1h (R/W) = Writing a 1 clears the INT flag and enable further interrupts to be generated if any of the event flags are set to 1

21.9.2.13 ECFRC Register (Offset = 19h) [Reset = 0h]

ECFRC is shown in Figure 21-30 and described in Table 21-17.

Return to the Summary Table.

Capture Interrupt Force Register

Figure 21-30. ECFRC Register

15	14	13	12	11	10	9	8
RESERVED							RESERVED
R-0h							R-0/W1S-0h
7	6	5	4	3	2	1	0
CTR_CMP	CTR_PRD	CTROVF	CEVT4	CEVT3	CEVT2	CEVT1	RESERVED
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0h

Table 21-17. ECFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-9	RESERVED	R	0h	Reserved
8	RESERVED	R-0/W1S	0h	Reserved
7	CTR_CMP	R-0/W1S	0h	Force Counter Equal Compare Interrupt. This event is only active in APWM mode. Reset type: SYSRSn 0h (R/W) = No effect. Always reads back a 0. 1h (R/W) = Writing a 1 sets the CTR_CMP flag.
6	CTR_PRD	R-0/W1S	0h	Force Counter Equal Period Interrupt. This event is only active in APWM mode. Reset type: SYSRSn 0h (R/W) = No effect. Always reads back a 0. 1h (R/W) = Writing a 1 sets the CTR_PRD flag.
5	CTROVF	R-0/W1S	0h	Force Counter Overflow Reset type: SYSRSn 0h (R/W) = No effect. Always reads back a 0. 1h (R/W) = Writing a 1 to this bit sets the CTROVF flag.
4	CEVT4	R-0/W1S	0h	Force Capture Event 4. This event is only active in CAP mode. Reset type: SYSRSn 0h (R/W) = No effect. Always reads back a 0. 1h (R/W) = Writing a 1 sets the CEVT4 flag.
3	CEVT3	R-0/W1S	0h	Force Capture Event 3. This event is only active in CAP mode. Reset type: SYSRSn 0h (R/W) = No effect. Always reads back a 0. 1h (R/W) = Writing a 1 sets the CEVT3 flag.
2	CEVT2	R-0/W1S	0h	Force Capture Event 2. This event is only active in CAP mode. Reset type: SYSRSn 0h (R/W) = No effect. Always reads back a 0. 1h (R/W) = Writing a 1 sets the CEVT2 flag.
1	CEVT1	R-0/W1S	0h	Force Capture Event 1. This event is only active in CAP mode. Reset type: SYSRSn 0h (R/W) = No effect. Always reads back a 0. 1h (R/W) = Sets the CEVT1 flag.
0	RESERVED	R	0h	Reserved

21.9.2.14 ECAPSYNCINSEL Register (Offset = 1Eh) [Reset = 1h]

ECAPSYNCINSEL is shown in Figure 21-31 and described in Table 21-18.

Return to the Summary Table.

SYNC source select register

Figure 21-31. ECAPSYNCINSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																											SEL				
R-0h																											R/W-1h				

Table 21-18. ECAPSYNCINSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reserved
4-0	SEL	R/W	1h	These bits determines the source of SYNCIN signal. Reset type: SYSRSn 0h (R/W) = Disable Syncin to eCAP 1h (R/W) = EPWM1SYNCOUT 2h (R/W) = EPWM2SYNCOUT 3h (R/W) = EPWM3SYNCOUT 4h (R/W) = EPWM4SYNCOUT 5h (R/W) = EPWM5SYNCOUT 6h (R/W) = EPWM6SYNCOUT 7h (R/W) = EPWM7SYNCOUT 8h (R/W) = EPWM8SYNCOUT 9h (R/W) = RSVD Ah (R/W) = RSVD Bh (R/W) = RSVD Ch (R/W) = RSVD Dh (R/W) = RSVD Eh (R/W) = RSVD Fh (R/W) = RSVD 10h (R/W) = RSVD 11h (R/W) = ECAP1SYNCOUT 12h (R/W) = ECAP2SYNCOUT 13h (R/W) = ECAP3SYNCOUT 14h (R/W) = RSVD 15h (R/W) = RSVD 16h (R/W) = RSVD 17h (R/W) = RSVD 18h (R/W) = INPUTXBAROUT5 19h (R/W) = INPUTXBAROUT6 1Ah (R/W) = RSVD 1Bh (R/W) = RSVD 1Ch (R/W) = RSVD 1Dh (R/W) = RSVD 1Eh (R/W) = RSVD 1Fh (R/W) = FSI_RXA_RX_TRIG1

21.9.3 ECAP Registers to Driverlib Functions

Table 21-19. ECAP Registers to Driverlib Functions

File	Driverlib Function
TSCTR	
ecap.h	ECAP_getTimeBaseCounter
CTRPHS	
ecap.h	ECAP_setPhaseShiftCount
CAP1	
ecap.h	ECAP_setAPWMPeriod

Table 21-19. ECAP Registers to Driverlib Functions (continued)

File	Driverlib Function
ecap.h	ECAP_getEventTimeStamp
CAP2	
ecap.h	ECAP_setAPWMCompare
ecap.h	ECAP_getEventTimeStamp
CAP3	
ecap.h	ECAP_setAPWMShadowPeriod
ecap.h	ECAP_getEventTimeStamp
CAP4	
ecap.h	ECAP_setAPWMShadowCompare
ecap.h	ECAP_getEventTimeStamp
ECCTL0	
ecap.h	ECAP_selectECAPInput
ECCTL1	
ecap.c	ECAP_setEmulationMode
ecap.h	ECAP_setEventPrescaler
ecap.h	ECAP_setEventPolarity
ecap.h	ECAP_enableCounterResetOnEvent
ecap.h	ECAP_disableCounterResetOnEvent
ecap.h	ECAP_enableTimeStampCapture
ecap.h	ECAP_disableTimeStampCapture
ECCTL2	
ecap.h	ECAP_setCaptureMode
ecap.h	ECAP_reArm
ecap.h	ECAP_enableCaptureMode
ecap.h	ECAP_enableAPWMMode
ecap.h	ECAP_enableLoadCounter
ecap.h	ECAP_disableLoadCounter
ecap.h	ECAP_loadCounter
ecap.h	ECAP_setSyncOutMode
ecap.h	ECAP_stopCounter
ecap.h	ECAP_startCounter
ecap.h	ECAP_setAPWMPolarity
ecap.h	ECAP_resetCounters
ecap.h	ECAP_setDMASource
ecap.h	ECAP_getModuloCounterStatus
ECEINT	
ecap.h	ECAP_enableInterrupt
ecap.h	ECAP_disableInterrupt
ECFLG	
ecap.h	ECAP_getInterruptSource
ecap.h	ECAP_getGlobalInterruptStatus
ECCLR	
ecap.h	ECAP_clearInterrupt
ecap.h	ECAP_clearGlobalInterrupt
ECFRC	

Table 21-19. ECAP Registers to Driverlib Functions (continued)

File	Driverlib Function
ecap.h	ECAP_forceInterrupt
SYNCINSEL	
ecap.h	ECAP_setSyncInPulseSource

This page intentionally left blank.

Chapter 22

High Resolution Capture (HRCAP)

This chapter describes the operation of the high resolution capture (HRCAP) module. The HRCAP submodule described here is part of the Type1 eCAP. HRCAP measures the width of external pulses to a higher degree of accuracy than the eCAP module. See the *C2000 Real-Time Control Peripheral Reference Guide* for a list of all devices with an HRCAP module of the same type, to determine the differences between types, and for a list of device-specific differences within a type. A detailed description of all referenced functions can be found in the C2000Ware documentation.

22.1 Introduction

Uses for the HRCAP module include:

- Capacitive touch applications
- High-resolution period and duty cycle measurements of pulse train cycles
- Instantaneous speed measurements
- Instantaneous frequency measurements
- Voltage measurements across an isolation boundary
- Distance/sonar measurement and scanning
- Measuring flow

22.1.1 HRCAP Related Collateral

Foundational Materials

- C2000 Academy - Control Peripherals

Getting Started Materials

- *Leveraging High Resolution Capture (HRCAP) for Single Wire Data Transfer Application Report*

22.1.2 Features

The HRCAP module includes the following features:

- Pulse-width capture in either non-high-resolution or high-resolution modes
- Absolute mode pulse-width capture
- Continuous or one-shot capture
- Interrupt on either falling or rising edge
- Continuous mode capture of pulse widths in 4-deep buffer
- Hardware calibration logic for precision high-resolution capture

All of the previous resources are available on any pin using the Input X-BAR.

22.1.3 Description

Improvements from the Type 0 HRCAP are:

- Simplified calibration scheme:
  - HRCAP is always functional; never offline to perform calibration
  - Calibration is always running in the background; drastically reduced software overhead to calibrate
- Reduced software overhead to compute fractional bits
- Fractional and integer portions are packed into 32 bits
- All eCAP hardware is accessible when using the HRCAP enhancements. See Section 22.3 for practical considerations.
- Usage of the HRCAP is now unified with the eCAP

The HRCAP enhancement has been added to eCAP 3 to allow signals to be captured asynchronously to SYSCLK. Each HRCAP submodule includes one capture channel in addition to a hardware calibration block. All eCAP hardware is accessible when using the HRCAP enhancements; however, using the Event Filter or the Input Qualifier is not valid, as these are synchronous to SYSCLK.

Each HRCAP-capable channel has the following independent key resources:

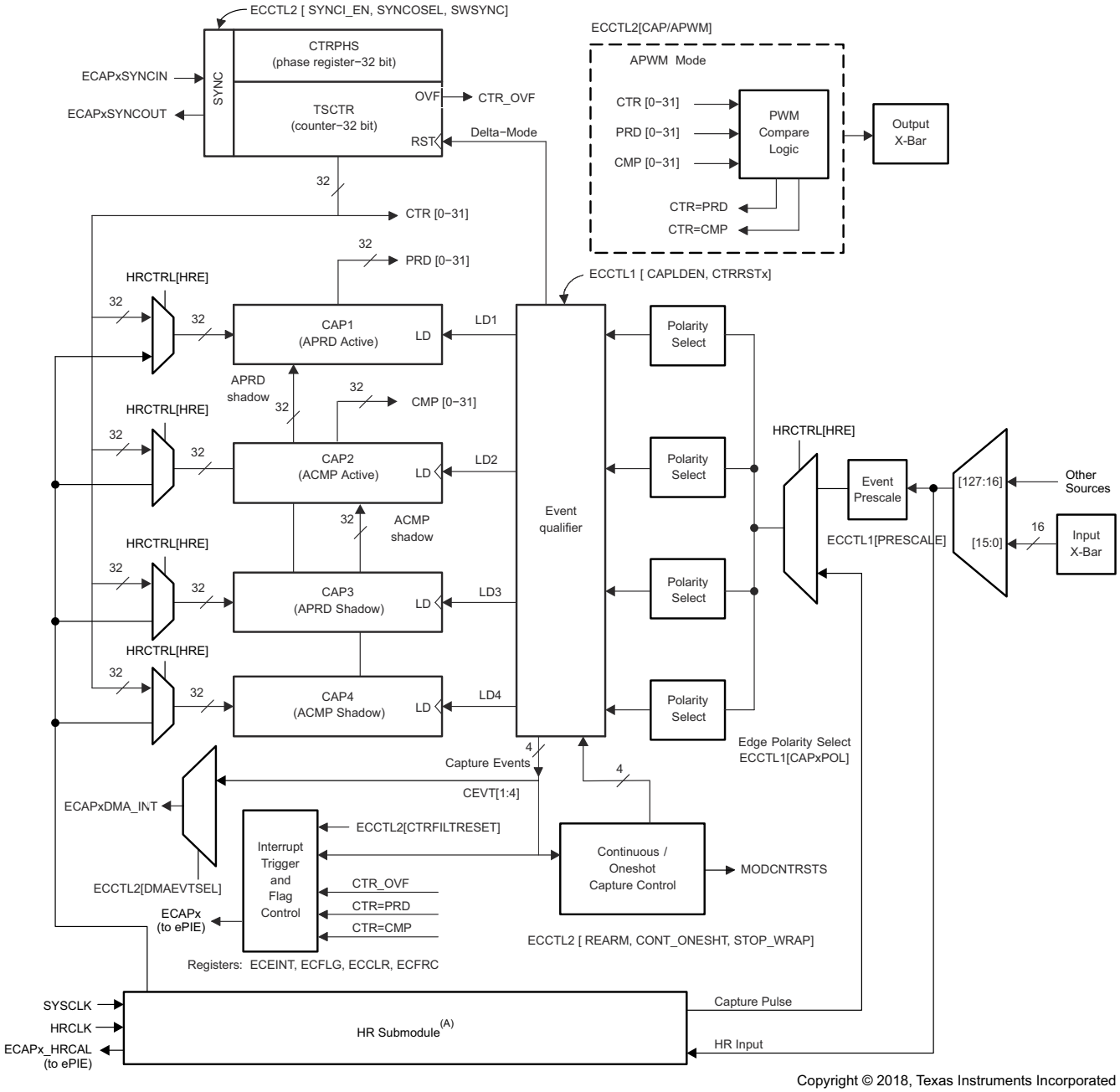
- All hardware of the respective eCAP
- High-resolution calibration logic
- Dedicated calibration interrupt

22.2 Operational Details

Figure 22-1 shows the various components that implement the high-resolution capture functionality of the eCAP module. Note that existing eCAP resources are reused, which requires that the eCAP module is set up before

using the HRCAP enhancements. For simplicity, absolute timestamp measurements are recommended. See Section 22.3 for more details.

All HRCAP measurements are relative-time measures, in terms of minimum step size. Calibration hardware as well as software functions, have been provided to convert relative-time measurements to time-converted measurements in terms of seconds. The calibration hardware and software is only required if time-converted measurements are required.



A. The HRCAP submodule is not available on all eCAP modules; in this case, the high-resolution muxes and hardware are not implemented.

Figure 22-1. HRCAP Operations Block Diagram

22.2.1 HRCAP Clocking

Unlike previous Type-0 HRCAP modules, the Type-1 eCAP, with HRCAP functionality, does not require a second PLL. However, the module still requires both SYSCLK and a second asynchronous clock source called HRCLK. The HRCLK is sensitive to changes in both temperature and voltage. For this reason, when using time-converted measurements, it is required to make periodic continuous calibrations.

22.2.2 HRCAP Initialization Sequence

Following are the HRCAP initialization sequence steps. When using the HRCAP to take relative-time measurements, only steps 1-5 are required. When using the HRCAP to take time-converted measurements, all steps are required.

1. Enable HRCLK using HRCAP_enableHighResolutionClock()
2. Delay 1 µS
3. Enable HR mode using HRCAP_enableHighResolution()
4. Delay 1 µS
5. Configure the eCAP module as desired, including interrupts
6. Set calibration period using HRCAP_setCalibrationPeriod()
7. Enable continuous calibration using HRCAP_setCalibrationMode()
8. Enable interrupts using HRCAP_enableCalibrationInterrupt()
9. Start calibration using HRCAP_startCalibration()

22.2.3 HRCAP Interrupts

The HRCAP enhancements leverage the existing eCAP interrupts (see *Interrupt Control* in the ECAP chapter) in addition to HRCALINT, which is used exclusively by the hardware calibration block. HRCALINT can be triggered by the following conditions:

1. SYSCLKCTR = HRCALIBPERIOD
2. SYSCLKCTR or HRCLKCTR experience an overflow condition

Figure 22-2 shows this logic.

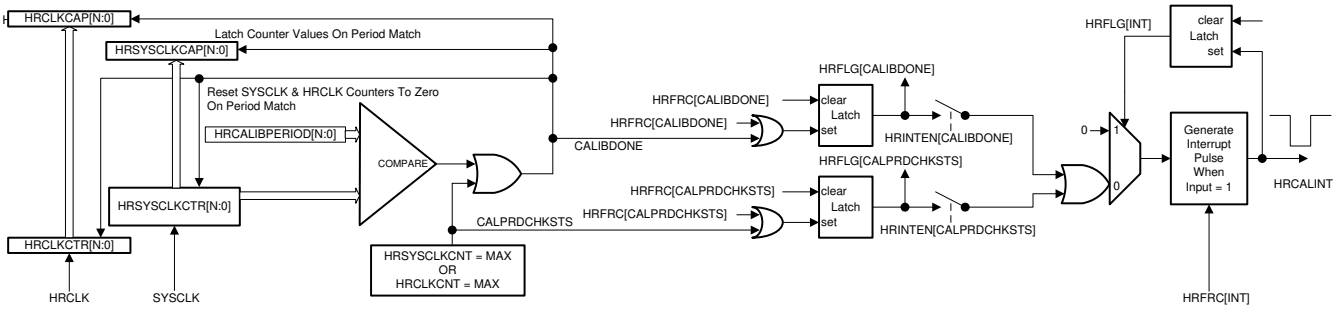


Figure 22-2. HRCAP Calibration

22.2.4 HRCAP Calibration

Note

For HRCAP calibration to work, the system needs to operate at SYSCLK frequency > 100 MHz.

The following section applies only to time-converted measurements; calibration for relative-time measurements is not required. All values captured by the HRCAP submodule are in number of HRCLK cycles. The HRCLK speed varies widely with temperature and voltage, thus a scale factor is required to convert the capture value to the SYSCLK domain. For the same reason, it is required to periodically recalculate the scale factor. The HRCAP submodule has a calibration block to reduce software overhead when calculating a scale factor between HRCLK and SYSCLK.

The calibration block contains the following key resources:

- **HRSYSCLKCNT:** A 32-bit counter connected to SYSCLK. The counter will start counting when CALIBSTART is set.
- **HRCLKCNT:** A 32-bit counter connected to HRCLK. The counter will start counting when CALIBSTART is set.
- **HRCALIBPERIOD:** Calibration period, calibration is stopped when HRSYSCLKCNT is equal to the value in this register.
- **HRSYSCLKCAP:** On a calibration period match, the value of HRSYSCLKCNT is captured into HRSYSCLKCAP.
- **HRCLKCAP:** On a calibration period match, the value of HRCLKCNT is captured into HRCLKCAP.
- **HRCALINT:** An interrupt that occurs on a calibration period match, or when one of the counter registers experiences an overflow condition.

The calibration logic consists of two free-running counters; one clocked by HRCLK(HRCLKCTR) and the other clocked by SYSCLK(HRSYSCLKCTR). When HRSYSCLKCTR is equal to HRCALIBPERIOD, the calibration block will capture and reset both counter values, then trigger an interrupt, indicating a new scale factor is ready to be calculated. The scale factor can be found by dividing HRSYSCLKCAP by HRCLKCAP (see Equation 17). A DriverLib function, HRCAP_getScaleFactor, has been provided to determine the scale factor. This function should be called inside of the calibration interrupt service routine. If one of the counters experiences overflow, the CALPRDCHKSTS flag will be set. The full details of the calibration block are described in Figure 22-2.

$$ScaleFactor = \frac{HRSYSCLKCAP}{HRCLKCAP} \tag{17}$$

Note

- Even with calibration, noise on the 1.2V VDD supply will negatively affect the standard deviation of the HRCAP submodule. Care should be taken to ensure that the 1.2V supply is clean, and that noisy internal events such as enabling and disabling clock trees have been minimized while using the HRCAP submodule.
- When HRCLK > SYSCLK, calibration stops immaturely, in order to mitigate improper calibration, SYSCLK must be set to at least 100 MHz.

22.2.4.1 Applying the Scale Factor

A DriverLib function has been provided to apply the scale factor to a capture value, HRCAP_getEventTimeStampNanoseconds(). Equation 2 shows how to convert a raw count to seconds without using the DriverLib function.

$$Measurement(ns) = \frac{RawCount \times scaleFactor}{128} * SysClkPrd(ns) \quad (18)$$

Table 22-1. Scale Factor

Parameter	Explanation
RawCount	Capture value as read from ECAP_REGS_CAP1-4
ScaleFactor(1)	The Scale factor as calculated from Equation 2
128	Constant determined by the hardware of the HRCAP submodule
SysClkPrd(nS)	Period of the system clock
Measurement(nS)	Signal converted to nS

(1) The scale factor is not automatically applied to captured values. You are required to apply the scale factor to all captured values as shown in Equation 2.

22.3 Known Exceptions

In HRCAP mode:

- Enabling and disabling core clocks negatively affects the standard deviation of the HRCAP submodule. Do not enable or disable core clocks while taking measurements.
- TSCTR is not writable; however, it can be reset using ECCTL2[CTRFILTRESET]
- Input synchronization is not applicable when using the HRCAP enhancements, because the HRCAP submodule is asychronous to SYSCLK.
- The Event Filter functionality is not applicable for HRCAP, which defeats the purpose of HRCAP as the Event Filter's output is synchronous to SYSCLK.
- The best practice is to use absolute time mode for high resolution mode. If time difference mode is used, it may lead to inaccurate results if the fractional value is not taken into consideration for capture events that have reset the time base counter.
  - Actual Capture Value = (Capture Value) – (fractional value of reference event that reset the counter)
- For high-frequency input signals, the CPU may not be able cope with the speed of the captures. In such a case, One-Shot mode is recommended. This mode allows the device to capture up to four edges before waiting to be serviced when the CPU is ready. This is applicable for the eCAP as well; however in that case the event filter can be used to reduce the rate of captures.

22.4 Software

22.4.1 HRCAP Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/f28003x/examples/hrcap

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

22.4.1.1 HRCAP Capture and Calibration Example

FILE: hrcap_ex1_capture.c

This example configures ECAP3 to use HRCAP functionality to capture time between edges on input GPIO2.

External Connections
The user must provide a signal to GPIO2. XCLKOUT has been configured to GPIO16 and can be externally jumped to serve this purpose.

Watch Variables
- onTime1, onTime2
- offTime1, offTime2
- period1, period2

22.5 HRCAP Registers

This section describes the High-Resolution Capture Registers.

22.5.1 HRCAP Base Address Table

Table 22-2. HRCAP Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
HRCap3Regs	HRCAP_REGS	HRCAP3_BASE	0x0000_52A0	YES	YES	YES	YES	YES

22.5.2 HRCAP_REGS Registers

Table 22-3 lists the memory-mapped registers for the HRCAP_REGS registers. All register offset addresses not listed in Table 22-3 should be considered as reserved locations and the register contents should not be modified.

Table 22-3. HRCAP_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	HRCTL	High-Res Control Register	EALLOW	Go
4h	HRINTEN	High-Res Calibration Interrupt Enable Register	EALLOW	Go
6h	HRFLG	High-Res Calibration Interrupt Flag Register		Go
8h	HRCLR	High-Res Calibration Interrupt Clear Register		Go
Ah	HRFRC	High-Res Calibration Interrupt Force Register	EALLOW	Go
Ch	HRCALPRD	High-Res Calibration Period Register	EALLOW	Go
Eh	HRSYSCLKCTR	High-Res Calibration SYSCLK Counter Register		Go
10h	HRSYSCLKCAP	High-Res Calibration SYSCLK Capture Register		Go
12h	HRCLKCTR	High-Res Calibration HRCLK Counter Register		Go
14h	HRCLKCAP	High-Res Calibration HRCLK Capture Register		Go

Complex bit access types are encoded to fit into small table cells. Table 22-4 shows the codes that are used for access types in this section.

Table 22-4. HRCAP_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

22.5.2.1 HRCTL Register (Offset = 0h) [Reset = 0h]

HRCTL is shown in Figure 22-3 and described in Table 22-5.

Return to the Summary Table.

High-Res Control Register

Figure 22-3. HRCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		CALIBCONT	CALIBSTS	CALIBSTART	PRDSEL	HRCLKE	HRE
R-0h		R/W-0h	R-0h	R-0/W1S-0h	R/W-0h	R/W-0h	R/W-0h

Table 22-5. HRCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	CALIBCONT	R/W	0h	Continuous mode Calibration Select Bit: 0 Continuous mode disabled. 1 Continuous mode enabled. Calibration automatically restarts at end of current calibration cycle. Reset type: CPU1.SYSRSn
4	CALIBSTS	R	0h	Calibration status Bit: 0 No active calibration cycle 1 Calibration cycle in progress Reset type: CPU1.SYSRSn
3	CALIBSTART	R-0/W1S	0h	Calibration start Bit: 0 No effect 1 Starts the calibration cycle Reset type: CPU1.SYSRSn
2	PRDSEL	R/W	0h	Calibration Period Match Select Bit: 0 Use SYSCLK Counter For Period Match (default at reset) 1 Reserved Reset type: CPU1.SYSRSn
1	HRCLKE	R/W	0h	High Resolution Clock Enable Bit: 0 High resolution clock disabled (default at reset) 1 High resolution clock enabled. The clock should be enabled before enabling the high res function via the HRE bit. Reset type: CPU1.SYSRSn
0	HRE	R/W	0h	High Resolution Enable Bit: 0 High resolution mode disabled (default at reset) 1 High resolution mode enabled. Enabling this mode will connect the capture registers and edge event modes of the ECAP to be accessed by the High Res function. Note: The High Res clock needs to be enabled (using the HRCLKE bit) first before enabling the module. Allow a certain start up stabilization period before enabling the module. Reset type: CPU1.SYSRSn

22.5.2.2 HRINTEN Register (Offset = 4h) [Reset = 0h]

HRINTEN is shown in Figure 22-4 and described in Table 22-6.

Return to the Summary Table.

High-Res Calibration Interrupt Enable Register

Figure 22-4. HRINTEN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					CALPRDCHKSTS	CALIBDONE	RESERVED
R-0h					R/W-0h	R/W-0h	R-0h

Table 22-6. HRINTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reserved
2	CALPRDCHKSTS	R/W	0h	Calibration Period Check status Interrupt Enable: 0 Disable Calibration Period Check interrupt status 1 Enable Calibration Period Check interrupt status Reset type: CPU1.SYSRSn
1	CALIBDONE	R/W	0h	Calibration done Interrupt Enable: 0 Disable Calibration done Interrupt 1 Enable Calibration done Interrupt Reset type: CPU1.SYSRSn
0	RESERVED	R	0h	Reserved

22.5.2.3 HRFLG Register (Offset = 6h) [Reset = 0h]

HRFLG is shown in Figure 22-5 and described in Table 22-7.

Return to the Summary Table.

High-Res Calibration Interrupt Flag Register

Figure 22-5. HRFLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					CALPRDCHKSTS	CALIBDONE	CALIBINT
R-0h					R-0h	R-0h	R-0h

Table 22-7. HRFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reserved
2	CALPRDCHKSTS	R	0h	Calibration period check status Flag Bit: 1 Indicates that calibration ended before PRDCHK due to overflow on one of the counters. 0 Indicates no event occurred. Note: This bit remains latched until cleared by the user using the HRCLR [CALPRDCHKSTS] bit. Reset type: CPU1.SYSRSn
1	CALIBDONE	R	0h	Calibration Done Interrupt Flag Bit: 1 Indicates calibration cycle is completed 0 Indicates calibration cycle has not completed. Note: This bit remains latched until cleared by the user using the HRCLR [CALIBDONE] bit. Reset type: CPU1.SYSRSn
0	CALIBINT	R	0h	Global calibration Interrupt Status Flag: 1 Indicates that an interrupt was generated from CALIBDONE or CALPRDCHKSTS. 0 Indicates no interrupt generated. Reset type: CPU1.SYSRSn

22.5.2.4 HRCLR Register (Offset = 8h) [Reset = 0h]

HRCLR is shown in Figure 22-6 and described in Table 22-8.

Return to the Summary Table.

High-Res Calibration Interrupt Clear Register

Figure 22-6. HRCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					CALPRDCHKSTS	CALIBDONE	CALIBINT
R-0h					R-0/W1C-0h	R-0/W1C-0h	R-0/W1C-0h

Table 22-8. HRCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reserved
2	CALPRDCHKSTS	R-0/W1C	0h	Clear Calibration period check status Flag Bit: 1 Clears the CALPRDCHKSTS flag register bit. 0 No effect. Note: H/W has priority over CPU writes if the user tries to clear a flag bit and an event occurs on the same cycle that tries to set the flag for the selected bit. Reset type: CPU1.SYSRSn
1	CALIBDONE	R-0/W1C	0h	Clear Calibration Done Interrupt Flag Bit: 1 Clears the CALIBDONE interrupt flag register bit. 0 No effect. Note: H/W has priority over CPU writes if the user tries to clear a flag bit and an event occurs on the same cycle that tries to set the flag for the selected bit. Reset type: CPU1.SYSRSn
0	CALIBINT	R-0/W1C	0h	Clear Global calibration Interrupt Flag 1 Clears the Global interrupt flag and enables further interrupts to be generated if any of the event flags are set. 0 No effect. Reset type: CPU1.SYSRSn

22.5.2.5 HRFRC Register (Offset = Ah) [Reset = 0h]

HRFRC is shown in Figure 22-7 and described in Table 22-9.

Return to the Summary Table.

High-Res Calibration Interrupt Force Register

Figure 22-7. HRFRC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					CALPRDCHKSTS	CALIBDONE	RESERVED
R-0h					R-0/W1S-0h	R-0/W1S-0h	R-0h

Table 22-9. HRFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reserved
2	CALPRDCHKSTS	R-0/W1S	0h	Force CALPRDCHKSTS flag: 0 No effect 1 Sets the CALPRDCHKSTS flag. Reset type: CPU1.SYSRSn
1	CALIBDONE	R-0/W1S	0h	Force CALIBDONE flag: 0 No effect 1 Sets the CALIBDONE flag. Reset type: CPU1.SYSRSn
0	RESERVED	R	0h	Reserved

22.5.2.6 HRCALPRD Register (Offset = Ch) [Reset = 003FFFFFh]

HRCALPRD is shown in Figure 22-8 and described in Table 22-10.

Return to the Summary Table.

High-Res Calibration Period Register

Figure 22-8. HRCALPRD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRD																															
R/W-003FFFFFh																															

Table 22-10. HRCALPRD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PRD	R/W	003FFFFFh	Register to program calibration period. The period value is matched against HRSYSCLKCTR. On a match an interrupt is generated and the counter registers values are captured. Reset type: CPU1.SYSRSn

22.5.2.7 HRSYSCLKCTR Register (Offset = Eh) [Reset = 0h]

HRSYSCLKCTR is shown in Figure 22-9 and described in Table 22-11.

Return to the Summary Table.

High-Res Calibration SYSCLK Counter Register

Figure 22-9. HRSYSCLKCTR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HRSYSCLKCTR																															
R-0h																															

Table 22-11. HRSYSCLKCTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	HRSYSCLKCTR	R	0h	Current SYSCLK counter value Reset type: CPU1.SYSRSn

22.5.2.8 HRSYSCLKCAP Register (Offset = 10h) [Reset = 0h]

HRSYSCLKCAP is shown in Figure 22-10 and described in Table 22-12.

Return to the Summary Table.

High-Res Calibration SYSCLK Capture Register

Figure 22-10. HRSYSCLKCAP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HRSYSCLKCAP																															
R-0h																															

Table 22-12. HRSYSCLKCAP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	HRSYSCLKCAP	R	0h	HRSYSCLKCTR is captures into this register at end of calibration cycle. Reset type: CPU1.SYSRSn

22.5.2.9 HRCLKCTR Register (Offset = 12h) [Reset = 0h]

HRCLKCTR is shown in Figure 22-11 and described in Table 22-13.

Return to the Summary Table.

High-Res Calibration HRCLK Counter Register

Figure 22-11. HRCLKCTR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HRCLKCTR																															
R-0h																															

Table 22-13. HRCLKCTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	HRCLKCTR	R	0h	Current HRCLK counter value Note: HRCLK is not synchronized to SYSCLK domain so reads may not be accurate Reset type: CPU1.SYSRSn

22.5.2.10 HRCLKCAP Register (Offset = 14h) [Reset = 0h]

HRCLKCAP is shown in Figure 22-12 and described in Table 22-14.

Return to the Summary Table.

High-Res Calibration HRCLK Capture Register

Figure 22-12. HRCLKCAP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HRCLKCAP																															
R-0h																															

Table 22-14. HRCLKCAP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	HRCLKCAP	R	0h	HRCLKCTR is captures into this register at end of calibration cycle. Note: HRCLK is not synchronized to SYSCLK domain so reads may not be accurate Reset type: CPU1.SYSRSn

22.5.3 HRCAP Registers to Driverlib Functions

Table 22-15. HRCAP Registers to Driverlib Functions

File	Driverlib Function
TSCTR	
-	
CTRPHS	
-	
CAP1	
-	
CAP2	
-	
CAP3	
-	
CAP4	
-	
ECCTL0	
-	
ECCTL1	
-	
ECCTL2	
-	
ECEINT	
-	
ECFLG	
-	
ECCLR	
-	
ECFRC	
-	
ECAPSYNCINSEL	
-	

Table 22-15. HRCAP Registers to Driverlib Functions (continued)

File	Driverlib Function
HRCTL	
hrcap.h	HRCAP_enableHighResolution
hrcap.h	HRCAP_disableHighResolution
hrcap.h	HRCAP_enableHighResolutionClock
hrcap.h	HRCAP_disbleHighResolutionClock
hrcap.h	HRCAP_startCalibration
hrcap.h	HRCAP_setCalibrationMode
hrcap.h	HRCAP_isCalibrationBusy
HRINTEN	
hrcap.h	HRCAP_enableCalibrationInterrupt
hrcap.h	HRCAP_disableCalibrationInterrupt
HRFLG	
hrcap.h	HRCAP_getCalibrationFlags
HRCLR	
hrcap.h	HRCAP_clearCalibrationFlags
HRFRC	
hrcap.h	HRCAP_forceCalibrationFlags
HRCALPRD	
hrcap.h	HRCAP_setCalibrationPeriod
HRSYSCLKCTR	
-	
HRSYSCLKCAP	
hrcap.h	HRCAP_getCalibrationClockPeriod
HRCLKCTR	
-	
HRCLKCAP	
hrcap.h	HRCAP_getCalibrationClockPeriod

This page intentionally left blank.

Chapter 23

Enhanced Quadrature Encoder Pulse (eQEP)

The enhanced Quadrature Encoder Pulse (eQEP) module described here is a Type 2 eQEP. See the *C2000 Real-Time Control Peripheral Reference Guide* for a list of all devices with a module of the same type to determine the differences between types and for a list of device-specific differences within a type.

The enhanced quadrature encoder pulse (eQEP) module is used for direct interface with a linear or rotary incremental encoder to get position, direction, and speed information from a rotating machine for use in a high-performance motion and position-control system.

23.1 Introduction

An incremental encoder disk is patterned with a track of slots along its periphery, as shown in Figure 23-1. These slots create an alternating pattern of dark and light lines. The disk count is defined as the number of dark and light line pairs that occur per revolution (lines per revolution). As a rule, a second track is added to generate a signal that occurs once per revolution (index signal: QEPI), which can be used to indicate an absolute position. Encoder manufacturers identify the index pulse using different terms such as index, marker, home position, and zero reference

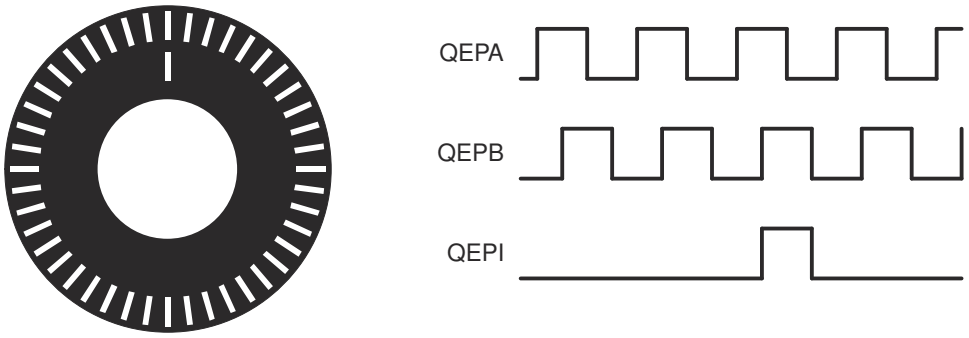
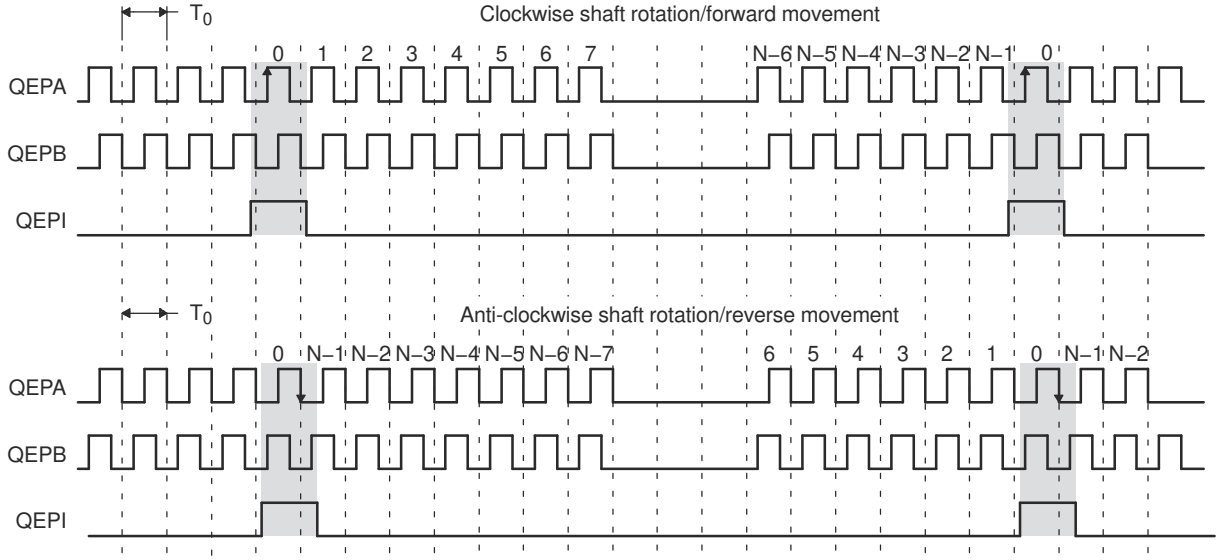


Figure 23-1. Optical Encoder Disk

To derive direction information, the lines on the disk are read out by two different photo-elements that "look" at the disk pattern with a mechanical shift of 1/4 the pitch of a line pair between them. This shift is detected with a reticle or mask that restricts the view of the photo-element to the desired part of the disk lines. As the disk rotates, the two photo-elements generate signals that are shifted 90° out of phase from each other. These are commonly called the quadrature QEPA and QEPB signals. The clockwise direction for most encoders is defined as the QEPA channel going positive before the QEPB channel and conversely, as shown in Figure 23-2.



Legend: N = lines per revolution

Figure 23-2. QEP Encoder Output Signal for Forward/Reverse Movement

The encoder wheel typically makes one revolution for every revolution of the motor, or the wheel may be at a geared rotation ratio with respect to the motor. Therefore, the frequency of the digital signal coming from the QEPA and QEPB outputs varies proportionally with the velocity of the motor. For example, a 2000-line encoder directly coupled to a motor running at 5000 revolutions per minute (rpm) results in a frequency of 166.6 kHz, so

by measuring the frequency of either the QEPA or QEPB output, the processor can determine the velocity of the motor.

Quadrature encoders from different manufacturers come with two forms of index pulse (gated index pulse or ungated index pulse) as shown in Figure 23-3. A nonstandard form of index pulse is ungated. In the ungated configuration, the index edges are not necessarily coincident with A and B signals. The gated index pulse is aligned to any of the four quadrature edges and width of the index pulse and can be equal to a quarter, half, or full period of the quadrature signal.

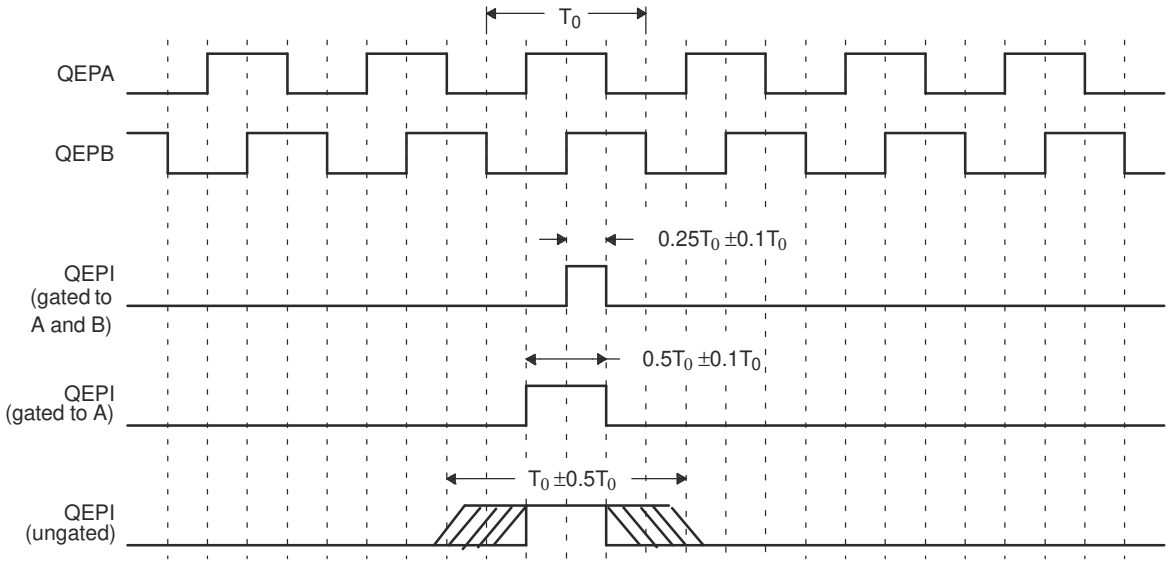


Figure 23-3. Index Pulse Example

Some typical applications of shaft encoders include robotics and computer input in the form of a mouse. Inside your mouse you can see where the mouse ball spins a pair of axles (a left/right, and an up/down axle). These axles are connected to optical shaft encoders that effectively tell the computer how fast and in what direction the mouse is moving.

General Issues: Estimating velocity from a digital position sensor is a cost-effective strategy in motor control. Two different first order approximations for velocity may be written as:

$$v(k) \approx \frac{x(k) - x(k-1)}{T} = \frac{\Delta X}{T} \tag{19}$$

$$v(k) \approx \frac{X}{t(k) - t(k-1)} = \frac{X}{\Delta T} \tag{20}$$

where:

- v(k) = Velocity at time instant k
- x(k) = Position at time instant k
- x(k-1) = Position at time instant k-1
- T = Fixed unit time or inverse of velocity calculation rate
- ΔX = Incremental position movement in unit time
- t(k) = Time instant "k"
- t(k-1) = Time instant "k-1"
- X = Fixed unit position
- ΔT = Incremental time elapsed for unit position movement

Equation 19 is the conventional approach to velocity estimation and it requires a time base to provide a unit time event for velocity calculation. Unit time is basically the inverse of the velocity calculation rate.

The encoder count (position) is read once during each unit time event. The quantity [x(k) - x(k-1)] is formed by subtracting the previous reading from the current reading. Then the velocity estimate is computed by multiplying by the known constant 1/T (where T is the constant time between unit time events and is known in advance).

Estimation based on Equation 19 has an inherent accuracy limit directly related to the resolution of the position sensor and the unit time period T. For example, consider a 500-line per revolution quadrature encoder with a velocity calculation rate of 400 Hz. When used for position, the quadrature encoder gives a four-fold increase in resolution; in this case, 2000 counts per revolution. The minimum rotation that can be detected is therefore 0.0005 revolutions, which gives a velocity resolution of 12 rpm when sampled at 400 Hz. While this resolution may be satisfactory at moderate or high speeds, for example 1% error at 1200 rpm, it would clearly prove inadequate at low speeds. In fact, at speeds below 12 rpm, the speed estimate would erroneously be zero much of the time.

At low speed, Equation 20 provides a more accurate approach. It requires a position sensor that outputs a fixed interval pulse train, such as the aforementioned quadrature encoder. The width of each pulse is defined by motor speed for a given sensor resolution. Equation 20 can be used to calculate motor speed by measuring the elapsed time between successive quadrature pulse edges. However, this method suffers from the opposite limitation, as does Equation 19. A combination of relatively large motor speeds and high sensor resolution makes the time interval ΔT small, and thus more greatly influenced by the timer resolution. This can introduce considerable error into high-speed estimates.

For systems with a large speed range (that is, speed estimation is needed at both low and high speeds), one approach is to use Equation 20 at low speed and have the DSP software switch over to Equation 19 when the motor speed rises above some specified threshold.

23.2 EQEP Related Collateral

Foundational Materials

- C2000 Academy - Control Peripherals
- *Real-Time Control Reference Guide*
  - Refer to the Encoders section

Getting Started Materials

- *C2000 Position Manager PTO API Reference Guide Application Report*

Expert Materials

- *CW/CCW Support on the C2000 eQEP Module Application Report*

23.3 Configuring Device Pins

The GPIO mux registers must be configured to connect this peripheral to the device pins. To avoid glitches on the pins, the GPyGMUX bits must be configured first (while keeping the corresponding GPyMUX bits at the default of zero), followed by writing the GPyMUX register to the desired value.

For proper operation of the eQEP module, input GPIO pins must be configured via the GPxQSELn registers for synchronous input mode (with or without qualification). The asynchronous mode should not be used for eQEP input pins. The internal pullups can be configured in the GPyPUD register.

See the *GPIO* chapter for more details on GPIO mux and settings.

23.4 Description

This section provides the eQEP inputs, memory map, and functional description.

23.4.1 EQEP Inputs

The eQEP inputs include two pins for quadrature-clock mode or direction-count mode, an index (or 0 marker), and a strobe input. The eQEP module requires that the QEPA, QEPB, and QEPI inputs are synchronized to SYSCLK prior to entering the module. The application code should enable the synchronous GPIO input feature on any eQEP-enabled GPIO pins (see the *General-Purpose Input/Output (GPIO)* chapter for more details).

- **QEPA/XCLK and QEPB/XDIR**

  These two pins can be used in quadrature-clock mode or direction-count mode.
  - Quadrature-clock Mode

    The eQEP encoders provide two square wave signals (A and B) 90 electrical degrees out of phase. This phase relationship is used to determine the direction of rotation of the input shaft and number of eQEP pulses from the index position to derive the relative position information. For forward or clockwise rotation, QEPA signal leads QEPB signal and conversely. The quadrature decoder uses these two inputs to generate quadrature-clock and direction signals.
  - Direction-count Mode

    In direction-count mode, direction and clock signals are provided directly from the external source. Some position encoders have this type of output instead of quadrature output. The QEPA pin provides the clock input and the QEPB pin provides the direction input.
- **QEPI: Index or Zero Marker**

  The eQEP encoder uses an index signal to assign an absolute start position from which position information is incrementally encoded using quadrature pulses. This pin is connected to the index output of the eQEP encoder to optionally reset the position counter for each revolution. This signal can be used to initialize or latch the position counter on the occurrence of a desired event on the index pin.
- **QEPS: Strobe Input**

  This general-purpose strobe signal can initialize or latch the position counter on the occurrence of a desired event on the strobe pin. This signal is typically connected to a sensor or limit switch to notify that the motor has reached a defined position.

Input signals to the eQEP (QEPA, QEPB, QEPI and QEPS) can come from multiple sources; that is, device pin, CMPSSx, or PWMXBARx. One typical used case is if SinCos transducers are used in the motor control system to estimate the position of motor shaft and Index signal is coming from traditional rotary encoder, source of the eQEP signals (QEPA, QEPB and QEPI) can be configured as output of CMPSSx which decodes the Sin, Cos and Index signals. Figure 23-4 illustrates the use case.

Selection of the source of Input signals (QEPA, QEPB, and QEPI) is user-configurable through the QEPSRCSEL register as shown in Table 23-1.

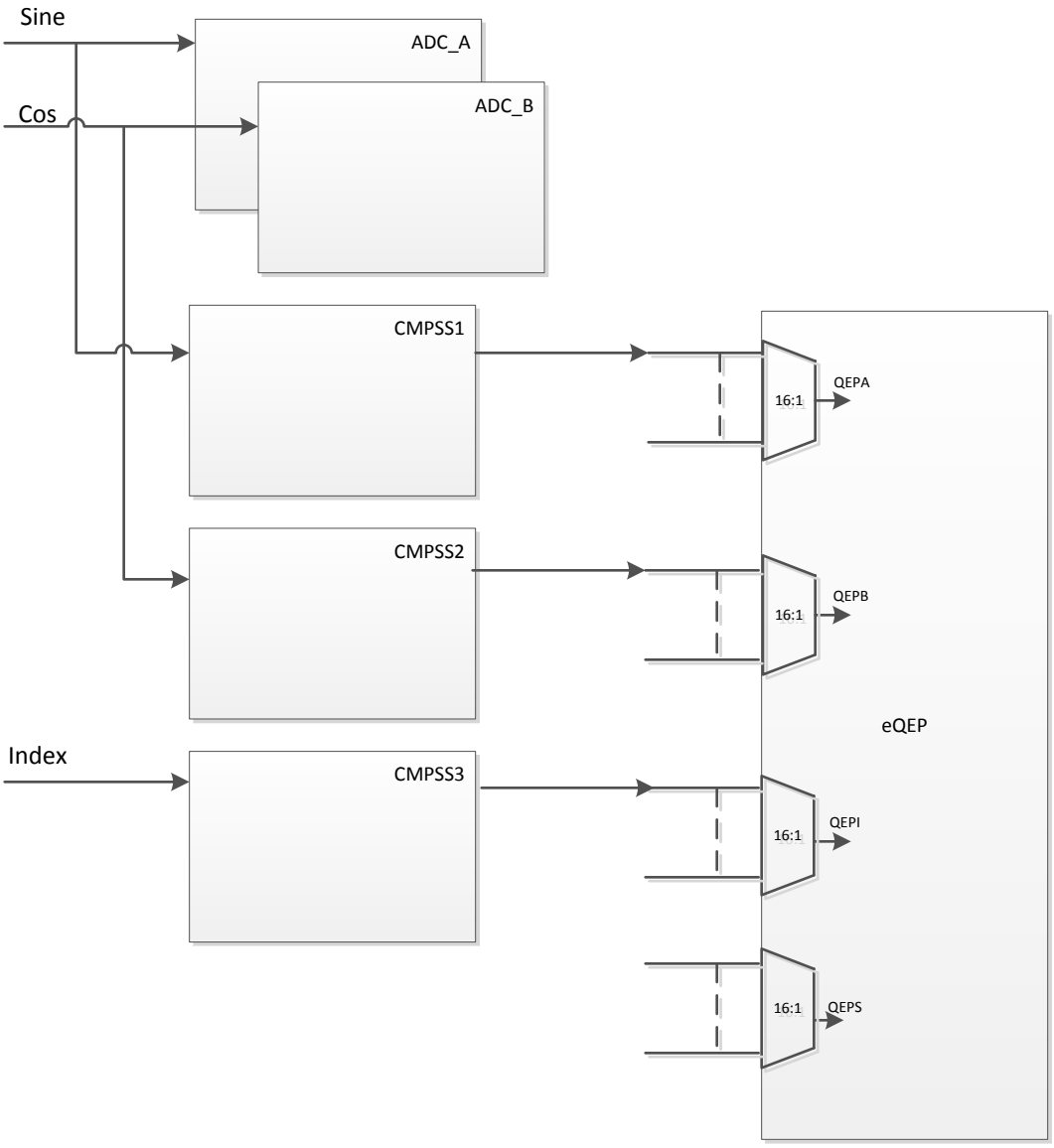


Figure 23-4. Using eQEP to Decode Signals from SinCos Transducer

Table 23-1. eQEP Input Source Select Table

QEPSRCSEL. QEPASEL	Source for QEPA	QEPSRCSEL. QEPBSEL	Source for QEPB	QEPSRCSEL. QEPISEL	Source for QEPI
0	Device Pin	0	Device Pin	0	Device Pin
1	CMPSS1.CTRIPH	1	CMPSS1.CTRIPH	1	CMPSS1.CTRIPH
2	CMPSS2.CTRIPH	2	CMPSS2.CTRIPH	2	CMPSS2.CTRIPH
3	CMPSS3.CTRIPH	3	CMPSS3.CTRIPH	3	CMPSS3.CTRIPH
4	CMPSS4.CTRIPH	4	CMPSS4.CTRIPH	4	CMPSS4.CTRIPH
5	Reserved	5	Reserved	5	Reserved
6	Reserved	6	Reserved	6	Reserved
7	Reserved	7	Reserved	7	Reserved
8	Reserved	8	Reserved	8	Reserved
9	PWMXBAR.1	9	PWMXBAR.1	9	PWMXBAR.1
10	PWMXBAR.2	10	PWMXBAR.2	10	PWMXBAR.2
11	PWMXBAR.3	11	PWMXBAR.3	11	PWMXBAR.3
12	PWMXBAR.4	12	PWMXBAR.4	12	PWMXBAR.4
13	PWMXBAR.5	13	PWMXBAR.5	13	PWMXBAR.5
14	PWMXBAR.6	14	PWMXBAR.6	14	PWMXBAR.6
15	PWMXBAR.7	15	PWMXBAR.7	15	PWMXBAR.7

Note

Configuration of QEPSRCSEL register to select the source of QEPA, QEPB, and QEPI signals can lead to unexpected transition on these signals, which can cause an undesirable outcome if eQEP is already running. You need to make sure that the eQEP is disabled before configuring the QEPSRCSEL register for input signals.

23.4.2 Functional Description

The eQEP peripheral contains the following major functional units (as shown in Figure 23-5):

- Programmable input qualification for each pin (part of the GPIO MUX)
- Quadrature decoder unit (QDU)
- Position counter and control unit for position measurement (PCCU)
- Quadrature edge-capture unit for low-speed measurement (QCAP)
- Unit time base for speed/frequency measurement (UTIME)
- Watchdog timer for detecting stalls (QWDOG)
- Quadrature Mode Adapter (QMA)

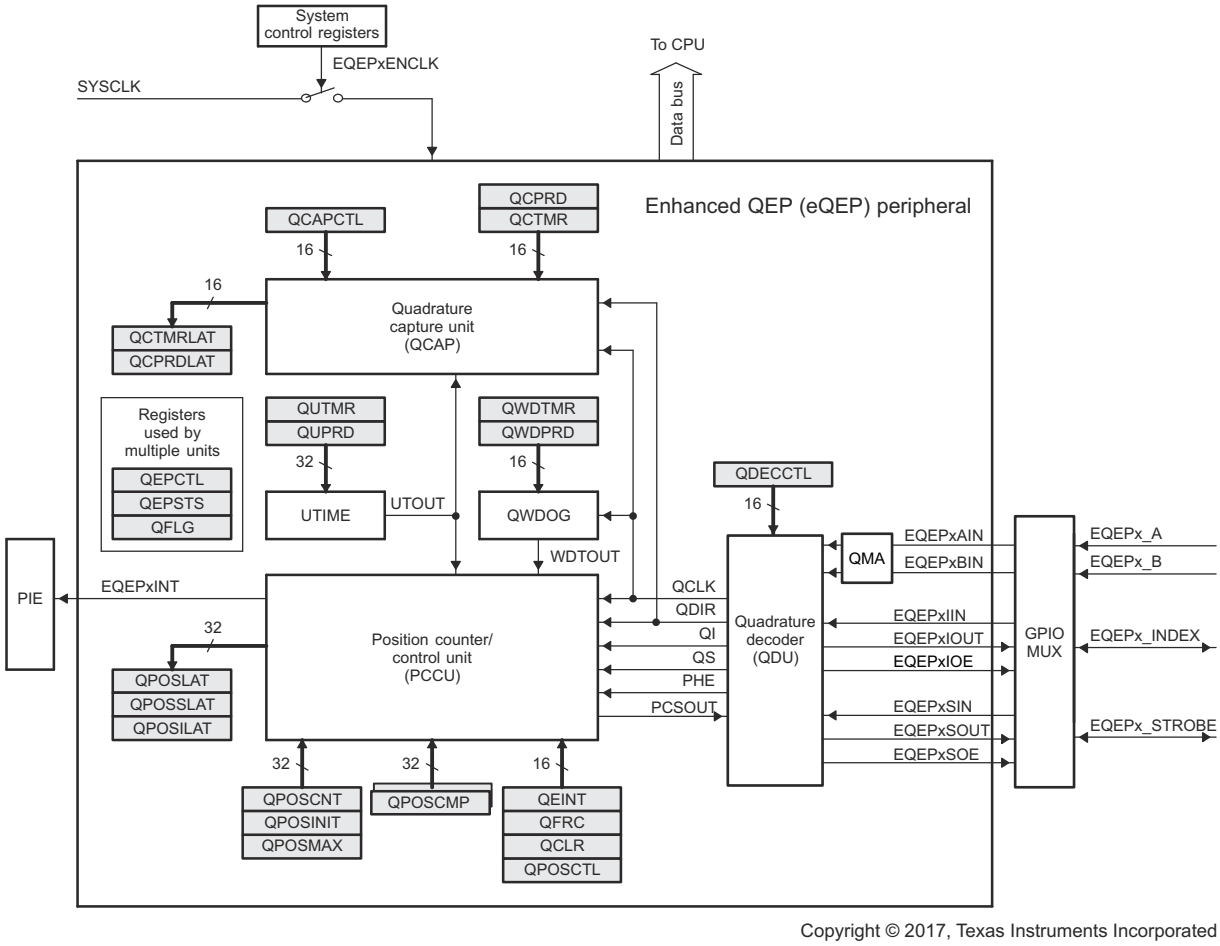


Figure 23-5. Functional Block Diagram of the eQEP Peripheral

23.4.3 eQEP Memory Map

Table 23-2 lists the registers with their memory locations, sizes, and reset values.

Table 23-2. EQEP Memory Map

Name	Offset	Size(x16)/ #shadow	Reset	Register Description
QPOSCNT	0x00	2/0	0x00000000	eQEP Position Counter
QPOSINIT	0x02	2/0	0x00000000	eQEP Initialization Position Count
QPOSMAX	0x04	2/0	0x00000000	eQEP Maximum Position Count
QPOSCMP	0x06	2/1	0x00000000	eQEP Position-compare
QPOSILAT	0x08	2/0	0x00000000	eQEP Index Position Latch
QPOSSLAT	0x0A	2/0	0x00000000	eQEP Strobe Position Latch
QPOSLAT	0x0C	2/0	0x00000000	eQEP Position Latch
QUTMR	0x0E	2/0	0x00000000	QEP Unit Timer
QUPRD	0x10	2/0	0x00000000	eQEP Unit Period Register
QWDTMR	0x12	1/0	0x0000	eQEP Watchdog Timer
QWDPRD	0x13	1/0	0x0000	eQEP Watchdog Period Register
QDECCTL	0x14	1/0	0x0000	eQEP Decoder Control Register
QEPCTL	0x15	1/0	0x0000	eQEP Control Register
QCAPCTL	0x16	1/0	0x0000	eQEP Capture Control Register
QPOSCTL	0x17	1/0	0x00000	eQEP Position-compare Control Register
QEINT	0x18	1/0	0x0000	eQEP Interrupt Enable Register
QFLG	0x19	1/0	0x0000	eQEP Interrupt Flag Register
QCLR	0x1A	1/0	0x0000	eQEP Interrupt Clear Register
QFRC	0x1B	1/0	0x0000	eQEP Interrupt Force Register
QEPSTS	0x1C	1/0	0x0000	eQEP Status Register
QCTMR	0x1D	1/0	0x0000	eQEP Capture Timer
QCPRD	0x1E	1/0	0x0000	eQEP Capture Period Register
QCTMRLAT	0x1F	1/0	0x0000	eQEP Capture Timer Latch
QCPRDLAT	0x20	1/0	0x0000	eQEP Capture Period Latch
reserved	0x21 to 0x2F	15/0		
REV	0x30	2/0	0x0000	eQEP Revision Number
QEPSTROBESEL	0x32	2/0	0x0000	eQEP Strobe select register
QMACTRL	0x34	2/0	0x0000	eQEP QMA Control register
QEPSRCSEL	0x36	2/0	0x0000	eQEP Source Select Register
reserved	0x38 to 0x3F	8/0		

23.5 Quadrature Decoder Unit (QDU)

Figure 23-6 shows a functional block diagram of the QDU.

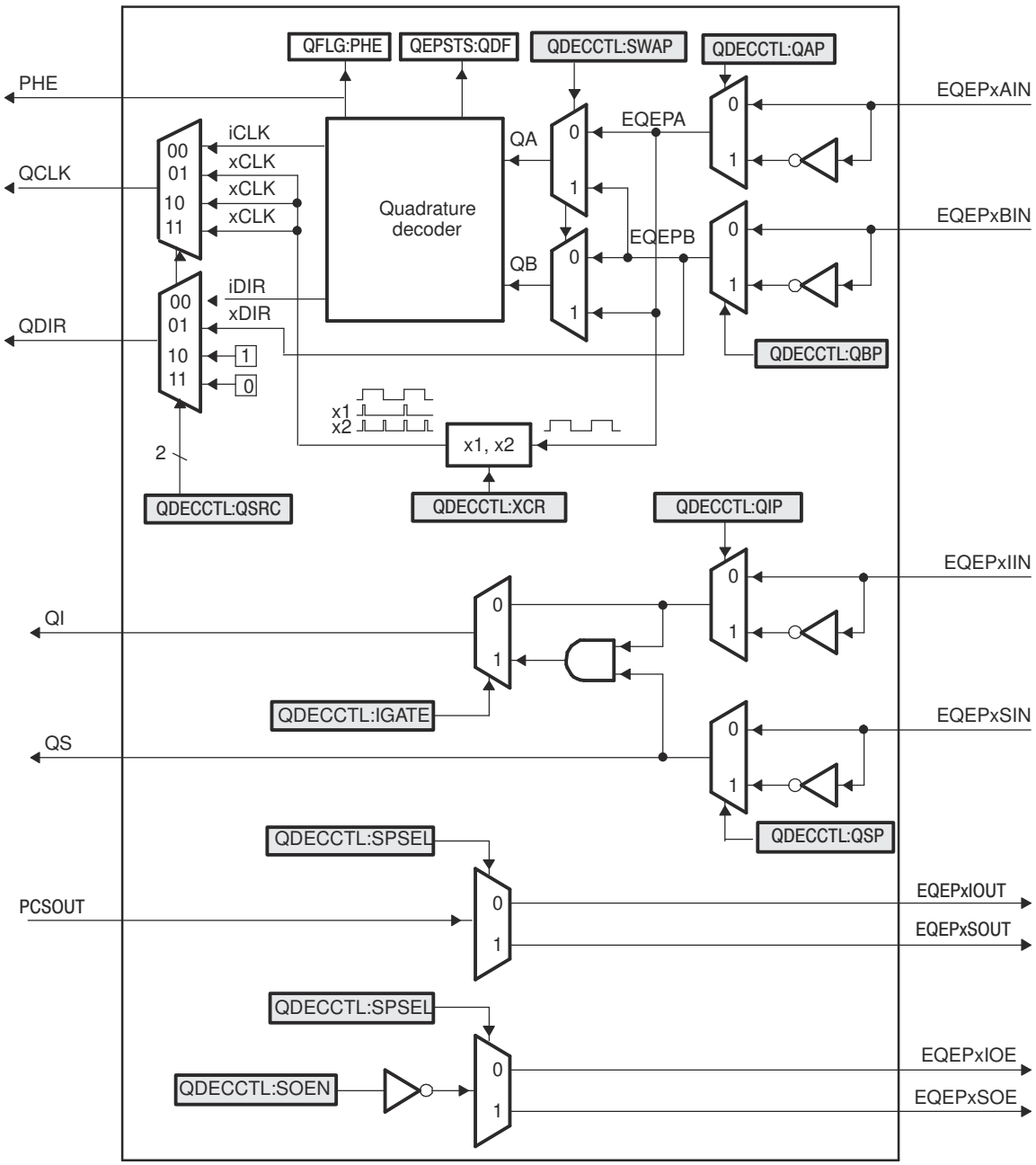


Figure 23-6. Functional Block Diagram of Decoder Unit

23.5.1 Position Counter Input Modes

Clock and direction input to the position counter is selected using QDECCTL[QSRC] bits, based on interface input requirement as follows:

- Quadrature-count mode
- Direction-count mode
- UP-count mode

- DOWN-count mode

23.5.1.1 Quadrature Count Mode

The quadrature decoder generates the direction and clock to the position counter in quadrature count mode.

Direction Decoding

The direction decoding logic of the eQEP circuit determines which one of the sequences (QEPA, QEPB) is the leading sequence and accordingly updates the direction information in the QEPSTS[QDF] bit. Table 23-3 and Figure 23-7 show the direction decoding logic in truth table and state machine form. Both edges of the QEPA and QEPB signals are sensed to generate count pulses for the position counter. Therefore, the frequency of the clock generated by the eQEP logic is four times that of each input sequence. Figure 23-8 shows the direction decoding and clock generation from the eQEP input signals.

Table 23-3. Quadrature Decoder Truth Table

Previous Edge	Present Edge	QDIR	QPOSCNT
QA↑	QB↑	UP	Increment
	QB↓	DOWN	Decrement
	QA↓	TOGGLE	Increment or Decrement
QA↓	QB↓	UP	Increment
	QB↑	DOWN	Decrement
	QA↑	TOGGLE	Increment or Decrement
QB↑	QA↑	DOWN	Decrement
	QA↓	UP	Increment
	QB↓	TOGGLE	Increment or Decrement
QB↓	QA↓	DOWN	Decrement
	QA↑	UP	Increment
	QB↑	TOGGLE	Increment or Decrement

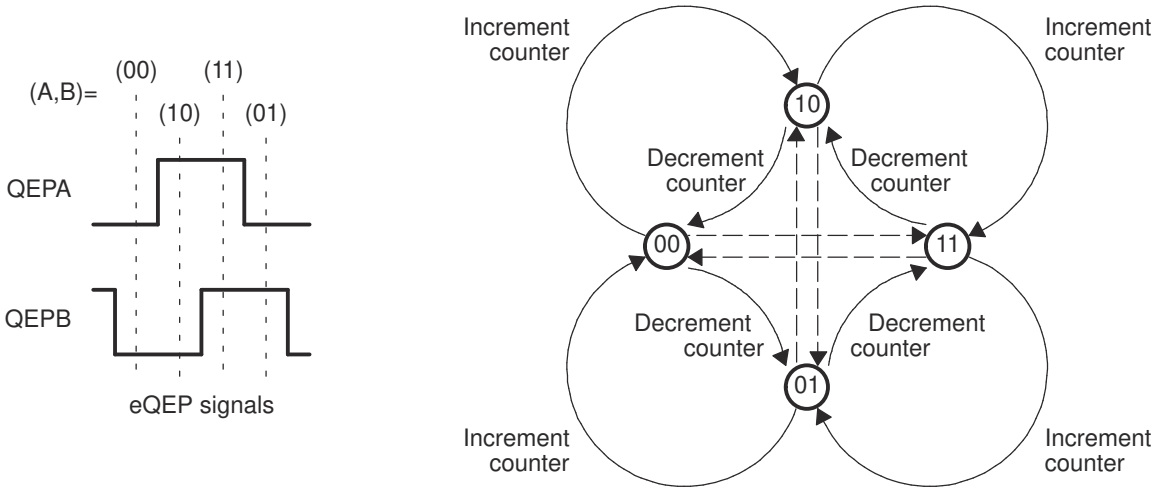


Figure 23-7. Quadrature Decoder State Machine

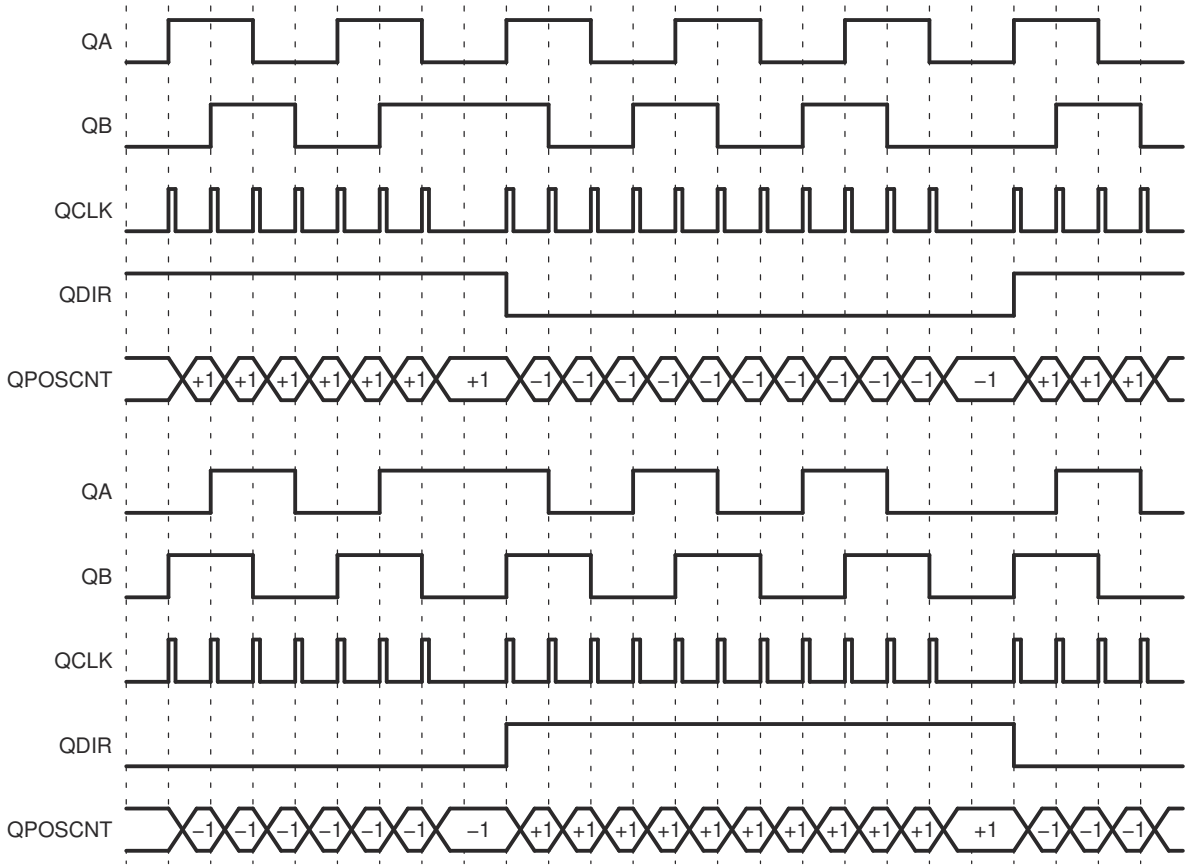


Figure 23-8. Quadrature-clock and Direction Decoding

Phase Error Flag In normal operating conditions, quadrature inputs QEPA and QEPB will be 90 degrees out of phase. The phase error flag (PHE) is set in the QFLG register and the QPOSCNT value can be incorrect and offset by multiples of 1 or 3. That is, when edge transition is detected simultaneously on the QEPA and QEPB signals to optionally generate interrupts. State transitions marked by dashed lines in Figure 23-7 are invalid transitions that generate a phase error.

Count Multiplication The eQEP position counter provides 4x times the resolution of an input clock by generating a quadrature-clock (QCLK) on the rising/falling edges of both eQEP input clocks (QEPA and QEPB) as shown in Figure 23-8.

Reverse Count In normal quadrature count operation, QEPA input is fed to the QA input of the quadrature decoder and the QEPB input is fed to the QB input of the quadrature decoder. Reverse counting is enabled by setting the SWAP bit in the QDECCTL register. This will swap the input to the quadrature decoder, thereby reversing the counting direction.

23.5.1.2 Direction-Count Mode

Some position encoders provide direction and clock outputs, instead of quadrature outputs. In such cases, direction-count mode can be used. QEPA input will provide the clock for the position counter and the QEPB input will have the direction information. The position counter is incremented on every rising edge of a QEPA input when the direction input is high, and decremented when the direction input is low.

23.5.1.3 Up-Count Mode

The counter direction signal is hard-wired for up-count and the position counter is used to measure the frequency of the QEPA input. Clearing the QDECCTL[XCR] bit enables clock generation to the position counter on both edges of the QEPA input, thereby increasing the measurement resolution by a factor of 2x. In up-count mode, it is recommended that the application not configure QEPB as a GPIO mux option, or ensure that a signal edge is not generated on the QEPB input.

23.5.1.4 Down-Count Mode

The counter direction signal is hardwired for a down-count and the position counter is used to measure the frequency of the QEPA input. Clearing the QDECCTL[XCR] bit enables clock generation to the position counter on both edges of a QEPA input, thereby increasing the measurement resolution by a factor of 2x. In down-count mode, it is recommended that the application not configure QEPB as a GPIO mux option, or ensure that a signal edge is not generated on the QEPB input.

23.5.2 eQEP Input Polarity Selection

Each eQEP input can be inverted using QDECCTL[8:5] control bits. As an example, setting the QDECCTL[QIP] bit will invert the index input.

23.5.3 Position-Compare Sync Output

The enhanced eQEP peripheral includes a position-compare unit that is used to generate the position-compare sync signal on compare match between the position-counter register (QPOSCNT) and the position- compare register (QPOSCMP). This sync signal can be output using an index pin or strobe pin of the EQEP peripheral.

Setting the QDECCTL[SOEN] bit enables the position-compare sync output and the QDECCTL[SPSEL] bit selects either an eQEP index pin or an eQEP strobe pin.

23.6 Position Counter and Control Unit (PCCU)

The position-counter and control unit provides two configuration registers (QEPCTL and QPOSCTL) for setting up position-counter operational modes, position-counter initialization/latch modes and position-compare logic for sync signal generation.

23.6.1 Position Counter Operating Modes

Position-counter data may be captured in different manners. In some systems, the position counter is accumulated continuously for multiple revolutions and the position-counter value provides the position information with respect to the known reference. An example of this is the quadrature encoder mounted on the motor controlling the print head in the printer. Here the position counter is reset by moving the print head to the home position and then the position counter provides absolute position information with respect to home position.

In other systems, the position counter is reset on every revolution using index pulse, and the position counter provides a rotor angle with respect to the index pulse position.

The position counter can be configured to operate in following four modes

- Position-Counter Reset on Index Event
- Position-Counter Reset on Maximum Position
- Position-Counter Reset on the first Index Event
- Position-Counter Reset on Unit Time Out Event (Frequency Measurement)

In all the above operating modes, the position counter is reset to 0 on overflow and to the QPOSMAX register value on underflow. Overflow occurs when the position counter counts up after the QPOSMAX value. Underflow occurs when the position counter counts down after "0". The Interrupt flag is set to indicate overflow/underflow in QFLG register.

23.6.1.1 Position Counter Reset on Index Event (QEPCTL[PCRM]=00)

If the index event occurs during the forward movement, then the position counter is reset to 0 on the next eQEP clock. If the index event occurs during the reverse movement, then the position counter is reset to the value in the QPOSMAX register on the next eQEP clock.

First index marker is defined as the quadrature edge following the first index edge. The eQEP peripheral records the occurrence of the first index marker (QEPSTS[FIMF]) and direction on the first index event marker (QEPSTS[FIDF]) in QEPSTS registers, it also remembers the quadrature edge on the first index marker so that same relative quadrature transition is used for index event reset operation.

For example, if the first reset operation occurs on the falling edge of QEPB during the forward direction, then all the subsequent reset must be aligned with the falling edge of QEPB for the forward rotation and on the rising edge of QEPB for the reverse rotation as shown in Figure 23-9.

The position-counter value is latched to the QPOSILAT register and direction information is recorded in the QEPSTS[QDLF] bit on every index event marker. The position-counter error flag (QEPSTS[PCEF]) and error interrupt flag (QFLG[PCE]) are set if the latched value is not equal to 0 or QPOSMAX. The position-counter error flag (QEPSTS[PCEF]) is updated on every index event marker and an interrupt flag (QFLG[PCE]) will be set on error that can be cleared only through software.

The index event latch configuration QEPCTL[IEL] must be configured to '00' or '11' when pcrm=0 and the position counter error flag/interrupt flag are generated only in index event reset mode. The position counter value is latched into the IPOSLAT register on every index marker.

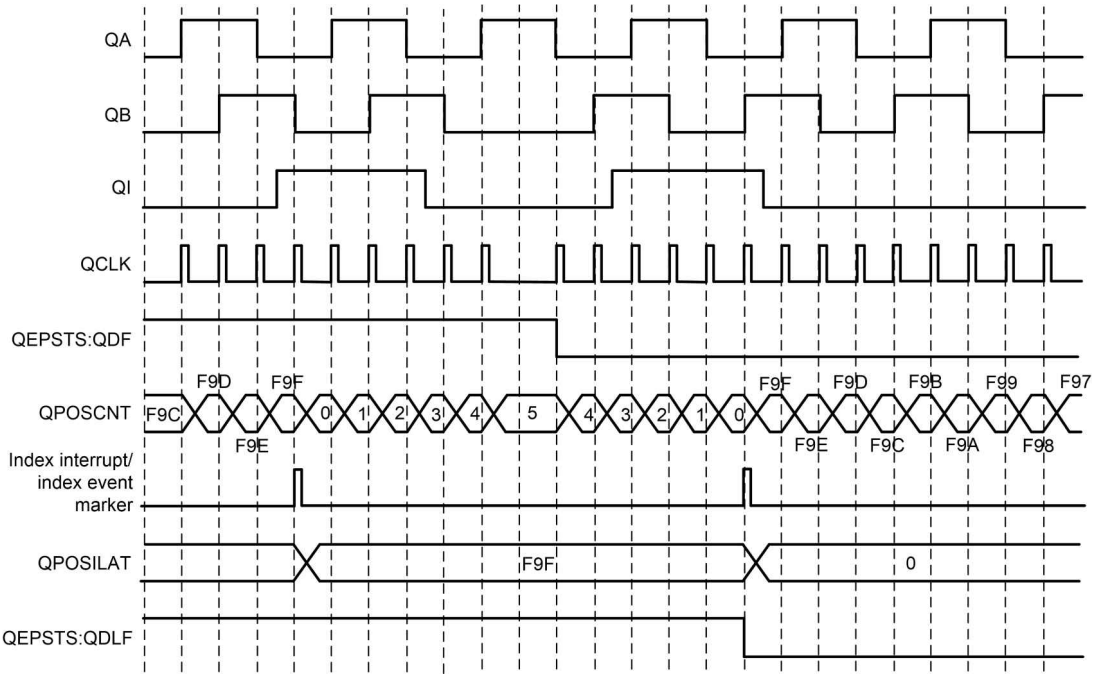


Figure 23-9. Position Counter Reset by Index Pulse for 1000 Line Encoder (QPOSMAX = 3999 or 0xF9F)

Note

In case of a boundary condition where the time period between the Index Event and the previous QCLK edge is less than SYSCLK period, then QPOSCNT gets reset to zero or QPOSMAX in the same SYSCLK cycle and does not wait for the next QCLK edge to occur.

23.6.1.2 Position Counter Reset on Maximum Position (QEPCTL[PCRM]=01)

If the position counter is equal to QPOSMAX, then the position counter is reset to 0 on the next eQEP clock for forward movement and position counter overflow flag is set. If the position counter is equal to ZERO, then the position counter is reset to QPOSMAX on the next QEP clock for reverse movement and position-counter underflow flag is set. Figure 23-10 shows the position-counter reset operation in this mode.

The first index marker fields (QEPSTS[FIDF] and QEPSTS[FIMF]) are not applicable in this mode.

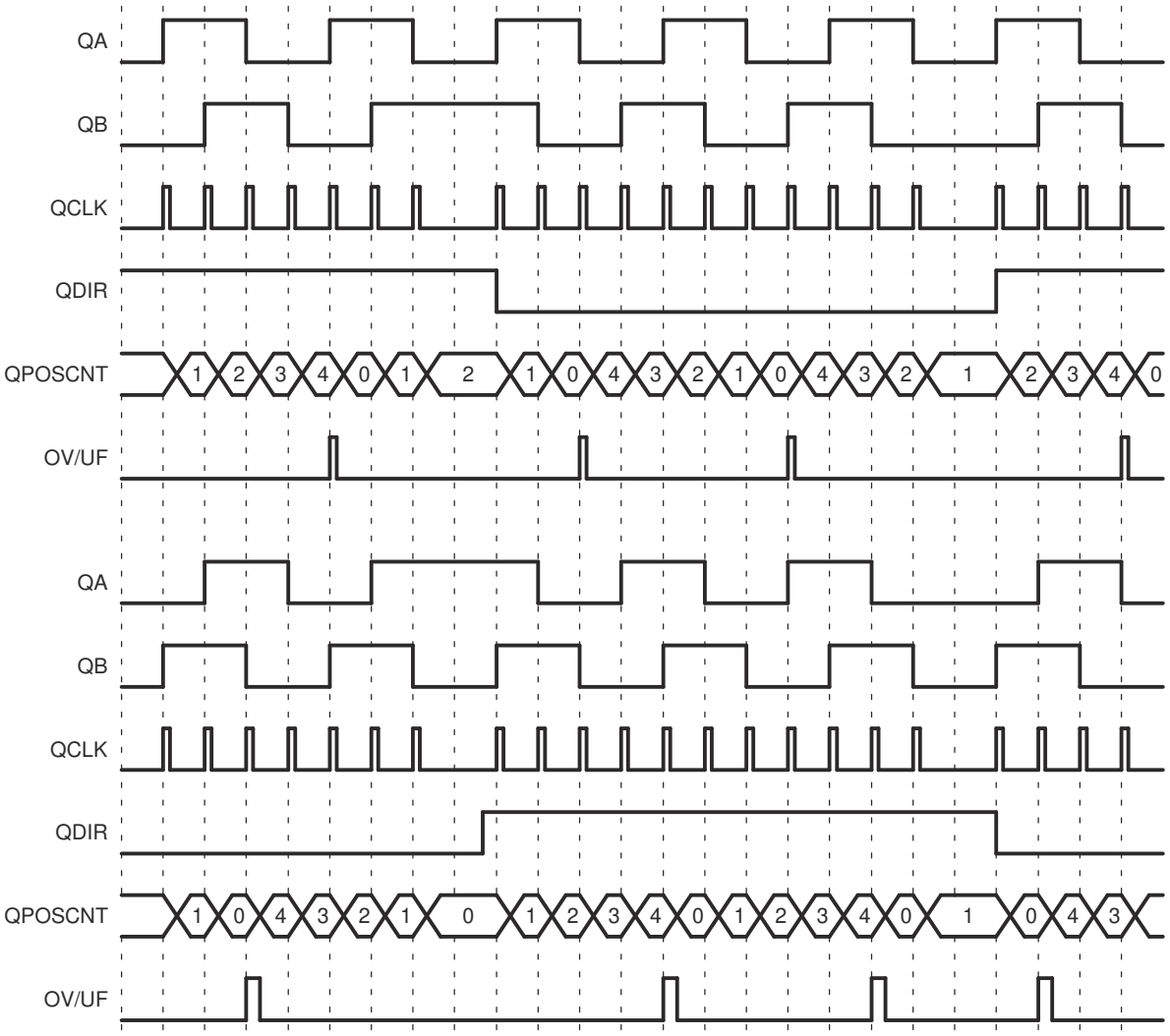


Figure 23-10. Position Counter Underflow/Overflow (QPOSMAX = 4)

23.6.1.3 Position Counter Reset on the First Index Event (QEPCTL[PCRM] = 10)

If the index event occurs during forward movement, then the position counter is reset to 0 on the next eQEP clock. If the index event occurs during the reverse movement, then the position counter is reset to the value in the QPOSMAX register on the next eQEP clock. Note that this is done only on the first occurrence and subsequently the position-counter value is not reset on an index event; rather, it is reset based on maximum position as described in Section 23.6.1.2.

The first index marker fields (QEPSTS[FIDF] and QEPSTS[FIMF]) are not applicable in this mode.

23.6.1.4 Position Counter Reset on Unit Time out Event (QEPCTL[PCRM] = 11)

In this mode, QPOSCNT is set to 0 or QPOMAX, depending on the direction mode selected by QDECCTL[QSRC] bits on a unit time event. This is useful for frequency measurement.

23.6.2 Position Counter Latch

The eQEP index and strobe input can be configured to latch the position counter (QPOSCNT) into QPOSILAT and QPOSSLAT, respectively, on occurrence of a definite event on these pins.

23.6.2.1 Index Event Latch

In some applications, it may not be desirable to reset the position counter on every index event and instead it may be required to operate the position counter in full 32-bit mode (QEPCTL[PCRM] = 01 and QEPCTL[PCRM] = 10 modes).

In such cases, the eQEP position counter can be configured to latch on the following events and direction information is recorded in the QEPSTS[QDLF] bit on every index event marker.

- Latch on Rising edge (QEPCTL[IEL]=01)
- Latch on Falling edge (QEPCTL[IEL]=10)
- Latch on Index Event Marker (QEPCTL[IEL]=11)

This is particularly useful as an error checking mechanism to check if the position counter accumulated the correct number of counts between index events. As an example, the 1000-line encoder must count 4000 times when moving in the same direction between the index events.

The index event latch interrupt flag (QFLG[IEL]) is set when the position counter is latched to the QPOSILAT register. The index event latch configuration bits (QEPCTZ[IEL]) are ignored when QEPCTL[PCRM] = 00.

Latch on Rising Edge (QEPCTL[IEL]=01)	The position-counter value (QPOSCNT) is latched to the QPOSILAT register on every rising edge of an index input.
Latch on Falling Edge (QEPCTL[IEL] = 10)	The position-counter value (QPOSCNT) is latched to the QPOSILAT register on every falling edge of index input.
Latch on Index Event Marker/Software Index Marker (QEPCTL[IEL] = 11	The first index marker is defined as the quadrature edge following the first index edge. The eQEP peripheral records the occurrence of the first index marker (QEPSTS[FIMF]) and direction on the first index event marker (QEPSTS[FIDF]) in the QEPSTS registers. It also remembers the quadrature edge on the first index marker so that same relative quadrature transition is used for latching the position counter (QEPCTL[IEL]=11).

Figure 23-11 shows the position counter latch using an index event marker.

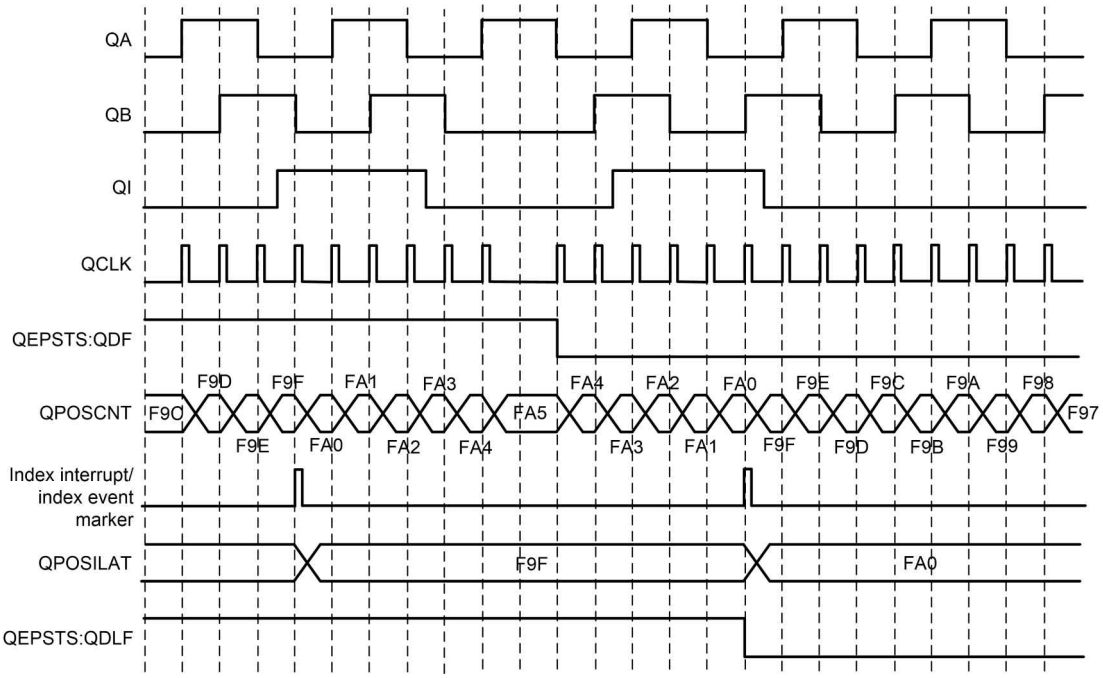


Figure 23-11. Software Index Marker for 1000-line Encoder (QEPCTL[IEL] = 1)

23.6.2.2 Strobe Event Latch

The position-counter value is latched to the QPOSSLAT register on the rising edge of the strobe input by clearing the QEPCTL[SEL] bit.

If the QEPCTL[SEL] bit is set, then the position-counter value is latched to the QPOSSLAT register on the rising edge of the strobe input for forward direction, and on the falling edge of the strobe input for reverse direction as shown in Figure 23-12.

The strobe event latch interrupt flag (QFLG[SEL) is set when the position counter is latched to the QPOSSLAT register.

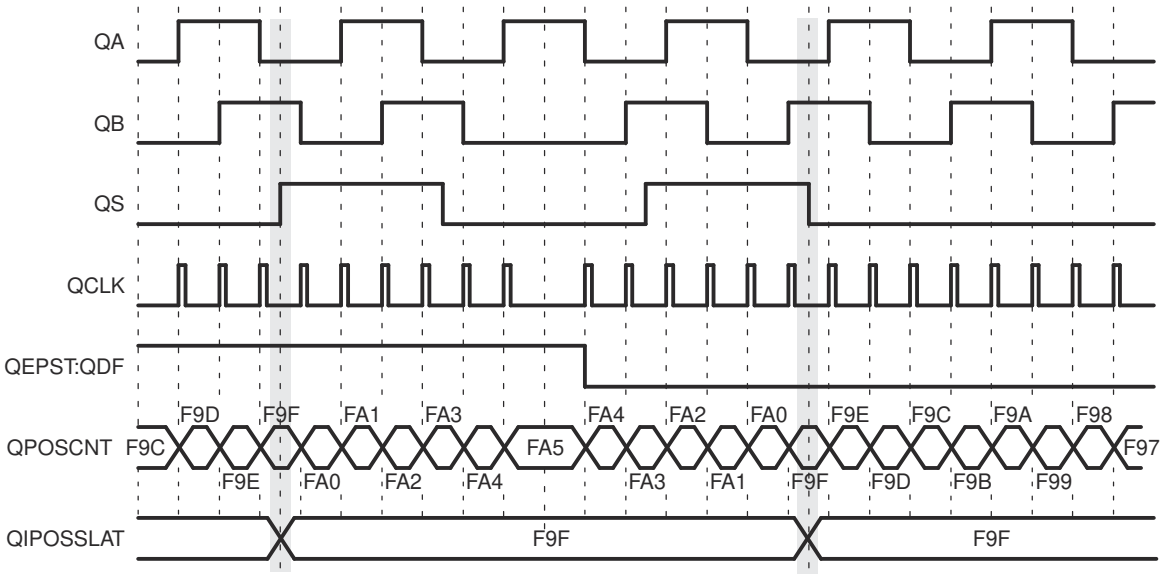


Figure 23-12. Strobe Event Latch (QEPCTL[SEL] = 1)

There is an added feature on Type 2.0 eQEP where position-counter value can also be latched on ADCSOCA and ADCSOSCB events by configuring the register QEPSTROBESEL.STROBESEL as shown in figure Figure 23-13

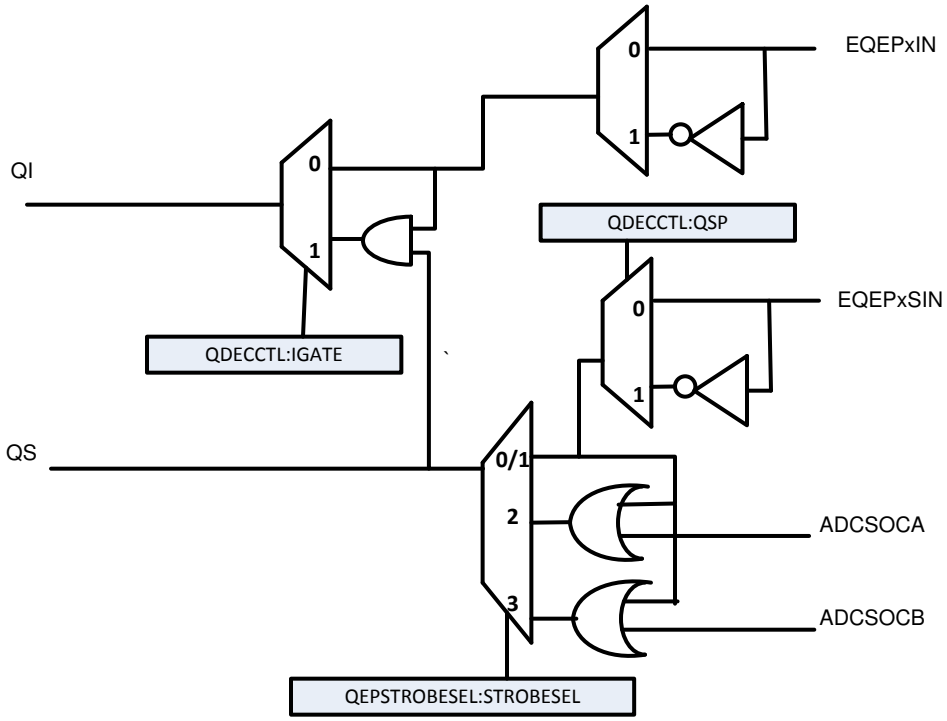


Figure 23-13. Latching Position Counter on ADCSOCA/ADCSOCB event

23.6.3 Position Counter Initialization

The position counter can be initialized using following events:

- Index event
- Strobe event
- Software initialization

Index Event Initialization (IEI)

The QEPI index input can be used to trigger the initialization of the position counter at the rising or falling edge of the index input. If the QEPCTL[IEI] bits are 10, then the position counter (QPOSCNT) is initialized with a value in the QPOSINIT register on the rising edge of index input. Conversely, if the QEPCTL[IEI] bits are 11, initialization will be on the falling edge of the index input.

Strobe Event Initialization (SEI)

If the QEPCTL[SEI] bits are 10, then the position counter is initialized with a value in the QPOSINIT register on the rising edge of strobe input.

If QEPCTL[SEL] bits are 11, then the position counter is initialized with a value in the QPOSINIT register on the rising edge of strobe input for forward direction and on the falling edge of strobe input for reverse direction.

Software Initialization (SWI)

The position counter can be initialized in software by writing a 1 to the QEPCTL[SWI] bit. This bit is not automatically cleared. While the bit is still set, if a 1 is written to it again, the position counter will be re-initialized.

23.6.4 eQEP Position-compare Unit

The eQEP peripheral includes a position-compare unit that is used to generate a sync output and/or interrupt on a position-compare match. Figure 23-14 shows a diagram. The position-compare (QPOSCMP) register is shadowed and shadow mode can be enabled or disabled using the QPOSCTL[PSSHDW] bit. If the shadow mode is not enabled, the CPU writes directly to the active position compare register.

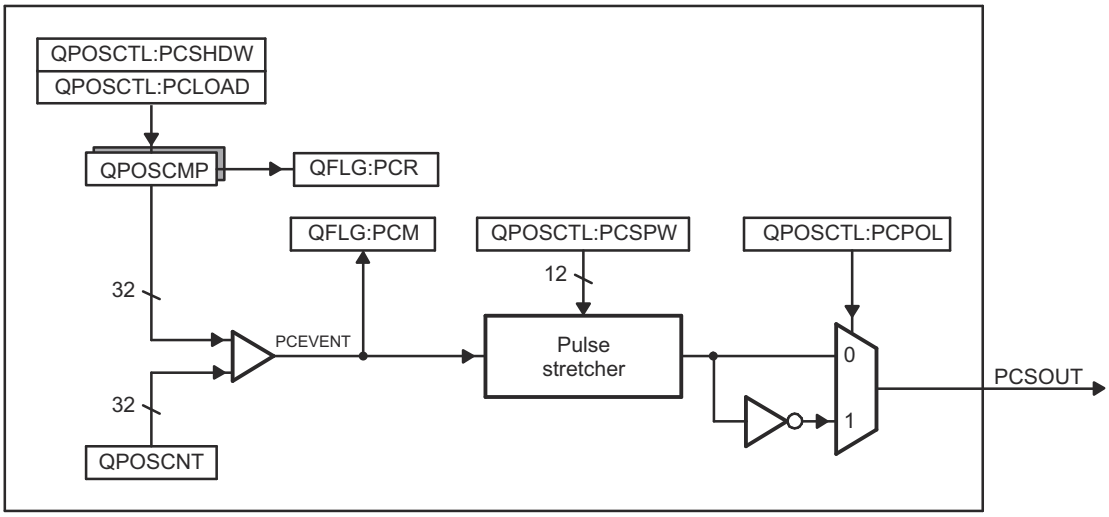


Figure 23-14. eQEP Position-compare Unit

In shadow mode, you can configure the position-compare unit (QPOSCTL[PCLOAD]) to load the shadow register value into the active register on the following events, and to generate the position-compare ready (QFLG[PCR]) interrupt after loading.

- Load on compare match
- Load on position-counter zero event

The position-compare match (QFLG[PCM]) is set when the position-counter value (QPOSCNT) matches with the active position-compare register (QPOSCMP) and the position-compare sync output of the programmable pulse width is generated on compare-match to trigger an external device.

For example, if QPOSCMP = 2, the position-compare unit generates a position-compare event on 1 to 2 transitions of the eQEP position counter for forward counting direction and on 3 to 2 transitions of the eQEP position counter for reverse counting direction (see Figure 23-15).

See the register section for the layout of the eQEP Position-Compare Control Register (QPOSCTL) and description of the QPOSCTL bit fields.

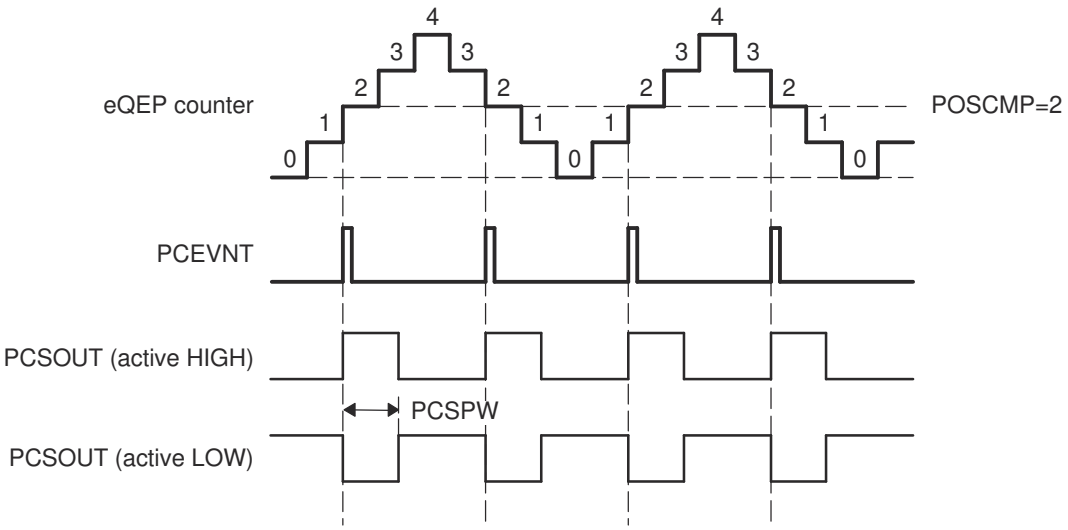


Figure 23-15. eQEP Position-compare Event Generation Points

The pulse stretcher logic in the position-compare unit generates a programmable position-compare sync pulse output on the position-compare match. In the event of a new position-compare match while a previous position-compare pulse is still active, then the pulse stretcher generates a pulse of specified duration from the new position-compare event as shown in Figure 23-16.

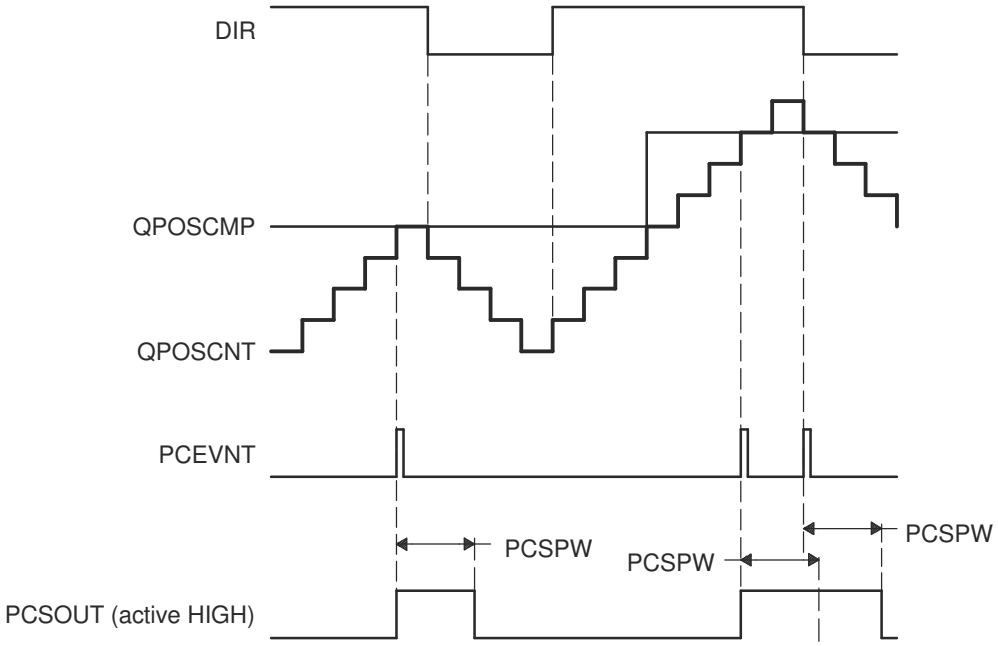


Figure 23-16. eQEP Position-compare Sync Output Pulse Stretcher

23.7 eQEP Edge Capture Unit

The eQEP peripheral includes an integrated edge capture unit to measure the elapsed time between the unit position events as shown in Figure 23-17. This feature is typically used for low-speed measurement using the following formula:

$$v(k) = \frac{X}{t(k) - t(k-1)} = \frac{X}{\Delta T} \quad (21)$$

where:

- X = Unit position is defined by integer multiple of quadrature edges (see Figure 23-18)
- ΔT = Elapsed time between unit position events
- v(k) = Velocity at time instant "k"

The eQEP capture timer (QCTMR) runs from prescaled SYSCLKOUT and the prescaler is programmed by the QCAPCTL[CCPS] bits. The capture timer (QCTMR) value is latched into the capture period register (QCPRD) on every unit position event and then the capture timer is reset, a flag is set in QEPSTS:UPEVNT to indicate that new value is latched into the QCPRD register. Software can check this status flag before reading the period register for low speed measurement, and clear the flag by writing 1.

Time measurement (ΔT) between unit position events will be correct if the following conditions are met:

- No more than 65,535 counts have occurred between unit position events.
- No direction change between unit position events.

If the QEP capture timer overflows between unit position events, then it sets the QEP capture overflow flag (QEPSTS[COEF]) in the status register and the QCPRDLAT register is set to 0xFFFF. If direction change occurs between the unit position events, then the error flag is set in the status register (QEPSTS[CDEF]) and the QCPRDLAT register is set to 0xFFFF.

The Capture Timer (QCTMR) and Capture Period register (QCPRD) can be configured to latch on following events.

- CPU read of QPOSCNT register
- Unit time-out event

If the QEPCTL[QCLM] bit is cleared, then the capture timer and capture period values are latched into the QCTMRLAT and QCPRDLAT registers, respectively, when the CPU reads the position counter (QPOSCNT).

If the QEPCTL[QCLM] bit is set, then the position counter, capture timer, and capture period values are latched into the QPOSLAT, QCTMRLAT and QCPRDLAT registers, respectively, on unit time out.

Figure 23-19 shows the capture unit operation along with the position counter.

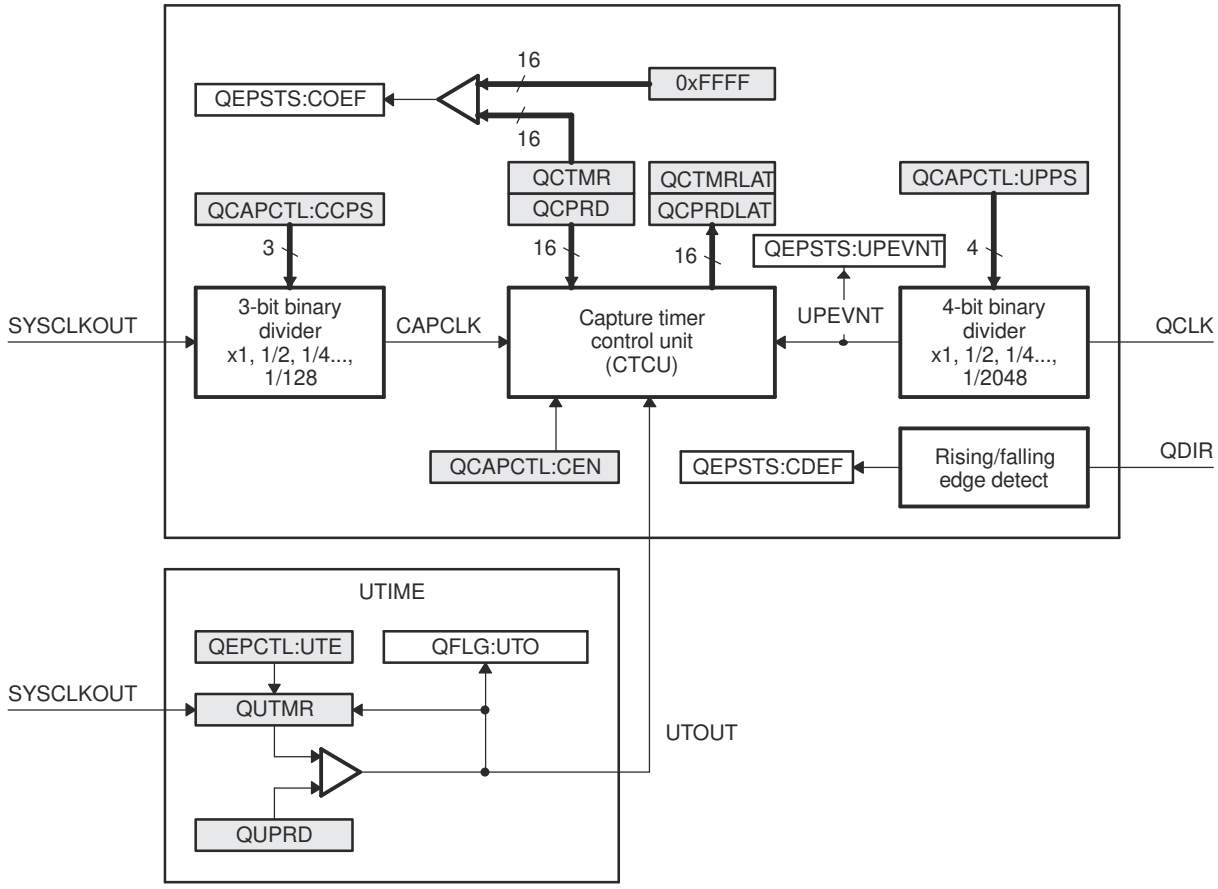
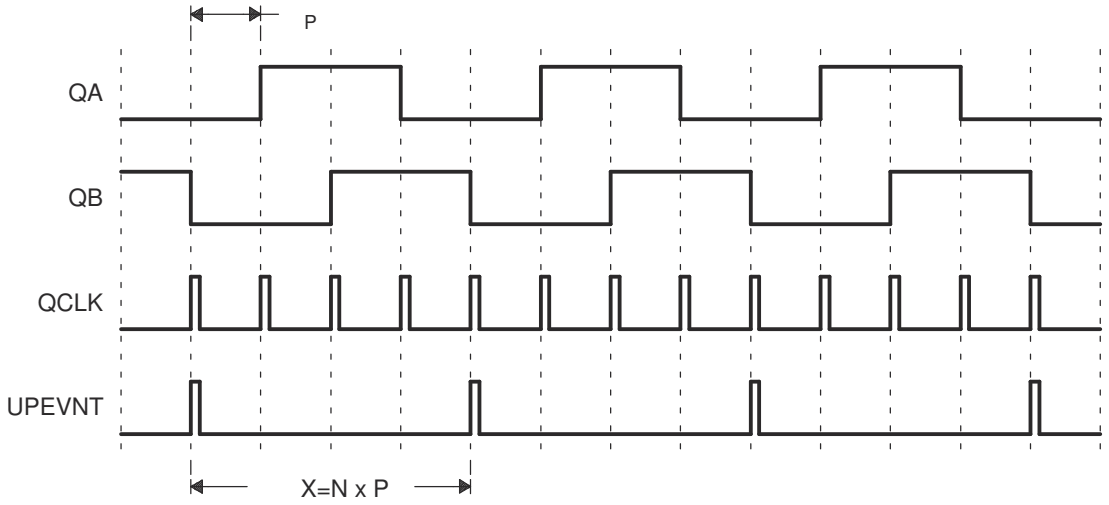


Figure 23-17. eQEP Edge Capture Unit

Note

The QCAPCTL[UPPS] prescaler should not be modified dynamically (such as switching the unit event prescaler from QCLK/4 to QCLK/8). Doing so may result in undefined behavior. The QCAPCTL[CPPS] prescaler can be modified dynamically (such as switching CAPCLK prescaling mode from SYSCLK/4 to SYSCLK/8) only after the capture unit is disabled.



N = Number of quadrature periods selected using QCAPCTL[UPPS] bits

Figure 23-18. Unit Position Event for Low Speed Measurement (QCAPCTL[UPPS] = 0010)

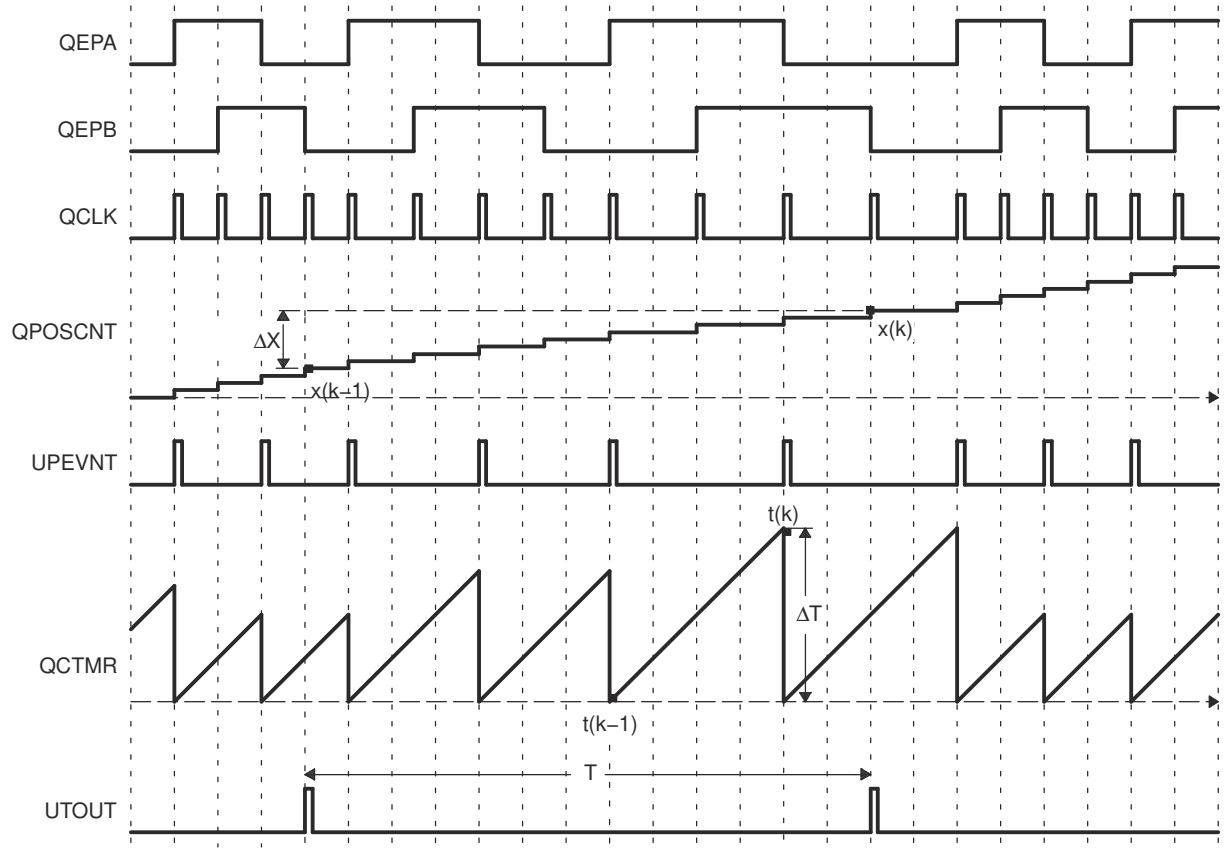


Figure 23-19. eQEP Edge Capture Unit - Timing Details

Velocity calculation equation:

$$v(k) = \frac{x(k) - x(k-1)}{T} = \frac{\Delta X}{T} \quad (22)$$

where:

- v(k) = Velocity at time instant k
- x(k) = Position at time instant k
- x(k-1) = Position at time instant k-1
- T = Fixed unit time or inverse of velocity calculation rate
- ΔX = Incremental position movement in unit time
- X = Fixed unit position
- ΔT = Incremental time elapsed for unit position movement
- t(k) = Time instant "k"
- t(k-1) = Time instant "k-1"

Unit time (T) and unit period (X) are configured using the QUPRD and QCAPCTL[UPPS] registers. Incremental position output and incremental time output is available in the QPOSLAT and QCPRDLAT registers.

Parameter	Relevant Register to Configure or Read the Information
T	Unit Period Register (QUPRD)
ΔX	Incremental Position = QPOSLAT(k) - QPOSLAT(K-1)
X	Fixed unit position defined by sensor resolution and ZCAPCTL[UPPS] bits
ΔT	Capture Period Latch (QCPRDLAT)

23.8 eQEP Watchdog

The eQEP peripheral contains a 16-bit watchdog timer (Figure 23-20) that monitors the quadrature-clock to indicate proper operation of the motion-control system. The eQEP watchdog timer is clocked from SYSCLKOUT/64 and the quadrate clock event (pulse) resets the watchdog timer. If no quadrature-clock event is detected until a period match (QWDPRD = QWDTMR), then the watchdog timer will time out and the watchdog interrupt flag will be set (QFLG[WTO]). The time-out value is programmable through the watchdog period register (QWDPRD).

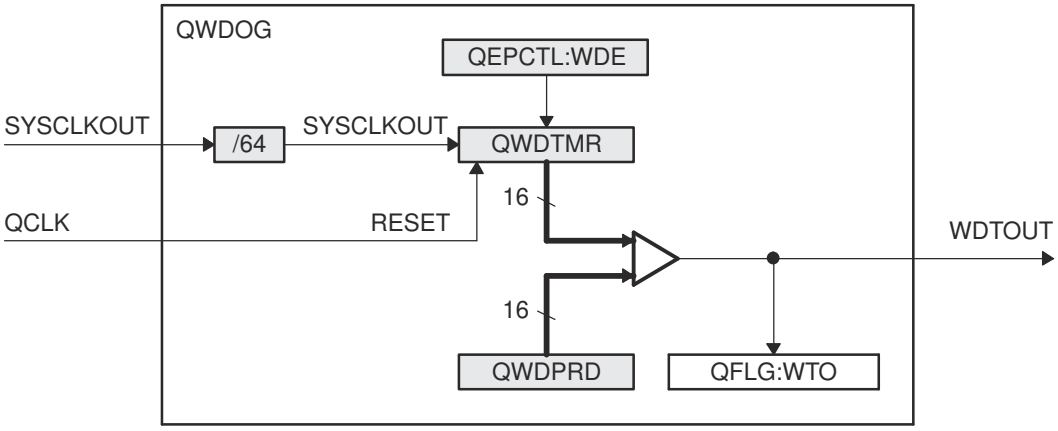


Figure 23-20. eQEP Watchdog Timer

23.9 eQEP Unit Timer Base

The eQEP peripheral includes a 32-bit timer (QUTMR) that is clocked by SYSCLKOUT to generate periodic interrupts for velocity calculations, see Figure 23-21. Whenever the unit timer (QUTMR) matches the unit period register (QUPRD), it resets the unit timer (QUTMR) and also generates the unit time out interrupt flag (QFLG[UTO]). The unit timer gets reset whenever timer value equals to configured period value.

The eQEP peripheral can be configured to latch the position counter, capture timer, and capture period values on a unit time out event so that latched values are used for velocity calculation as described in Section 23.7.

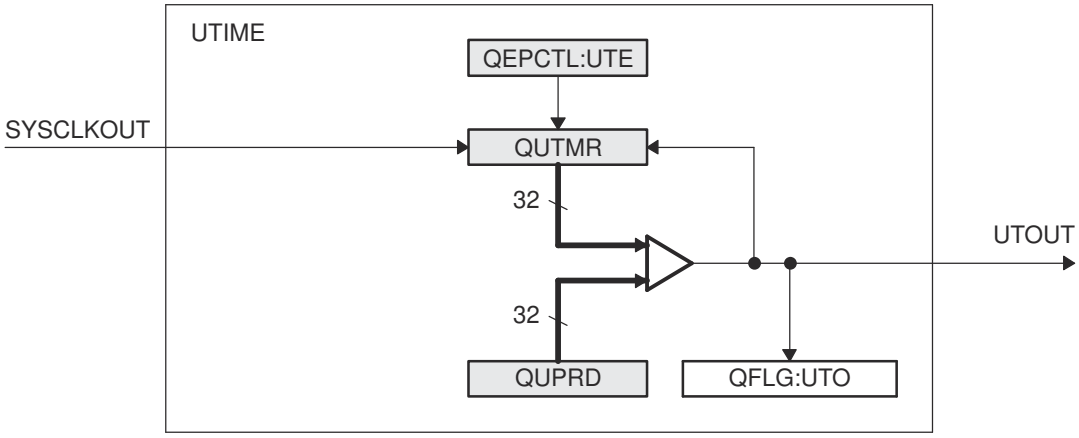


Figure 23-21. eQEP Unit Timer Base

23.10 QMA Module

The QEP Mode Adapter (QMA) is designed to extend the C2000™ eQEP module capabilities to support the additional modes described. Figure 23-22 depicts how the QMA module is integrated into to the eQEP module.

At reset, by default QMA logic is bypassed and the EQEPA and EQEPB inputs from the pins go directly into the eQEP module. When QMA module is enabled by configuring the QMACTRL[MODE] register, the EQEPA and EQEPB input are processed by this module and modified version of EQEPA and EQEPB signals are sent to the eQEP module. The QMA module requires the eQEP module to be configured in the Direction-Count mode and generates a clock signal on EQEPA input and direction signal on EQEPB input as needed for the proper operation of the intended mode.

- The xCLKMOD block inside the QMA module looks at the transitions on external EQEPA and EQEPB signals to generate the clock signal on the EQEPA input to the eQEP module.
- The xDIRMOD block inside the QMA module looks at the transitions on external EQEPA and EQEPB signals to generate the direction signal on the EQEPB input to the eQEP module.

The QMA module has error detection logic to detect illegal transitions on EQEPA and EQEPB input signals. The QMA module's error and interrupt are integrated inside the eQEP module as described in Section 23.11. In addition, the QMACTRL register configuration can be locked using the QMALOCK register. Refer to the register description for more details.

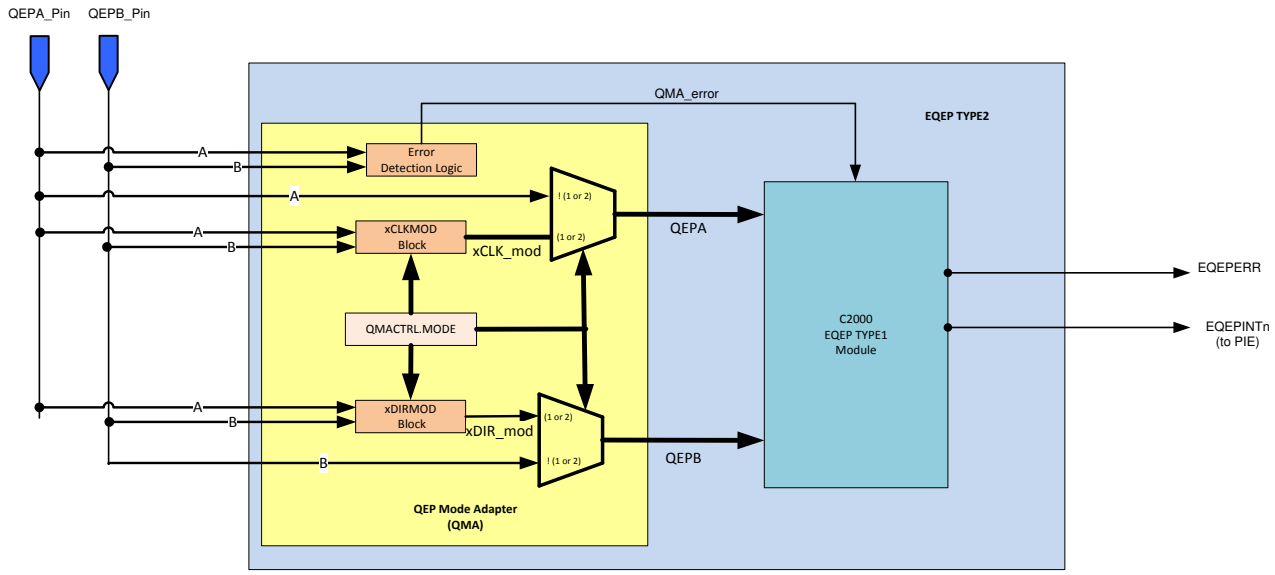


Figure 23-22. QMA Module Block Diagram

23.10.1 Modes of Operation

The QMA module can be operated in the following modes by configuring the QMACTRL register:

- QMA Mode-1 (QMACTRL[MODE]=1)
- QMA Mode-2 (QMACTRL[MODE]=2)

23.10.1.1 QMA Mode-1 (QMACTRL[MODE]=1)

This mode is used when the default state of EQEPA and EQEPB inputs is high. In this mode, outputs of QMA correspond to the following as shown in Figure 23-23:

- EQEPA Output of QMA is the AND of EQEPA and EQEPB inputs coming from the pin
- EQEPB Output of QMA is the direction signal generated by QMA based on EQEPA and EQEPB inputs

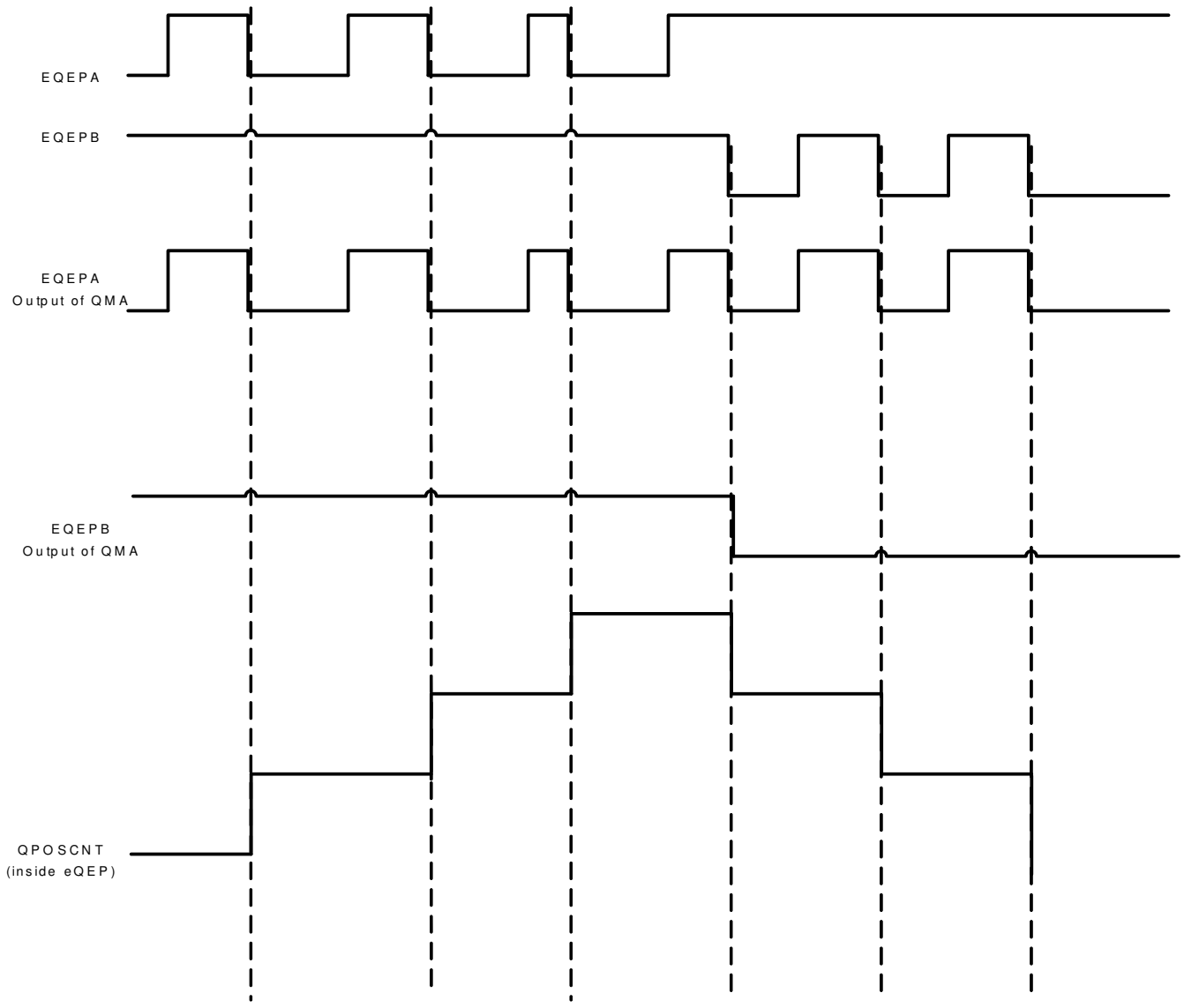


Figure 23-23. QMA Mode-1

23.10.1.2 QMA Mode-2 (QMACTRL[MODE]=2)

This mode is used when the default state of EQEPA and EQEPB inputs is low. In this mode, outputs of QMA correspond to the following as shown in Figure 23-24:

- EQEPA Output of QMA is the OR of EQEPA and EQEPB inputs coming from the pin
- EQEPB Output of QMA is the direction signal generated by QMA based on EQEPA and EQEPB inputs

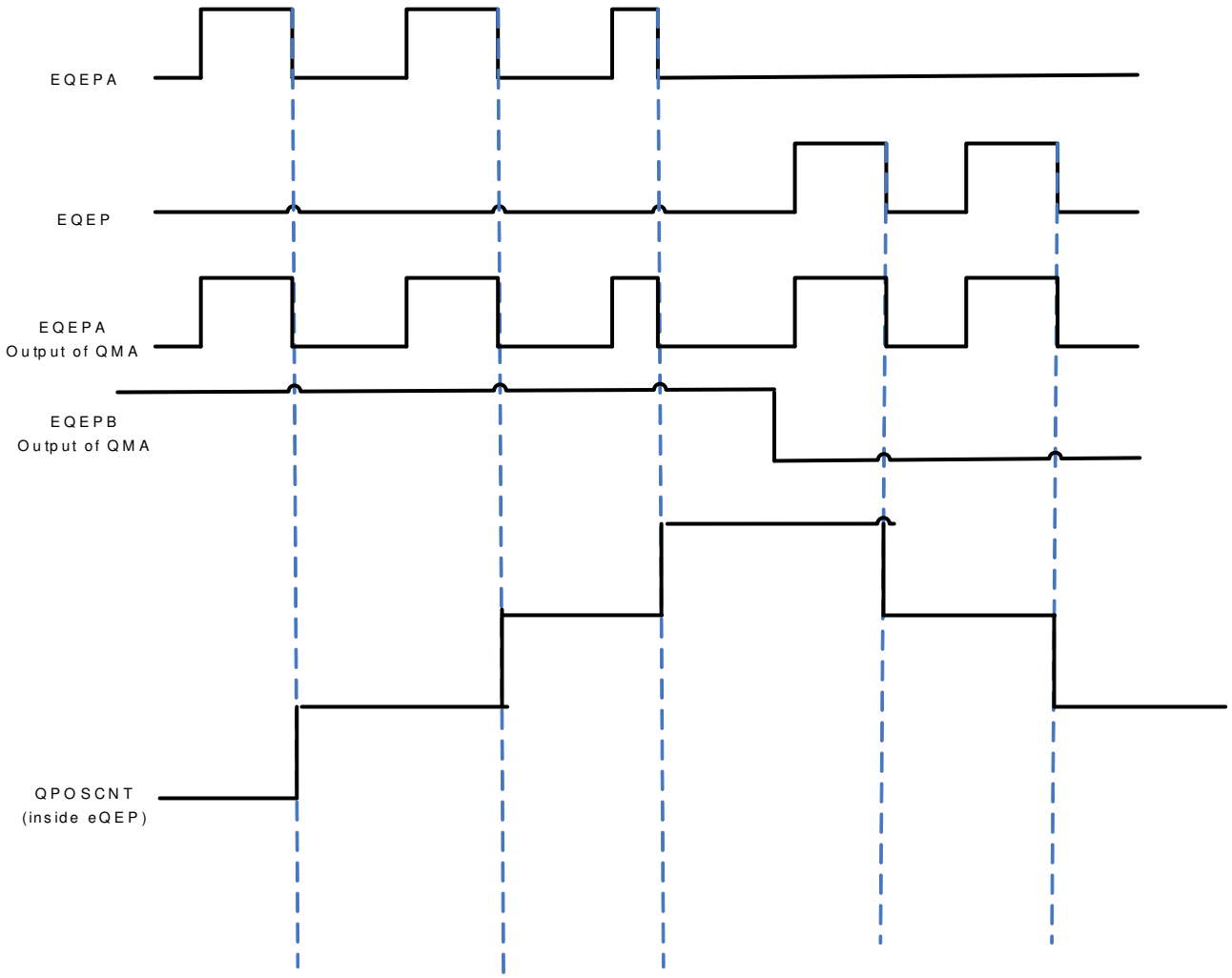


Figure 23-24. QMA Mode-2

23.10.2 Interrupt and Error Generation

The error detection logic detects illegal transitions on EQEPA and EQEPB signals and generates an error signal. This error signal can be used to generate eQEP interrupt and error output. Refer to Section 23.11 for details.

23.11 eQEP Interrupt Structure

Figure 23-25 shows how the interrupt mechanism works in the eQEP module.

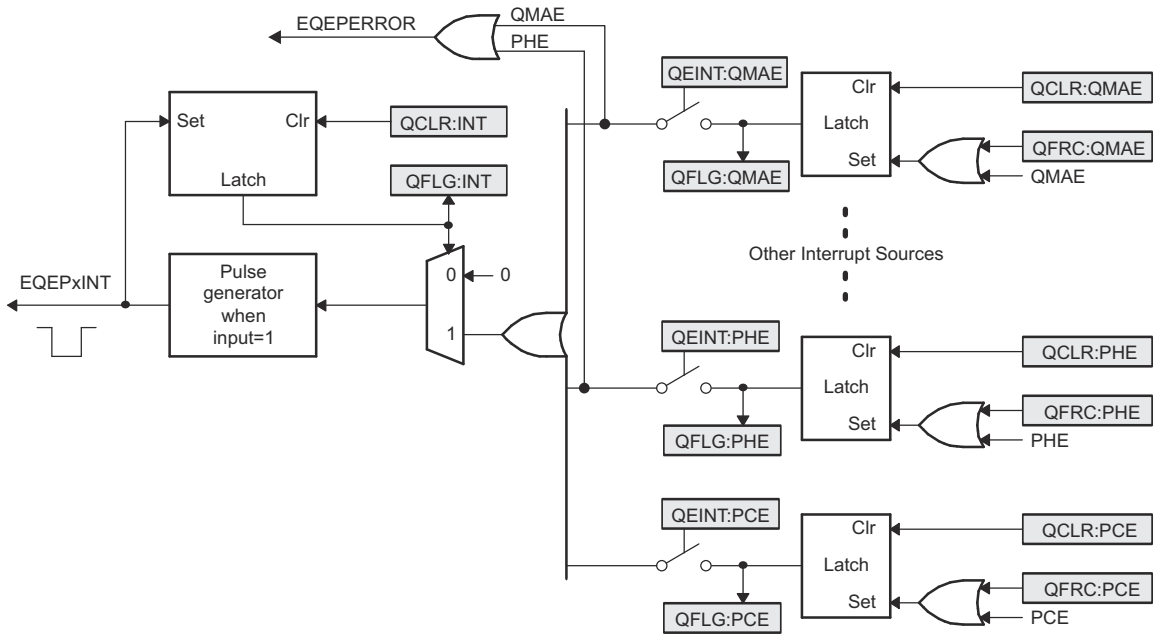


Figure 23-25. eQEP Interrupt Generation

Eleven interrupt events (PCE, PHE, QDC, WTO, PCU, PCO, PCR, PCM, SEL, IEL and UTO) can be generated. The interrupt control register (QEINT) is used to enable/disable individual interrupt event sources. The interrupt flag register (QFLG) indicates if any interrupt event has been latched and contains the global interrupt flag bit (INT).

An interrupt pulse is generated to PIE when:

1. Interrupt is enabled for eQEP event inside QEINT register
2. Interrupt flag for eQEP event inside QFLG register is set, and
3. Global interrupt status flag bit QFLG[INT] had been cleared for previously generated interrupt event. The interrupt service routine will need to clear the global interrupt flag bit and the serviced event, by way of the interrupt clear register (QCLR), before any other interrupt pulses are generated. If either flags inside the QFLG register are not cleared, further interrupt events will not generate an interrupt to PIE. You can force an interrupt event by way of the interrupt force register (QFRC), which is useful for test purposes.

23.12 Software

23.12.1 EQEP Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/eqep

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

23.12.1.1 Frequency Measurement Using eQEP

FILE: eqep_ex1_freq_cal.c

This example will calculate the frequency of an input signal using the eQEP module. ePWM1A is configured to generate this input signal with a frequency of 5 kHz. It will interrupt once every period and call the frequency calculation function. This example uses the IQMath library to simplify high-precision calculations.

In addition to the main example file, the following files must be included in this project:

- *eqep_ex1_calculation.c* - contains frequency calculation function
- *eqep_ex1_calculation.h* - includes initialization values for frequency structure

The configuration for this example is as follows

- Maximum frequency is configured to 10 kHz (baseFreq)
- Minimum frequency is assumed at 50 Hz for capture pre-scalar selection

SPEED_FR: High Frequency Measurement is obtained by counting the external input pulses for 10 ms (unit timer set to 100 Hz). \[SPEED_FR = \frac{Count\ Delta}{10 ms} \]

SPEED_PR: Low Frequency Measurement is obtained by measuring time period of input edges. Time measurement is averaged over 64 edges for better results and the capture unit performs the time measurement using pre-scaled SYSCLK.

Note that the pre-scaler for capture unit clock is selected such that the capture timer does not overflow at the required minimum frequency. This example runs indefinitely until the user stops it.

For more information about the frequency calculation see the comments at the beginning of eqep_ex1_calculation.c and the XLS file provided with the project, eqep_ex1_calculation.xls.

External Connections

- Connect GPIO25/eQEP1A to GPIO0/ePWM1A

Watch Variables

- *freq.freqHzFR* - Frequency measurement using position counter/unit time out
- *freq.freqHzPR* - Frequency measurement using capture unit

23.12.1.2 Position and Speed Measurement Using eQEP

FILE: eqep_ex2_pos_speed.c

This example provides position and speed measurement using the capture unit and speed measurement using unit time out of the eQEP module. ePWM1 and a GPIO are configured to generate simulated eQEP signals. The ePWM module will interrupt once every period and call the position/speed calculation function. This example uses the IQMath library to simplify high-precision calculations.

In addition to the main example file, the following files must be included in this project:

- *eqep_ex2_calculation.c* - contains position/speed calculation function
- *eqep_ex2_calculation.h* - includes initialization values for position/speed structure

The configuration for this example is as follows

- Maximum speed is configured to 6000 rpm (baseRPM)
- Minimum speed is assumed at 10 rpm for capture pre-scalar selection
- Pole pair is configured to 2 (polePairs)
- Encoder resolution is configured to 4000 counts/revolution (mechScaler)

- Which means: 4000 / 4 = 1000 line/revolution quadrature encoder (simulated by ePWM1)
- ePWM1 (simulating QEP encoder signals) is configured for a 5 kHz frequency or 300 rpm (= 4 * 5000 counts/sec * 60 sec/min) / 4000 counts/rev)

SPEEDRPM_FR: High Speed Measurement is obtained by counting the QEP input pulses for 10 ms (unit timer set to 100 Hz).

SPEEDRPM_FR = (Position Delta / 10 ms) * 60 rpm

SPEEDRPM_PR: Low Speed Measurement is obtained by measuring time period of QEP edges. Time measurement is averaged over 64 edges for better results and the capture unit performs the time measurement using pre-scaled SYSCLK.

Note that the pre-scaler for capture unit clock is selected such that the capture timer does not overflow at the required minimum frequency. This example runs indefinitely until the user stops it.

For more information about the position/speed calculation see the comments at the beginning of eqep_ex2_calculation.c and the XLS file provided with the project, eqep_ex2_calculation.xls.

External Connections

- Connect GPIO25/eQEP1A to GPIO0/ePWM1A (simulates eQEP Phase A signal)
- Connect GPIO29/eQEP1B to GPIO1/ePWM1B (simulates eQEP Phase B signal)
- Connect GPIO23/eQEP1I to GPIO2 (simulates eQEP Index Signal)

Watch Variables

- *posSpeed.speedRPMFR* - Speed meas. in rpm using QEP position counter
- *posSpeed.speedRPMPR* - Speed meas. in rpm using capture unit
- *posSpeed.thetaMech* - Motor mechanical angle (Q15)
- *posSpeed.thetaElec* - Motor electrical angle (Q15)

23.12.1.3 ePWM Frequency Measurement Using eQEP with X-BAR Connection

FILE: eqep_ex3_epwm_xbar.c

This example will calculate the frequency of an PWM signal using the eQEP module. ePWM1A is configured to generate this input signal with a frequency of 5 kHz. This ePWM signal is connected to input of eQEP using Input CrossBar and EPWM XBAR. ePWM module will interrupt once every period and call the frequency calculation function. This example uses the IQMath library to simplify high-precision calculations.

In addition to the main example file, the following files must be included in this project:

- *eqep_ex1_calculation.c* - contains frequency calculation function
- *eqep_ex1_calculation.h* - includes initialization values for frequency structure

The configuration for this example is as follows

- Maximum frequency is configured to 10 kHz (baseFreq)
- Minimum frequency is assumed at 50 Hz for capture pre-scalar selection
- GPIO0 is connected to output of INPUT_XBAR1
- INPUT_XBAR1 is connected to output of PWMXBAR at TRIP4
- eQEPA source is configured as PWMXBAR.1 output (TRIP4)

SPEED_FR: High Frequency Measurement is obtained by counting the external input pulses for 10 ms (unit timer set to 100 Hz). \[SPEED_FR = \frac{Count\ Delta}{10 ms} \]

SPEED_PR: Low Frequency Measurement is obtained by measuring time period of input edges. Time measurement is averaged over 64 edges for better results and the capture unit performs the time measurement using pre-scaled SYSCLK.

Note that the pre-scaler for capture unit clock is selected such that the capture timer does not overflow at the required minimum frequency. This example runs indefinitely until the user stops it.

For more information about the frequency calculation see the comments at the beginning of eqep_ex1_calculation.c and the XLS file provided with the project, eqep_ex1_calculation.xls.

Watch Variables

- *freq.freqHzFR* - Frequency measurement using position counter/unit time out
- *freq.freqHzPR* - Frequency measurement using capture unit

23.12.1.4 Frequency Measurement Using eQEP with Unit Timeout Interrupt

FILE: eqep_ex4_freq_cal_interrupt.c

This example will calculate the frequency of an input signal using the eQEP module. ePWM1A is configured to generate this input signal with a frequency of 5 kHz. EQEP unit timeout is set which will generate an interrupt every *UNIT_PERIOD* microseconds and frequency calculation occurs continuously

The configuration for this example is as follows

- PWM frequency is specified as 5000 Hz
- UNIT_PERIOD is specified as 10000 µs
- Minimum frequency is (1/(2*10ms)), that is 50 Hz
- Highest frequency can be (2^32)/ ((2*10 ms))
- Resolution of frequency measurement is 50 Hz

freq : Frequency Measurement is obtained by counting the external input pulses for UNIT_PERIOD (unit timer set to 10 ms).

External Connections

- Connect GPIO25/eQEP1A to GPIO0/ePWM1A

Watch Variables

- *freq* - Frequency measurement using position counter/unit time out
- *pass* - If measured frequency matches with PWM frequency then pass = 1, else 0

23.12.1.5 Motor Speed and Direction Measurement Using eQEP with Unit Timeout Interrupt

FILE: eqep_ex5_speed_dir_motor.c

This example can be used to sense the speed and direction of motor using eQEP in quadrature encoder mode. ePWM1A is configured to simulate motor encoder signals with frequency of 5 kHz on both A and B pins with 90 degree phase shift (so as to run this example without motor). EQEP unit timeout is set which will generate an interrupt every *UNIT_PERIOD* microseconds and speed calculation occurs continuously based on the direction of motor

The configuration for this example is as follows

- PWM frequency is specified as 5000 Hz
- UNIT_PERIOD is specified as 10000 µs
- Simulated quadrature signal frequency is 20000 Hz (4 * 5000)
- Encoder holes assumed as 1000
- Thus Simulated motor speed is 300 rpm (5000 * (60 / 1000))

freq : Simulated quadrature signal frequency measured by counting the external input pulses for UNIT_PERIOD (unit timer set to 10 ms).

speed : Measure motor speed in rpm

dir : Indicates clockwise (1) or anticlockwise (-1)

External Connections (if motor encoder signals are simulated by ePWM)

- Connect GPIO25/eQEP1A to GPIO0/ePWM1A
- Connect GPIO29/eQEP1B to GPIO1/ePWM1B with motor
- Comment in "MOTOR" includes
- Connect GPIO25/eQEP1A to encoder A output
- Connect GPIO29/eQEP1B to encoder B output

Watch Variables

- *freq* : Simulated motor frequency measurement is obtained by counting the external input pulses for UNIT_PERIOD (unit timer set to 10 ms).
- *speed* : Measure motor speed in rpm
- *dir* : Indicates clockwise (1) or anticlockwise (-1)
- *pass* : If measured qudrature frequency matches with input quadrature frequency (4 * PWM frequency) then pass = 1, else fail = 1 (** only when "MOTOR" is commented out)

23.13 eQEP Registers

This section describes the Enhanced Quadrature Encoder Pulse Registers.

23.13.1 EQEP Base Address Table

Table 23-4. EQEP Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
EQep1Regs	EQEP_REGS	EQEP1_BASE	0x0000_5100	YES	YES	YES	YES	YES
EQep2Regs	EQEP_REGS	EQEP2_BASE	0x0000_5140	YES	YES	YES	YES	YES

23.13.2 EQEP_REGS Registers

Table 23-5 lists the memory-mapped registers for the EQEP_REGS registers. All register offset addresses not listed in Table 23-5 should be considered as reserved locations and the register contents should not be modified.

Table 23-5. EQEP_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	QPOSCNT	Position Counter		Go
2h	QPOSINIT	Position Counter Init		Go
4h	QPOSMAX	Maximum Position Count		Go
6h	QPOSCMP	Position Compare		Go
8h	QPOSILAT	Index Position Latch		Go
Ah	QPOSSLAT	Strobe Position Latch		Go
Ch	QPOSLAT	Position Latch		Go
Eh	QUTMR	QEP Unit Timer		Go
10h	QUPRD	QEP Unit Period		Go
12h	QWDTMR	QEP Watchdog Timer		Go
13h	QWDPRD	QEP Watchdog Period		Go
14h	QDECCTL	Quadrature Decoder Control		Go
15h	QEPCTL	QEP Control		Go
16h	QCAPCTL	Qaudrature Capture Control		Go
17h	QPOSCTL	Position Compare Control		Go
18h	QEINT	QEP Interrupt Control		Go
19h	QFLG	QEP Interrupt Flag		Go
1Ah	QCLR	QEP Interrupt Clear		Go
1Bh	QFRC	QEP Interrupt Force		Go
1Ch	QEPSTS	QEP Status		Go
1Dh	QCTMR	QEP Capture Timer		Go
1Eh	QCPRD	QEP Capture Period		Go
1Fh	QCTMRLAT	QEP Capture Latch		Go
20h	QCPRDLAT	QEP Capture Period Latch		Go
30h	REV	QEP Revision Number		Go
32h	QEPSTROBESEL	QEP Strobe select register		Go
34h	QMACTRL	QMA Control register		Go
36h	QEPSRCSEL	QEP Source Select Register		Go

Complex bit access types are encoded to fit into small table cells. Table 23-6 shows the codes that are used for access types in this section.

Table 23-6. EQEP_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		

Table 23-6. EQEP_REGS Access Type Codes (continued)

Access Type	Code	Description
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

23.13.2.1 QPOSCNT Register (Offset = 0h) [Reset = 0h]

QPOSCNT is shown in Figure 23-25 and described in Table 23-7.

Return to the Summary Table.

Position Counter

Figure 23-25. QPOSCNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSCNT																															
R/W-0h																															

Table 23-7. QPOSCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSCNT	R/W	0h	Position Counter This 32-bit position counter register counts up/down on every eQEP pulse based on direction input. This counter acts as a position integrator whose count value is proportional to position from a give reference point. This Register acts as a Read ONLY register while counter is counting up/down. Note: It is recommended to only write to the position counter register (QPOSCNT) during initialization, i.e. when the eQEP position counter is disabled (QPEN bit of QEPCTL is zero). Once the position counter is enabled (QPEN bit is one), writing to the eQEP position counter register (QPOSCNT) may cause unexpected results. Reset type: SYSRSn

23.13.2.2 QPOSINIT Register (Offset = 2h) [Reset = 0h]

QPOSINIT is shown in Figure 23-26 and described in Table 23-8.

Return to the Summary Table.

Position Counter Init

Figure 23-26. QPOSINIT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSINIT																															
R/W-0h																															

Table 23-8. QPOSINIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSINIT	R/W	0h	Position Counter Init This register contains the position value that is used to initialize the position counter based on external strobe or index event. The position counter can be initialized through software. Writes to this register should always be full 32-bit writes. Reset type: SYSRSn

23.13.2.3 QPOSMAX Register (Offset = 4h) [Reset = 0h]

QPOSMAX is shown in Figure 23-27 and described in Table 23-9.

Return to the Summary Table.

Maximum Position Count

Figure 23-27. QPOSMAX Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSMAX																															
R/W-0h																															

Table 23-9. QPOSMAX Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSMAX	R/W	0h	Maximum Position Count This register contains the maximum position counter value. Writes to this register should always be full 32-bit writes. Reset type: SYSRSn

23.13.2.4 QPOSCMP Register (Offset = 6h) [Reset = 0h]

QPOSCMP is shown in Figure 23-28 and described in Table 23-10.

Return to the Summary Table.

Position Compare

Figure 23-28. QPOSCMP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSCMP																															
R/W-0h																															

Table 23-10. QPOSCMP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSCMP	R/W	0h	Position Compare The position-compare value in this register is compared with the position counter (QPOSCNT) to generate sync output and/or interrupt on compare match. Reset type: SYSRSn

23.13.2.5 QPOSILAT Register (Offset = 8h) [Reset = 0h]

QPOSILAT is shown in Figure 23-29 and described in Table 23-11.

Return to the Summary Table.

Index Position Latch

Figure 23-29. QPOSILAT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSILAT																															
R-0h																															

Table 23-11. QPOSILAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSILAT	R	0h	Index Position Latch The position-counter value is latched into this register on an index event as defined by the QEPCTL[IEL] bits. Reset type: SYSRSn

23.13.2.6 QPOSSLAT Register (Offset = Ah) [Reset = 0h]

QPOSSLAT is shown in Figure 23-30 and described in Table 23-12.

Return to the Summary Table.

Strobe Position Latch

Figure 23-30. QPOSSLAT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSSLAT																															
R-0h																															

Table 23-12. QPOSSLAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSSLAT	R	0h	Strobe Position Latch The position-counter value is latched into this register on a strobe event as defined by the QEPCTL[SEL] bits. Reset type: SYSRSn

23.13.2.7 QPOSLAT Register (Offset = Ch) [Reset = 0h]

QPOSLAT is shown in Figure 23-31 and described in Table 23-13.

Return to the Summary Table.

Position Latch

Figure 23-31. QPOSLAT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QPOSLAT																															
R-0h																															

Table 23-13. QPOSLAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QPOSLAT	R	0h	Position Latch The position-counter value is latched into this register on a unit time out event. Reset type: SYSRSn

23.13.2.8 QUTMR Register (Offset = Eh) [Reset = 0h]

QUTMR is shown in Figure 23-32 and described in Table 23-14.

Return to the Summary Table.

QEP Unit Timer

Figure 23-32. QUTMR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QUTMR																															
R/W-0h																															

Table 23-14. QUTMR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QUTMR	R/W	0h	QEP Unit Timer This register acts as time base for unit time event generation. When this timer value matches the unit time period value a unit time event is generated. Reset type: SYSRSn

23.13.2.9 QUPRD Register (Offset = 10h) [Reset = 0h]

QUPRD is shown in Figure 23-33 and described in Table 23-15.

Return to the Summary Table.

QEP Unit Period

Figure 23-33. QUPRD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QUPRD																															
R/W-0h																															

Table 23-15. QUPRD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	QUPRD	R/W	0h	QEP Unit Period This register contains the period count for the unit timer to generate periodic unit time events. These events latch the eQEP position information at periodic intervals and optionally generate an interrupt. Writes to this register should always be full 32-bit writes. Reset type: SYSRSn

23.13.2.10 QWDTMR Register (Offset = 12h) [Reset = 0h]

QWDTMR is shown in Figure 23-34 and described in Table 23-16.

Return to the Summary Table.

QEP Watchdog Timer

Figure 23-34. QWDTMR Register

15	14	13	12	11	10	9	8
QWDTMR							
R/W-0h							
7	6	5	4	3	2	1	0
QWDTMR							
R/W-0h							

Table 23-16. QWDTMR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	QWDTMR	R/W	0h	QEP Watchdog Timer This register acts as time base for the watchdog to detect motor stalls. When this timer value matches with the watchdog's period value a watchdog timeout interrupt is generated. This register is reset upon edge transition in quadrature-clock indicating the motion. Reset type: SYSRSn

23.13.2.11 QWDPRD Register (Offset = 13h) [Reset = 0h]

QWDPRD is shown in Figure 23-35 and described in Table 23-17.

Return to the Summary Table.

QEP Watchdog Period

Figure 23-35. QWDPRD Register

15	14	13	12	11	10	9	8
QWDPRD							
R/W-0h							
7	6	5	4	3	2	1	0
QWDPRD							
R/W-0h							

Table 23-17. QWDPRD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	QWDPRD	R/W	0h	QEP Watchdog Period This register contains the time-out count for the eQEP peripheral watch dog timer. When the watchdog timer value matches the watchdog period value, a watchdog timeout interrupt is generated. Reset type: SYSRSn

23.13.2.12 QDECCTL Register (Offset = 14h) [Reset = 0h]

QDECCTL is shown in Figure 23-36 and described in Table 23-18.

Return to the Summary Table.

Quadrature Decoder Control

Figure 23-36. QDECCTL Register

15	14	13	12	11	10	9	8
QSRC		SOEN	SPSEL	XCR	SWAP	IGATE	QAP
R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
QBP	QIP	QSP	RESERVED				QIDIRE
R/W-0h	R/W-0h	R/W-0h	R-0h				R/W-0h

Table 23-18. QDECCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	QSRC	R/W	0h	Position-counter source selection Reset type: SYSRSn
13	SOEN	R/W	0h	Sync output-enable Reset type: SYSRSn 0h (R/W) = Disable position-compare sync output 1h (R/W) = Enable position-compare sync output
12	SPSEL	R/W	0h	Sync output pin selection Reset type: SYSRSn 0h (R/W) = Index pin is used for sync output 1h (R/W) = Strobe pin is used for sync output
11	XCR	R/W	0h	External Clock Rate Reset type: SYSRSn 0h (R/W) = 2x resolution: Count the rising/falling edge 1h (R/W) = 1x resolution: Count the rising edge only
10	SWAP	R/W	0h	CLK/DIR Signal Source for Position Counter Reset type: SYSRSn 0h (R/W) = Quadrature-clock inputs are not swapped 1h (R/W) = Quadrature-clock inputs are swapped
9	IGATE	R/W	0h	Index pulse gating option Reset type: SYSRSn 0h (R/W) = Disable gating of Index pulse 1h (R/W) = Gate the index pin with strobe
8	QAP	R/W	0h	QEPA input polarity Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Negates QEPA input
7	QBP	R/W	0h	QEPB input polarity Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Negates QEPB input
6	QIP	R/W	0h	QEPI input polarity Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Negates QEPI input
5	QSP	R/W	0h	QEPS input polarity Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Negates QEPS input
4-1	RESERVED	R	0h	Reserved

Table 23-18. QDECCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	QIDIRE	R/W	0h	0 - Compatible mode, Behavior same as existing devices 1 - Enhancement for Direction change during Index will be enabled Reset type: SYSRSn

23.13.2.13 QEPCTL Register (Offset = 15h) [Reset = 0h]

QEPCTL is shown in Figure 23-37 and described in Table 23-19.

Return to the Summary Table.

QEP Control

Figure 23-37. QEPCTL Register

15	14	13	12	11	10	9	8
FREE_SOFT		PCRM		SEI		IEI	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
SWI	SEL	IEL		QPEN	QCLM	UTE	WDE
R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 23-19. QEPCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	FREE_SOFT	R/W	0h	Emulation mode Reset type: SYSRSn 0h (R/W) = QPOSCNT behavior Position counter stops immediately on emulation suspend 0h (R/W) = QWDTMR behavior Watchdog counter stops immediately 0h (R/W) = QUTMR behavior Unit timer stops immediately 0h (R/W) = QCTMR behavior Capture Timer stops immediately 1h (R/W) = QPOSCNT behavior Position counter continues to count until the rollover 1h (R/W) = QWDTMR behavior Watchdog counter counts until WD period match roll over 1h (R/W) = QUTMR behavior Unit timer counts until period rollover 1h (R/W) = QCTMR behavior Capture Timer counts until next unit period event 2h (R/W) = QPOSCNT behavior Position counter is unaffected by emulation suspend 2h (R/W) = QWDTMR behavior Watchdog counter is unaffected by emulation suspend 2h (R/W) = QUTMR behavior Unit timer is unaffected by emulation suspend 2h (R/W) = QCTMR behavior Capture Timer is unaffected by emulation suspend 3h (R/W) = Same as FREE_SOFT_2
13-12	PCRM	R/W	0h	Postion counter reset Reset type: SYSRSn 0h (R/W) = Position counter reset on an index event 1h (R/W) = Position counter reset on the maximum position 2h (R/W) = Position counter reset on the first index event 3h (R/W) = Position counter reset on a unit time event
11-10	SEI	R/W	0h	Strobe event initialization of position counter Reset type: SYSRSn 0h (R/W) = Does nothing (action disabled) 1h (R/W) = Does nothing (action disabled) 2h (R/W) = Initializes the position counter on rising edge of the QEPS signal 3h (R/W) = Clockwise Direction: Initializes the position counter on the rising edge of QEPS strobe Counter Clockwise Direction: Initializes the position counter on the falling edge of QEPS strobe

Table 23-19. QEPCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9-8	IEI	R/W	0h	Index event init of position count Reset type: SYSRSn 0h (R/W) = Do nothing (action disabled) 1h (R/W) = Do nothing (action disabled) 2h (R/W) = Initializes the position counter on the rising edge of the QEPI signal (QPOSCNT = QPOSINIT) 3h (R/W) = Initializes the position counter on the falling edge of QEPI signal (QPOSCNT = QPOSINIT)
7	SWI	R/W	0h	Software init position counter Reset type: SYSRSn 0h (R/W) = Do nothing (action disabled) 1h (R/W) = Initialize position counter (QPOSCNT=QPOSINIT). This bit is not cleared automatically
6	SEL	R/W	0h	Strobe event latch of position counter Reset type: SYSRSn 0h (R/W) = The position counter is latched on the rising edge of QEPS strobe (QPOSSLAT = POSCCNT). Latching on the falling edge can be done by inverting the strobe input using the QSP bit in the QDECCTL register 1h (R/W) = Clockwise Direction: Position counter is latched on rising edge of QEPS strobe Counter Clockwise Direction: Position counter is latched on falling edge of QEPS strobe
5-4	IEL	R/W	0h	Index event latch of position counter (software index marker) Reset type: SYSRSn 0h (R/W) = Reserved 1h (R/W) = Latches position counter on rising edge of the index signal 2h (R/W) = Latches position counter on falling edge of the index signal 3h (R/W) = Software index marker. Latches the position counter and quadrature direction flag on index event marker. The position counter is latched to the QPOSILAT register and the direction flag is latched in the QEPSTS[QDLF] bit. This mode is useful for software index marking.
3	QPEN	R/W	0h	Quadrature position counter enable/software reset Reset type: SYSRSn 0h (R/W) = Reset the eQEP peripheral internal operating flags/read-only registers. Control/configuration registers are not disturbed by a software reset. When QPEN is disabled, some flags in the QFLG register do not get reset or cleared and show the actual state of that flag. 1h (R/W) = eQEP position counter is enabled
2	QCLM	R/W	0h	QEP capture latch mode Reset type: SYSRSn 0h (R/W) = Latch on position counter read by CPU. Capture timer and capture period values are latched into QCTMRLAT and QCPRDLAT registers when CPU reads the QPOSCNT register. 1h (R/W) = Latch on unit time out. Position counter, capture timer and capture period values are latched into QPOSLAT, QCTMRLAT and QCPRDLAT registers on unit time out.
1	UTE	R/W	0h	QEP unit timer enable Reset type: SYSRSn 0h (R/W) = Disable eQEP unit timer 1h (R/W) = Enable unit timer
0	WDE	R/W	0h	QEP watchdog enable Reset type: SYSRSn 0h (R/W) = Disable the eQEP watchdog timer 1h (R/W) = Enable the eQEP watchdog timer

23.13.2.14 QCAPCTL Register (Offset = 16h) [Reset = 0h]

QCAPCTL is shown in Figure 23-38 and described in Table 23-20.

Return to the Summary Table.

Qaudrature Capture Control

Figure 23-38. QCAPCTL Register

15	14	13	12	11	10	9	8
CEN	RESERVED						
R/W-0h	R-0h						
7	6	5	4	3	2	1	0
RESERVED	CCPS			UPPS			
R-0h	R/W-0h			R/W-0h			

Table 23-20. QCAPCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	CEN	R/W	0h	Enable eQEP capture Reset type: SYSRSn 0h (R/W) = eQEP capture unit is disabled 1h (R/W) = eQEP capture unit is enabled
14-7	RESERVED	R	0h	Reserved
6-4	CCPS	R/W	0h	eQEP capture timer clock prescaler Reset type: SYSRSn 0h (R/W) = CAPCLK = SYSCLKOUT/1 1h (R/W) = CAPCLK = SYSCLKOUT/2 2h (R/W) = CAPCLK = SYSCLKOUT/4 3h (R/W) = CAPCLK = SYSCLKOUT/8 4h (R/W) = CAPCLK = SYSCLKOUT/16 5h (R/W) = CAPCLK = SYSCLKOUT/32 6h (R/W) = CAPCLK = SYSCLKOUT/64 7h (R/W) = CAPCLK = SYSCLKOUT/128
3-0	UPPS	R/W	0h	Unit position event prescaler Reset type: SYSRSn 0h (R/W) = UPEVNT = QCLK/1 1h (R/W) = UPEVNT = QCLK/2 2h (R/W) = UPEVNT = QCLK/4 3h (R/W) = UPEVNT = QCLK/8 4h (R/W) = UPEVNT = QCLK/16 5h (R/W) = UPEVNT = QCLK/32 6h (R/W) = UPEVNT = QCLK/64 7h (R/W) = UPEVNT = QCLK/128 8h (R/W) = UPEVNT = QCLK/256 9h (R/W) = UPEVNT = QCLK/512 Ah (R/W) = UPEVNT = QCLK/1024 Bh (R/W) = UPEVNT = QCLK/2048 Ch (R/W) = Reserved Dh (R/W) = Reserved Eh (R/W) = Reserved Fh (R/W) = Reserved

23.13.2.15 QPOSCTL Register (Offset = 17h) [Reset = 0h]

QPOSCTL is shown in Figure 23-39 and described in Table 23-21.

Return to the Summary Table.

Position Compare Control

Figure 23-39. QPOSCTL Register

15	14	13	12	11	10	9	8
PCSHDW	PCLOAD	PCPOL	PCE	PCSPW			
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h			
7	6	5	4	3	2	1	0
PCSPW							
R/W-0h							

Table 23-21. QPOSCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	PCSHDW	R/W	0h	Position compare of shadow enable Reset type: SYSRSn 0h (R/W) = Shadow disabled, load Immediate 1h (R/W) = Shadow enabled
14	PCLOAD	R/W	0h	Position compare of shadow load Reset type: SYSRSn 0h (R/W) = Load on QPOSCNT = 0 1h (R/W) = Load when QPOSCNT = QPOSCMP
13	PCPOL	R/W	0h	Polarity of sync output Reset type: SYSRSn 0h (R/W) = Active HIGH pulse output 1h (R/W) = Active LOW pulse output
12	PCE	R/W	0h	Position compare enable/disable Reset type: SYSRSn 0h (R/W) = Disable position compare unit 1h (R/W) = Enable position compare unit
11-0	PCSPW	R/W	0h	Select-position-compare sync output pulse width Reset type: SYSRSn 0h (R/W) = 1 * 4 * SYSCLKOUT cycles 1h (R/W) = 2 * 4 * SYSCLKOUT cycles FFFh (R/W) = 4096 * 4 * SYSCLKOUT cycles

23.13.2.16 QEINT Register (Offset = 18h) [Reset = 0h]

QEINT is shown in Figure 23-40 and described in Table 23-22.

Return to the Summary Table.

QEP Interrupt Control

Figure 23-40. QEINT Register

15	14	13	12	11	10	9	8
RESERVED			QMAE	UTO	IEL	SEL	PCM
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
PCR	PCO	PCU	WTO	QDC	QPE	PCE	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h

Table 23-22. QEINT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12	QMAE	R/W	0h	QMA Error Interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt is disabled 1h (R/W) = Interrupt is enabled
11	UTO	R/W	0h	Unit time out interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt is disabled 1h (R/W) = Interrupt is enabled
10	IEL	R/W	0h	Index event latch interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt is disabled 1h (R/W) = Interrupt is enabled
9	SEL	R/W	0h	Strobe event latch interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt is disabled 1h (R/W) = Interrupt is enabled
8	PCM	R/W	0h	Position-compare match interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt is disabled 1h (R/W) = Interrupt is enabled
7	PCR	R/W	0h	Position-compare ready interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt is disabled 1h (R/W) = Interrupt is enabled
6	PCO	R/W	0h	Position counter overflow interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt is disabled 1h (R/W) = Interrupt is enabled
5	PCU	R/W	0h	Position counter underflow interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt is disabled 1h (R/W) = Interrupt is enabled
4	WTO	R/W	0h	Watchdog time out interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt is disabled 1h (R/W) = Interrupt is enabled
3	QDC	R/W	0h	Quadrature direction change interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt is disabled 1h (R/W) = Interrupt is enabled

Table 23-22. QEINT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	QPE	R/W	0h	Quadrature phase error interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt is disabled 1h (R/W) = Interrupt is enabled
1	PCE	R/W	0h	Position counter error interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt is disabled 1h (R/W) = Interrupt is enabled
0	RESERVED	R	0h	Reserved

23.13.2.17 QFLG Register (Offset = 19h) [Reset = 0h]

QFLG is shown in Figure 23-41 and described in Table 23-23.

Return to the Summary Table.

QEP Interrupt Flag

Figure 23-41. QFLG Register

15	14	13	12	11	10	9	8
RESERVED			QMAE	UTO	IEL	SEL	PCM
R-0h			R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
PCR	PCO	PCU	WTO	QDC	PHE	PCE	INT
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 23-23. QFLG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12	QMAE	R	0h	QMA Error interrupt flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = Interrupt was generated
11	UTO	R	0h	Unit time out interrupt flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = Set by eQEP unit timer period match
10	IEL	R	0h	Index event latch interrupt flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = This bit is set after latching the QPOSCNT to QPOSILAT
9	SEL	R	0h	Strobe event latch interrupt flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = This bit is set after latching the QPOSCNT to QPOSSLAT
8	PCM	R	0h	eQEP compare match event interrupt flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = This bit is set on position-compare match
7	PCR	R	0h	Position-compare ready interrupt flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = This bit is set after transferring the shadow register value to the active position compare register
6	PCO	R	0h	Position counter overflow interrupt flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = This bit is set on position counter overflow.
5	PCU	R	0h	Position counter underflow interrupt flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = This bit is set on position counter underflow.
4	WTO	R	0h	Watchdog timeout interrupt flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = Set by watchdog timeout
3	QDC	R	0h	Quadrature direction change interrupt flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = Interrupt was generated

Table 23-23. QFLG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	PHE	R	0h	Quadrature phase error interrupt flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = Set on simultaneous transition of QEPA and QEPB
1	PCE	R	0h	Position counter error interrupt flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = Position counter error
0	INT	R	0h	Global interrupt status flag Reset type: SYSRSn 0h (R/W) = No interrupt generated 1h (R/W) = Interrupt was generated

23.13.2.18 QCLR Register (Offset = 1Ah) [Reset = 0h]

QCLR is shown in Figure 23-42 and described in Table 23-24.

Return to the Summary Table.

QEP Interrupt Clear

Figure 23-42. QCLR Register

15	14	13	12	11	10	9	8
RESERVED			QMAE	UTO	IEL	SEL	PCM
R-0h			R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h
7	6	5	4	3	2	1	0
PCR	PCO	PCU	WTO	QDC	PHE	PCE	INT
R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h

Table 23-24. QCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12	QMAE	R-0/W1S	0h	Clear QMA Error interrupt flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag
11	UTO	R-0/W1S	0h	Clear unit time out interrupt flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag
10	IEL	R-0/W1S	0h	Clear index event latch interrupt flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag
9	SEL	R-0/W1S	0h	Clear strobe event latch interrupt flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag
8	PCM	R-0/W1S	0h	Clear eQEP compare match event interrupt flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag
7	PCR	R-0/W1S	0h	Clear position-compare ready interrupt flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag
6	PCO	R-0/W1S	0h	Clear position counter overflow interrupt flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag
5	PCU	R-0/W1S	0h	Clear position counter underflow interrupt flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag
4	WTO	R-0/W1S	0h	Clear watchdog timeout interrupt flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag
3	QDC	R-0/W1S	0h	Clear quadrature direction change interrupt flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag

Table 23-24. QCLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	PHE	R-0/W1S	0h	Clear quadrature phase error interrupt flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag
1	PCE	R-0/W1S	0h	Clear position counter error interrupt flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag
0	INT	R-0/W1S	0h	Global interrupt clear flag Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Clears the interrupt flag

23.13.2.19 QFRC Register (Offset = 1Bh) [Reset = 0h]

QFRC is shown in Figure 23-43 and described in Table 23-25.

Return to the Summary Table.

QEP Interrupt Force

Figure 23-43. QFRC Register

15	14	13	12	11	10	9	8
RESERVED			QMAE	UTO	IEL	SEL	PCM
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
PCR	PCO	PCU	WTO	QDC	PHE	PCE	RESERVED
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h

Table 23-25. QFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12	QMAE	R/W	0h	Force QMA error interrupt Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Force the interrupt
11	UTO	R/W	0h	Force unit time out interrupt Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Force the interrupt
10	IEL	R/W	0h	Force index event latch interrupt Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Force the interrupt
9	SEL	R/W	0h	Force strobe event latch interrupt Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Force the interrupt
8	PCM	R/W	0h	Force position-compare match interrupt Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Force the interrupt
7	PCR	R/W	0h	Force position-compare ready interrupt Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Force the interrupt
6	PCO	R/W	0h	Force position counter overflow interrupt Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Force the interrupt
5	PCU	R/W	0h	Force position counter underflow interrupt Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Force the interrupt
4	WTO	R/W	0h	Force watchdog time out interrupt Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Force the interrupt
3	QDC	R/W	0h	Force quadrature direction change interrupt Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Force the interrupt

Table 23-25. QFRC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	PHE	R/W	0h	Force quadrature phase error interrupt Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Force the interrupt
1	PCE	R/W	0h	Force position counter error interrupt Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Force the interrupt
0	RESERVED	R	0h	Reserved

23.13.2.20 QEPSTS Register (Offset = 1Ch) [Reset = 80h]

QEPSTS is shown in Figure 23-44 and described in Table 23-26.

Return to the Summary Table.

QEP Status

Figure 23-44. QEPSTS Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
UPEVNT	FIDF	QDF	QDLF	COEF	CDEF	FIMF	PCEF
R/W1S-1h	R-0h	R-0h	R-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R-0h

Table 23-26. QEPSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	UPEVNT	R/W1S	1h	Unit position event flag Reset type: SYSRSn 0h (R/W) = No unit position event detected 1h (R/W) = Unit position event detected. Write 1 to clear
6	FIDF	R	0h	Direction on the first index marker Status of the direction is latched on the first index event marker. Reset type: SYSRSn 0h (R/W) = Counter-clockwise rotation (or reverse movement) on the first index event 1h (R/W) = Clockwise rotation (or forward movement) on the first index event
5	QDF	R	0h	Quadrature direction flag Reset type: SYSRSn 0h (R/W) = Counter-clockwise rotation (or reverse movement) 1h (R/W) = Clockwise rotation (or forward movement)
4	QDLF	R	0h	eQEP direction latch flag Reset type: SYSRSn 0h (R/W) = Counter-clockwise rotation (or reverse movement) on index event marker 1h (R/W) = Clockwise rotation (or forward movement) on index event marker
3	COEF	R/W1S	0h	Capture overflow error flag Reset type: SYSRSn 0h (R/W) = Overflow has not occurred. 1h (R/W) = Overflow occurred in eQEP Capture timer (QEPCTMR). This bit is cleared by writing a '1'.
2	CDEF	R/W1S	0h	Capture direction error flag Reset type: SYSRSn 0h (R/W) = Capture direction error has not occurred. 1h (R/W) = Direction change occurred between the capture position event. This bit is cleared by writing a '1'.
1	FIMF	R/W1S	0h	First index marker flag Reset type: SYSRSn 0h (R/W) = First index pulse has not occurred. 1h (R/W) = Set by first occurrence of index pulse. This bit is cleared by writing a '1'.
0	PCEF	R	0h	Position counter error flag. This bit is not sticky and it is updated for every index event. Reset type: SYSRSn 0h (R/W) = No error occurred during the last index transition 1h (R/W) = Position counter error

23.13.2.21 QCTMR Register (Offset = 1Dh) [Reset = 0h]

QCTMR is shown in Figure 23-45 and described in Table 23-27.

Return to the Summary Table.

QEP Capture Timer

Figure 23-45. QCTMR Register

15	14	13	12	11	10	9	8
QCTMR							
R/W-0h							
7	6	5	4	3	2	1	0
QCTMR							
R/W-0h							

Table 23-27. QCTMR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	QCTMR	R/W	0h	This register provides time base for edge capture unit. Reset type: SYSRSn

23.13.2.22 QCPRD Register (Offset = 1Eh) [Reset = 0h]

QCPRD is shown in Figure 23-46 and described in Table 23-28.

Return to the Summary Table.

QEP Capture Period

Figure 23-46. QCPRD Register

15	14	13	12	11	10	9	8
QCPRD							
R/W-0h							
7	6	5	4	3	2	1	0
QCPRD							
R/W-0h							

Table 23-28. QCPRD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	QCPRD	R/W	0h	This register holds the period count value between the last successive eQEP position events Reset type: SYSRSn

23.13.2.23 QCTMRLAT Register (Offset = 1Fh) [Reset = 0h]

QCTMRLAT is shown in Figure 23-47 and described in Table 23-29.

Return to the Summary Table.

QEP Capture Latch

Figure 23-47. QCTMRLAT Register

15	14	13	12	11	10	9	8
QCTMRLAT							
R-0h							
7	6	5	4	3	2	1	0
QCTMRLAT							
R-0h							

Table 23-29. QCTMRLAT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	QCTMRLAT	R	0h	The eQEP capture timer value can be latched into this register on two events viz., unit timeout event, reading the eQEP position counter. Reset type: SYSRSn

23.13.2.24 QCPRDLAT Register (Offset = 20h) [Reset = 0h]

QCPRDLAT is shown in Figure 23-48 and described in Table 23-30.

Return to the Summary Table.

QEP Capture Period Latch

Figure 23-48. QCPRDLAT Register

15	14	13	12	11	10	9	8
QCPRDLAT							
R-0h							
7	6	5	4	3	2	1	0
QCPRDLAT							
R-0h							

Table 23-30. QCPRDLAT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	QCPRDLAT	R	0h	eQEP capture period value can be latched into this register on two events viz., unit timeout event, reading the eQEP position counter. Reset type: SYSRSn

23.13.2.25 REV Register (Offset = 30h) [Reset = 9h]

REV is shown in Figure 23-49 and described in Table 23-31.

Return to the Summary Table.

QEP Revision Number

Figure 23-49. REV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED										MINOR			MAJOR		
R-0h										R-1h			R-1h		

Table 23-31. REV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5-3	MINOR	R	1h	This field specifies the Minor Revision number for the eQEP IP. Reset type: N/A
2-0	MAJOR	R	1h	This field specifies the Major Revision number for the eQEP IP. Reset type: N/A

23.13.2.26 QEPSTROBESEL Register (Offset = 32h) [Reset = 0h]

QEPSTROBESEL is shown in Figure 23-50 and described in Table 23-32.

Return to the Summary Table.

QEP Strobe select register

Figure 23-50. QEPSTROBESEL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						STROBESEL	
R-0h						R/W-0h	

Table 23-32. QEPSTROBESEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1-0	STROBESEL	R/W	0h	Strobe source select: Reset type: SYSRSn 0h (R/W) = QEP Strobe after polarity mux 1h (R/W) = QEP Strobe after polarity mux 2h (R/W) = QEP Strobe after polarity mux ORed with ADCSOCA 3h (R/W) = QEP Strobe after polarity mux ORed with ADCSOCB

23.13.2.27 QMACTRL Register (Offset = 34h) [Reset = 0h]

QMACTRL is shown in Figure 23-51 and described in Table 23-33.

Return to the Summary Table.

QMA Control register

Figure 23-51. QMACTRL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													MODE		
R-0h													R/W-0h		

Table 23-33. QMACTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reserved
2-0	MODE	R/W	0h	Select Mode for QMA mode: 000 : QMA Module is bypassed. 001 : QMA Mode-1 operation selected 010 : QMA Mode-2 operation selected 011 : QMA Module is bypassed (reserved) 1xx : QMA Module is bypassed (reserved) Reset type: SYSRSn

23.13.2.28 QEPSRCSEL Register (Offset = 36h) [Reset = 0h]

QEPSRCSEL is shown in Figure 23-52 and described in Table 23-34.

Return to the Summary Table.

QEP Source Select Register

Figure 23-52. QEPSRCSEL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
QEPSSEL				QEPISEL				QEPBSEL				QEPASEL			
R/W-0h				R/W-0h				R/W-0h				R/W-0h			

Table 23-34. QEPSRCSEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-12	QEPSSEL	R/W	0h	QEP Strobe source select: 0x0: From device Pins (Default). Others: Tied to zero. Reset type: SYSRSn
11-8	QEPISEL	R/W	0h	QEP Index source select: 0x0: Device Pin (Default) 0x1: CMPSS1.CTRIPH 0x2: CMPSS2.CTRIPH 0x3: CMPSS3.CTRIPH 0x4: CMPSS4.CTRIPH 0x5: RSVD 0x6: RSVD 0x7: RSVD 0x8: RSVD 0x9: PWMXBAR.1 0xA:PWMXBAR.2 0xB:PWMXBAR.3 0xC:PWMXBAR.4 0xD:PWMXBAR.5 0xE:PWMXBAR.6 0xF:PWMXBAR.7 Note: eQEP needs to be disabled before configuring these bits as it can lead to unexpected behavior if eQEP is running. Reset type: SYSRSn
7-4	QEPBSEL	R/W	0h	QEPB source select: 0x0: Device Pin (Default) 0x1: CMPSS1.CTRIPH 0x2: CMPSS2.CTRIPH 0x3: CMPSS3.CTRIPH 0x4: CMPSS4.CTRIPH 0x5: RSVD 0x6: RSVD 0x7: RSVD 0x8: RSVD 0x9: PWMXBAR.1 0xA:PWMXBAR.2 0xB:PWMXBAR.3 0xC:PWMXBAR.4 0xD:PWMXBAR.5 0xE:PWMXBAR.6 0xF:PWMXBAR.7 Note: eQEP needs to be disabled before configuring these bits as it can lead to unexpected behavior if eQEP is running. Reset type: SYSRSn

Table 23-34. QEPSRCSEL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	QEPASEL	R/W	0h	QEPA source select: 0x0: Device Pin (Default) 0x1: CMPSS1.CTRIPH 0x2: CMPSS2.CTRIPH 0x3: CMPSS3.CTRIPH 0x4: CMPSS4.CTRIPH 0x5: RSVD 0x6: RSVD 0x7: RSVD 0x8: RSVD 0x9: PWMXBAR.1 0xA:PWMXBAR.2 0xB:PWMXBAR.3 0xC:PWMXBAR.4 0xD:PWMXBAR.5 0xE:PWMXBAR.6 0xF:PWMXBAR.7 Note: eQEP needs to be disabled before configuring these bits as it can lead to unexpected behavior if eQEP is running. Reset type: SYSRSn

23.13.3 EQEP Registers to Driverlib Functions

Table 23-35. EQEP Registers to Driverlib Functions

File	Driverlib Function
QPOSCNT	
eqep.h	EQEP_getPosition
eqep.h	EQEP_setPosition
QPOSINIT	
eqep.h	EQEP_setInitialPosition
QPOSMAX	
eqep.h	EQEP_setPositionCounterConfig
QPOSCMP	
eqep.c	EQEP_setCompareConfig
QPOSILAT	
eqep.h	EQEP_getIndexPositionLatch
QPOSSLAT	
eqep.h	EQEP_getStrobePositionLatch
QPOSLAT	
eqep.h	EQEP_getPositionLatch
QUTMR	
-	
QUPRD	
eqep.h	EQEP_loadUnitTimer
eqep.h	EQEP_enableUnitTimer
QWDTMR	
eqep.h	EQEP_setWatchdogTimerValue
eqep.h	EQEP_getWatchdogTimerValue
QWDPRD	
eqep.h	EQEP_enableWatchdog
QDECCTL	
eqep.c	EQEP_setCompareConfig

Table 23-35. EQEP Registers to Driverlib Functions (continued)

File	Driverlib Function
eqep.c	EQEP_setInputPolarity
eqep.h	EQEP_setDecoderConfig
eqep.h	EQEP_enableDirectionChangeDuringIndex
eqep.h	EQEP_disableDirectionChangeDuringIndex
QEPCTL	
eqep.h	EQEP_enableModule
eqep.h	EQEP_disableModule
eqep.h	EQEP_setPositionCounterConfig
eqep.h	EQEP_enableUnitTimer
eqep.h	EQEP_disableUnitTimer
eqep.h	EQEP_enableWatchdog
eqep.h	EQEP_disableWatchdog
eqep.h	EQEP_setPositionInitMode
eqep.h	EQEP_setSWPositionInit
eqep.h	EQEP_setLatchMode
eqep.h	EQEP_setEmulationMode
QCAPCTL	
eqep.h	EQEP_setCaptureConfig
eqep.h	EQEP_enableCapture
eqep.h	EQEP_disableCapture
QPOSCTL	
eqep.c	EQEP_setCompareConfig
eqep.h	EQEP_enableCompare
eqep.h	EQEP_disableCompare
eqep.h	EQEP_setComparePulseWidth
QEINT	
eqep.h	EQEP_enableInterrupt
eqep.h	EQEP_disableInterrupt
QFLG	
eqep.h	EQEP_getInterruptStatus
eqep.h	EQEP_getError
QCLR	
eqep.h	EQEP_clearInterruptStatus
QFRC	
eqep.h	EQEP_forceInterrupt
QEPSTS	
eqep.h	EQEP_getDirection
eqep.h	EQEP_getStatus
eqep.h	EQEP_clearStatus
QCTMR	
eqep.h	EQEP_getCaptureTimer
eqep.h	EQEP_getCaptureTimerLatch
QCPRD	
eqep.h	EQEP_getCapturePeriod
eqep.h	EQEP_getCapturePeriodLatch

Table 23-35. EQEP Registers to Driverlib Functions (continued)

File	Driverlib Function
QCTMRLAT	
eqep.h	EQEP_getCaptureTimerLatch
QCPRDLAT	
eqep.h	EQEP_getCapturePeriodLatch
REV	
-	
QEPSTROBESEL	
eqep.h	EQEP_setStrobeSource
QMACTRL	
eqep.h	EQEP_setQMAModuleMode
QEPSRCSEL	
eqep.h	EQEP_selectSource

Chapter 24

Serial Peripheral Interface (SPI)

This chapter describes the serial peripheral interface (SPI) which is a high-speed synchronous serial input and output (I/O) port that allows a serial bit stream of programmed length (one to 16 bits) to be shifted into and out of the device at a programmed bit-transfer rate. The SPI is normally used for communications between the MCU controller and external peripherals or another controller. Typical applications include external I/O or peripheral expansion via devices such as shift registers, display drivers, and analog-to-digital converters (ADCs). Multi-device communications are supported by the master or slave operation of the SPI. The port supports a 16-level, receive and transmit FIFO for reducing CPU servicing overhead.

24.1 Introduction

24.1.1 Features

The SPI module features include:

- SPISOMI: SPI slave-output/master-input pin
- SPISIMO: SPI slave-input/master-output pin
- $\overline{\text{SPISTE}}$: SPI slave transmit-enable pin
- SPICLK: SPI serial-clock pin

Note

All four pins can be used as GPIO if the SPI module is not used.

- Two operational modes: Master and Slave
- Baud rate: 125 different programmable rates. The maximum baud rate that can be employed is limited by the maximum speed of the I/O buffers used on the SPI pins. See the device-specific data manual for more details.
- Data word length: one to sixteen data bits
- Four clocking schemes (controlled by clock polarity and clock phase bits) include:
  - Falling edge without phase delay: SPICLK active-high. SPI transmits data on the falling edge of the SPICLK signal and receives data on the rising edge of the SPICLK signal.
  - Falling edge with phase delay: SPICLK active-high. SPI transmits data one half-cycle ahead of the falling edge of the SPICLK signal and receives data on the falling edge of the SPICLK signal.
  - Rising edge without phase delay: SPICLK inactive-low. SPI transmits data on the rising edge of the SPICLK signal and receives data on the falling edge of the SPICLK signal.
  - Rising edge with phase delay: SPICLK inactive-low. SPI transmits data one half-cycle ahead of the rising edge of the SPICLK signal and receives data on the rising edge of the SPICLK signal.
- Simultaneous receive and transmit operation (transmit function can be disabled in software)
- Transmitter and receiver operations are accomplished through either interrupt- driven or polled algorithm
- 16-level transmit/receive FIFO
- DMA support
- High-speed mode
- Delayed transmit control
- 3-wire SPI mode
- $\overline{\text{SPISTE}}$ inversion for digital audio interface receive mode on devices with two SPI modules

24.1.2 Block Diagram

Figure 24-1 shows the SPI CPU interfaces.

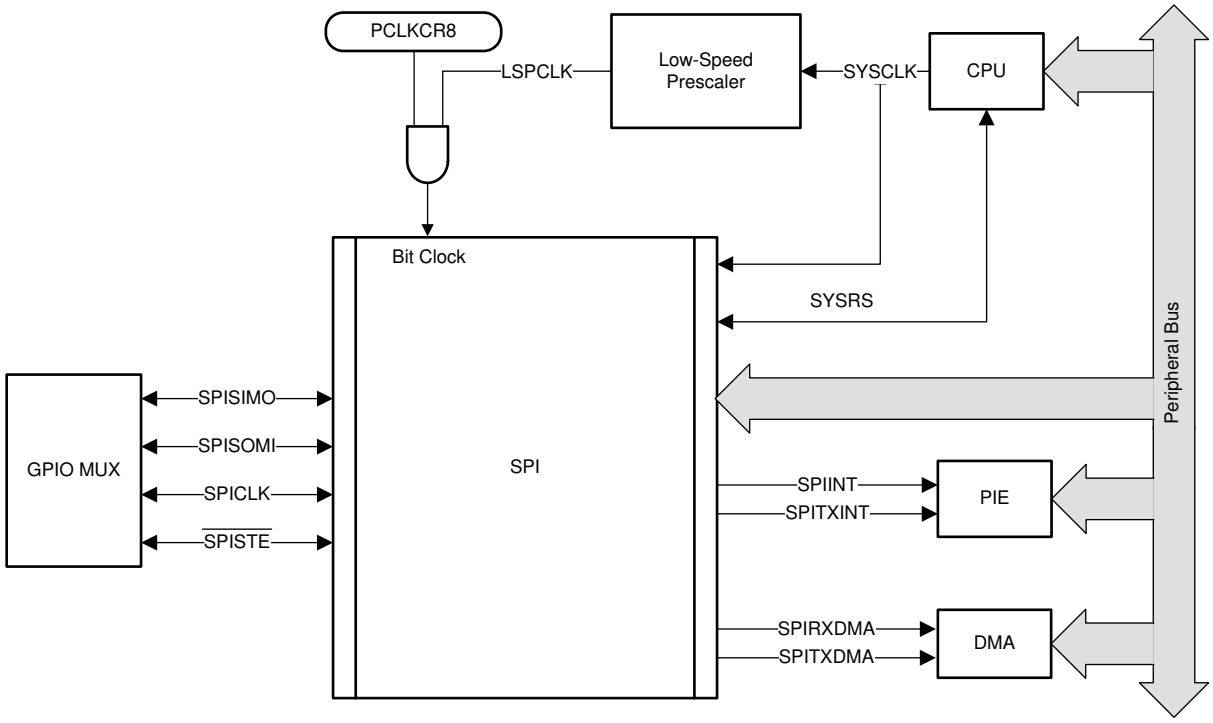


Figure 24-1. SPI CPU Interface

24.2 System-Level Integration

This section describes the various functionality that is applicable to the device integration. These features require configuration of other modules in the device that are not within the scope of this chapter.

24.2.1 SPI Module Signals

Table 24-1 classifies and provides a summary of the SPI module signals.

Table 24-1. SPI Module Signal Summary

Signal Name	Description
External Signals	
SPICLK	SPI clock
SPISIMO	SPI slave in, master out
SPISOMI	SPI slave out, master in
$\overline{\text{SPISTE}}$	SPI slave transmit enable
Control	
SPI Clock Rate	LSPCLK
Interrupt Signals	
SPIINT/SPIRXINT	Transmit interrupt/ Receive Interrupt in non FIFO mode (referred to as SPIINT)
	Receive interrupt in FIFO mode
SPITXINT	Transmit interrupt in FIFO mode
DMA Triggers	
SPITXDMA	Transmit request to DMA
SPIRXDMA	Receive request to DMA

Special Considerations

The $\overline{\text{SPISTE}}$ signal provides the ability to gate any spurious clock and data pulses when the SPI is in slave mode. A HIGH logic signal on $\overline{\text{SPISTE}}$ will not allow the slave to receive data. This prevents the SPI slave from losing synchronization with the master. It is this reason that TI does not recommend that the $\overline{\text{SPISTE}}$ always be tied to the active state.

If the SPI slave does ever lose synchronization with the master, toggling SPISWRESET resets the internal bit counter as well as the various status flags in the module. By resetting the bit counter, the SPI interprets the next clock transition as the first bit of a new transmission. The register bit fields that are reset by SPISWRESET are found in Section 24.6.

Configuring a GPIO to Emulate $\overline{\text{SPISTE}}$

In many systems, a SPI master may be connected to multiple SPI slaves using multiple instances of $\overline{\text{SPISTE}}$. Though this SPI module does not natively support multiple $\overline{\text{SPISTE}}$ signals, it is possible to emulate this behavior in software using GPIOs. In this configuration, the SPI must be configured as the master. Rather than using the GPIO Mux to select $\overline{\text{SPISTE}}$, the application would configure pins to be GPIO outputs, one GPIO per SPI slave. Before transmitting any data, the application would drive the desired GPIO to the active state. Immediately after the transmission has been completed, the GPIO chip select would be driven to the inactive state. This process can be repeated for many slaves which share the SPICLK, SPISIMO, and SPISOMI lines.

24.2.2 Configuring Device Pins

The GPIO mux registers must be configured to connect this peripheral to the device pins. To avoid glitches on the pins, the GPyGMUX bits must be configured first (while keeping the corresponding GPyMUX bits at the default of zero), followed by writing the GPyMUX register to the desired value.

Some IO functionality is defined by GPIO register settings independent of this peripheral. For input signals, the GPIO input qualification should be set to asynchronous mode by setting the appropriate GPxQSELn register bits to 11b. The internal pullups can be configured in the GPyPUD register.

See the *GPIO* chapter for more details on GPIO mux and settings.

24.2.2.1 GPIOs Required for High-Speed Mode

The high-speed mode of the SPI is available on all GPIO mux options. To enable the high-speed enhancements, set SPICCR.HS_MODE to 1. Ensure that the capacitive loading on the pin does not exceed the value stated in the device Data Manual.

When not operating in high-speed mode, or if the capacitive loading on the pins exceed the value stated in the device Data Manual, SPICCR.HS_MODE should be set to 0.

24.2.3 SPI Interrupts

This section includes information on the available interrupts present in the SPI module.

The SPI module contains two interrupt lines: SPIINT/SPIRXINT and SPITXINT. When the SPI is operating in non-FIFO mode, all available interrupts are routed together to generate the single SPIINT interrupt. When FIFO mode is used, both SPIRXINT and SPITXINT can be generated.

SPIINT/SPIRXINT

When the SPI is operating in non-FIFO mode, the interrupt generated is called SPIINT. If FIFO enhancements are enabled, the interrupt is called SPIRXINT. These interrupts share the same interrupt vector in the Peripheral Interrupt Expansion (PIE) block.

In non-FIFO mode, two conditions can trigger an interrupt: a transmission is complete (INT_FLAG), or there is overrun in the receiver (OVERRUN_FLAG). Both of these conditions share the same interrupt vector: SPIINT.

The transmission complete flag (INT_FLAG) indicates that the SPI has completed sending or receiving the last bit and is ready to be serviced. At the same time this bit is set, the received character is placed in the receiver buffer (SPIRXBUF). The INT_FLAG will generate an interrupt on the SPIINT vector if the SPIINTENA bit is set.

The receiver overrun flag (OVERRUN_FLAG) indicates that a transmit or receive operation has completed before the previous character has been read from the buffer. The OVERRUN_FLAG will generate an interrupt on the SPIINT vector if the OVERRUNINTENA bit is set and OVERRUN_FLAG was previously cleared.

In FIFO mode, the SPI can interrupt the CPU upon a match condition between the current receive FIFO status (RXFFST) and the receive FIFO interrupt level (RXFFIL). If RXFFST is greater than or equal to RXFFIL, the receive FIFO interrupt flag (RXFFINT) will be set. SPIRXINT will be triggered in the PIE block if RXFFINT is set and the receive FIFO interrupt is enabled (RXFFIENA = 1).

SPITXINT

The SPITXINT interrupt is not available when the SPI is operating in non-FIFO mode.

In FIFO mode, the SPITXINT behavior is similar to the SPIRXINT. SPITXINT is generated upon a match condition between the current transmit FIFO status (TXFFST) and the transmit FIFO interrupt level (TXFFIL). If TXFFST is less than or equal to TXFFIL, the transmit FIFO interrupt flag (TXFFINT) will be set. SPITXINT will be triggered in the PIE block if TXFFINT is set and the transmit FIFO interrupt is enabled in the SPI module (TXFFIENA = 1).

Figure 24-2 and Table 24-2 show how these control bits influence the SPI interrupt generation.

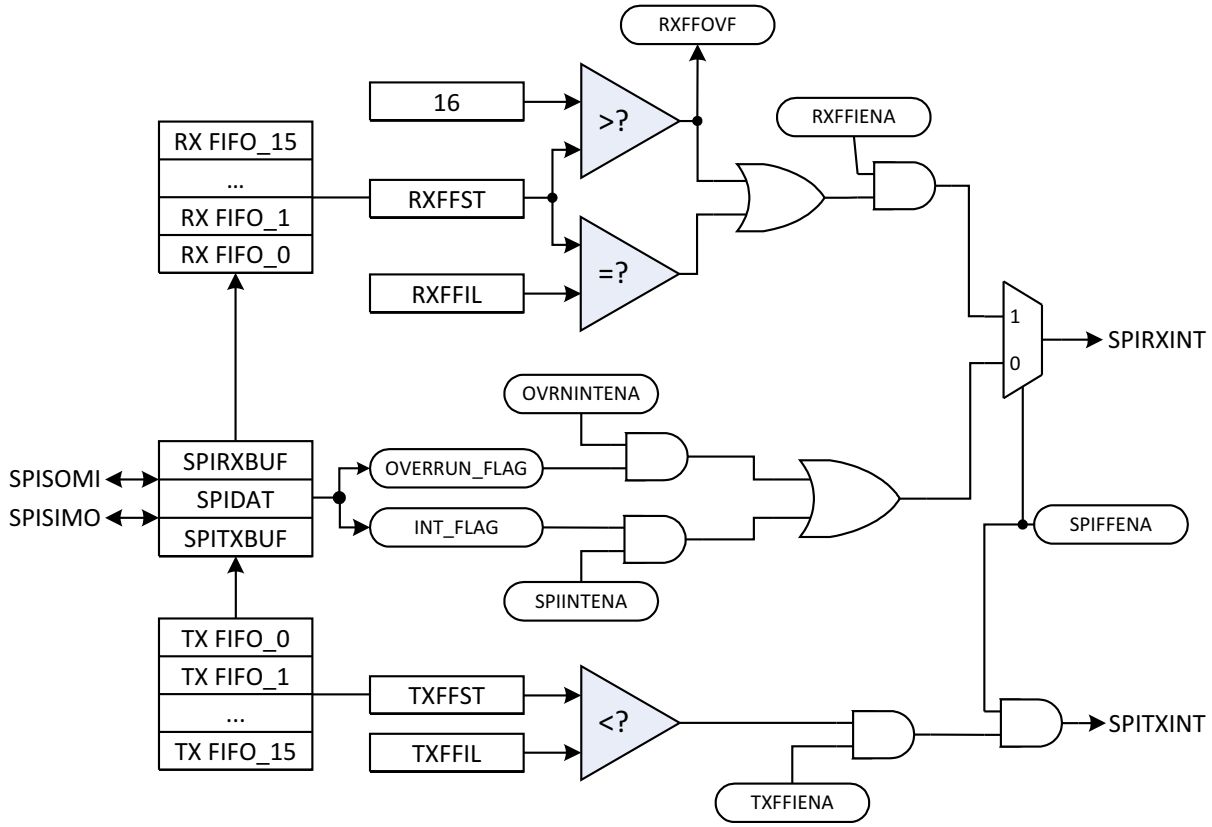


Figure 24-2. SPI Interrupt Flags and Enable Logic Generation

Table 24-2. SPI Interrupt Flag Modes

FIFO Options	SPI Interrupt Source	Interrupt Flags	Interrupt Enables	FIFO Enable (SPIFFENA)	Interrupt Line(1)
SPI without FIFO	Receive overrun	RXOVRN	OVRNINTENA	0	SPIRXINT
	Data receive	SPIINT	SPIINTENA	0	SPIRXINT
	Transmit empty	SPIINT	SPIINTENA	0	SPIRXINT
SPI FIFO mode	FIFO receive	RXFFIL	RXFFIENA	1	SPIRXINT
	Transmit empty	TXFFIL	TXFFIENA	1	SPITXINT

(1) In non-FIFO mode, SPIRXINT is the same name as the SPIINT interrupt in C28x devices.

24.2.4 DMA Support

Both the CPU and DMA have access to the SPI data registers via the internal peripheral bus. This access is limited to 16-bit register read/writes. Each SPI module can generate two DMA events, SPITXDMA and SPIRXDMA. The DMA events are controlled by configuring the SPIFFTX.TXFFIL and SPIFFRX.RXFFIL appropriately. SPITXDMA activates when TXFFST is less than the interrupt level (TXFFIL). SPIRXDMA activates when RXFFST is greater than or equal to the interrupt level (RXFFIL).

The SPI must have FIFO enhancements enabled in order for the DMA triggers to be generated.

For more information on configuring the SPI for DMA transfers, refer to Section 24.3.8.

Figure 24-3 is a block diagram showing the DMA trigger generation from the SPI module.

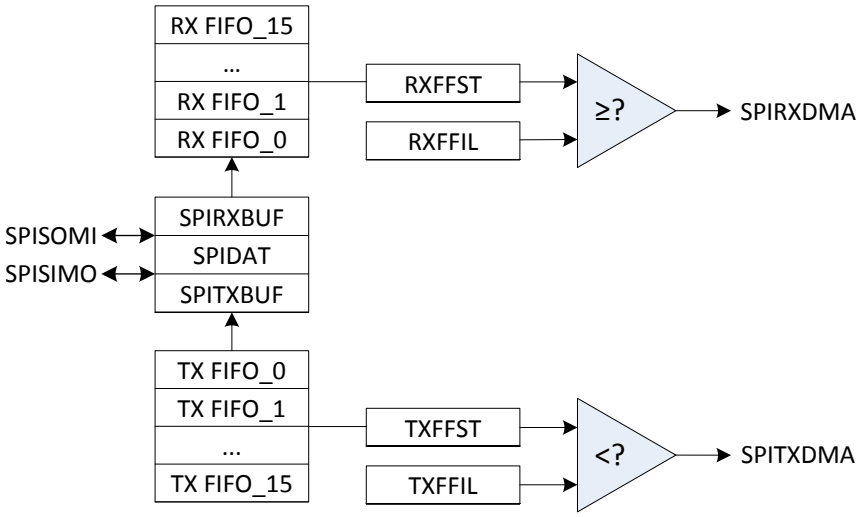


Figure 24-3. SPI DMA Trigger Diagram

24.3 SPI Operation

This section describes the various modes of operation of the SPI. Included are explanations of the operational modes, interrupts, data format, clock sources, and initialization. Typical timing diagrams for data transfers are given.

24.3.1 Introduction to Operation

Figure 24-4 shows typical connections of the SPI for communications between two controllers: a master and a slave.

The master transfers data by sending the SPICLK signal. For both the slave and the master, data is shifted out of the shift registers on one edge of the SPICLK and latched into the shift register on the opposite SPICLK clock edge. If the CLK_PHASE bit is high, data is transmitted and received a half-cycle before the SPICLK transition. As a result, both controllers send and receive data simultaneously. The application software determines whether the data is meaningful or dummy data. There are three possible methods for data transmission:

- Master sends data; slave sends dummy data.
- Master sends data; slave sends data.
- Master sends dummy data; slave sends data.

The master can initiate data transfer at any time because it controls the SPICLK signal. The software, however, determines how the master detects when the slave is ready to broadcast data.

The SPI can operate in master or slave mode. The MASTER_SLAVE bit selects the operating mode and the source of the SPICLK signal.

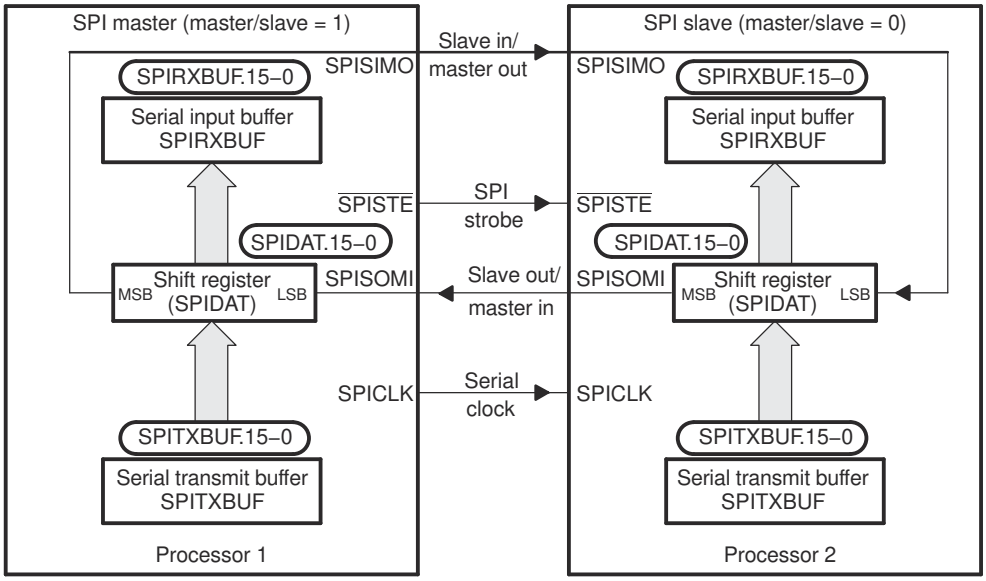


Figure 24-4. SPI Master/Slave Connection

24.3.2 Master Mode

In master mode (MASTER_SLAVE = 1), the SPI provides the serial clock on the SPICLK pin for the entire serial communications network. Data is output on the SPISIMO pin and latched from the SPISOMI pin.

The SPIBRR register determines both the transmit and receive bit transfer rate for the network. SPIBRR can select 125 different data transfer rates.

Data written to SPIDAT or SPITXBUF initiates data transmission on the SPISIMO pin, MSB (most significant bit) first. Simultaneously, received data is shifted through the SPISOMI pin into the LSB (least significant bit)

of SPIDAT. When the selected number of bits has been transmitted, the received data is transferred to the SPIRXBUF (buffered receiver) for the CPU to read. Data is stored right-justified in SPIRXBUF.

When the specified number of data bits has been shifted through SPIDAT, the following events occur:

- SPIDAT contents are transferred to SPIRXBUF.
- INT_FLAG bit is set to 1.
- If there is valid data in the transmit buffer SPITXBUF, as indicated by the Transmit Buffer Full Flag (BUFFULL_FLAG), this data is transferred to SPIDAT and is transmitted; otherwise, SPICLK stops after all bits have been shifted out of SPIDAT.
- If the SPIINTENA bit is set to 1, an interrupt is asserted.

In a typical application, the $\overline{\text{SPISTE}}$ pin serves as a chip-enable pin for a slave SPI device. This pin is driven low by the master before transmitting data to the slave and is taken high after the transmission is complete.

Figure 24-5 is a block diagram of the SPI in master mode. It shows the basic control blocks available in SPI master mode.

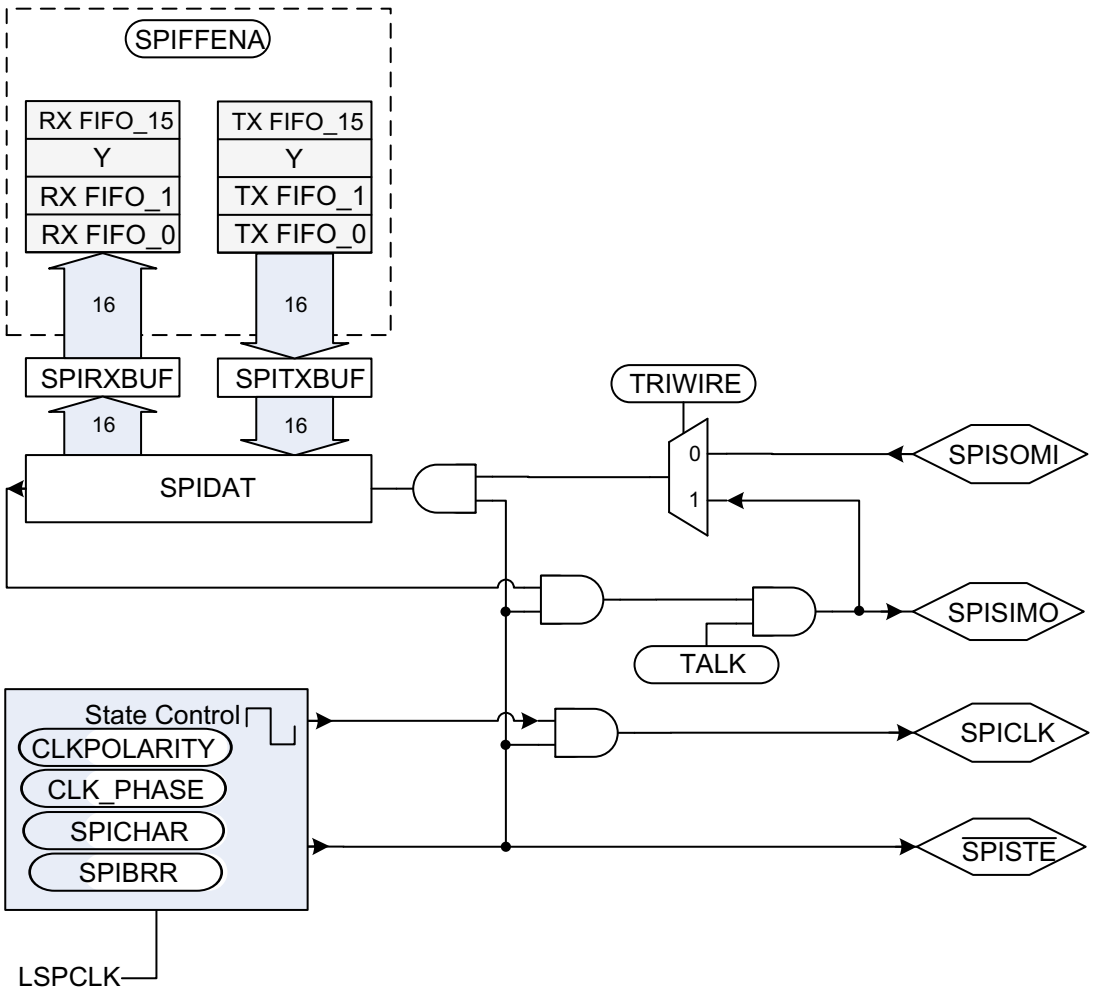


Figure 24-5. SPI Module Master Configuration

24.3.3 Slave Mode

In slave mode (MASTER_SLAVE = 0), data shifts out on the SPISOMI pin and in on the SPISIMO pin. The SPICLK pin is used as the input for the serial shift clock, which is supplied from the external network master. The transfer rate is defined by this clock. The SPICLK input frequency should be no greater than the LSPCLK frequency divided by 4.

Data written to SPIDAT or SPITXBUF is transmitted to the network when appropriate edges of the SPICLK signal are received from the network master. Data written to the SPITXBUF register will be transferred to the SPIDAT register when all bits of the character to be transmitted have been shifted out of SPIDAT. If no character is currently being transmitted when SPITXBUF is written to, the data will be transferred immediately to SPIDAT. To receive data, the SPI waits for the network master to send the SPICLK signal and then shifts the data on the SPISIMO pin into SPIDAT. If data is to be transmitted by the slave simultaneously, and SPITXBUF has not been previously loaded, the data must be written to SPITXBUF or SPIDAT before the beginning of the SPICLK signal.

When the TALK bit is cleared, data transmission is disabled, and the output line (SPISOMI) is put into the high-impedance state. If this occurs while a transmission is active, the current character is completely transmitted even though SPISOMI is forced into the high-impedance state. This ensures that the SPI is still able to receive incoming data correctly. This TALK bit allows many slave devices to be tied together on the network, but only one slave at a time is allowed to drive the SPISOMI line.

The $\overline{\text{SPISTE}}$ pin operates as the slave-select pin. An active-low signal on the $\overline{\text{SPISTE}}$ pin allows the slave SPI to transfer data to the serial data line; an inactive- high signal causes the slave SPI serial shift register to stop and its serial output pin to be put into the high-impedance state. This allows many slave devices to be tied together on the network, although only one slave device is selected at a time.

Figure 24-6 is a block diagram of the SPI in slave mode. It shows the basic control blocks available in SPI slave mode.

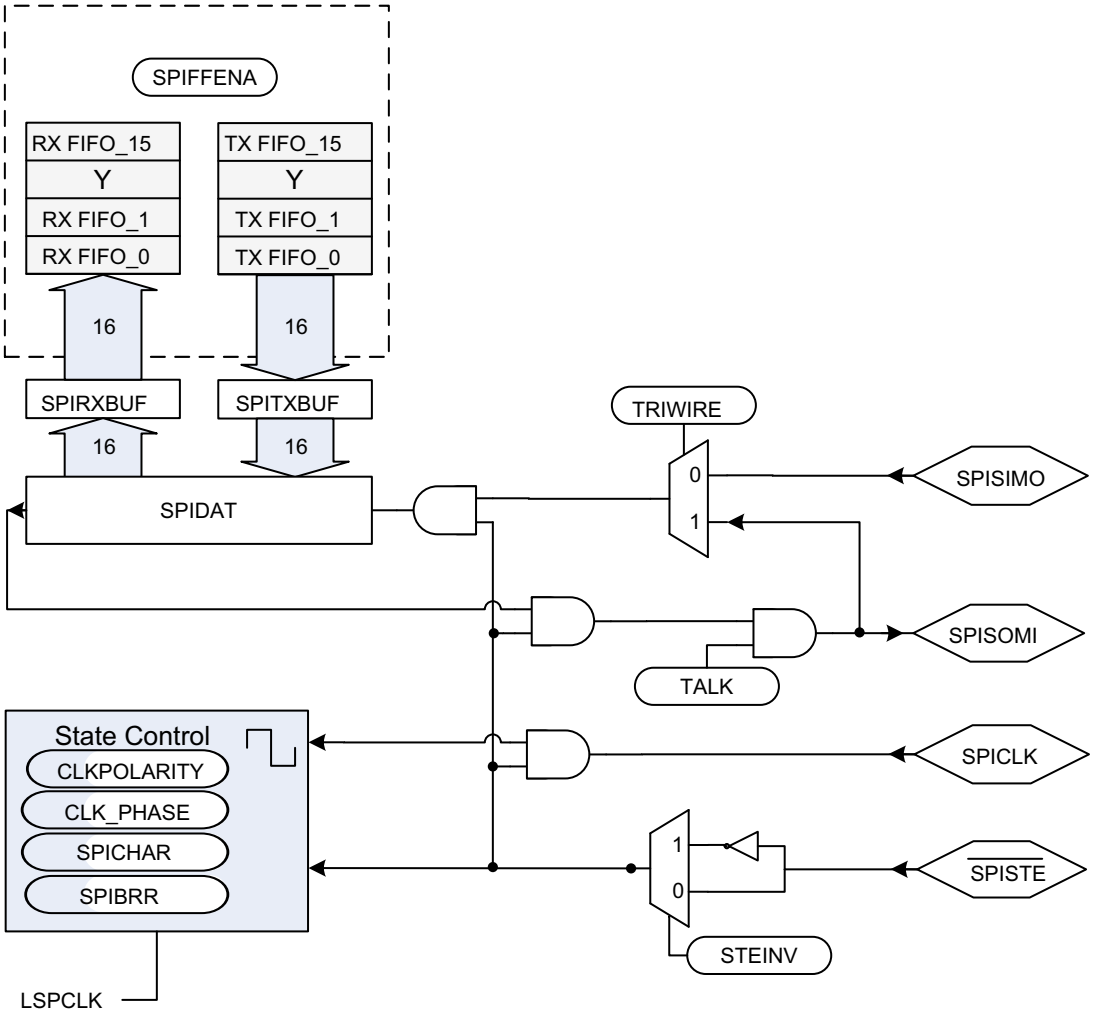


Figure 24-6. SPI Module Slave Configuration

24.3.4 Data Format

The four-bit SPICHAR register field specifies the number of bits in the data character (1 to 16). This information directs the state control logic to count the number of bits received or transmitted to determine when a complete character has been processed.

The following statements apply to characters with fewer than 16 bits:

- Data must be left-justified when written to SPIDAT and SPITXBUF.
- Data read back from SPIRXBUF is right-justified.
- SPIRXBUF contains the most recently received character, right-justified, plus any bits that remain from previous transmission(s) that have been shifted to the left (shown in Example 24-1).

Example 24-1. Transmission of Bit from SPIRXBUF

Conditions:

1. Transmission character length = 1 bit (specified in bits SPICHAR)
2. The current value of SPIDAT = 737Bh

SPIDAT (before transmission)																	
	0	1	1	1	0	0	1	1	0	1	1	1	1	0	1	1	
SPIDAT (after transmission)																	
(TXed) 0 ←	1	1	1	0	0	1	1	0	1	1	1	1	0	1	1	x[(1)]	← (RXed)
SPIRXBUF (after transmission)																	
	1	1	1	0	0	1	1	0	1	1	1	1	0	1	1	x[(1)]	

(1) x = 1 if SPISOMI data is high; x = 0 if SPISOMI data is low; master mode is assumed.

24.3.5 Baud Rate Selection

The SPI module supports 125 different baud rates and four different clock schemes. Depending on whether the SPI clock is in slave or master mode, the SPICLK pin can receive an external SPI clock signal or provide the SPI clock signal, respectively.

- In the slave mode, the SPI clock is received on the SPICLK pin from the external source, and can be no greater than the LSPCLK frequency divided by 4.
- In the master mode, the SPI clock is generated by the SPI and is output on the SPICLK pin, and can be no greater than the LSPCLK frequency divided by 4.

Note

The baud rate should be configured to not exceed the maximum rated GPIO toggle frequency. Refer to the device Data Manual for the maximum GPIO toggle frequency

Example 24-2 shows how to determine the SPI baud rates.

Example 24-2. Baud Rate Determination

For SPIBRR = 3 to 127:

$$\text{SPI Baud Rate} = \frac{\text{LSPCLK}}{(\text{SPIBRR} + 1)} \quad (23)$$

For SPIBRR = 0, 1, or 2:

$$\text{SPI Baud Rate} = \frac{\text{LSPCLK}}{4} \quad (24)$$

where:

LSPCLK = Low-speed peripheral clock frequency of the device

SPIBRR = Contents of the SPIBRR in the master SPI device

To determine what value to load into SPIBRR, you must know the device system clock (LSPCLK) frequency (which is device-specific) and the baud rate at which you will be operating.

Example 24-3 shows how to calculate the baud rate of the SPI module in standard SPI mode (HS_MODE=0).

Example 24-3. Baud Rate Calculation in Non-High Speed Mode (HS_MODE=0)

$$\begin{aligned} SPI\ Baud\ Rate &= \frac{LSPCLK}{SPIBRR + 1'} \quad LSPCLK = 50\ MHz \\ &= \frac{50 \times 10^6}{3 + 1} \\ &= 12.5\ Mbps \end{aligned} \quad (25)$$

24.3.6 SPI Clocking Schemes

The clock polarity select bit (CLKPOLARITY) and the clock phase select bit (CLK_PHASE) control four different clocking schemes on the SPICLK pin. CLKPOLARITY selects the active edge, either rising or falling, of the clock. CLK_PHASE selects a half-cycle delay of the clock. The four different clocking schemes are as follows:

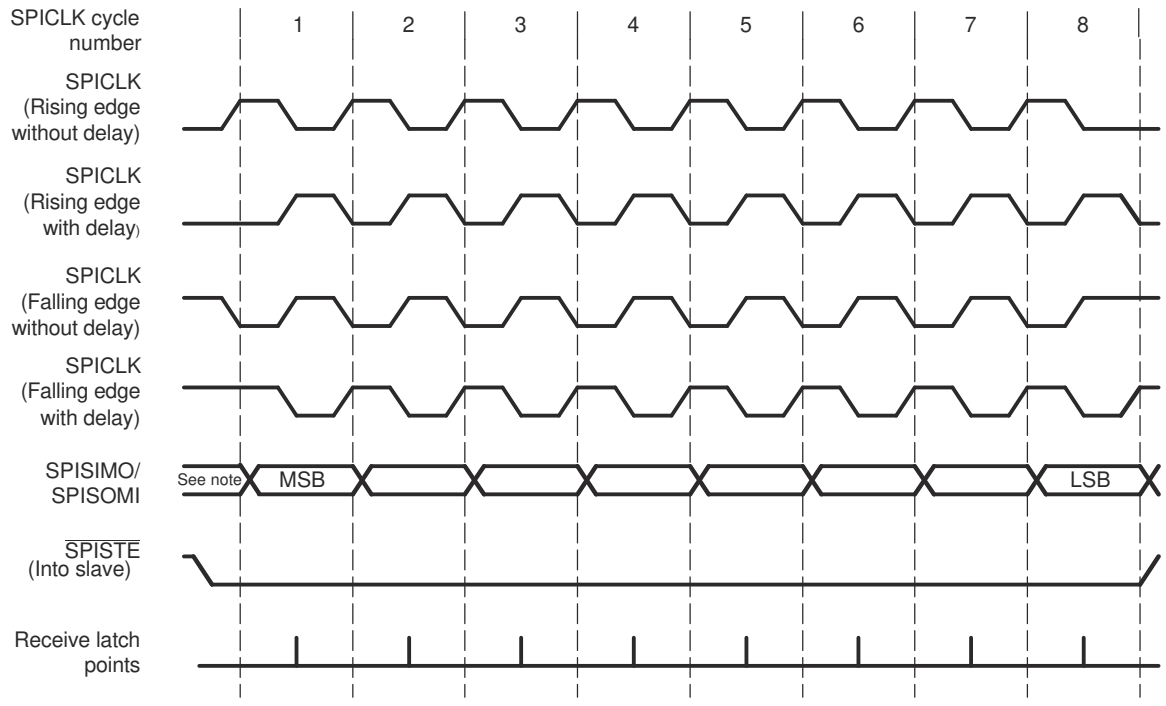
- Falling Edge Without Delay. The SPI transmits data on the falling edge of the SPICLK and receives data on the rising edge of the SPICLK.
- Falling Edge With Delay. The SPI transmits data one half-cycle ahead of the falling edge of the SPICLK signal and receives data on the falling edge of the SPICLK signal.
- Rising Edge Without Delay. The SPI transmits data on the rising edge of the SPICLK signal and receives data on the falling edge of the SPICLK signal.
- Rising Edge With Delay. The SPI transmits data one half-cycle ahead of the rising edge of the SPICLK signal and receives data on the rising edge of the SPICLK signal.

The selection procedure for the SPI clocking scheme is shown in Table 24-3. Examples of these four clocking schemes relative to transmitted and received data are shown in Figure 24-7.

Table 24-3. SPI Clocking Scheme Selection Guide

SPICLK Scheme	CLKPOLARITY	CLK_PHASE(1)
Rising edge without delay	0	0
Rising edge with delay	0	1
Falling edge without delay	1	0
Falling edge with delay	1	1

(1) The description of CLK_PHASE and CLKPOLARITY differs between manufacturers. For proper operation, select the desired waveform to determine the clock phase and clock polarity settings.



Note: Previous data bit

Figure 24-7. SPICLK Signal Options

SPICLK symmetry is retained only when the result of (SPIBRR+1) is an even value. When (SPIBRR + 1) is an odd value and SPIBRR is greater than 3, SPICLK becomes asymmetrical. The low pulse of SPICLK is one LSPCLK cycle longer than the high pulse when CLKPOLARITY bit is clear (0). When CLKPOLARITY bit is set to 1, the high pulse of the SPICLK is one LSPCLK cycle longer than the low pulse, as shown in Figure 24-8.

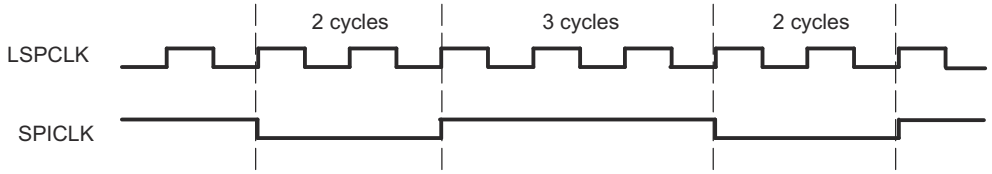


Figure 24-8. SPI: SPICLK-LSPCLK Characteristic When (BRR + 1) is Odd, BRR > 3, and CLKPOLARITY = 1

24.3.7 SPI FIFO Description

The following steps explain the FIFO features and help with programming the SPI FIFOs:

1. **Reset.** At reset the SPI powers up in standard SPI mode and the FIFO function is disabled. The FIFO registers SPIFFTX, SPIFFRX and SPIFFCT remain inactive.
2. **Standard SPI.** The standard 28x SPI mode will work with SPIINT/SPIRXINT as the interrupt source.
3. **Mode change.** FIFO mode is enabled by setting the SPIFFENA bit to 1 in the SPIFFTX register. SPIRST can reset the FIFO mode at any stage of its operation.
4. **Active registers.** All the SPI registers and SPI FIFO registers SPIFFTX, SPIFFRX, and SPIFFCT will be active.
5. **Interrupts.** FIFO mode has two interrupts one for the transmit FIFO, SPITXINT and one for the receive FIFO, SPIRXINT. SPIRXINT is the common interrupt for SPI FIFO receive, receive error and receive FIFO overflow conditions. The single SPIINT for both transmit and receive sections of the standard SPI will be disabled and this interrupt will service as SPI receive FIFO interrupt. For more information, refer to Section 24.2.3.
6. **Buffers.** Transmit and receive buffers are each supplemented with a 16 word FIFO. The one-word transmit buffer (SPITXBUF) of the standard SPI functions as a transition buffer between the transmit FIFO and shift register. The one-word transmit buffer will be loaded from transmit FIFO only after the last bit of the shift register is shifted out.
7. **Delayed transfer.** The rate at which transmit words in the FIFO are transferred to transmit shift register is programmable. The SPIFFCT register bits (7−0) FFTXDLY7−FFTXDLY0 define the delay between the word transfer. The delay is defined in number SPI serial clock cycles. The 8-bit register could define a minimum delay of 0 SPICLK cycles and a maximum of 255 SPICLK cycles. With zero delay, the SPI module can transmit data in continuous mode with the FIFO words shifting out back to back. With the 255 clock delay, the SPI module can transmit data in a maximum delayed mode with the FIFO words shifting out with a delay of 255 SPICLK cycles between each words. The programmable delay facilitates glueless interface to various slow SPI peripherals, such as EEPROMs, ADC, DAC, and so on.
8. **FIFO status bits.** Both transmit and receive FIFOs have status bits TXFFST or RXFFST that define the number of words available in the FIFOs at any time. The transmit FIFO reset bit (TXFIFO) and receive reset bit (RXFIFO) will reset the FIFO pointers to zero when these bits are set to 1. The FIFOs will resume operation from start once these bits are cleared to zero.
9. **Programmable interrupt levels.** Both transmit and receive FIFOs can generate CPU interrupts and DMA triggers. The transmit interrupt (SPITXINT) is generated whenever the transmit FIFO status bits (TXFFST) match (less than or equal to) the interrupt trigger level bits (TXFFIL). The receive interrupt (SPIRXINT) is generated whenever the receive FIFO status bits (RXFFST) match (greater than or equal to) the interrupt trigger level RXFFIL. This provides a programmable interrupt trigger for transmit and receive sections of the SPI. The default value for these trigger level bits will be 0x11111 for receive FIFO and 0x00000 for transmit FIFO, respectively.

24.3.8 SPI DMA Transfers

24.3.8.1 Transmitting Data Using SPI with DMA

When using the DMA with the TX FIFO, the DMA Burst Size (DMA_BURST_SIZE) should be no greater than 16 – TXFFIL in order to prevent the DMA from writing to an already full FIFO. This will lead to data loss and is not recommended.

For complete data transmission, follow these steps:

1. Calculate the total number or words to be transmitted. [NUM_WORDS]
2. Decide the transmit FIFO level. [TXFFIL]
3. Calculate the number of DMA transfers. [DMA_TRANSFER_SIZE]
4. Calculate the size of the DMA Burst. [DMA_BURST_SIZE]
5. Configure DMA using calculated values.
6. Configure SPI with FIFO using the calculated values.

To transfer 128 words to SPI using the DMA:

NUM_WORDS: 128

TXFFIL: 8

DMA_TRANSFER_SIZE: (NUM_WORDS /TXFFIL) – 1 = (128/8) – 1 = 15 (16 transfers)

DMA_BURST_SIZE: (16 – TXFFIL) – 1 = (16 – 8) – 1 = 7 (8 words per burst)

Note

Avoid setting TXFFIL to 0h or 10h to ensure proper DMA configuration.

24.3.8.2 Receiving Data Using SPI with DMA

When using the DMA with the RX FIFO, the DMA Burst Size (BURST_SIZE) should be no greater than RXFFIL in order to prevent the DMA from reading from an empty FIFO. To ensure that the DMA correctly receives all data from the RX FIFO, the DMA Burst Size should equal RXFFIL and also be an integer divisor of the total number of SPI transmissions.

For complete data reception, follow these steps:

1. Calculate the number of words to be received. [NUM_WORDS]
2. Calculate the necessary FIFO level [RXFFIL]
3. Calculate the total number of DMA transfers. [DMA_TRANSFER_SIZE]
4. Calculate the size of the DMA Burst. [DMA_BURST_SIZE]
5. Configure DMA using the calculated values.
6. Configure SPI with FIFO using the calculated values.

To receive 200 words from SPI using the DMA:

NUM_WORDS = 200

RXFFIL: 4

DMA_TRANSFER_SIZE: (NUM_WORDS /RXFFIL) – 1 = (200/4) – 1 = 49 (50 transfers)

DMA_BURST_SIZE = RXFFIL-1 = 3 (4 words per burst)

Note

Avoid setting RXFFIL to 0h to ensure proper DMA configuration.

24.3.9 SPI High-Speed Mode

The SPI module is capable of reaching full-duplex communication speeds up to LSPCLK/4 (where LSPCLK equals SYSCLK). For the maximum rated speed, refer the device Data Manual.

In order to achieve the maximum full-duplex speeds, the following restrictions are placed on the design:

- Single master to single slave configuration is supported.
- Loading on the pins must not exceed the value stated in the device Data Manual.

When configuring the GPIOs to support High-Speed mode, refer to Section 24.2.2.1 for more information.

24.3.10 SPI 3-Wire Mode Description

SPI 3-wire mode allows for SPI communication over three pins instead of the normal four pins.

In master mode, if the TRIWIRE bit is set, enabling 3-wire SPI mode, SPISIMOx becomes the bi-directional SPIMOMIx (SPI master out, master in) pin, and SPISOMIx is no longer used by the SPI. In slave mode, if the TRIWIRE bit is set, SPISOMIx becomes the bi-directional SPISISOx (SPI slave in, slave out) pin, and SPISIMOx is no longer used by the SPI.

Table 24-4 indicates the pin function differences between 3-wire and 4-wire SPI mode for a master and slave SPI.

Table 24-4. 4-wire vs. 3-wire SPI Pin Functions

4-wire SPI	3-wire SPI (Master)	3-wire SPI (Slave)
SPICLKx	SPICLKx	SPICLKx
SPISTEx	SPISTEx	SPISTEx
SPISIMOx	SPIMOMIx	Free
SPISOMIx	Free	SPISISOx

Because in 3-wire mode, the receive and transmit paths within the SPI are connected, any data transmitted by the SPI module is also received by itself. The application software must take care to perform a dummy read to clear the SPI data register of the additional received data.

The TALK bit plays an important role in 3-wire SPI mode. The bit must be set to transmit data and cleared prior to reading data. In master mode, in order to initiate a read, the application software must write dummy data to the SPI data register (SPIDAT or SPIRXBUF) while the TALK bit is cleared (no data is transmitted out the SPIMOMI pin) before reading from the data register.

Figure 24-9 and Figure 24-10 illustrate 3-wire master and slave mode.

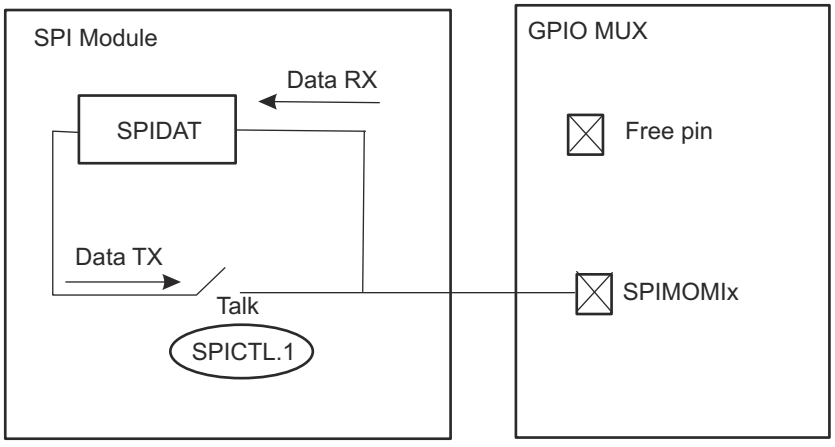


Figure 24-9. SPI 3-wire Master Mode

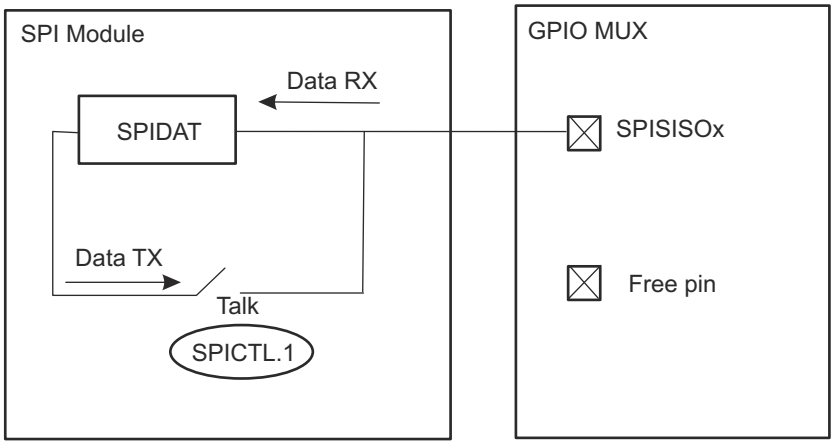


Figure 24-10. SPI 3-wire Slave Mode

Table 24-5 indicates how data is received or transmitted in the various SPI modes while the TALK bit is set or cleared.

Table 24-5. 3-Wire SPI Pin Configuration

Pin Mode	SPIPRI[TRIWIRE]	SPICTL[TALK]	SPISIMO	SPISOMI
		Master Mode		
4-wire	0	X	TX	RX
3-pin mode	1	0	RX	Disconnect from SPI
		1	TX/RX	
		Slave Mode		
4-wire	0	X	RX	TX
3-pin mode	1	0	Disconnect from SPI	RX
		1		TX/RX

24.4 Programming Procedure

This section describes the procedure for configuring the SPI for the various modes of operation.

24.4.1 Initialization Upon Reset

A system reset forces the SPI peripheral into the following default configuration:

- Unit is configured as a slave module (MASTER_SLAVE = 0)
- Transmit capability is disabled (TALK = 0)
- Data is latched at the input on the falling edge of the SPICLK signal
- Character length is assumed to be one bit
- SPI interrupts are disabled
- Data in SPIDAT is reset to 0000h

24.4.2 Configuring the SPI

This section describes the procedure in which to configure the SPI module for operation. To prevent unwanted and unforeseen events from occurring during or as a result of initialization changes, clear the SPISWRESET bit before making initialization changes, and then set this bit after initialization is complete. While the SPI is held in reset (SPISWRESET = 0), configuration may be changed in any order. The following list shows the SPI configuration procedure in a logical order. However, the SPI registers can be written with single 16-bit writes, so the order is not required with the exception of SPISWRESET.

To change the SPI configuration:

1. Clear the SPI Software Reset bit (SPISWRESET) to 0 to force the SPI to the reset state.
2. Configure the SPI as desired:
   - Select either master or slave mode (MASTER_SLAVE).
   - Choose SPICLK polarity and phase (CLKPOLARITY and CLK_PHASE).
   - Set the desired baud rate (SPIBRR).
   - Enable high speed mode if desired (HS_MODE).
   - Set the SPI character length (SPICHAR).
   - Clear the SPI Flags (OVERRUN_FLAG, INT_FLAG).
   - Enable $\overline{\text{SPISTE}}$ inversion (STEINV) if needed.
   - Enable 3-wire mode (TRIWIRE) if needed.
   - If using FIFO enhancements:
     - Enable the FIFO enhancements (SPIFFENA).
     - Clear the FIFO Flags (TXFFINTCLR, RXFFOVFCLR, and RXFFINTCLR).
     - Release transmit and receive FIFO resets (TXFIFO and RXFIFORESET).
     - Release SPI FIFO channels from reset (SPIRST).
3. If interrupts are used:
   - In non-FIFO mode, enable the receiver overrun and/or SPI interrupts (OVERRUNINTENA and SPIINTENA).
   - In FIFO mode, set the transmit and receive interrupt levels (TXFFIL and RXFFIL) then enable the interrupts (TXFFIENA and RXFFIENA).
4. Set SPISWRESET to 1 to release the SPI from the reset state.

Note

Do not change the SPI configuration when communication is in progress.

24.4.3 Configuring the SPI for High-Speed Mode

In order to achieve the maximum rated speeds, the following settings must be made. This example assumes that the device is operating at 100 MHz.

Set LSPCLK equal to SYSCLK:

```
ClkCfgRegs.LOSPCP.bit.LSPCLKDIV = 0;
```

Select the appropriate Pin Mux options in GPIO_CTRL_REGS.

During the SPI configuration procedure:

Set HS_MODE to 1.

```
SpiaRegs.SPICCR.bit.HS_MODE = 0x1;
```

Set SPIBRR to 3. SPICLK = LSPCLK/(SPIBRR+1) = 25- MHz

```
SpiaRegs.SPIBRR = 0x3;
```

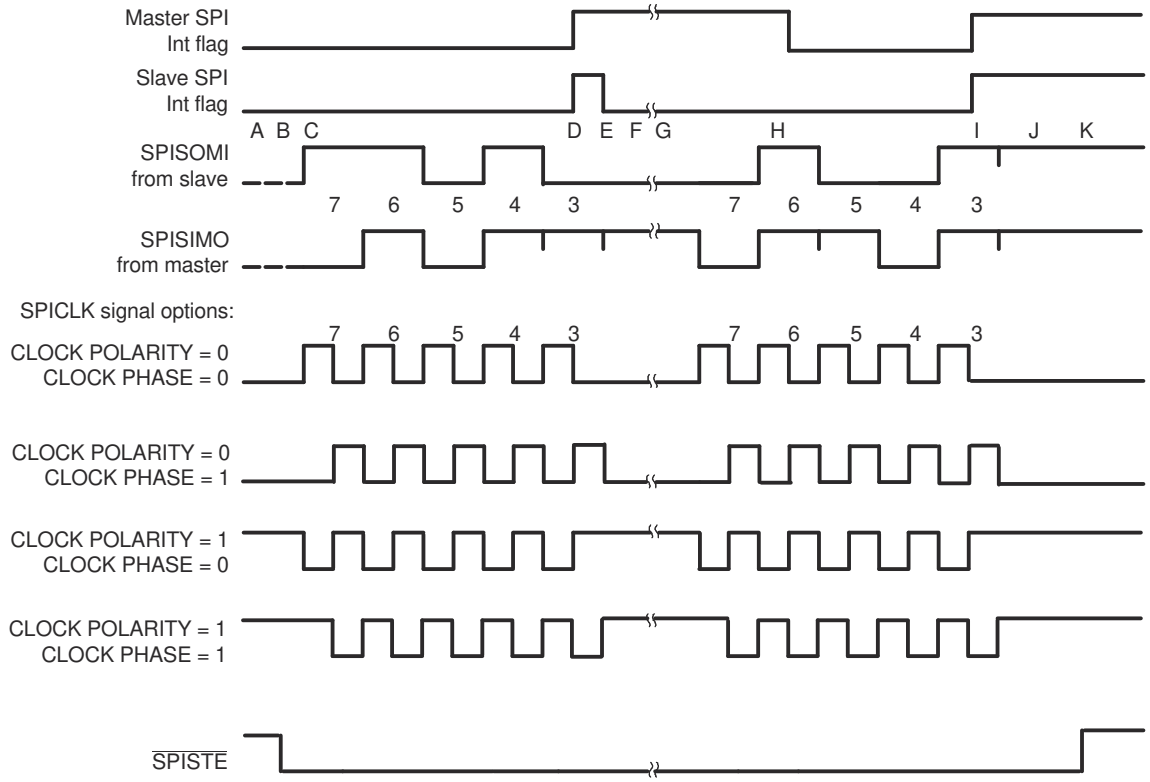
There are no other differences in the configuration from normal SPI operation. Sending and receiving data, DMA operation, and interrupts will operate without change.

24.4.4 Data Transfer Example

The timing diagram shown in Figure 24-11 illustrates an SPI data transfer between two devices using a character length of five bits with the SPICLK being symmetrical.

The timing diagram with SPICLK asymmetrical (Figure 24-8) shares similar characterizations with Figure 24-11 except that the data transfer is one LSPCLK cycle longer per bit during the low pulse (CLKPOLARITY = 0) or during the high pulse (CLKPOLARITY = 1) of the SPICLK.

Figure 24-11 is applicable for 8-bit SPI only and is not for C28x devices that are capable of working with 16-bit data. The figure is shown for illustrative purposes only.



A. Slave writes 0D0h to SPIDAT and waits for the master to shift out the data.
B. Master sets the slave $\overline{\text{SPISTE}}$ signal low (active).
C. Master writes 058h to SPIDAT, which starts the transmission procedure.
D. First byte is finished and sets the interrupt flags.
E. Slave reads 0Bh from its SPIRXBUF (right-justified).
F. Slave writes 04Ch to SPIDAT and waits for the master to shift out the data.
G. Master writes 06Ch to SPIDAT, which starts the transmission procedure.
H. Master reads 01Ah from the SPIRXBUF (right−justified).
I. Second byte is finished and sets the interrupt flags.
J. Master reads 89h and the slave reads 8Dh from their respective SPIRXBUF. After the user's software masks off the unused bits, the master receives 09h and the slave receives 0Dh.
K. Master clears the slave $\overline{\text{SPISTE}}$ signal high (inactive).

Figure 24-11. Five Bits per Character

24.4.5 SPI 3-Wire Mode Code Examples

In addition to the normal SPI initialization, to configure the SPI module for 3-wire mode, the TRIWIRE bit (SPIPRI.0) must be set to 1. After initialization, there are several considerations to take into account when transmitting and receiving data in 3-wire master and slave mode. The following examples demonstrate these considerations.

In 3-wire master mode, SPICLKx, SPISTEx, and SPISIMOx pins must be configured as SPI pins (SPISOMIx pin can be configured as non-SPI pin). When the master transmits, it receives the data it transmits (because SPISIMOx and SPISOMIx are connected internally in 3-wire mode). Therefore, the junk data received must be cleared from the receive buffer every time data is transmitted.

Example 24-4. 3-Wire Master Mode Transmit

```
Uint16 data;
Uint16 dummy;
     SpiaRegs.SPICTL.bit.TALK = 1;              // Enable Transmit path
     SpiaRegs.SPITXBUF = data; // Master transmits data
     while(SpiaRegs.SPISTS.bit.INT_FLAG !=1) {} // Waits until data rx'd
     dummy = SpiaRegs.SPIRXBUF;                 // Clears junk data from itself
                                                // bc it rx'd same data tx'd
```

To receive data in 3-wire master mode, the master must clear the TALK (SPICTL.1) bit to 0 to close the transmit path and then transmit dummy data in order to initiate the transfer from the slave. Because the TALK bit is 0, unlike in transmit mode, the master dummy data does not appear on the SPISIMOx pin, and the master does not receive its own dummy data. Instead, the data from the slave is received by the master.

Example 24-5. 3-Wire Master Mode Receive

```
Uint16 rdata;
Uint16 dummy;
     SpiaRegs.SPICTL.bit.TALK = 0;              // Disable Transmit path
     SpiaRegs.SPITXBUF = dummy;                 // Send dummy to start tx
     // NOTE: because TALK = 0, data does not tx onto SPISIMOA pin
     while(SpiaRegs.SPISTS.bit.INT_FLAG !=1) {} // Wait until data received
     rdata = SpiaRegs.SPIRXBUF;                 // Master reads data
```

In 3-wire slave mode, SPICLKx, SPISTEx, and SPISOMIx pins must be configured as SPI pins (SPISIMOx pin can be configured as non-SPI pin). Like in master mode, when transmitting, the slave receives the data it transmits and must clear this junk data from its receive buffer.

Example 24-6. 3-Wire Slave Mode Transmit

```
Uint16 data;
Uint16 dummy;
     SpiaRegs.SPICTL.bit.TALK = 1;              // Enable Transmit path
     SpiaRegs.SPITXBUF = data;                  // Slave transmits data
     while(SpiaRegs.SPISTS.bit.INT_FLAG !=1) {} // Wait until data rx'd
     dummy = SpiaRegs.SPIRXBUF;                 // Clears junk data from itself
```

As in 3-wire master mode, the TALK bit must be cleared to 0. Otherwise, the slave receives data normally.

Example 24-7. 3-Wire Slave Mode Receive

```
Uint16 rdata;
    SpiaRegs.SPICTL.bit.TALK = 0;             // Disable Transmit path
    while(SpiaRegs.SPISTS.bit.INT_FLAG !=1) {} // Waits until data rx'd
    rdata = SpiaRegs.SPIRXBUF;                // Slave reads data
```

24.4.6 SPI STEINV Bit in Digital Audio Transfers

On those devices with two SPI modules, enabling the STEINV bit on one of the SPI modules allows the pair of SPIs to receive both left and right-channel digital audio data in slave mode. The SPI module that receives a normal active-low $\overline{\text{SPISTE}}$ signal stores right-channel data, and the SPI module that receives an inverted active-high $\overline{\text{SPISTE}}$ signal stores left-channel data from the master. To receive digital audio data from a digital audio interface receiver, the SPI modules can be connected as shown in Figure 24-12.

Note

This configuration is only applicable to slave mode (MASTER_SLAVE = 0). When the SPI is configured as master (MASTER_SLAVE = 1), the STEINV bit will have no effect on the $\overline{\text{SPISTE}}$ pin.

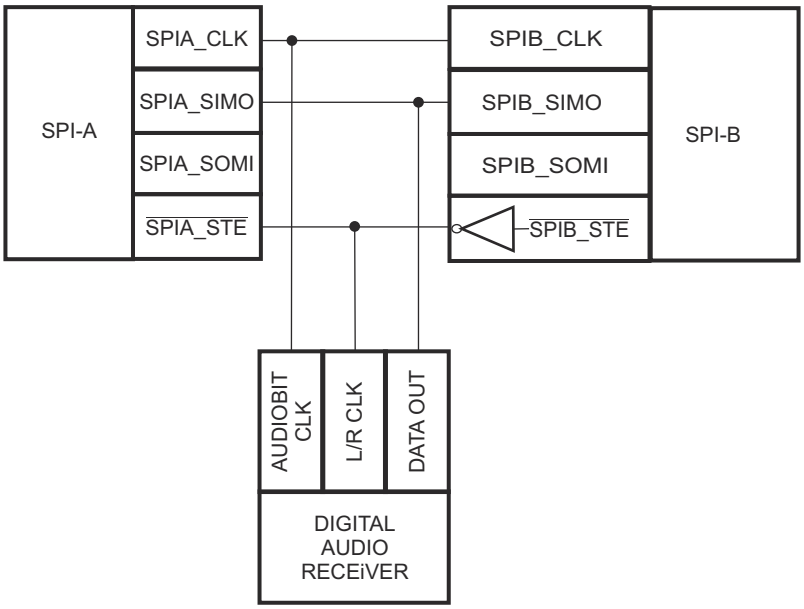


Figure 24-12. SPI Digital Audio Receiver Configuration Using Two SPIs

Standard C28x SPI timing requirements limit the number of digital audio interface formats supported using the 2-SPI configuration with the STEINV bit. See your device-specific data sheet electrical specifications for SPI timing requirements. With the SPI clock phase configured such that the CLKPOLARITY bit is 0 and the CLK_PHASE bit is 1 (data latched on rising edge of clock), standard right-justified digital audio interface data format is supported as shown in Figure 24-13.

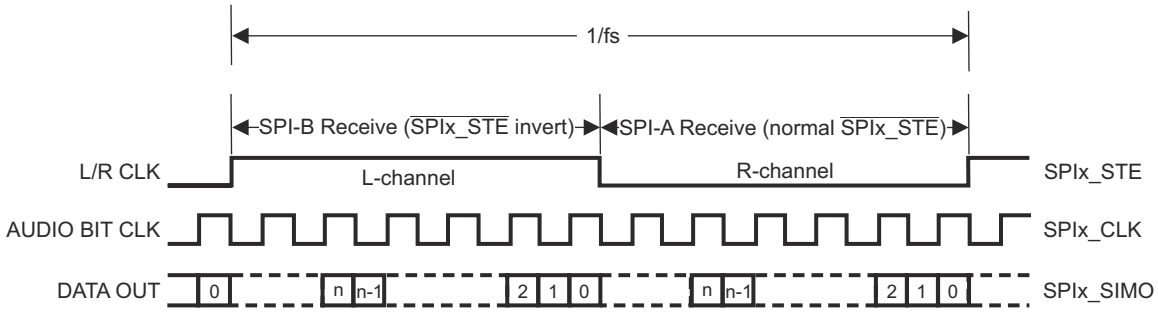


Figure 24-13. Standard Right-Justified Digital Audio Data Format

24.5 Software

24.5.1 SPI Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/f28003x/examples/spi

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

24.5.1.1 SPI Digital Loopback

FILE: spi_ex1_loopback.c

This program uses the internal loopback test mode of the SPI module. This is a very basic loopback that does not use the FIFOs or interrupts. A stream of data is sent and then compared to the received stream. The pinmux and SPI modules are configure through the sysconfig file.

The sent data looks like this:
0000 0001 0002 0003 0004 0005 0006 0007 .... FFFE FFFF 0000

This pattern is repeated forever.

External Connections
- None

Watch Variables
- *sData* - Data to send
- *rData* - Received data

24.5.1.2 SPI Digital Loopback with FIFO Interrupts

FILE: spi_ex2_loopback_fifo_interrupts.c

This program uses the internal loopback test mode of the SPI module. Both the SPI FIFOs and their interrupts are used.

A stream of data is sent and then compared to the received stream. The sent data looks like this:
0000 0001
0001 0002
0002 0003
....
FFFE FFFF
FFFF 0000
etc..
This pattern is repeated forever.

External Connections
- None

Watch Variables
- *sData* - Data to send
- *rData* - Received data
- *rDataPoint* - Used to keep track of the last position in the receive stream for error checking

24.5.1.3 SPI Digital External Loopback without FIFO Interrupts

FILE: spi_ex3_external_loopback.c

This program uses the external loopback between two SPI modules. Both the SPI FIFOs and interrupts are not used in this example. SPIA is configured as a slave and SPI B is configured as master. This example demonstrates full duplex communication where both master and slave transmits and receives data simultaneously.

External Connections

- GPIO40 and GPIO8 - SPISIMO
- GPIO41 and GPIO10 - SPISOMI
- GPIO22 and GPIO9 - SPICLK
- GPIO23 and GPIO11 - SPISTE

Watch Variables

- *TxData_SPIA* - Data send from SPIA (slave)
- *TxData_SPIB* - Data send from SPIB (master)
- *RxData_SPIA* - Data received by SPIA (slave)
- *RxData_SPIB* - Data received by SPIB (master)

24.5.1.4 SPI Digital External Loopback with FIFO Interrupts

FILE: spi_ex4_external_loopback_fifo_interrupts.c

This program uses the external loopback between two SPI modules. Both the SPI FIFOs and their interrupts are used. SPIA is configured as a slave and receives data from SPI B which is configured as a master.

A stream of data is sent and then compared to the received stream. The sent data looks like this:
0000 0001
0001 0002
0002 0003
....
FFFE FFFF
FFFF 0000
etc..
This pattern is repeated forever.

External Connections

- GPIO40 and GPIO8 - SPISIMO
- GPIO41 and GPIO10 - SPISOMI
- GPIO22 and GPIO9 - SPICLK
- GPIO23 and GPIO11 - SPISTE

Watch Variables

- *sData* - Data to send
- *rData* - Received data
- *rDataPoint* - Used to keep track of the last position in the receive stream for error checking

24.5.1.5 SPI Digital Loopback with DMA

FILE: spi_ex5_loopback_dma.c

This program uses the internal loopback test mode of the SPI module. Both DMA interrupts and the SPI FIFOs are used. When the SPI transmit FIFO has enough space (as indicated by its FIFO level interrupt signal), the DMA will transfer data from global variable sData into the FIFO. This will be transmitted to the receive FIFO via the internal loopback.

When enough data has been placed in the receive FIFO (as indicated by its FIFO level interrupt signal), the DMA will transfer the data from the FIFO into global variable rData.

When all data has been placed into rData, a check of the validity of the data will be performed in one of the DMA channels' ISRs.

External Connections

- None

Watch Variables

- *sData* - Data to send
- *rData* - Received data

24.5.1.6 SPI EEPROM

FILE: spi_ex6_eeprom.c

This program will write 8 bytes to EEPROM and read them back. The device communicates with the EEPROM via SPI and specific opcodes. This example is written to work with the SPI Serial EEPROM AT25128/256.

External Connections

- Connect external SPI EEPROM
- Connect GPIO08 (SIMO) to external EEPROM SI pin
- Connect GPIO10 (SOMI) to external EEPROM SO pin
- Connect GPIO09 (CLK) to external EEPROM SCK pin
- Connect GPIO11 (CS) to external EEPROM CS pin
- Connect the external EEPROM VCC and GND pins

Watch Variables

- writeBuffer - Data that is written to external EEPROM
- readBuffer - Data that is read back from EEPROM
- error - Error count

24.5.1.7 SPI DMA EEPROM

FILE: spi_ex7_eeprom_dma.c

This program will write 8 bytes to EEPROM and read them back. The device communicates with the EEPROM via SPI using DMA and specific opcodes. This example is written to work with the SPI Serial EEPROM AT25128/256.

External Connections

- Connect external SPI EEPROM
- Connect GPIO08 (SIMO) to external EEPROM SI pin
- Connect GPIO10 (SOMI) to external EEPROM SO pin
- Connect GPIO09 (CLK) to external EEPROM SCK pin
- Connect GPIO11 (CS) to external EEPROM CS pin
- Connect the external EEPROM VCC and GND pins

Watch Variables

- writeBuffer - Data that is written to external EEPROM
- SPI_DMA_Handle.RXdata - Data that is read back from EEPROM when number of received bytes is less than 4
- SPI_DMA_Handle.pSPIRXDMA->pbuffer - Start address of received data from EEPROM
- error - Error count

24.6 SPI Registers

This section describes the Serial Peripheral Interface registers. It is important to note that the SPI registers only allow 16-bit accesses.

24.6.1 SPI Base Address Table

Table 24-6. SPI Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
SpiaRegs	SPI_REGS	SPIA_BASE	0x0000_6100	YES	YES	YES	YES	YES
SpibRegs	SPI_REGS	SPIB_BASE	0x0000_6110	YES	YES	YES	YES	YES

24.6.2 SPI_REGS Registers

Table 24-7 lists the memory-mapped registers for the SPI_REGS registers. All register offset addresses not listed in Table 24-7 should be considered as reserved locations and the register contents should not be modified.

Table 24-7. SPI_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	SPICCR	SPI Configuration Control Register		Go
1h	SPICTL	SPI Operation Control Register		Go
2h	SPISTS	SPI Status Register		Go
4h	SPIBRR	SPI Baud Rate Register		Go
6h	SPIRXEMU	SPI Emulation Buffer Register		Go
7h	SPIRXBUF	SPI Serial Input Buffer Register		Go
8h	SPITXBUF	SPI Serial Output Buffer Register		Go
9h	SPIDAT	SPI Serial Data Register		Go
Ah	SPIFFTX	SPI FIFO Transmit Register		Go
Bh	SPIFFRX	SPI FIFO Receive Register		Go
Ch	SPIFFCT	SPI FIFO Control Register		Go
Fh	SPIPRI	SPI Priority Control Register		Go

Complex bit access types are encoded to fit into small table cells. Table 24-8 shows the codes that are used for access types in this section.

Table 24-8. SPI_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
RC	R C	Read to Clear
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

24.6.2.1 SPICCR Register (Offset = 0h) [Reset = 0h]

SPICCR is shown in Figure 24-14 and described in Table 24-9.

Return to the Summary Table.

SPICCR controls the setup of the SPI for operation.

Figure 24-14. SPICCR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
SPISWRESET	CLKPOLARITY	HS_MODE	SPILBK	SPICHAR			
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h			

Table 24-9. SPICCR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	SPISWRESET	R/W	0h	SPI Software Reset When changing configuration, you should clear this bit before the changes and set this bit before resuming operation. Reset type: SYSRSn 0h (R/W) = Initializes the SPI operating flags to the reset condition. Specifically, the RECEIVER OVERRUN Flag bit (SPISTS.7), the SPI INT FLAG bit (SPISTS.6), and the TXBUF FULL Flag bit (SPISTS.5) are cleared. SPISTE will become inactive. SPICLK will be immediately driven to 0 regardless of the clock polarity. The SPI configuration remains unchanged. 1h (R/W) = SPI is ready to transmit or receive the next character. When the SPI SW RESET bit is a 0, a character written to the transmitter will not be shifted out when this bit is set. A new character must be written to the serial data register. SPICLK will be returned to its inactive state one SPICLK cycle after this bit is set.
6	CLKPOLARITY	R/W	0h	Shift Clock Polarity This bit controls the polarity of the SPICLK signal. CLOCK POLARITY and POLARITY CLOCK PHASE (SPICTL.3) control four clocking schemes on the SPICLK pin. Reset type: SYSRSn 0h (R/W) = Data is output on rising edge and input on falling edge. When no SPI data is sent, SPICLK is at low level. The data input and output edges depend on the value of the CLOCK PHASE bit (SPICTL.3) as follows: - CLOCK PHASE = 0: Data is output on the rising edge of the SPICLK signal. Input data is latched on the falling edge of the SPICLK signal. - CLOCK PHASE = 1: Data is output one half-cycle before the first rising edge of the SPICLK signal and on subsequent falling edges of the SPICLK signal. Input data is latched on the rising edge of the SPICLK signal. 1h (R/W) = Data is output on falling edge and input on rising edge. When no SPI data is sent, SPICLK is at high level. The data input and output edges depend on the value of the CLOCK PHASE bit (SPICTL.3) as follows: - CLOCK PHASE = 0: Data is output on the falling edge of the SPICLK signal. Input data is latched on the rising edge of the SPICLK signal. - CLOCK PHASE = 1: Data is output one half-cycle before the first falling edge of the SPICLK signal and on subsequent rising edges of the SPICLK signal. Input data is latched on the falling edge of the SPICLK signal.

Table 24-9. SPICCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	HS_MODE	R/W	0h	High Speed Mode Enable Bits This bit determines if the High Speed mode is enabled. The correct GPIOs should be selected in the GPxGMUX/GPxMUX registers. Reset type: SYSRSn 0h (R/W) = SPI High Speed mode disabled. This is the default value after reset. 1h (R/W) = SPI High Speed mode enabled,
4	SPILBK	R/W	0h	SPI Loopback Mode Select Loopback mode allows module validation during device testing. This mode is valid only in master mode of the SPI. Reset type: SYSRSn 0h (R/W) = SPI loopback mode disabled. This is the default value after reset. 1h (R/W) = SPI loopback mode enabled, SIMO/SOMI lines are connected internally. Used for module self-tests.
3-0	SPICHAR	R/W	0h	Character Length Control Bits These four bits determine the number of bits to be shifted in or SPI CHAR0 out as a single character during one shift sequence. SPICHAR = Word length - 1 Reset type: SYSRSn 0h (R/W) = 1-bit word 1h (R/W) = 2-bit word 7h (R/W) = 8-bit word Fh (R/W) = 16-bit word

24.6.2.2 SPICTL Register (Offset = 1h) [Reset = 0h]

SPICTL is shown in Figure 24-15 and described in Table 24-10.

Return to the Summary Table.

SPICTL controls data transmission, the SPI's ability to generate interrupts, the SPICLK phase, and the operational mode (slave or master).

Figure 24-15. SPICTL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			OVERRUNINTENA	CLK_PHASE	MASTER_SLAVE	TALK	SPIINTENA
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 24-10. SPICTL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R	0h	Reserved
4	OVERRUNINTENA	R/W	0h	Overrun Interrupt Enable Overrun Interrupt Enable. Setting this bit causes an interrupt to be generated when the RECEIVER OVERRUN Flag bit (SPISTS.7) is set by hardware. Interrupts generated by the RECEIVER OVERRUN Flag bit and the SPI INT FLAG bit (SPISTS.6) share the same interrupt vector. Reset type: SYSRSn 0h (R/W) = Disable RECEIVER OVERRUN interrupts. 1h (R/W) = Enable RECEIVER_OVERRUN interrupts.
3	CLK_PHASE	R/W	0h	SPI Clock Phase Select This bit controls the phase of the SPICLK signal. CLOCK PHASE and CLOCK POLARITY (SPICCR.6) make four different clocking schemes possible (see clocking figures in SPI chapter). When operating with CLOCK PHASE high, the SPI (master or slave) makes the first bit of data available after SPIDAT is written and before the first edge of the SPICLK signal, regardless of which SPI mode is being used. Reset type: SYSRSn 0h (R/W) = Normal SPI clocking scheme, depending on the CLOCK POLARITY bit (SPICCR.6). 1h (R/W) = SPICLK signal delayed by one half-cycle. Polarity determined by the CLOCK POLARITY bit.
2	MASTER_SLAVE	R/W	0h	SPI Network Mode Control This bit determines whether the SPI is a network master or slave. SLAVE During reset initialization, the SPI is automatically configured as a network slave. Reset type: SYSRSn 0h (R/W) = SPI is configured as a slave. 1h (R/W) = SPI is configured as a master.

Table 24-10. SPICTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	TALK	R/W	0h	Transmit Enable The TALK bit can disable data transmission (master or slave) by placing the serial data output in the high-impedance state. If this bit is disabled during a transmission, the transmit shift register continues to operate until the previous character is shifted out. When the TALK bit is disabled, the SPI is still able to receive characters and update the status flags. TALK is cleared (disabled) by a system reset. Reset type: SYSRSn 0h (R/W) = Disables transmission: - Slave mode operation: If not previously configured as a general-purpose I/O pin, the SPISOMI pin will be put in the high-impedance state. - Master mode operation: If not previously configured as a general-purpose I/O pin, the SPISIMO pin will be put in the high-impedance state. 1h (R/W) = Enables transmission For the 4-pin option, ensure to enable the receiver's SPISTEn input pin.
0	SPIINTENA	R/W	0h	SPI Interrupt Enable This bit controls the SPI's ability to generate a transmit/receive interrupt. The SPI INT FLAG bit (SPISTS.6) is unaffected by this bit. Reset type: SYSRSn 0h (R/W) = Disables the interrupt. 1h (R/W) = Enables the interrupt.

24.6.2.3 SPISTS Register (Offset = 2h) [Reset = 0h]

SPISTS is shown in Figure 24-16 and described in Table 24-11.

Return to the Summary Table.

SPISTS contrains interrupt and status bits.

Figure 24-16. SPISTS Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
OVERRUN_FLAG	INT_FLAG	BUFFULL_FLAG	RESERVED				
W1C-0h	RC-0h	R-0h	R-0h				

Table 24-11. SPISTS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	OVERRUN_FLAG	W1C	0h	SPI Receiver Overrun Flag This bit is a read/clear-only flag. The SPI hardware sets this bit when a receive or transmit operation completes before the previous character has been read from the buffer. The bit is cleared in one of three ways: - Writing a 1 to this bit - Writing a 0 to SPI SW RESET (SPICCR.7) - Resetting the system If the OVERRUN INT ENA bit (SPICTL.4) is set, the SPI requests only one interrupt upon the first occurrence of setting the RECEIVER OVERRUN Flag bit. Subsequent overruns will not request additional interrupts if this flag bit is already set. This means that in order to allow new overrun interrupt requests the user must clear this flag bit by writing a 1 to SPISTS.7 each time an overrun condition occurs. In other words, if the RECEIVER OVERRUN Flag bit is left set (not cleared) by the interrupt service routine, another overrun interrupt will not be immediately re-entered when the interrupt service routine is exited. Reset type: SYSRSn 0h (R/W) = A receive overrun condition has not occurred. 1h (R/W) = The last received character has been overwritten and therefore lost (when the SPIRXBUF was overwritten by the SPI module before the previous character was read by the user application). Writing a '1' will clear this bit. The RECEIVER OVERRUN Flag bit should be cleared during the interrupt service routine because the RECEIVER OVERRUN Flag bit and SPI INT FLAG bit (SPISTS.6) share the same interrupt vector. This will alleviate any possible doubt as to the source of the interrupt when the next byte is received.

Table 24-11. SPISTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	INT_FLAG	RC	0h	SPI Interrupt Flag SPI INT FLAG is a read-only flag. Hardware sets this bit to indicate that the SPI has completed sending or receiving the last bit and is ready to be serviced. This flag causes an interrupt to be requested if the SPI INT ENA bit (SPICTL.0) is set. The received character is placed in the receiver buffer at the same time this bit is set. This bit is cleared in one of three ways: - Reading SPIRXBUF - Writing a 0 to SPI SW RESET (SPICCR.7) - Resetting the system Note: This bit should not be used if FIFO mode is enabled. The internal process of copying the received word from SPIRXBUF to the Receive FIFO will clear this bit. Use the FIFO status, or FIFO interrupt bits for similar functionality. Reset type: SYSRSn 0h (R/W) = No full words have been received or transmitted. 1h (R/W) = Indicates that the SPI has completed sending or receiving the last bit and is ready to be serviced.
5	BUFFULL_FLAG	R	0h	SPI Transmit Buffer Full Flag This read-only bit gets set to 1 when a character is written to the SPI Transmit buffer SPITXBUF. It is cleared when the character is automatically loaded into SPIDAT when the shifting out of a previous character is complete. Reset type: SYSRSn 0h (R/W) = Transmit buffer is not full. 1h (R/W) = Transmit buffer is full.
4-0	RESERVED	R	0h	Reserved

24.6.2.4 SPIBRR Register (Offset = 4h) [Reset = 0h]

SPIBRR is shown in Figure 24-17 and described in Table 24-12.

Return to the Summary Table.

SPIBRR contains the bits used for baud-rate selection.

Figure 24-17. SPIBRR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	SPI_BIT_RATE						
R-0h	R/W-0h						

Table 24-12. SPIBRR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R	0h	Reserved
6-0	SPI_BIT_RATE	R/W	0h	SPI Baud Rate Control These bits determine the bit transfer rate if the SPI is the network SPI BIT RATE 0 master. There are 125 data-transfer rates (each a function of the CPU clock, LSPCLK) that can be selected. One data bit is shifted per SPICLK cycle. (SPICLK is the baud rate clock output on the SPICLK pin.) If the SPI is a network slave, the module receives a clock on the SPICLK pin from the network master. Therefore, these bits have no effect on the SPICLK signal. The frequency of the input clock from the master should not exceed the slave SPI's LSPCLK signal divided by 4. In master mode, the SPI clock is generated by the SPI and is output on the SPICLK pin. The SPI baud rates are determined by the following formula: For SPIBRR = 3 to 127: SPI Baud Rate = LSPCLK / (SPIBRR + 1) For SPIBRR = 0, 1, or 2: SPI Baud Rate = LSPCLK / 4 Reset type: SYSRSn 3h (R/W) = SPI Baud Rate = LSPCLK/4 4h (R/W) = SPI Baud Rate = LSPCLK/5 7Eh (R/W) = SPI Baud Rate = LSPCLK/127 7Fh (R/W) = SPI Baud Rate = LSPCLK/128

24.6.2.5 SPIRXEMU Register (Offset = 6h) [Reset = 0h]

SPIRXEMU is shown in Figure 24-18 and described in Table 24-13.

Return to the Summary Table.

SPIRXEMU contains the received data. Reading SPIRXEMU does not clear the SPI INT FLAG bit of SPISTS. This is not a real register but a dummy address from which the contents of SPIRXBUF can be read by the emulator without clearing the SPI INT FLAG.

Figure 24-18. SPIRXEMU Register

15	14	13	12	11	10	9	8
ERXBn							
R-0h							
7	6	5	4	3	2	1	0
ERXBn							
R-0h							

Table 24-13. SPIRXEMU Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	ERXBn	R	0h	Emulation Buffer Received Data SPIRXEMU functions almost identically to SPIRXBUF, except that reading SPIRXEMU does not clear the SPI INT FLAG bit (SPISTS.6). Once the SPIDAT has received the complete character, the character is transferred to SPIRXEMU and SPIRXBUF, where it can be read. At the same time, SPI INT FLAG is set. This mirror register was created to support emulation. Reading SPIRXBUF clears the SPI INT FLAG bit (SPISTS.6). In the normal operation of the emulator, the control registers are read to continually update the contents of these registers on the display screen. SPIRXEMU was created so that the emulator can read this register and properly update the contents on the display screen. Reading SPIRXEMU does not clear the SPI INT FLAG bit, but reading SPIRXBUF clears this flag. In other words, SPIRXEMU enables the emulator to emulate the true operation of the SPI more accurately. It is recommended that you view SPIRXEMU in the normal emulator run mode. Reset type: SYSRSn

24.6.2.6 SPIRXBUF Register (Offset = 7h) [Reset = 0h]

SPIRXBUF is shown in Figure 24-19 and described in Table 24-14.

Return to the Summary Table.

SPIRXBUF contains the received data. Reading SPIRXBUF clears the SPI INT FLAG bit in SPISTS. If FIFO mode is enabled, reading this register will also decrement the RXFFST counter in SPIFFRX.

Figure 24-19. SPIRXBUF Register

15	14	13	12	11	10	9	8
RXBn							
R-0h							
7	6	5	4	3	2	1	0
RXBn							
R-0h							

Table 24-14. SPIRXBUF Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	RXBn	R	0h	Received Data Once SPIDAT has received the complete character, the character is transferred to SPIRXBUF, where it can be read. At the same time, the SPI INT FLAG bit (SPISTS.6) is set. Since data is shifted into the SPI's most significant bit first, it is stored right-justified in this register. Reset type: SYSRSn

24.6.2.7 SPITXBUF Register (Offset = 8h) [Reset = 0h]

SPITXBUF is shown in Figure 24-20 and described in Table 24-15.

Return to the Summary Table.

SPITXBUF stores the next character to be tranmitted. Writing to this register sets the TX BUF FULL Flag bit in SPISTS. When the transmission of the current character is complete, the contents of this register are automatically loaded in SPIDAT and the TX BUF FULL Flag is cleared. If no tranmission is currently active, data written to this register falls through into the SPIDAT register and the TX BUF FULL Flag is not set.
In master mode, if no tranmission is currently active, writing to this register initiates a transmission in the same manner that writing to SPIDAT does.

Figure 24-20. SPITXBUF Register

15	14	13	12	11	10	9	8
TXBn							
R/W-0h							
7	6	5	4	3	2	1	0
TXBn							
R/W-0h							

Table 24-15. SPITXBUF Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	TXBn	R/W	0h	Transmit Data Buffer This is where the next character to be transmitted is stored. When the transmission of the current character has completed, if the TX BUF FULL Flag bit is set, the contents of this register is automatically transferred to SPIDAT, and the TX BUF FULL Flag is cleared. Writes to SPITXBUF must be left-justified. Reset type: SYSRSn

24.6.2.8 SPIDAT Register (Offset = 9h) [Reset = 0h]

SPIDAT is shown in Figure 24-21 and described in Table 24-16.

Return to the Summary Table.

SPIDAT is the transmit and receive shift register. Data written to SPIDAT is shifted out (MSB) on subsequent SPICLK cycles. For every bit (MSB) shifted out of the SPI, a bit is shifted into the LSB end of the shift register.

Figure 24-21. SPIDAT Register

15	14	13	12	11	10	9	8
SDATn							
R/W-0h							
7	6	5	4	3	2	1	0
SDATn							
R/W-0h							

Table 24-16. SPIDAT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	SDATn	R/W	0h	Serial Data Shift Register - It provides data to be output on the serial output pin if the TALK bit (SPICTL.1) is set. - When the SPI is operating as a master, a data transfer is initiated. When initiating a transfer, check the CLOCK POLARITY bit (SPICCR.6) described in Section 10.2.1.1 and the CLOCK PHASE bit (SPICTL.3) described in Section 10.2.1.2, for the requirements. In master mode, writing dummy data to SPIDAT initiates a receiver sequence. Since the data is not hardware-justified for characters shorter than sixteen bits, transmit data must be written in left-justified form, and received data read in right-justified form. Reset type: SYSRSn

24.6.2.9 SPIFFTX Register (Offset = Ah) [Reset = A000h]

SPIFFTX is shown in Figure 24-22 and described in Table 24-17.

Return to the Summary Table.

SPIFFTX contains both control and status bits related to the output FIFO buffer. This includes FIFO reset control, FIFO interrupt level control, FIFO level status, as well as FIFO interrupt enable and clear bits.

Figure 24-22. SPIFFTX Register

15	14	13	12	11	10	9	8
SPIRST	SPIFFENA	TXFIFO	TXFFST				
R/W-1h	R/W-0h	R/W-1h	R-0h				
7	6	5	4	3	2	1	0
TXFFINT	TXFFINTCLR	TXFFIENA	TXFFIL				
R-0h	W-0h	R/W-0h	R/W-0h				

Table 24-17. SPIFFTX Register Field Descriptions

Bit	Field	Type	Reset	Description
15	SPIRST	R/W	1h	SPI Reset Reset type: SYSRSn 0h (R/W) = Write 0 to reset the SPI transmit and receive channels. The SPI FIFO register configuration bits will be left as is. 1h (R/W) = SPI FIFO can resume transmit or receive. No effect to the SPI registers bits.
14	SPIFFENA	R/W	0h	SPI FIFO Enhancements Enable Reset type: SYSRSn 0h (R/W) = SPI FIFO enhancements are disabled. 1h (R/W) = SPI FIFO enhancements are enabled.
13	TXFIFO	R/W	1h	TX FIFO Reset Reset type: SYSRSn 0h (R/W) = Write 0 to reset the FIFO pointer to zero, and hold in reset. 1h (R/W) = Release transmit FIFO from reset.
12-8	TXFFST	R	0h	Transmit FIFO Status Reset type: SYSRSn 0h (R/W) = Transmit FIFO is empty. 1h (R/W) = Transmit FIFO has 1 word. 2h (R/W) = Transmit FIFO has 2 words. 10h (R/W) = Transmit FIFO has 16 words, which is the maximum. 1Fh (R/W) = Reserved.
7	TXFFINT	R	0h	TX FIFO Interrupt Flag Reset type: SYSRSn 0h (R/W) = TXFIFO interrupt has not occurred, This is a read-only bit. 1h (R/W) = TXFIFO interrupt has occurred, This is a read-only bit.
6	TXFFINTCLR	W	0h	TXFIFO Interrupt Clear Reset type: SYSRSn 0h (R/W) = Write 0 has no effect on TXFIFINT flag bit, Bit reads back a zero. 1h (R/W) = Write 1 to clear SPIFFTX[TXFFINT] flag.
5	TXFFIENA	R/W	0h	TX FIFO Interrupt Enable Reset type: SYSRSn 0h (R/W) = TX FIFO interrupt based on TXFFIL match (less than or equal to) will be disabled. 1h (R/W) = TX FIFO interrupt based on TXFFIL match (less than or equal to) will be enabled.

Table 24-17. SPIFFTX Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-0	TXFFIL	R/W	0h	Transmit FIFO Interrupt Level Bits Transmit FIFO will generate interrupt when the FIFO status bits (TXFFST4-0) and FIFO level bits (TXFFIL4-0) match (less than or equal to). Reset type: SYSRSn 0h (R/W) = A TX FIFO interrupt request is generated when there are no words remaining in the TX buffer. 1h (R/W) = A TX FIFO interrupt request is generated when there is 1 word or no words remaining in the TX buffer. 2h (R/W) = A TX FIFO interrupt request is generated when there is 2 words or fewer remaining in the TX buffer. 10h (R/W) = A TX FIFO interrupt request is generated when there are 16 words or fewer remaining in the TX buffer. 1Fh (R/W) = Reserved.

24.6.2.10 SPIFFRX Register (Offset = Bh) [Reset = 201Fh]

SPIFFRX is shown in Figure 24-23 and described in Table 24-18.

Return to the Summary Table.

SPIFFRX contains both control and status bits related to the input FIFO buffer. This includes FIFO reset control, FIFO interrupt level control, FIFO level status, as well as FIFO interrupt enable and clear bits.

Figure 24-23. SPIFFRX Register

15	14	13	12	11	10	9	8
RXFFOVF	RXFFOVFCLR	RXFIFORESET	RXFFST				
R-0h	W-0h	R/W-1h	R-0h				
7	6	5	4	3	2	1	0
RXFFINT	RXFFINTCLR	RXFFIENA	RXFFIL				
R-0h	W-0h	R/W-0h	R/W-1Fh				

Table 24-18. SPIFFRX Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RXFFOVF	R	0h	Receive FIFO Overflow Flag Reset type: SYSRSn 0h (R/W) = Receive FIFO has not overflowed. This is a read-only bit. 1h (R/W) = Receive FIFO has overflowed, read-only bit. More than 16 words have been received in to the FIFO, and the first received word is lost.
14	RXFFOVFCLR	W	0h	Receive FIFO Overflow Clear Reset type: SYSRSn 0h (R/W) = Write 0 does not affect RXFFOVF flag bit, Bit reads back a zero. 1h (R/W) = Write 1 to clear SPIFFRX[RXFFOVF].
13	RXFIFORESET	R/W	1h	Receive FIFO Reset Reset type: SYSRSn 0h (R/W) = Write 0 to reset the FIFO pointer to zero, and hold in reset. 1h (R/W) = Re-enable receive FIFO operation.
12-8	RXFFST	R	0h	Receive FIFO Status Reset type: SYSRSn 0h (R/W) = Receive FIFO is empty. 1h (R/W) = Receive FIFO has 1 word. 2h (R/W) = Receive FIFO has 2 words. 10h (R/W) = Receive FIFO has 16 words, which is the maximum. 1Fh (R/W) = Reserved.
7	RXFFINT	R	0h	Receive FIFO Interrupt Flag Reset type: SYSRSn 0h (R/W) = RXFIFO interrupt has not occurred. This is a read-only bit. 1h (R/W) = RXFIFO interrupt has occurred. This is a read-only bit.
6	RXFFINTCLR	W	0h	Receive FIFO Interrupt Clear Reset type: SYSRSn 0h (R/W) = Write 0 has no effect on RXFIFINT flag bit, Bit reads back a zero. 1h (R/W) = Write 1 to clear SPIFFRX[RXFFINT] flag
5	RXFFIENA	R/W	0h	RX FIFO Interrupt Enable Reset type: SYSRSn 0h (R/W) = RX FIFO interrupt based on RXFFIL match (greater than or equal to) will be disabled. 1h (R/W) = RX FIFO interrupt based on RXFFIL match (greater than or equal to) will be enabled.

Table 24-18. SPIFFRX Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-0	RXFFIL	R/W	1Fh	Receive FIFO Interrupt Level Bits Receive FIFO generates an interrupt when the FIFO status bits (RXFFST4-0) are greater than or equal to the FIFO level bits (RXFFIL4-0). The default value of these bits after reset is 11111. This avoids frequent interrupts after reset, as the receive FIFO will be empty most of the time. Reset type: SYSRSn 0h (R/W) = A RX FIFO interrupt request is generated when there is 0 or more words in the RX buffer. 1h (R/W) = A RX FIFO interrupt request is generated when there are 1 or more words in the RX buffer. 2h (R/W) = A RX FIFO interrupt request is generated when there are 2 or more words in the RX buffer. 10h (R/W) = A RX FIFO interrupt request is generated when there are 16 words in the RX buffer. 1Fh (R/W) = Reserved.

24.6.2.11 SPIFFCT Register (Offset = Ch) [Reset = 0h]

SPIFFCT is shown in Figure 24-24 and described in Table 24-19.

Return to the Summary Table.

SPIFFCT controls the FIFO transmit delay bits.

Figure 24-24. SPIFFCT Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
TXDLY							
R/W-0h							

Table 24-19. SPIFFCT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-0	TXDLY	R/W	0h	FIFO Transmit Delay Bits These bits define the delay between every transfer from FIFO transmit buffer to transmit shift register. The delay is defined in number SPI serial clock cycles. The 8-bit register could define a minimum delay of 0 serial clock cycles and a maximum of 255 serial clock cycles. In FIFO mode, the buffer (TXBUF) between the shift register and the FIFO should be filled only after the shift register has completed shifting of the last bit. This is required to pass on the delay between transfers to the data stream. In the FIFO mode TXBUF should not be treated as one additional level of buffer. Reset type: SYSRSn 0h (R/W) = The next word in the TX FIFO buffer is transferred to SPITXBUF immediately upon completion of transmission of the previous word. 1h (R/W) = The next word in the TX FIFO buffer is transferred to SPITXBUF1 serial clock cycle after completion of transmission of the previous word. 2h (R/W) = The next word in the TX FIFO buffer is transferred to SPITXBUF 2 serial clock cycles after completion of transmission of the previous word. FFh (R/W) = The next word in the TX FIFO buffer is transferred to SPITXBUF 255 serial clock cycles after completion of transmission of the previous word.

24.6.2.12 SPIPRI Register (Offset = Fh) [Reset = 0h]

SPIPRI is shown in Figure 24-25 and described in Table 24-20.

Return to the Summary Table.

SPIPRI controls auxillary functions for the SPI including emulation control, SPISTE inversion, and 3-wire control.

Figure 24-25. SPIPRI Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	RESERVED	SOFT	FREE	RESERVED		STEINV	TRIWIRE
R-0h	R/W-0h	R/W-0h	R/W-0h	R-0h		R/W-0h	R/W-0h

Table 24-20. SPIPRI Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R	0h	Reserved
6	RESERVED	R/W	0h	Reserved
5	SOFT	R/W	0h	Emulation Soft Run This bit only has an effect when the FREE bit is 0. Reset type: SYSRSn 0h (R/W) = Transmission stops midway in the bit stream while TSUSPEND is asserted. Once TSUSPEND is deasserted without a system reset, the remainder of the bits pending in the DATBUF are shifted. Example: If SPIDAT has shifted 3 out of 8 bits, the communication freezes right there. However, if TSUSPEND is later deasserted without resetting the SPI, SPI starts transmitting from where it had stopped (fourth bit in this case) and will transmit 8 bits from that point. 1h (R/W) = If the emulation suspend occurs before the start of a transmission, (that is, before the first SPICLK pulse) then the transmission will not occur. If the emulation suspend occurs after the start of a transmission, then the data will be shifted out to completion. When the start of transmission occurs is dependent on the baud rate used. Standard SPI mode: Stop after transmitting the words in the shift register and buffer. That is, after TXBUF and SPIDAT are empty. In FIFO mode: Stop after transmitting the words in the shift register and buffer. That is, after TX FIFO and SPIDAT are empty.
4	FREE	R/W	0h	Emulation Free Run These bits determine what occurs when an emulation suspend occurs (for example, when the debugger hits a breakpoint). The peripheral can continue whatever it is doing (free-run mode) or, if in stop mode, it can either stop immediately or stop when the current operation (the current receive/transmit sequence) is complete. Reset type: SYSRSn 0h (R/W) = Emulation mode is selected by the SOFT bit 1h (R/W) = Free run, continue SPI operation regardless of suspend or when the suspend occurred.
3-2	RESERVED	R	0h	Reserved
1	STEINV	R/W	0h	SPISTEn Inversion Bit On devices with 2 SPI modules, inverting the SPISTE signal on one of the modules allows the device to receive left and right- channel digital audio data. This bit is only applicable to slave mode. Writing to this bit while configured as master (MASTER_SLAVE = 1) has no effect Reset type: SYSRSn 0h (R/W) = SPISTEn is active low (normal) 1h (R/W) = SPISTE is active high (inverted)

Table 24-20. SPIPRI Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	TRIWIRE	R/W	0h	SPI 3-wire Mode Enable Reset type: SYSRSn 0h (R/W) = Normal 4-wire SPI mode. 1h (R/W) = 3-wire SPI mode enabled. The unused pin becomes a GPIO pin. In master mode, the SPISIMO pin becomes the SPIMOMI (master receive and transmit) pin and SPISOMI is free for non-SPI use. In slave mode, the SPISOMI pin becomes the SPISISO (slave receive and transmit) pin and SPISIMO is free for non-SPI use.

24.6.3 SPI Registers to Driverlib Functions

Table 24-21. SPI Registers to Driverlib Functions

File	Driverlib Function
SPICCR	
spi.c	SPI_setConfig
spi.c	SPI_clearInterruptStatus
spi.h	SPI_enableModule
spi.h	SPI_disableModule
spi.h	SPI_setcharLength
spi.h	SPI_enableLoopback
spi.h	SPI_disableLoopback
spi.h	SPI_enableHighSpeedMode
spi.h	SPI_disableHighSpeedMode
SPICTL	
spi.c	SPI_setConfig
spi.c	SPI_enableInterrupt
spi.c	SPI_disableInterrupt
spi.h	SPI_enableTalk
spi.h	SPI_disableTalk
SPISTS	
spi.c	SPI_getInterruptStatus
spi.c	SPI_clearInterruptStatus
spi.h	SPI_writeDataBlockingNonFIFO
spi.h	SPI_readDataBlockingNonFIFO
SPIBRR	
spi.c	SPI_setConfig
spi.c	SPI_setBaudRate
SPIRXEMU	
spi.h	SPI_readRxEmulationBuffer
SPIRXBUF	
spi.h	SPI_readDataNonBlocking
spi.h	SPI_readDataBlockingFIFO
spi.h	SPI_readDataBlockingNonFIFO
SPITXBUF	
spi.h	SPI_writeDataNonBlocking
spi.h	SPI_writeDataBlockingFIFO
spi.h	SPI_writeDataBlockingNonFIFO
SPIDAT	

Table 24-21. SPI Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
SPIFFTX	
spi.c	SPI_enableInterrupt
spi.c	SPI_disableInterrupt
spi.c	SPI_getInterruptStatus
spi.c	SPI_clearInterruptStatus
spi.h	SPI_enableFIFO
spi.h	SPI_disableFIFO
spi.h	SPI_resetTxFIFO
spi.h	SPI_setFIFOInterruptLevel
spi.h	SPI_getFIFOInterruptLevel
spi.h	SPI_getTxFIFOStatus
spi.h	SPI_isBusy
spi.h	SPI_reset
SPIFFRX	
spi.c	SPI_enableInterrupt
spi.c	SPI_disableInterrupt
spi.c	SPI_getInterruptStatus
spi.c	SPI_clearInterruptStatus
spi.h	SPI_enableFIFO
spi.h	SPI_disableFIFO
spi.h	SPI_resetRxFIFO
spi.h	SPI_setFIFOInterruptLevel
spi.h	SPI_getFIFOInterruptLevel
spi.h	SPI_getRxFIFOStatus
SPIFFCT	
spi.h	SPI_setTxFifoTransmitDelay
SPIPRI	
spi.h	SPI_enableTriWire
spi.h	SPI_disableTriWire
spi.h	SPI_setSTESignalPolarity
spi.h	SPI_setEmulationMode

Chapter 25

Serial Communications Interface (SCI)

This chapter describes the features and operation of the serial communication interface (SCI) module. SCI is a two-wire asynchronous serial port, commonly known as a UART. The SCI modules support digital communications between the CPU and other asynchronous peripherals that use the standard non-return-to-zero (NRZ) format. The SCI receiver and transmitter each have a 16-level deep FIFO for reducing servicing overhead, and each has its own separate enable and interrupt bits. Both can be operated independently for half-duplex communication, or simultaneously for full-duplex communication.

To specify data integrity, the SCI checks received data for break detection, parity, overrun, and framing errors. The bit rate is programmable to different speeds through a 16-bit baud-select register.

25.1 Introduction

The SCI interfaces are shown in Figure 25-1.

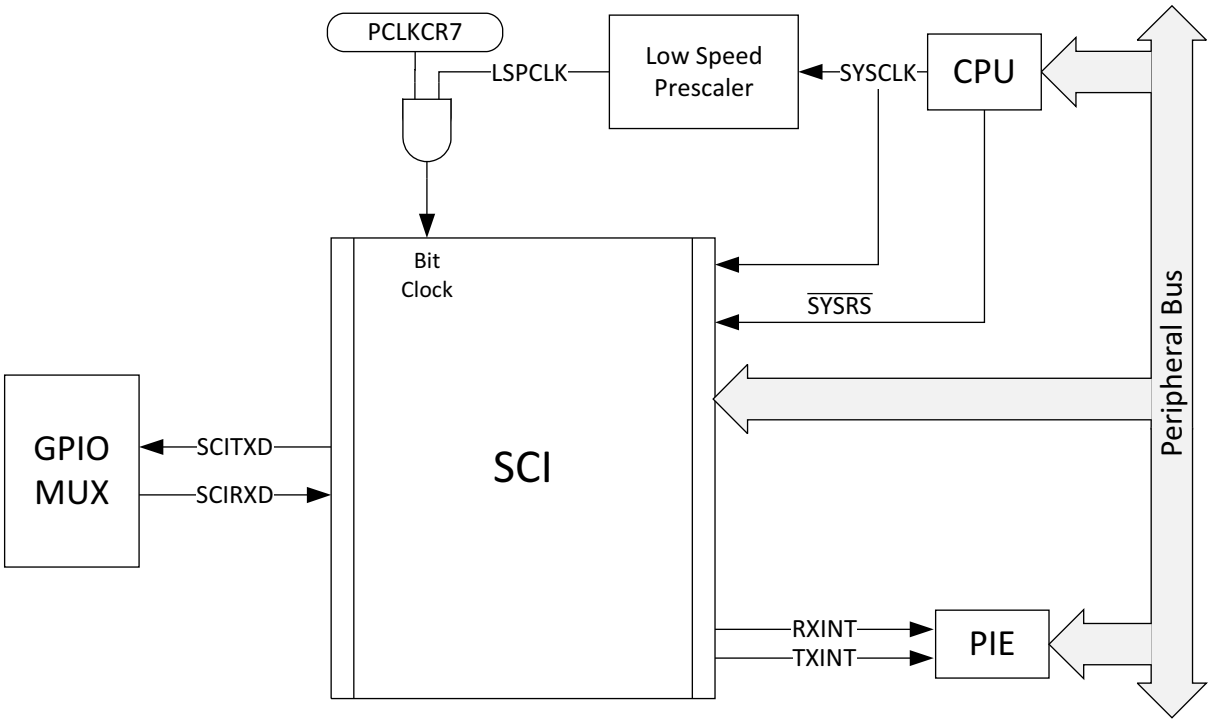


Figure 25-1. SCI CPU Interface

25.1.1 SCI Related Collateral

Foundational Materials

- C2000 Academy - Communications
- One Minute RS-485 Introduction (Video)
- RS-232, RS-422, RS-485: What Are the Differences? (Video)

Getting Started Materials

- [FAQ] My C2000 SCI is not Transmitting and/or Receiving data correctly, how do I fix this?

25.1.2 Features

Features of the SCI module include:

- Two external pins:
  - SCITXD: SCI transmit-output pin
  - SCIRXD: SCI receive-input pin

  Both pins can be used as GPIO if not used for SCI.
- Baud rate programmable to 64K different rates
- Data-word format
  - One start bit
  - Data-word length programmable from one to eight bits
  - Optional even/odd/no parity bit
  - One or two stop bits
  - An extra bit to distinguish addresses from data (address bit mode only)

- Four error-detection flags: parity, overrun, framing, and break detection
- Two wake-up multiprocessor modes: idle-line and address bit
- Half- or full-duplex operation
- Double-buffered receive and transmit functions
- Transmitter and receiver operations can be accomplished through interrupt-driven or polled algorithms with status flags
- Separate enable bits for transmitter and receiver interrupts (except BRKDT)
- NRZ (non-return-to-zero) format

Enhanced features include:

- Auto-baud-detect hardware logic
- 16-level transmit/receive FIFO

25.1.3 Block Diagram

Figure 25-2 shows the SCI module block diagram. The SCI port operation is configured and controlled by the registers listed in Section 25.15.

25.2 Architecture

The major elements used in full-duplex operation are shown in Figure 25-2 and include:

- A transmitter (TX) and its major registers (upper half of Figure 25-2)
  - SCITXBUF — transmitter data buffer register. Contains data (loaded by the CPU) to be transmitted
  - TXSHF register — transmitter shift register. Accepts data from register SCITXBUF and shifts data onto the SCITXD pin, one bit at a time
- A receiver (RX) and its major registers (lower half of Figure 25-2)
  - RXSHF register — receiver shift register. Shifts data in from SCIRXD pin, one bit at a time
  - SCIRXBUF — receiver data buffer register. Contains data to be read by the CPU. Data from a remote processor is loaded into register RXSHF and then into registers SCIRXBUF and SCIRXEMU
- A programmable baud generator
- Control and status registers

The SCI receiver and transmitter can operate either independently or simultaneously.

25.3 SCI Module Signal Summary

A summarized description of each SCI signal name is shown in Table 25-1.

Table 25-1. SCI Module Signal Summary

Signal Name	Description
External signals	
SCIRXD	SCI Asynchronous Serial Port receive data
SCITXD	SCI Asynchronous Serial Port transmit data
Control	
Baud clock	LSPCLK Prescaled clock
Interrupt signals	
TXINT	Transmit interrupt
RXINT	Receive Interrupt

25.4 Configuring Device Pins

The GPIO mux registers must be configured to connect this peripheral to the device pins. To avoid glitches on the pins, the GPyGMUX bits must be configured first (while keeping the corresponding GPyMUX bits at the default of zero), followed by writing the GPyMUX register to the desired value.

Some IO functionality is defined by GPIO register settings independent of this peripheral. For input signals, the GPIO input qualification should be set to asynchronous mode by setting the appropriate GPxQSELn register bits to 11b. The internal pullups can be configured in the GPyPUD register.

See the *General-Purpose Input/Output (GPIO)* chapter for more details on GPIO mux and settings.

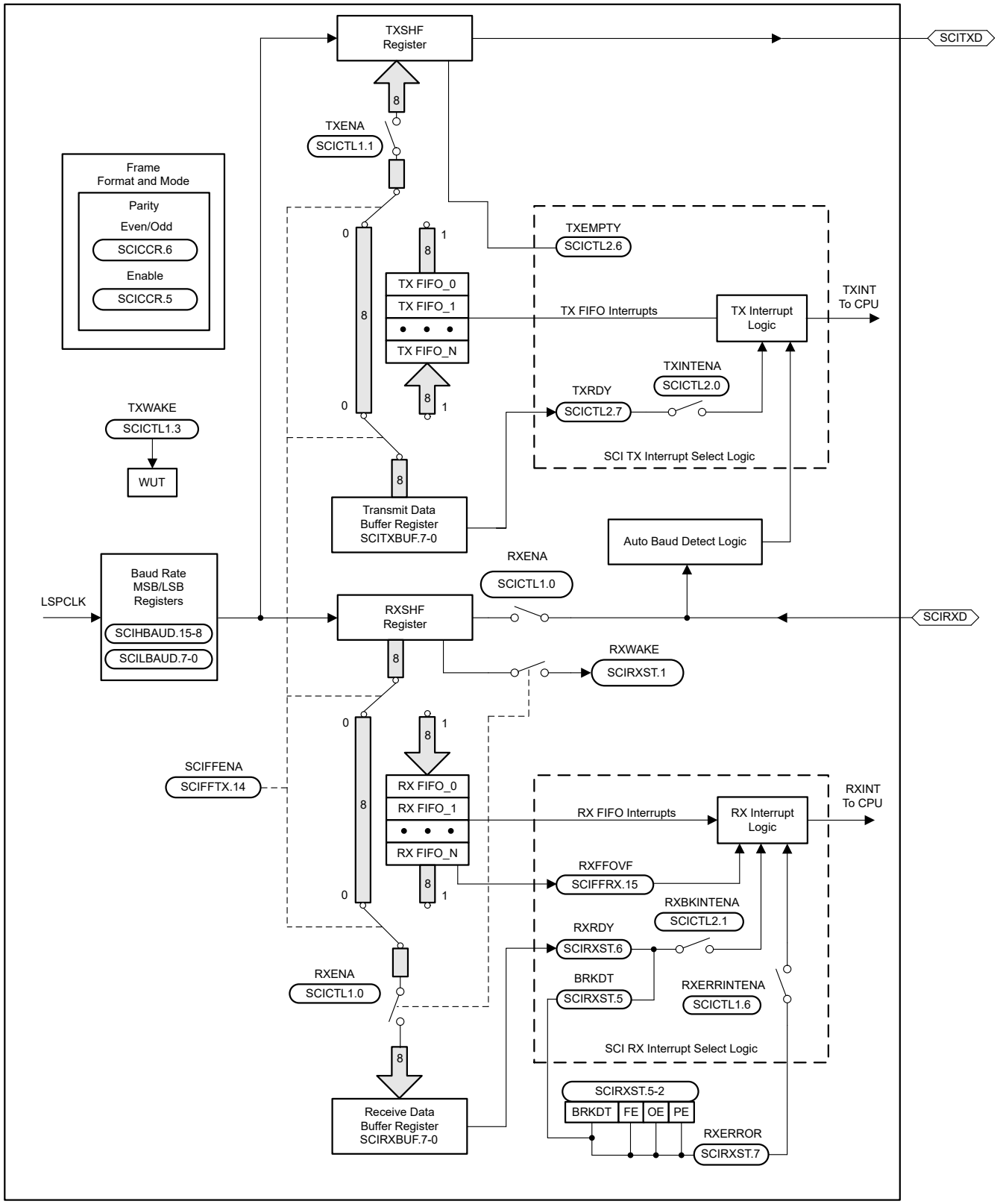


Figure 25-2. Serial Communications Interface (SCI) Module Block Diagram

25.5 Multiprocessor and Asynchronous Communication Modes

The SCI has two multiprocessor protocols, the idle-line multiprocessor mode (see Section 25.8) and the address-bit multiprocessor mode (see Section 25.9). These protocols allow efficient data transfer between multiple processors.

The SCI offers the universal asynchronous receiver/transmitter (UART) communications mode for interfacing with many popular peripherals. The asynchronous mode (see Section 25.10) requires two lines to interface with many standard devices such as terminals and printers that use RS-232-C formats. Data transmission characteristics include:

- One start bit
- One to eight data bits
- An even/odd parity bit or no parity bit
- One or two stop bits

25.6 SCI Programmable Data Format

SCI data, both receive and transmit, is in NRZ (non-return-to-zero) format. The NRZ data format, shown in Figure 25-3, consists of:

- One start bit
- One to eight data bits
- An even/odd parity bit (optional)
- One or two stop bits
- An extra bit to distinguish addresses from data (address-bit mode only)

The basic unit of data is called a character and is one to eight bits in length. Each character of data is formatted with a start bit, one or two stop bits, and optional parity and address bits. A character of data with its formatting information is called a frame and is shown in Figure 25-3.

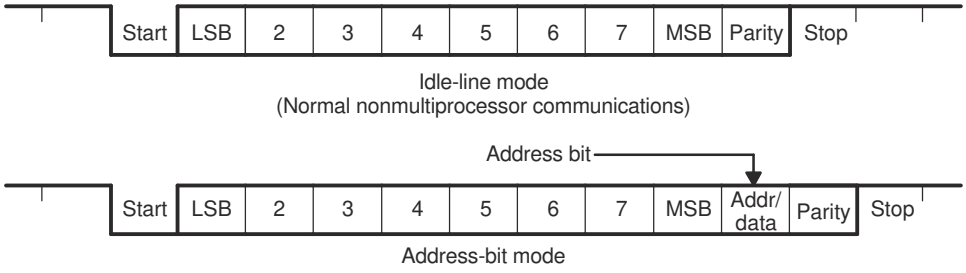


Figure 25-3. Typical SCI Data Frame Formats

To program the data format, use the SCICCR register. The bits used to program the data format are shown in Table 25-2.

Table 25-2. Programming the Data Format Using SCICCR

Bit(s)	Bit Name	Designation	Functions
2-0	SCICHAR	SCICCR.2:0	Select the character (data) length (one to eight bits).
5	PARITYENA (ENABLE)	SCICCR.5	Enables the parity function if set to 1, or disables the parity function if cleared to 0.
6	PARITY (EVEN/ODD)	SCICCR.6	If parity is enabled, selects odd parity if cleared to 0; even parity if set to 1.
7	STOPBITS	SCICCR.7	Determines the number of stop bits transmitted—one stop bit if cleared to 0 or two stop bits if set to 1.

25.7 SCI Multiprocessor Communication

The multiprocessor communication format allows one processor to efficiently send blocks of data to other processors on the same serial link. On one serial line, there should be only one transfer at a time. In other words, there can be only one talker on a serial line at a time.

Address Byte

The first byte of a block of information that the talker sends contains an address byte that is read by all listeners. Only listeners with the correct address can be interrupted by the data bytes that follow the address byte. The listeners with an incorrect address remain uninterrupted until the next address byte.

Sleep Bit

All processors on the serial link set the SCI SLEEP bit (bit 2 of SCICTL1) to 1 so that they are interrupted only when the address byte is detected. When a processor reads a block address that corresponds to the CPU device address as set by your application software, your program must clear the SLEEP bit to enable the SCI to generate an interrupt on receipt of each data byte.

Although the receiver still operates when the SLEEP bit is 1, it does not set RXRDY, RXINT, or any of the receiver error status bits to 1 unless the address byte is detected and the address bit in the received frame is a 1 (applicable to address-bit mode). The SCI does not alter the SLEEP bit; your software must alter the SLEEP bit.

25.7.1 Recognizing the Address Byte

A processor recognizes an address byte differently, depending on the multiprocessor mode used. For example:

- The idle-line mode (Section 25.8) leaves a quiet space before the address byte. This mode does not have an extra address/data bit and is more efficient than the address-bit mode for handling blocks that contain more than ten bytes of data. The idle-line mode should be used for typical non-multiprocessor SCI communication.
- The address-bit mode (Section 25.9) adds an extra bit (that is, an address bit) into every byte to distinguish addresses from data. This mode is more efficient in handling many small blocks of data because, unlike the idle mode, it does not have to wait between blocks of data. However, at a high transmit speed, the program is not fast enough to avoid a 10-bit idle in the transmission stream.

25.7.2 Controlling the SCI TX and RX Features

The multiprocessor mode is software selectable via the ADDR/IDLE MODE bit (SCICCR, bit 3). Both modes use the TXWAKE flag bit (SCICTL1, bit 3), RXWAKE flag bit (SCIRXST, bit1), and the SLEEP flag bit (SCICTL1, bit 2) to control the SCI transmitter and receiver features of these modes.

25.7.3 Receipt Sequence

In both multiprocessor modes, the receive sequence is as follows:

1. At the receipt of an address block, the SCI port wakes up and requests an interrupt (bit number 1 RX/BK INT ENA-of SCICTL2 must be enabled to request an interrupt). It reads the first frame of the block, which contains the destination address.
2. A software routine is entered through the interrupt and checks the incoming address. This address byte is checked against its device address byte stored in memory.
3. If the check shows that the block is addressed to the device CPU, the CPU clears the SLEEP bit and reads the rest of the block. If not, the software routine exits with the SLEEP bit still set, and does not receive interrupts until the next block start.

25.8 Idle-Line Multiprocessor Mode

In the idle-line multiprocessor protocol (ADDR/IDLE MODE bit=0), blocks are separated by having a longer idle time between the blocks than between frames in the blocks. An idle time of ten or more high-level bits after a frame indicates the start of a new block. The time of a single bit is calculated directly from the baud value (bits per second). The idle-line multiprocessor communication format is shown in Figure 25-4 (ADDR/IDLE MODE bit is bit 3 of SCICCR).

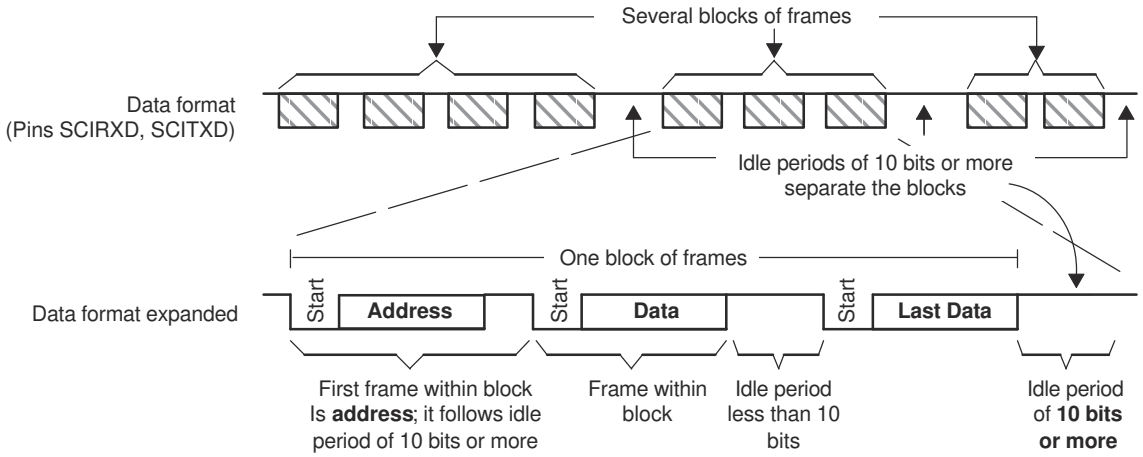


Figure 25-4. Idle-Line Multiprocessor Communication Format

25.8.1 Idle-Line Mode Steps

The steps followed by the idle-line mode:

1. SCI wakes up after receipt of the block-start signal.
2. The processor recognizes the next SCI interrupt.
3. The interrupt service routine compares the received address (sent by a remote transmitter) to its own.
4. If the CPU is being addressed, the service routine clears the SLEEP bit and receives the rest of the data block.
5. If the CPU is not being addressed, the SLEEP bit remains set. This lets the CPU continue to execute its main program without being interrupted by the SCI port until the next detection of a block start.

25.8.2 Block Start Signal

There are two ways to send a block-start signal:

- **Method 1:** Deliberately leave an idle time of ten bits or more by delaying the time between the transmission of the last frame of data in the previous block and the transmission of the address frame of the new block.
- **Method 2:** The SCI port first sets the TXWAKE bit (SCICTL1, bit 3) to 1 before writing to the SCITXBUF register. This sends an idle time of exactly 11 bits. In this method, the serial communications line is not idle any longer than necessary. (A don't care byte has to be written to SCITXBUF after setting TXWAKE, and before sending the address, so as to transmit the idle time.)

25.8.3 Wake-UP Temporary (WUT) Flag

Associated with the TXWAKE bit is the wake-up temporary (WUT) flag. WUT is an internal flag, double-buffered with TXWAKE. When TXSHF is loaded from SCITXBUF, WUT is loaded from TXWAKE, and the TXWAKE bit is cleared to 0. This arrangement is shown in Figure 25-5.

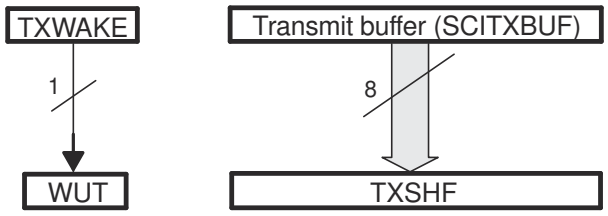


Figure 25-5. Double-Buffered WUT and TXSHF

25.8.3.1 Sending a Block Start Signal

To send out a block-start signal of exactly one frame time during a sequence of block transmissions:

1. Write a 1 to the TXWAKE bit.
2. Write a data word (content not important: a don't care) to the SCITXBUF register (transmit data buffer) to send a block-start signal. (The first data word written is suppressed while the block-start signal is sent out and ignored after that.) When the TXSHF (transmit shift register) is free again, SCITXBUF contents are shifted to TXSHF, the TXWAKE value is shifted to WUT, and then TXWAKE is cleared.

   Because TXWAKE was set to a 1, the start, data, and parity bits are replaced by an idle period of 11 bits transmitted following the last stop bit of the previous frame.
3. Write a new address value to SCITXBUF.

   A don't-care data word must first be written to register SCITXBUF so that the TXWAKE bit value can be shifted to WUT. After the don't-care data word is shifted to the TXSHF register, the SCITXBUF (and TXWAKE, if necessary) can be written to again because TXSHF and WUT are both double-buffered.

25.8.4 Receiver Operation

The receiver operates regardless of the SLEEP bit. However, the receiver neither sets RXRDY nor the error status bits, nor does it request a receive interrupt until an address frame is detected.

25.9 Address-Bit Multiprocessor Mode

In the address-bit protocol (ADDR/IDLE MODE bit=1), frames have an extra bit called an address bit that immediately follows the last data bit. The address bit is set to 1 in the first frame of the block and to 0 in all other frames. The idle period timing is irrelevant (see Figure 25-6).

25.9.1 Sending an Address

The TXWAKE bit value is placed in the address bit. During transmission, when the SCITXBUF register and TXWAKE are loaded into the TXSHF register and WUT respectively, TXWAKE is reset to 0 and WUT becomes the value of the address bit of the current frame. Thus, to send an address:

1. Set the TXWAKE bit to 1 and write the appropriate address value to the SCITXBUF register.

   When this address value is transferred to the TXSHF register and shifted out, its address bit is sent as a 1. This flags the other processors on the serial link to read the address.
2. Write to SCITXBUF and TXWAKE after TXSHF and WUT are loaded. (Can be written to immediately since both TXSHF and WUT are both double-buffered.
3. Leave the TXWAKE bit set to 0 to transmit non-address frames in the block.

Note

As a general rule, the address-bit format is typically used for data frames of 11 bytes or less. This format adds one bit value (1 for an address frame, 0 for a data frame) to all data bytes transmitted. The idle-line format is typically used for data frames of 12 bytes or more.

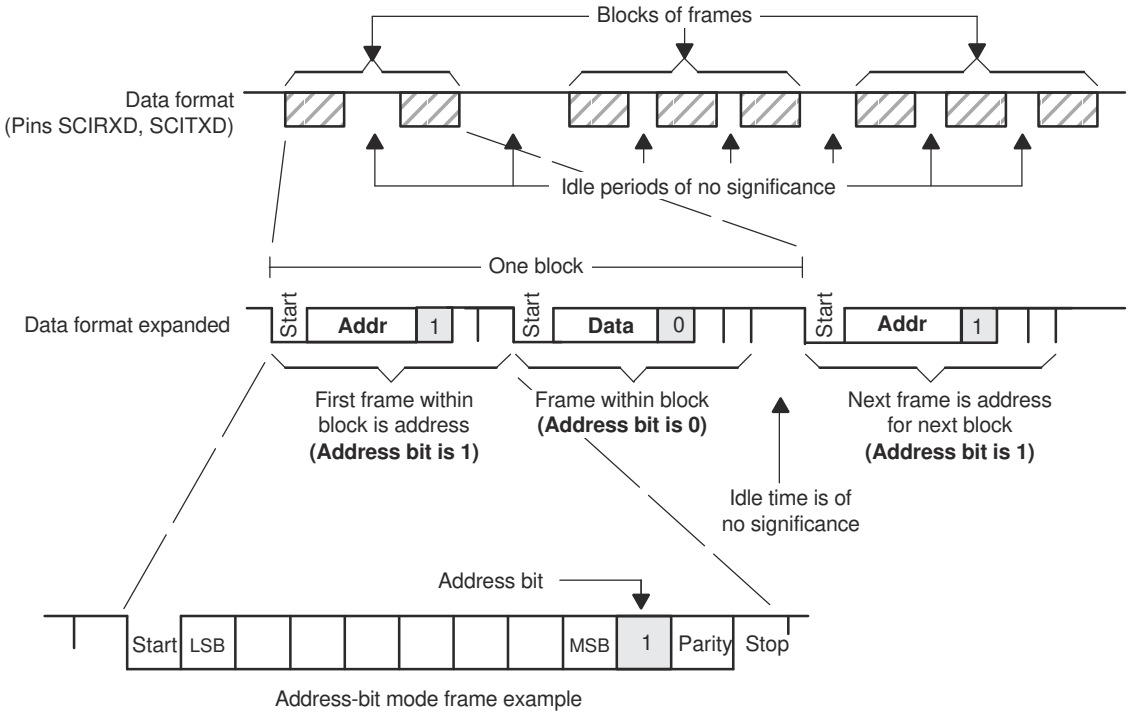


Figure 25-6. Address-Bit Multiprocessor Communication Format

25.10 SCI Communication Format

The SCI asynchronous communication format uses either single line (one way) or two line (two way) communications. In this mode, the frame consists of a start bit, one to eight data bits, an optional even/odd parity bit, and one or two stop bits (shown in Figure 25-7). There are eight SCICLK periods per data bit.

The receiver begins operation on receipt of a valid start bit. A valid start bit is identified by four consecutive internal SCICLK periods of zero bits as shown in Figure 25-7. If any bit is not zero, then the processor starts over and begins looking for another start bit.

For the bits following the start bit, the processor determines the bit value by making three samples in the middle of the bits. These samples occur on the fourth, fifth, and sixth SCICLK periods, and bit-value determination is on a majority (two out of three) basis. Figure 25-7 illustrates the asynchronous communication format for this with a start bit showing where a majority vote is taken.

Since the receiver synchronizes itself to frames, the external transmitting and receiving devices do not have to use a synchronized serial clock. The clock can be generated locally.

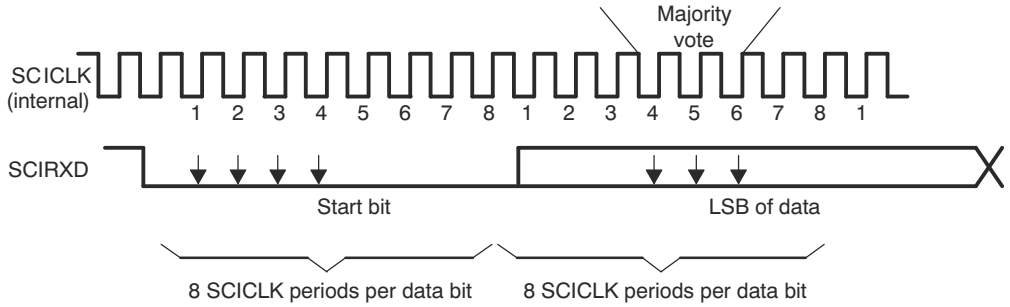


Figure 25-7. SCI Asynchronous Communications Format

25.10.1 Receiver Signals in Communication Modes

Figure 25-8 illustrates an example of receiver signal timing that assumes the following conditions:

- Address-bit wake-up mode (address bit does not appear in idle-line mode)
- Six bits per character

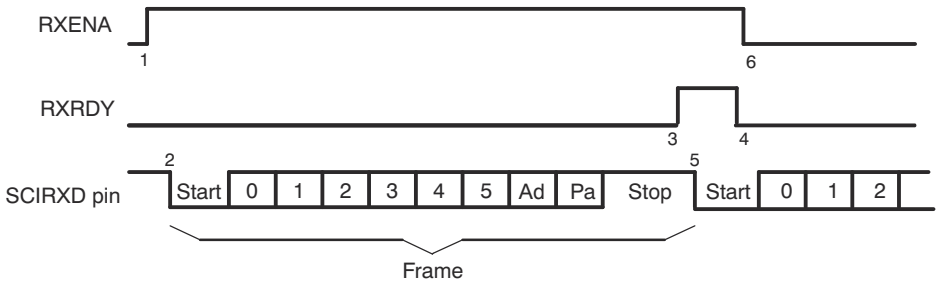


Figure 25-8. SCI RX Signals in Communication Modes

Notes:

1. Flag bit RXENA (SCICTL1, bit 0) goes high to enable the receiver.
2. Data arrives on the SCIRXD pin, start bit detected.
3. Data is shifted from RXSHF to the receiver buffer register (SCIRXBUF); an interrupt is requested. Flag bit RXRDY (SCIRXST, bit 6) goes high to signal that a new character has been received.
4. The program reads SCIRXBUF; flag RXRDY is automatically cleared.
5. The next byte of data arrives on the SCIRXD pin; the start bit is detected, then cleared.
6. Bit RXENA is brought low to disable the receiver. Data continues to be assembled in RXSHF but is not transferred to the receiver buffer register.

25.10.2 Transmitter Signals in Communication Modes

Figure 25-9 illustrates an example of transmitter signal timing that assumes the following conditions:

- Address-bit wake-up mode (address bit does not appear in idle-line mode)
- Three bits per character

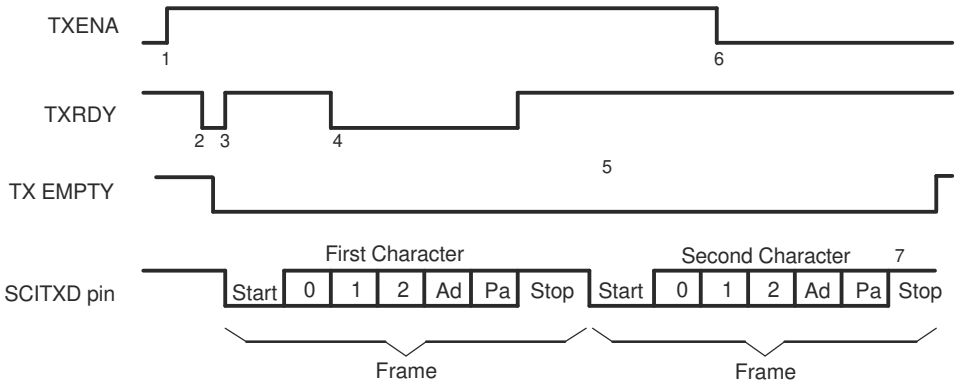


Figure 25-9. SCI TX Signals in Communications Mode

Notes:

1. Bit TXENA (SCICTL1, bit 1) goes high, enabling the transmitter to send data.
2. SCITXBUF is written to; thus, (1) the transmitter is no longer empty, and (2) TXRDY goes low.
3. The SCI transfers data to the shift register (TXSHF). The transmitter is ready for a second character (TXRDY goes high), and it requests an interrupt (to enable an interrupt, bit TX INT ENA — SCICTL2, bit 0 — must be set).
4. The program writes a second character to SCITXBUF after TXRDY goes high (item 3). (TXRDY goes low again after the second character is written to SCITXBUF.)
5. Transmission of the first character is complete. Transfer of the second character to shift register TXSHF begins.
6. Bit TXENA goes low to disable the transmitter; the SCI finishes transmitting the current character.
7. Transmission of the second character is complete; transmitter is empty and ready for new character.

25.11 SCI Port Interrupts

The SCI receiver and transmitter can be interrupt controlled. The SCICTL2 register has one flag bit (TXRDY) that indicates active interrupt conditions, and the SCIRXST register has two interrupt flag bits (RXRDY and BRKDT), plus the RX ERROR interrupt flag that is a logical-OR of the FE, OE, BRKDT, and PE conditions. The transmitter and receiver have separate interrupt-enable bits. When not enabled, the interrupts are not asserted; however, the condition flags remain active, reflecting transmission and receipt status.

The SCI has independent peripheral interrupt vectors for the receiver and transmitter. Peripheral interrupt requests can be either high priority or low priority. This is indicated by the priority bits that are output from the peripheral to the PIE controller. When both RX and TX interrupt requests are made at the same priority level, the receiver always has higher priority than the transmitter, reducing the possibility of receiver overrun.

The operation of peripheral interrupts is described in the Peripheral Interrupts section of the *System Control and Interrupts* chapter.

- If the RX/BK INT ENA bit (SCICTL2, bit 1) is set, the receiver peripheral interrupt request is asserted when one of the following events occurs:
  - The SCI receives a complete frame and transfers the data in the RXSHF register to the SCIRXBUF register. This action sets the RXRDY flag (SCIRXST, bit 6) and initiates an interrupt.
  - A break detect condition occurs (the SCIRXD is low for 9.625 bit periods following a missing stop bit). This action sets the BRKDT flag bit (SCIRXST, bit 5) and initiates an interrupt.
- If the TX INT ENA bit (SCICTL2.0) is set, the transmitter peripheral interrupt request is asserted whenever the data in the SCITXBUF register is transferred to the TXSHF register, indicating that the CPU can write to SCITXBUF; this action sets the TXRDY flag bit (SCICTL2, bit 7) and initiates an interrupt.

Note

Interrupt generation due to the RXRDY and BRKDT bits is controlled by the RX/BK INT ENA bit (SCICTL2, bit 1). Interrupt generation due to the RX ERROR bit is controlled by the RX ERR INT ENA bit (SCICTL1, bit 6).

25.12 SCI Baud Rate Calculations

The internally generated serial clock is determined by the low-speed peripheral clock LSPCLK) and the baud-select registers. The SCI uses the 16-bit value of the baud-select registers to select one of the 64K different serial clock rates possible for a given LSPCLK.

See the bit descriptions in the baud-select registers, for the formula to use when calculating the SCI asynchronous baud. Table 25-3 shows the baud-select values for common SCI bit rates. LSPCLK/16 is the maximum baud rate. For example, if LSPCLK is 100 MHz, then the maximum baud rate is 6.25 Mbps.

Table 25-3. Asynchronous Baud Register Values for Common SCI Bit Rates

	LSPCLK Clock Frequency, 100 MHz		
Ideal Baud	**BRR**	**Actual Baud**	**% Error**
2400	5207 (1457h)	2400	0
4800	2603 (A2Bh)	4800	0
9600	1301 (515h)	9601	0.01
19200	650 (28Ah)	19201	0.01
38400	324 (144h)	38462	0.16

25.13 SCI Enhanced Features

The C28x SCI features autobaud detection and transmit/receive FIFO. The following section explains the FIFO operation.

25.13.1 SCI FIFO Description

The following steps explain the FIFO features and help with programming the SCI with FIFOs.

1. *Reset.* At reset the SCI powers up in standard SCI mode and the FIFO function is disabled. The FIFO registers SCIFFTX, SCIFFRX, and SCIFFCT remain inactive.
2. *Standard SCI.* The standard SCI modes will work normally with TXINT/RXINT interrupts as the interrupt source for the module.
3. *FIFO enable.* FIFO mode is enabled by setting the SCIFFEN bit in the SCIFFTX register. SCIRST can reset the FIFO mode at any stage of its operation.
4. *Active registers.* All the SCI registers and SCI FIFO registers (SCIFFTX, SCIFFRX, and SCIFFCT) are active.
5. *Interrupts.* FIFO mode has two interrupts; one for transmit FIFO, TXINT and one for receive FIFO, RXINT. RXINT is the common interrupt for SCI FIFO receive, receive error, and receive FIFO overflow conditions. The TXINT of the standard SCI will be disabled and this interrupt will service as SCI transmit FIFO interrupt.
6. *Buffers.* Transmit and receive buffers are supplemented with two 16-level FIFOs. The transmit FIFO registers are 8 bits wide and receive FIFO registers are 10 bits wide. The one-word transmit buffer (SCITXBUF) of the standard SCI functions as a transition buffer before the transmit FIFO and shift register. SCITXBUF is loaded into either the FIFO (when FIFO is enabled) or TXSHF (when FIFO is disabled). When FIFO is enabled, SCITXBUF loads into the FIFO only after the last bit of the shift register is shifted out, so SCITXBUF should not be treated as an additional level of buffer. With the FIFO enabled, TXSHF is directly loaded from the FIFO (not TXBUF) after an optional delay value (SCIFFCT). When FIFO mode is enabled for SCI, characters written to SCITXBUF are queued in to SCI-TXFIFO and the characters received in SCI-RXFIFO can be read using SCIRXBUF.
7. *Delayed transfer.* The rate at which words in the FIFO are transferred to the transmit shift register is programmable. The SCIFFCT register bits (7−0) FFTXDLY7−FFTXDLY0 define the delay between the word transfer. The delay is defined in the number SCI baud clock cycles. The 8 bit register can define a minimum delay of 0 baud clock cycles and a maximum of 256-baud clock cycles. With zero delay, the SCI module can transmit data in continuous mode with the FIFO words shifting out back to back. With the 256 clock delay the SCI module can transmit data in a maximum delayed mode with the FIFO words shifting out with a delay of 256 baud clocks between each words. The programmable delay facilitates communication with slow SCI/UARTs with little CPU intervention.
8. *FIFO status bits.* Both the transmit and receive FIFOs have status bits TXFFST or RXFFST (bits 12−8) that define the number of words available in the FIFOs at any time. The transmit FIFO reset bit TXFIFO and receive reset bit RXFIFO reset the FIFO pointers to zero when these bits are cleared to 0. The FIFOs resumes operation from start once these bits are set to 1.
9. *Programmable interrupt levels.* Both transmit and receive FIFO can generate CPU interrupts. The interrupt trigger is generated whenever the transmit FIFO status bits TXFFST (bits 12−8) match (less than or equal to) the interrupt trigger level bits TXFFIL (bits 4−0). This provides a programmable interrupt trigger for transmit and receive sections of the SCI. Default value for these trigger level bits will be 0x11111 for receive FIFO and 0x00000 for transmit FIFO, respectively.

Figure 25-10 and Table 25-4 explain the operation/configuration of SCI interrupts in nonFIFO/FFO mode.

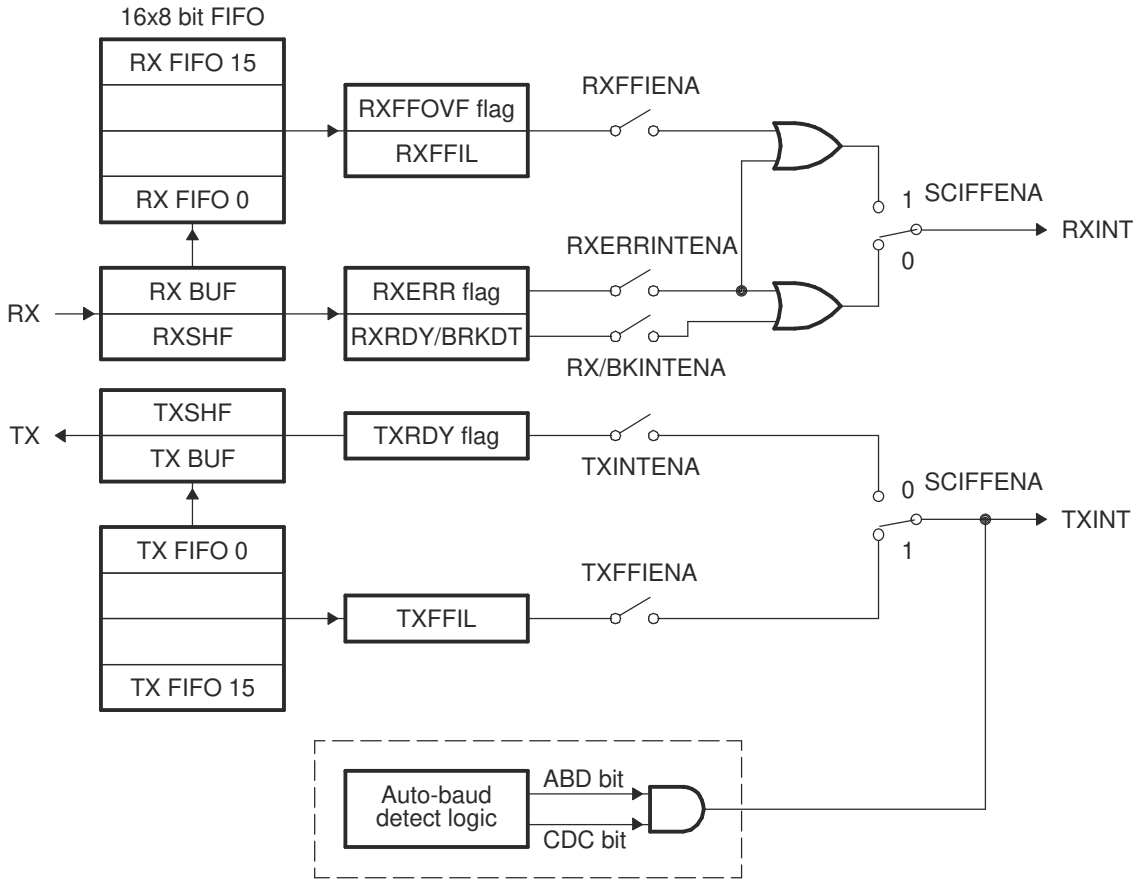


Figure 25-10. SCI FIFO Interrupt Flags and Enable Logic

Table 25-4. SCI Interrupt Flags

FIFO Options[1]	SCI Interrupt Source	Interrupt Flags	Interrupt Enables	FIFO Enable SCIFFENA	Interrupt Line
SCI without FIFO	Receive error	RXERR[2]	RXERRINTENA	0	RXINT
	Receive break	BRKDT	RX/BKINTENA	0	RXINT
	Data receive	RXRDY	RX/BKINTENA	0	RXINT
	Transmit empty	TXRDY	TXINTENA	0	TXINT
SCI with FIFO	Receive error and receive break	RXERR	RXERRINTENA	1	RXINT
	FIFO receive	RXFFIL	RXFFIENA	1	RXINT
	Transmit empty	TXFFIL	TXFFIENA	1	TXINT
Auto-baud	Auto-baud detected	ABD	Don't care	x	TXINT

(1) FIFO mode TXSHF is directly loaded after delay value, TXBUF is not used.
(2) RXERR can be set by BRKDT, FE, OE, PE flags. In FIFO mode, BRKDT interrupt is only through RXERR flag.

25.13.2 SCI Auto-Baud

Most SCI modules do not have an auto-baud detect logic built-in hardware. These SCI modules are integrated with embedded controllers whose clock rates are dependent on PLL reset values. Often embedded controller clocks change after final design. In the enhanced feature set this module supports an autobaud-detect logic in hardware. The following section explains the enabling sequence for autobaud-detect feature.

25.13.3 Autobaud-Detect Sequence

Bits ABD and CDC in SCIFFCT control the autobaud logic. The SCIRST bit should be enabled to make autobaud logic work.

If ABD is set while CDC is 1, which indicates auto-baud alignment, SCI transmit FIFO interrupt will occur (TXINT). After the interrupt service CDC bit has to be cleared by software. If CDC remains set even after interrupt service, there should be no repeat interrupts.

1. Enable autobaud-detect mode for the SCI by setting the CDC bit (bit 13) in SCIFFCT and clearing the ABD bit (Bit 15) by writing a 1 to ABDCLR bit (bit 14).
2. Initialize the baud register to be 1 or less than a baud rate limit of 500 Kbps.
3. Allow SCI to receive either character "A" or "a" from a host at the desired baud rate. If the first character is either "A" or "a", the autobaud-detect hardware will detect the incoming baud rate and set the ABD bit.
4. The auto-detect hardware will update the baud rate register with the equivalent baud value hex. The logic will also generate an interrupt to the CPU.
5. Respond to the interrupt clear ADB bit by writing a 1 to ABD CLR (bit 14) of SCIFFCT register and disable further autobaud locking by clearing CDC bit by writing a 0.
6. Read the receive buffer for character "A" or "a" to empty the buffer and buffer status.
7. If ABD is set while CDC is 1, which indicates autobaud alignment, the SCI transmit FIFO interrupt will occur (TXINT). After the interrupt service CDC bit must be cleared by software.

Note

At higher baud rates, the slew rate of the incoming data bits can be affected by transceiver and connector performance. While normal serial communications may work well, this slew rate may limit reliable autobaud detection at higher baud rates (typically beyond 100k baud) and cause the auto-baudlock feature to fail.

To avoid this, the following is recommended:

- Achieve a baud-lock between the host and C28x SCI boot loader using a lower baud rate.
- The host may then handshake with the loaded C28x application to set the SCI baud rate register to the desired higher baud rate.

25.14 Software

25.14.1 SCI Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/f28003x/examples/sci

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

25.14.1.1 Tune Baud Rate via UART Example

FILE: baud_tune_via_uart.c

This example demonstrates the process of tuning the UART/SCI baud rate of a C2000 device based on the UART input from another device. As UART does not have a clock signal, reliable communication requires baud rates to be reasonably matched. This example addresses cases where a clock mismatch between devices is greater than is acceptable for communications, requiring baud compensation between boards. As reliable communication only requires matching the EFFECTIVE baud rate, it does not matter which of the two boards executes the tuning (the board with the less-accurate clock source does not need to be the one to tune; as long as one of the two devices tunes to the other, then proper communication can be established).

To tune the baud rate of this device, SCI data (of the desired baud rate) must be sent to this device.
The input SCI baud rate must be within the +/- MARGINPERCENT of the TARGETBAUD chosen below.
These two variables are defined below, and should be chosen based on the application requirements. Higher MARGINPERCENT will allow more data to be considered "correct" in noisy conditions, and may decrease accuracy. The TARGETBAUD is what was expected to be the baud rate, but due to clock differences, needs to be tuned for better communication robustness with the other device.

NOTE: Lower baud rates have more granularity in register options, and therefore tuning is more affective at these speeds.

External Connections for Control Card
- SCIA_RX/eCAP1 is on GPIO9, connect to incoming SCI communications
- SCIA_TX is on GPIO8, for observation externally

Watch Variables
- *avgBaud* - Baud rate that was detected and set after tuning

25.14.1.2 SCI FIFO Digital Loop Back

FILE: sci_ex1_loopback.c

This program uses the internal loop back test mode of the peripheral. Other then boot mode pin configuration, no other hardware configuration is required. The pinmux and SCI modules are configured through the sysconfig file.

This test uses the loopback test mode of the SCI module to send characters starting with 0x00 through 0xFF. The test will send a character and then check the receive buffer for a correct match.

Watch Variables
- *loopCount* - Number of characters sent
- *errorCount* - Number of errors detected
- *sendChar* - Character sent
- *receivedChar* - Character received

25.14.1.3 SCI Digital Loop Back with Interrupts

FILE: sci_ex2_loopback_interrupts.c

This test uses the internal loop back test mode of the peripheral. Other then boot mode pin configuration, no other hardware configuration is required. Both interrupts and the SCI FIFOs are used.

A stream of data is sent and then compared to the received stream. The SCI-A sent data looks like this:
00 01
01 02
02 03
....
FE FF
FF 00
etc.

The pattern is repeated forever.

Watch Variables
- *sDataA* - Data being sent
- *rDataA* - Data received
- *rDataPointA* - Keep track of where we are in the data stream. This is used to check the incoming data

25.14.1.4 SCI Echoback

FILE: sci_ex3_echoback.c

This test receives and echo-backs data through the SCI-A port.

A terminal such as 'putty' can be used to view the data from the SCI and to send information to the SCI. Characters received by the SCI port are sent back to the host.

Running the Application Open a COM port with the following settings using a terminal:
- Find correct COM port
- Bits per second = 9600
- Data Bits = 8
- Parity = None
- Stop Bits = 1
- Hardware Control = None

The program will print out a greeting and then ask you to enter a character which it will echo back to the terminal.

Watch Variables
- loopCounter - the number of characters sent

External Connections
Connect the USB cable from Control card J1:A to PC

25.15 SCI Registers

The section describes the Serial Communication Interface module registers.

25.15.1 SCI Base Address Table

Table 25-5. SCI Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
SciaRegs	SCI_REGS	SCIA_BASE	0x0000_7200	YES	-	YES	-	YES
ScibRegs	SCI_REGS	SCIB_BASE	0x0000_7210	YES	-	YES	-	YES

25.15.2 SCI_REGS Registers

Table 25-6 lists the memory-mapped registers for the SCI_REGS registers. All register offset addresses not listed in Table 25-6 should be considered as reserved locations and the register contents should not be modified.

Table 25-6. SCI_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	SCICCR	Communications control register		Go
1h	SCICTL1	Control register 1		Go
2h	SCIHBAUD	Baud rate (high) register		Go
3h	SCILBAUD	Baud rate (low) register		Go
4h	SCICTL2	Control register 2		Go
5h	SCIRXST	Receive status register		Go
6h	SCIRXEMU	Receive emulation buffer register		Go
7h	SCIRXBUF	Receive data buffer		Go
9h	SCITXBUF	Transmit data buffer		Go
Ah	SCIFFTX	FIFO transmit register		Go
Bh	SCIFFRX	FIFO receive register		Go
Ch	SCIFFCT	FIFO control register		Go
Fh	SCIPRI	SCI priority control		Go

Complex bit access types are encoded to fit into small table cells. Table 25-7 shows the codes that are used for access types in this section.

Table 25-7. SCI_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

25.15.2.1 SCICCR Register (Offset = 0h) [Reset = 0h]

SCICCR is shown in Figure 25-11 and described in Table 25-8.

Return to the Summary Table.

SCICCR defines the character format, protocol, and communications mode used by the SCI.

Figure 25-11. SCICCR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
STOPBITS	PARITY	PARITYENA	LOOPBKENA	ADDRIDLE_MODE	SCICHAR		
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h		

Table 25-8. SCICCR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	STOPBITS	R/W	0h	SCI number of stop bits. This bit specifies the number of stop bits transmitted. The receiver checks for only one stop bit. Reset type: SYSRSn 0h (R/W) = One stop bit 1h (R/W) = Two stop bits
6	PARITY	R/W	0h	SCI parity odd/even selection. If the PARITY ENABLE bit (SCICCR, bit 5) is set, PARITY (bit 6) designates odd or even parity (odd or even number of bits with the value of 1 in both transmitted and received characters). Reset type: SYSRSn 0h (R/W) = Odd parity 1h (R/W) = Even parity
5	PARITYENA	R/W	0h	SCI parity enable. This bit enables or disables the parity function. If the SCI is in the addressbit multiprocessor mode (set using bit 3 of this register), the address bit is included in the parity calculation (if parity is enabled). For characters of less than eight bits, the remaining unused bits should be masked out of the parity calculation. Reset type: SYSRSn 0h (R/W) = Parity disabled no parity bit is generated during transmission or is expected during reception 1h (R/W) = Parity is enabled
4	LOOPBKENA	R/W	0h	Loop Back test mode enable. This bit enables the Loop Back test mode where the Tx pin is internally connected to the Rx pin. Reset type: SYSRSn 0h (R/W) = Loop Back test mode disabled 1h (R/W) = Loop Back test mode enabled
3	ADDRIDLE_MODE	R/W	0h	SCI multiprocessor mode control bit. This bit selects one of the multiprocessor protocols.Multiprocessor communication is different from the other communication modes because it uses SLEEP and TXWAKE functions (bits SCICTL1, bit 2 and SCICTL1, bit 3, respectively). The idle-line mode is usually used for normal communications because the address-bit mode adds an extra bit to the frame. The idle-line mode does not add this extra bit and is compatible with RS-232 type communications. Reset type: SYSRSn 0h (R/W) = Idle-line mode protocol selected 1h (R/W) = Address-bit mode protocol selected

Table 25-8. SCICCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-0	SCICHAR	R/W	0h	Character-length control bits 2-0. These bits select the SCI character length from one to eight bits. Characters of less than eight bits are right-justified in SCIRXBUF and SCIRXEMU and are padded with leading zeros in SCIRXBUF. SCITXBUF doesn't need to be padded with leading zeros. Reset type: SYSRSn 0h (R/W) = SCICHAR_LENGTH_1 1h (R/W) = SCICHAR_LENGTH_2 2h (R/W) = SCICHAR_LENGTH_3 3h (R/W) = SCICHAR_LENGTH_4 4h (R/W) = SCICHAR_LENGTH_5 5h (R/W) = SCICHAR_LENGTH_6 6h (R/W) = SCICHAR_LENGTH_7 7h (R/W) = SCICHAR_LENGTH_8

25.15.2.2 SCICTL1 Register (Offset = 1h) [Reset = 0h]

SCICTL1 is shown in Figure 25-12 and described in Table 25-9.

Return to the Summary Table.

SCICTL1 controls the receiver/transmitter enable, TXWAKE and SLEEP functions, and the SCI software reset.

Figure 25-12. SCICTL1 Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	RXERRINTENA	SWRESET	RESERVED	TXWAKE	SLEEP	TXENA	RXENA
R-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 25-9. SCICTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R	0h	Reserved
6	RXERRINTENA	R/W	0h	SCI receive error interrupt enable. Setting this bit enables an interrupt if the RX ERROR bit (SCIRXST, bit 7) becomes set because of errors occurring. Reset type: SYSRSn 0h (R/W) = Receive error interrupt disabled 1h (R/W) = Receive error interrupt enabled
5	SWRESET	R/W	0h	SCI software reset (active low). Writing a 0 to this bit initializes the SCI state machines and operating flags (registers SCICTL2 and SCIRXST) to the reset condition. The SW RESET bit does not affect any of the configuration bits. All affected logic is held in the specified reset state until a 1 is written to SW RESET (the bit values following a reset are shown beneath each register diagram in this section). Thus, after a system reset, re-enable the SCI by writing a 1 to this bit. Clear this bit after a receiver break detect (BRKDT flag, bit SCIRXST, bit 5). SW RESET affects the operating flags of the SCI, but it neither affects the configuration bits nor restores the reset values. Once SW RESET is asserted, the flags are frozen until the bit is deasserted. The affected flags are as follows: Value After SW SCI Flag Register Bit RESET 1 TXRDY SCICTL2, bit 7 1 TX EMPTY SCICTL2, bit 6 0 RXWAKE SCIRXST, bit 1 0 PE SCIRXST, bit 2 0 OE SCIRXST, bit 3 0 FE SCIRXST, bit 4 0 BRKDT SCIRXST, bit 5 0 RXRDY SCIRXST, bit 6 0 RX ERROR SCIRXST, bit 7 Reset type: SYSRSn 0h (R/W) = Writing a 0 to this bit initializes the SCI state machines and operating flags (registers SCICTL2 and SCIRXST) to the reset condition. 1h (R/W) = After a system reset, re-enable the SCI by writing a 1 to this bit.
4	RESERVED	R	0h	Reserved

Table 25-9. SCICTL1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3	TXWAKE	R/W	0h	SCI transmitter wake-up method select. The TXWAKE bit controls selection of the data-transmit feature, depending on which transmit mode (idle-line or address-bit) is specified at the ADDR/IDLE MODE bit (SCICCR, bit 3) Reset type: SYSRSn 0h (R/W) = Transmit feature is not selected. In idle-line mode: write a 1 to TXWAKE, then write data to register SCITXBUF to generate an idle period of 11 data bits In address-bit mode: write a 1 to TXWAKE, then write data to SCITXBUF to set the address bit for that frame to 1 1h (R/W) = Transmit feature selected is dependent on the mode, idle-line or address-bit: TXWAKE is not cleared by the SW RESET bit (SCICTL1, bit 5) it is cleared by a system reset or the transfer of TXWAKE to the WUT flag.
2	SLEEP	R/W	0h	SCI sleep. The TXWAKE bit controls selection of the data-transmit feature, depending on which transmit mode (idle-line or address-bit) is specified at the ADDR/IDLE MODE bit (SCICCR, bit 3). In a multiprocessor configuration, this bit controls the receiver sleep function. Clearing this bit brings the SCI out of the sleep mode. The receiver still operates when the SLEEP bit is set however, operation does not update the receiver buffer ready bit (SCIRXST, bit 6, RXRDY) or the error status bits (SCIRXST, bit 5-2: BRKDT, FE, OE, and PE) unless the address byte is detected. SLEEP is not cleared when the address byte is detected. Reset type: SYSRSn 0h (R/W) = Sleep mode disabled 1h (R/W) = Sleep mode enabled
1	TXENA	R/W	0h	SCI transmitter enable. Data is transmitted through the SCITXD pin only when TXENA is set. If reset, transmission is halted but only after all data previously written to SCITXBUF has been sent. Data written into SCITXBUF when TXENA is disabled will not be transmitted even if the TXENA is enabled later. Reset type: SYSRSn 0h (R/W) = Transmitter disabled 1h (R/W) = Transmitter enabled
0	RXENA	R/W	0h	SCI receiver enable. Data is received on the SCIRXD pin and is sent to the receiver shift register and then the receiver buffers. This bit enables or disables the receiver (transfer to the buffers). Clearing RXENA stops received characters from being transferred to the two receiver buffers and also stops the generation of receiver interrupts. However, the receiver shift register can continue to assemble characters. Thus, if RXENA is set during the reception of a character, the complete character will be transferred into the receiver buffer registers, SCIRXEMU and SCIRXBUF. Reset type: SYSRSn 0h (R/W) = Prevent received characters from transfer into the SCIRXEMU and SCIRXBUF receiver buffers 1h (R/W) = Send received characters to SCIRXEMU and SCIRXBUF

25.15.2.3 SCIHBAUD Register (Offset = 2h) [Reset = 0h]

SCIHBAUD is shown in Figure 25-13 and described in Table 25-10.

Return to the Summary Table.

The values in SCIHBAUD and SCILBAUD specify the baud rate for the SCI.

Figure 25-13. SCIHBAUD Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
BAUD							
R/W-0h							

Table 25-10. SCIHBAUD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-0	BAUD	R/W	0h	SCI 16-bit baud selection Registers SCIHBAUD (MSbyte). The internally-generated serial clock is determined by the low speed peripheral clock (LSPCLK) signal and the two baud-select registers. The SCI uses the 16-bit value of these registers to select one of 64K serial clock rates for the communication modes. BRR = (SCIHBAUD << 8) + (SCILBAUD) The SCI baud rate is calculated using the following equation: SCI Asynchronous Baud = LSPCLK / ((BRR + 1) *8) Alternatively, BRR = LSPCLK / (SCI Asynchronous Baud * 8) - 1 Note that the above formulas are applicable only when 0 < BRR < 65536. If BRR = 0, then SCI Asynchronous Baud = LSPCLK / 16 Where: BRR = the 16-bit value (in decimal) in the baud-select registers Reset type: SYSRSn

25.15.2.4 SCILBAUD Register (Offset = 3h) [Reset = 0h]

SCILBAUD is shown in Figure 25-14 and described in Table 25-11.

Return to the Summary Table.

The values in SCIHBAUD and SCILBAUD specify the baud rate for the SCI.

Figure 25-14. SCILBAUD Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
BAUD							
R/W-0h							

Table 25-11. SCILBAUD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-0	BAUD	R/W	0h	See SCIHBAUD Detailed Description Reset type: SYSRSn

25.15.2.5 SCICTL2 Register (Offset = 4h) [Reset = C0h]

SCICTL2 is shown in Figure 25-15 and described in Table 25-12.

Return to the Summary Table.

SCICTL2 enables the receive-ready, break-detect, and transmit-ready interrupts as well as transmitter-ready and -empty flags.

Figure 25-15. SCICTL2 Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
TXRDY	TXEMPTY	RESERVED				RXBKINTENA	TXINTENA
R-1h	R-1h	R-0h				R/W-0h	R/W-0h

Table 25-12. SCICTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	TXRDY	R	1h	Transmitter buffer register ready flag. When set, this bit indicates that the transmit data buffer register, SCITXBUF, is ready to receive another character. Writing data to the SCITXBUF automatically clears this bit. When set, this flag asserts a transmitter interrupt request if the interrupt-enable bit, TX INT ENA (SCICTL2.0), is also set. TXRDY is set to 1 by enabling the SW RESET bit (SCICTL1.5) or by a system reset. Reset type: SYSRSn 0h (R/W) = SCITXBUF is full 1h (R/W) = SCITXBUF is ready to receive the next character
6	TXEMPTY	R	1h	Transmitter empty flag. This flag's value indicates the contents of the transmitter's buffer register (SCITXBUF) and shift register (TXSHF). An active SW RESET (SCICTL1.5), or a system reset, sets this bit. This bit does not cause an interrupt request. Reset type: SYSRSn 0h (R/W) = Transmitter buffer or shift register or both are loaded with data 1h (R/W) = Transmitter buffer and shift registers are both empty
5-2	RESERVED	R	0h	Reserved
1	RXBKINTENA	R/W	0h	Receiver-buffer/break interrupt enable. This bit controls the interrupt request caused by either the RXRDY flag or the BRKDT flag (bits SCIRXST.6 and .5) being set. However, RX/BK INT ENA does not prevent the setting of these flags. Reset type: SYSRSn 0h (R/W) = Disable RXRDY/BRKDT interrupt 1h (R/W) = Enable RXRDY/BRKDT interrupt

Table 25-12. SCICTL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	TXINTENA	R/W	0h	SCITXBUF-register interrupt enable. This bit controls the interrupt request caused by the setting of TXRDY flag bit (SCICTL2.7). However, it does not prevent the TXRDY flag from being set (which indicates SCITXBUF is ready to receive another character). 0 Disable TXRDY interrupt 1 Enable TXRDY interrupt. In non-FIFO mode, a dummy (or a valid) data has to be written to SCITXBUF for the first transmit interrupt to occur. This is the case when you enable the transmit interrupt for the first time and also when you re-enable (disable and then enable) the transmit interrupt. If TXINTENA is enabled after writing the data to SCITXBUF, it will not generate an interrupt. Reset type: SYSRSn 0h (R/W) = Disable TXRDY interrupt 1h (R/W) = Enable TXRDY interrupt

25.15.2.6 SCIRXST Register (Offset = 5h) [Reset = 0h]

SCIRXST is shown in Figure 25-16 and described in Table 25-13.

Return to the Summary Table.

SCIRXST contains seven bits that are receiver status flags (two of which can generate interrupt requests). Each time a complete character is transferred to the receiver buffers (SCIRXEMU and SCIRXBUF), the status flags are updated.

Figure 25-16. SCIRXST Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RXERROR	RXRDY	BRKDT	FE	OE	PE	RXWAKE	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 25-13. SCIRXST Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7	RXERROR	R	0h	SCI receiver error flag. The RX ERROR flag indicates that one of the error flags in the receiver status register is set. RX ERROR is a logical OR of the break detect, framing error, overrun, and parity error enable flags (bits 5-2: BRKDT, FE, OE, and PE). A 1 on this bit will cause an interrupt if the RX ERR INT ENA bit (SCICTL1.6) is set. This bit can be used for fast error-condition checking during the interrupt service routine. This error flag cannot be cleared directly it is cleared by an active SW RESET or by a system reset. Reset type: SYSRSn 0h (R/W) = No error flags set 1h (R/W) = Error flag(s) set
6	RXRDY	R	0h	SCI receiver-ready flag. When a new character is ready to be read from the SCIRXBUF register, the receiver sets this bit, and a receiver interrupt is generated if the RX/BK INT ENA bit (SCICTL2.1) is a 1. RXRDY is cleared by a reading of the SCIRXBUF register, by an active SW RESET, or by a system reset. Reset type: SYSRSn 0h (R/W) = No new character in SCIRXBUF 1h (R/W) = Character ready to be read from SCIRXBUF
5	BRKDT	R	0h	SCI break-detect flag. The SCI sets this bit when a break condition occurs. A break conditionoccurs when the SCI receiver data line (SCIRXD) remains continuously low for at least 9.625 bits, beginning after a missing first stop bit. The occurrence of a break causes a receiver interrupt to be generated if the RX/BK INT ENA bit is a 1, but it does not cause the receiver buffer to be loaded. A BRKDT interrupt can occur even if the receiver SLEEP bit is set to 1. BRKDT is cleared by an active SW RESET or by a system reset. It is not cleared by receipt of a character after the break is detected. In order to receive more characters, the SCI must be reset by toggling the SW RESET bit or by a system reset. Reset type: SYSRSn 0h (R/W) = No break condition 1h (R/W) = Break condition occurred

Table 25-13. SCIRXST Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	FE	R	0h	SCI framing-error flag. The SCI sets this bit when an expected stop bit is not found. Only the first stop bit is checked. The missing stop bit indicates that synchronization with the start bit has been lost and that the character is incorrectly framed. The FE bit is reset by a clearing of the SW RESET bit or by a system reset. Reset type: SYSRSn 0h (R/W) = No framing error detected 1h (R/W) = Framing error detected
3	OE	R	0h	SCI overrun-error flag. The SCI sets this bit when a character is transferred into registers SCIRXEMU and SCIRXBUF before the previous character is fully read by the CPU or DMAC. The previous character is overwritten and lost. The OE flag bit is reset by an active SW RESET or by a system reset. Reset type: SYSRSn 0h (R/W) = No overrun error detected 1h (R/W) = Overrun error detected
2	PE	R	0h	SCI parity-error flag. This flag bit is set when a character is received with a mismatch between the number of 1s and its parity bit. The address bit is included in the calculation. If parity generation and detection is not enabled, the PE flag is disabled and read as 0. The PE bit is reset by an active SW RESET or a system reset. Reset type: SYSRSn 0h (R/W) = No parity error or parity is disabled 1h (R/W) = Parity error is detected
1	RXWAKE	R	0h	Receiver wake-up-detect flag Reset type: SYSRSn 0h (R/W) = No detection of a receiver wake-up condition 1h (R/W) = A value of 1 in this bit indicates detection of a receiver wake-up condition. In the address-bit multiprocessor mode (SCICCR.3 = 1), RXWAKE reflects the value of the address bit for the character contained in SCIRXBUF. In the idle-line multiprocessor mode, RXWAKE is set if the SCIRXD data line is detected as idle. RXWAKE is a read-only flag, cleared by one of the following: - The transfer of the first byte after the address byte to SCIRXBUF (only in non-FIFO mode) - The reading of SCIRXBUF - An active SW RESET - A system reset
0	RESERVED	R	0h	Reserved

25.15.2.7 SCIRXEMU Register (Offset = 6h) [Reset = 0h]

SCIRXEMU is shown in Figure 25-17 and described in Table 25-14.

Return to the Summary Table.

Normal SCI data-receive operations read the data received from the SCIRXBUF register. The SCIRXEMU register is used principally by the emulator (EMU) because it can continuously read the data received for screen updates without clearing the RXRDY flag. SCIRXEMU is cleared by a system reset. This is the register that should be used in an emulator watch window to view the contents of the SCIRXBUF register. SCIRXEMU is not physically implemented
it is just a different address location to access the SCIRXBUF register without clearing the RXRDY flag.

Figure 25-17. SCIRXEMU Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
ERXDT							
R-0h							

Table 25-14. SCIRXEMU Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-0	ERXDT	R	0h	Receive emulation buffer data Reset type: SYSRSn

25.15.2.8 SCIRXBUF Register (Offset = 7h) [Reset = 0h]

SCIRXBUF is shown in Figure 25-18 and described in Table 25-15.

Return to the Summary Table.

When the current data received is shifted from RXSHF to the receiver buffer, flag bit RXRDY is set and the data is ready to be read. If the RXBKINTENA bit (SCICTL2.1) is set, this shift also causes an interrupt. When SCIRXBUF is read, the RXRDY flag is reset. SCIRXBUF is cleared by a system reset.

Figure 25-18. SCIRXBUF Register

15	14	13	12	11	10	9	8
SCIFFFE	SCIFFPE	RESERVED					
R-0h	R-0h	R-0h					
7	6	5	4	3	2	1	0
SAR							
R-0h							

Table 25-15. SCIRXBUF Register Field Descriptions

Bit	Field	Type	Reset	Description
15	SCIFFFE	R	0h	SCIFFFE. SCI FIFO Framing error flag bit (applicable only if the FIFO is enabled) Reset type: SYSRSn 0h (R/W) = No frame error occurred while receiving the character, in bits 7-0. This bit is associated with the character on the top of the FIFO. 1h (R/W) = A frame error occurred while receiving the character in bits 7-0. This bit is associated with the character on the top of the FIFO.
14	SCIFFPE	R	0h	SCIFFPE. SCI FIFO parity error flag bit (applicable only if the FIFO is enabled) Reset type: SYSRSn 0h (R/W) = No parity error occurred while receiving the character, in bits 7-0. This bit is associated with the character on the top of the FIFO. 1h (R/W) = A parity error occurred while receiving the character in bits 7-0. This bit is associated with the character on the top of the FIFO.
13-8	RESERVED	R	0h	Reserved
7-0	SAR	R	0h	Receive Character bits Reset type: SYSRSn

25.15.2.9 SCITXBUF Register (Offset = 9h) [Reset = 0h]

SCITXBUF is shown in Figure 25-19 and described in Table 25-16.

Return to the Summary Table.

Data bits to be transmitted are written to SCITXBUF. These bits must be rightjustified because the leftmost bits are ignored for characters less than eight bits long. The transfer of data from this register to the TXSHF transmitter shift register sets the TXRDY flag (SCICTL2.7), indicating that SCITXBUF is ready to receive another set of data. If bit TXINTENA (SCICTL2.0) is set, this data transfer also causes an interrupt.

Figure 25-19. SCITXBUF Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
TXDT							
R/W-0h							

Table 25-16. SCITXBUF Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-0	TXDT	R/W	0h	Transmit data buffer Reset type: SYSRSn

25.15.2.10 SCIFFTX Register (Offset = Ah) [Reset = A000h]

SCIFFTX is shown in Figure 25-20 and described in Table 25-17.

Return to the Summary Table.

SCIFFTX controls the transmit FIFO interrupt, FIFO enhancements, and reset for the SCI transmit and receive channels.

Figure 25-20. SCIFFTX Register

15	14	13	12	11	10	9	8
SCIRST	SCIFFENA	TXFIFORESET	TXFFST				
R/W-1h	R/W-0h	R/W-1h	R-0h				
7	6	5	4	3	2	1	0
TXFFINT	TXFFINTCLR	TXFFIENA	TXFFIL				
R-0h	R-0/W1S-0h	R/W-0h	R/W-0h				

Table 25-17. SCIFFTX Register Field Descriptions

Bit	Field	Type	Reset	Description
15	SCIRST	R/W	1h	SCI Reset 0 Write 0 to reset the SCI transmit and receive channels. SCI FIFO register configuration bits will be left as is. 1 SCI FIFO can resume transmit or receive. SCIRST should be 1 even for Autobaud logic to work. Reset type: SYSRSn
14	SCIFFENA	R/W	0h	SCI FIFO enable Reset type: SYSRSn 0h (R/W) = SCI FIFO enhancements are disabled 1h (R/W) = SCI FIFO enhancements are enabled
13	TXFIFORESET	R/W	1h	Transmit FIFO reset Reset type: SYSRSn 0h (R/W) = Reset the FIFO pointer to zero and hold in reset 1h (R/W) = Re-enable transmit FIFO operation
12-8	TXFFST	R	0h	FIFO status Reset type: SYSRSn 0h (R/W) = Transmit FIFO is empty 1h (R/W) = Transmit FIFO has 1 words 2h (R/W) = Transmit FIFO has 2 words 3h (R/W) = Transmit FIFO has 3 words 4h (R/W) = Transmit FIFO has 4 words 5h (R/W) = Transmit FIFO has 5 words 6h (R/W) = Transmit FIFO has 6 words 7h (R/W) = Transmit FIFO has 7 words 8h (R/W) = Transmit FIFO has 8 words 9h (R/W) = Transmit FIFO has 9 words Ah (R/W) = Transmit FIFO has 10 words Bh (R/W) = Transmit FIFO has 11 words Ch (R/W) = Transmit FIFO has 12 words Dh (R/W) = Transmit FIFO has 13 words Eh (R/W) = Transmit FIFO has 14 words Fh (R/W) = Transmit FIFO has 15 words 10h (R/W) = Transmit FIFO has 16 words
7	TXFFINT	R	0h	Transmit FIFO interrupt Reset type: SYSRSn 0h (R/W) = TXFIFO interrupt has not occurred, read-only bit 1h (R/W) = TXFIFO interrupt has occurred, read-only bit
6	TXFFINTCLR	R-0/W1S	0h	Transmit FIFO clear Reset type: SYSRSn 0h (R/W) = Write 0 has no effect on TXFIFINT flag bit, Bit reads back a zero 1h (R/W) = Write 1 to clear TXFFINT flag in bit 7

Table 25-17. SCIFFTX Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	TXFFIENA	R/W	0h	Transmit FIFO interrrupt enable Reset type: SYSRSn 0h (R/W) = TX FIFO interrupt is disabled 1h (R/W) = TX FIFO interrupt is enabled. This interrupt is triggered whenever the transmit FIFO status (TXFFST) bits match (equal to or less than) the interrupt trigger level bits TXFFIL (bits 4-0).
4-0	TXFFIL	R/W	0h	TXFFIL4-0 Transmit FIFO interrupt level bits. The transmit FIFO generates an interrupt whenever the FIFO status bits (TXFFST4-0) are less than or equal to the FIFO level bits (TXFFIL4-0). The maximum value that can be assigned to these bits to generate an interrupt cannot be more than the depth of the TX FIFO. The default value of these bits after reset is 00000b. Users should set TXFFIL to best fit their application needs by weighing between the CPU overhead to service the ISR and the best possible usage of SCI bus bandwidth. Reset type: SYSRSn

25.15.2.11 SCIFFRX Register (Offset = Bh) [Reset = 201Fh]

SCIFFRX is shown in Figure 25-21 and described in Table 25-18.

Return to the Summary Table.

SCIFFRX controls the receive FIFO interrupt, receive FIFO reset, and status of the receive FIFO overflow.

Figure 25-21. SCIFFRX Register

15	14	13	12	11	10	9	8
RXFFOVF	RXFFOVRCLR	RXFIFORESET	RXFFST				
R-0h	R-0/W1S-0h	R/W-1h	R-0h				
7	6	5	4	3	2	1	0
RXFFINT	RXFFINTCLR	RXFFIENA	RXFFIL				
R-0h	W-0h	R/W-0h	R/W-1Fh				

Table 25-18. SCIFFRX Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RXFFOVF	R	0h	Receive FIFO overflow. This will function as flag, but cannot generate interrupt by itself. This condition will occur while receive interrupt is active. Receive interrupts should service this flag condition. Reset type: SYSRSn 0h (R/W) = Receive FIFO has not overflowed, read-only bit 1h (R/W) = Receive FIFO has overflowed, read-only bit. More than 16 words have been received in to the FIFO, and the first received word is lost
14	RXFFOVRCLR	R-0/W1S	0h	RXFFOVF clear Reset type: SYSRSn 0h (R/W) = Write 0 has no effect on RXFFOVF flag bit, Bit reads back a zero 1h (R/W) = Write 1 to clear RXFFOVF flag in bit 15
13	RXFIFORESET	R/W	1h	Receive FIFO reset Reset type: SYSRSn 0h (R/W) = Write 0 to reset the FIFO pointer to zero, and hold in reset. 1h (R/W) = Re-enable receive FIFO operation
12-8	RXFFST	R	0h	FIFO status Reset type: SYSRSn 0h (R/W) = Receive FIFO is empty 1h (R/W) = Receive FIFO has 1 words 2h (R/W) = Receive FIFO has 2 words 3h (R/W) = Receive FIFO has 3 words 4h (R/W) = Receive FIFO has 4 words 5h (R/W) = Receive FIFO has 5 words 6h (R/W) = Receive FIFO has 6 words 7h (R/W) = Receive FIFO has 7 words 8h (R/W) = Receive FIFO has 8 words 9h (R/W) = Receive FIFO has 9 words Ah (R/W) = Receive FIFO has 10 words Bh (R/W) = Receive FIFO has 11 words Ch (R/W) = Receive FIFO has 12 words Dh (R/W) = Receive FIFO has 13 words Eh (R/W) = Receive FIFO has 14 words Fh (R/W) = Receive FIFO has 15 words 10h (R/W) = Receive FIFO has 16 words
7	RXFFINT	R	0h	Receive FIFO interrupt Reset type: SYSRSn 0h (R/W) = RXFIFO interrupt has not occurred, read-only bit 1h (R/W) = RXFIFO interrupt has occurred, read-only bit

Table 25-18. SCIFFRX Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	RXFFINTCLR	W	0h	Receive FIFO interrupt clear Reset type: SYSRSn 0h (R/W) = Write 0 has no effect on RXFIFINT flag bit. Bit reads back a zero. 1h (R/W) = Write 1 to clear RXFFINT flag in bit 7
5	RXFFIENA	R/W	0h	Receive FIFO interrupt enable Reset type: SYSRSn 0h (R/W) = RX FIFO interrupt is disabled 1h (R/W) = RX FIFO interrupt is enabled. This interrupt is triggered whenever the receive FIFO status (RXFFST) bits match (equal to or greater than) the interrupt trigger level bits RXFFIL (bits 4-0).
4-0	RXFFIL	R/W	1Fh	Receive FIFO interrupt level bits The receive FIFO generates an interrupt whenever the FIFO status bits (RXFFST4-0) are greater than or equal to the FIFO level bits (RXFFIL4-0). The maximum value that can be assigned to these bits to generate an interrupt cannot be more than the depth of the RX FIFO. The default value of these bits after reset is 11111b. Users should set RXFFIL to best fit their application needs by weighing between the CPU overhead to service the ISR and the best possible usage of received SCI data. Reset type: SYSRSn

25.15.2.12 SCIFFCT Register (Offset = Ch) [Reset = 0h]

SCIFFCT is shown in Figure 25-22 and described in Table 25-19.

Return to the Summary Table.

SCIFFCT contains the status of auto-baud detect, clears the auto-baud flag, and calibrate for A-detect bit.

Figure 25-22. SCIFFCT Register

15	14	13	12	11	10	9	8
ABD	ABDCLR	CDC	RESERVED				
R-0h	W-0h	R/W-0h	R-0h				
7	6	5	4	3	2	1	0
FFTXDLY							
R/W-0h							

Table 25-19. SCIFFCT Register Field Descriptions

Bit	Field	Type	Reset	Description
15	ABD	R	0h	Auto-baud detect (ABD) bit Reset type: SYSRSn 0h (R/W) = Auto-baud detection is not complete. "A","a" character has not been received successfully. 1h (R/W) = Auto-baud hardware has detected "A" or "a" character on the SCI receive register. Auto-detect is complete.
14	ABDCLR	W	0h	ABD-clear bit Reset type: SYSRSn 0h (R/W) = Write 0 has no effect on ABD flag bit. Bit reads back a zero. 1h (R/W) = Write 1 to clear ABD flag in bit 15.
13	CDC	R/W	0h	CDC calibrate A-detect bit Reset type: SYSRSn 0h (R/W) = Disables auto-baud alignment 1h (R/W) = Enables auto-baud alignment
12-8	RESERVED	R	0h	Reserved
7-0	FFTXDLY	R/W	0h	FIFO transfer delay. These bits define the delay between every transfer from FIFO transmit bufferto transmit shift register. The delay is defined in the number of SCI serial baud clock cycles. The 8 bit register could define a minimum delay of 0 baud clock cycles and a maximum of 256 baud clock cycles In FIFO mode, the buffer (TXBUF) between the shift register and the FIFO should be filled only after the shift register has completed shifting of the last bit. This is required to pass on the delay between transfers to the data stream. In FIFO mode, TXBUF should not be treated as one additional level of buffer. The delayed transmit feature will help to create an auto-flow scheme without RTS/CTS controls as in standard UARTS. When SCI is configured for one stop-bit, delay introduced by FFTXDLY between one frame and the next frame is equal to number of baud clock cycles that FFTXDLY is set to. When SCI is configured for two stop-bits, delay introduced by FFTXDLY between one frame and the next frame is equal to number of baud clock cycles that FFTXDLY is set to minus 1. Reset type: SYSRSn

25.15.2.13 SCIPRI Register (Offset = Fh) [Reset = 0h]

SCIPRI is shown in Figure 25-23 and described in Table 25-20.

Return to the Summary Table.

SCIPRI determines what happens when an emulation suspend event occurs.

Figure 25-23. SCIPRI Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			FREESOFT		RESERVED		
R-0h			R/W-0h		R-0h		

Table 25-20. SCIPRI Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-5	RESERVED	R	0h	Reserved
4-3	FREESOFT	R/W	0h	These bits determine what occurs when an emulation suspend event occurs (for example, when the debugger hits a breakpoint). The peripheral can continue whatever it is doing (free-run mode), or if in stop mode, it can either stop immediately or stop when the current operation (the current receive/transmit sequence) is complete. Reset type: SYSRSn 0h (R/W) = Immediate stop on suspend 1h (R/W) = Complete current receive/transmit sequence before stopping 2h (R/W) = Free run 3h (R/W) = Free run
2-0	RESERVED	R	0h	Reserved

25.15.3 SCI Registers to Driverlib Functions

Table 25-21. SCI Registers to Driverlib Functions

File	Driverlib Function
SCICCR	
sci.c	SCI_setConfig
sci.h	SCI_setParityMode
sci.h	SCI_getParityMode
sci.h	SCI_setAddrMultiProcessorMode
sci.h	SCI_setIdleMultiProcessorMode
sci.h	SCI_getConfig
sci.h	SCI_enableLoopback
sci.h	SCI_disableLoopback
SCICTL1	
sci.c	SCI_enableInterrupt
sci.c	SCI_disableInterrupt
sci.c	SCI_setWakeFlag
sci.h	SCI_enableModule
sci.h	SCI_disableModule
sci.h	SCI_performSoftwareReset
SCIHBAUD	

Table 25-21. SCI Registers to Driverlib Functions (continued)

File	Driverlib Function
sci.c	SCI_setConfig
sci.c	SCI_setBaud
sci.h	SCI_lockAutobaud
sci.h	SCI_getConfig
SCILBAUD	
sci.c	SCI_setConfig
sci.c	SCI_setBaud
sci.h	SCI_lockAutobaud
sci.h	SCI_getConfig
SCICTL2	
sci.c	SCI_enableInterrupt
sci.c	SCI_disableInterrupt
sci.c	SCI_getInterruptStatus
sci.h	SCI_isSpaceAvailableNonFIFO
sci.h	SCI_isTransmitterBusy
SCIRXST	
sci.c	SCI_getInterruptStatus
sci.h	SCI_isDataAvailableNonFIFO
sci.h	SCI_getRxStatus
SCIRXEMU	
-	
SCIRXBUF	
sci.c	SCI_readCharArray
sci.h	SCI_readCharBlockingFIFO
sci.h	SCI_readCharBlockingNonFIFO
sci.h	SCI_readCharNonBlocking
SCITXBUF	
sci.c	SCI_writeCharArray
sci.h	SCI_writeCharBlockingFIFO
sci.h	SCI_writeCharBlockingNonFIFO
sci.h	SCI_writeCharNonBlocking
SCIFFTX	
sci.c	SCI_enableInterrupt
sci.c	SCI_disableInterrupt
sci.c	SCI_getInterruptStatus
sci.c	SCI_clearInterruptStatus
sci.h	SCI_setFIFOInterruptLevel
sci.h	SCI_getFIFOInterruptLevel
sci.h	SCI_disableModule
sci.h	SCI_enableFIFO
sci.h	SCI_disableFIFO
sci.h	SCI_isFIFOEnabled
sci.h	SCI_resetTxFIFO
sci.h	SCI_resetChannels
sci.h	SCI_getTxFIFOStatus

Table 25-21. SCI Registers to Driverlib Functions (continued)

File	Driverlib Function
sci.h	SCI_isTransmitterBusy
SCIFFRX	
sci.c	SCI_enableInterrupt
sci.c	SCI_disableInterrupt
sci.c	SCI_getInterruptStatus
sci.c	SCI_clearInterruptStatus
sci.h	SCI_setFIFOInterruptLevel
sci.h	SCI_getFIFOInterruptLevel
sci.h	SCI_enableFIFO
sci.h	SCI_resetRxFIFO
sci.h	SCI_getRxFIFOStatus
sci.h	SCI_getOverflowStatus
sci.h	SCI_clearOverflowStatus
SCIFFCT	
sci.h	SCI_lockAutobaud
SCIPRI	
-	

Chapter 26

Inter-Integrated Circuit Module (I2C)

This chapter describes the features and operation of the inter-integrated circuit (I2C) module. The I2C module provides an interface between one of these devices and devices compliant with the NXP Semiconductors Inter-IC bus (I2C bus) specification version 2.1, and connected by way of an I2C bus. External components attached to this 2-wire serial bus can transmit/receive 1- to 8-bit data to/from the device through the I2C module. This chapter assumes the reader is familiar with the I2C bus specification.

Note

A unit of data transmitted or received by the I2C module can have fewer than 8 bits; however, for convenience, a unit of data is called a data byte throughout this document. The number of bits in a data byte is selectable by way of the BC bits of the mode register, I2CMDR.

26.1 Introduction

The I2C module supports any slave or master I2C-compatible device. Figure 26-1 shows an example of multiple I2C modules connected for a two-way transfer from one device to other devices.

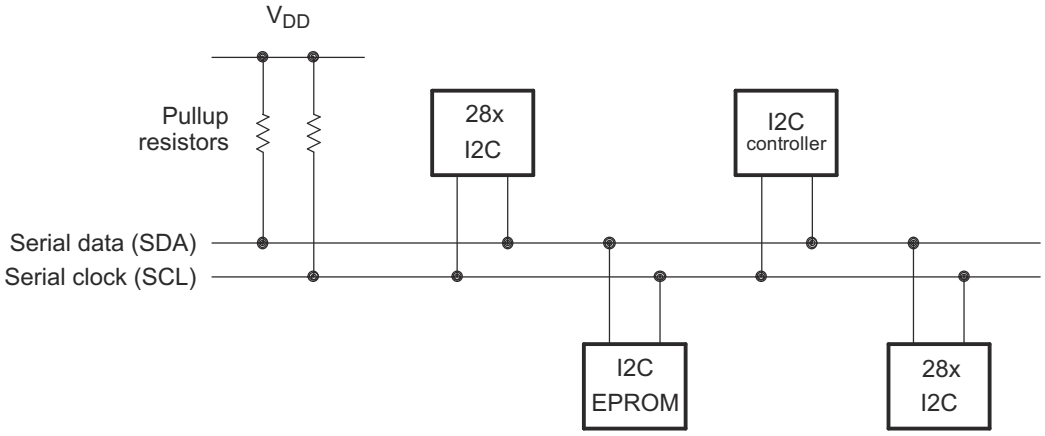


Figure 26-1. Multiple I2C Modules Connected

26.1.1 I2C Related Collateral

Foundational Materials

- C2000 Academy - Communications
- I2C Hardware Overview (Video)
- I2C Protocol Overview (Video)
- *Understanding the I2C Bus Application Report*

Getting Started Materials

- *Configuring the TMS320F280x DSP as an I2C Processor Application Report*
- I2C Buffers Overview (Video)
- *I2C Dynamic Addressing Application Report*
- I2C translators overview (Video)
- *Why, When, and How to use I2C Buffers Application Report*

Expert Materials

- *I2C Bus Pull-Up Resistor Calculation Application Report*
- *Maximum Clock Frequency of I2C Bus Using Repeaters Application Report*

26.1.2 Features

The I2C module has the following features:

- Compliance with the NXP Semiconductors I2C bus specification (version 2.1):
  - Support for 8-bit format transfers
  - 7-bit and 10-bit addressing modes
  - General call
  - START byte mode
  - Support for multiple master-transmitters and slave-receivers
  - Support for multiple slave-transmitters and master-receivers
  - Combined master transmit/receive and receive/transmit mode
  - Data transfer rate from 10 kbps up to 400 kbps (Fast-mode)
- Receive FIFO and Transmitter FIFO (16-deep x 8-bit FIFO)
- Supports two ePIE interrupts:
  - I2Cx Interrupt – Any of the following events can be configured to generate an I2Cx interrupt:
    - Transmit-data ready
    - Receive-data ready
    - Register-access ready
    - No-acknowledgment received
    - Arbitration lost
    - Stop condition detected
    - Addressed as slave
  - I2Cx_FIFO interrupts:
    - Transmit FIFO interrupt
    - Receive FIFO interrupt
- Module enable/disable capability
- Free data format mode

26.1.3 Features Not Supported

The I2C module does not support:

- High-speed mode (Hs-mode)
- CBUS-compatibility mode

26.1.4 Functional Overview

Each device connected to an I2C bus is recognized by a unique address. Each device can operate as either a transmitter or a receiver, depending on the function of the device. A device connected to the I2C bus can also be considered as the master or the slave when performing data transfers. A master device is the device that initiates a data transfer on the bus and generates the clock signals to permit that transfer. During this transfer, any device addressed by this master is considered a slave. The I2C module supports the multi-master mode, in which one or more devices capable of controlling an I2C bus can be connected to the same I2C bus.

For data communication, the I2C module has a serial data pin (SDA) and a serial clock pin (SCL), as shown in Figure 26-2. These two pins carry information between the C28x device and other devices connected to the I2C bus. The SDA and SCL pins both are bidirectional. They each must be connected to a positive supply voltage using a pull-up resistor. When the bus is free, both pins are high. The driver of these two pins has an open-drain configuration to perform the required wired-AND function.

There are two major transfer techniques:

- Standard Mode: Send exactly n data values, where n is a value you program in an I2C module register. See the I2CCNT register in Section 26.7 for more information.
- Repeat Mode: Keep sending data values until you use software to initiate a STOP condition or a new START condition. See the I2CMDR register in Section 26.7 for RM bit information.

The I2C module consists of the following primary blocks:

- A serial interface: one data pin (SDA) and one clock pin (SCL)
- Data registers and FIFOs to temporarily hold receive data and transmit data traveling between the SDA pin and the CPU
- Control and status registers
- A peripheral bus interface to enable the CPU to access the I2C module registers and FIFOs.
- A clock synchronizer to synchronize the I2C input clock (from the device clock generator) and the clock on the SCL pin, and to synchronize data transfers with masters of different clock speeds
- A prescaler to divide down the input clock that is driven to the I2C module
- A noise filter on each of the two pins, SDA and SCL
- An arbitrator to handle arbitration between the I2C module (when it is a master) and another master
- Interrupt generation logic, so that an interrupt can be sent to the CPU
- FIFO interrupt generation logic, so that FIFO access can be synchronized to data reception and data transmission in the I2C module

Figure 26-2 shows the four registers used for transmission and reception in non-FIFO mode. The CPU writes data for transmission to I2CDXR and reads received data from I2CDRR. When the I2C module is configured as a transmitter, data written to I2CDXR is copied to I2CXSR and shifted out on the SDA pin one bit at a time. When the I2C module is configured as a receiver, received data is shifted into I2CRSR and then copied to I2CDRR.

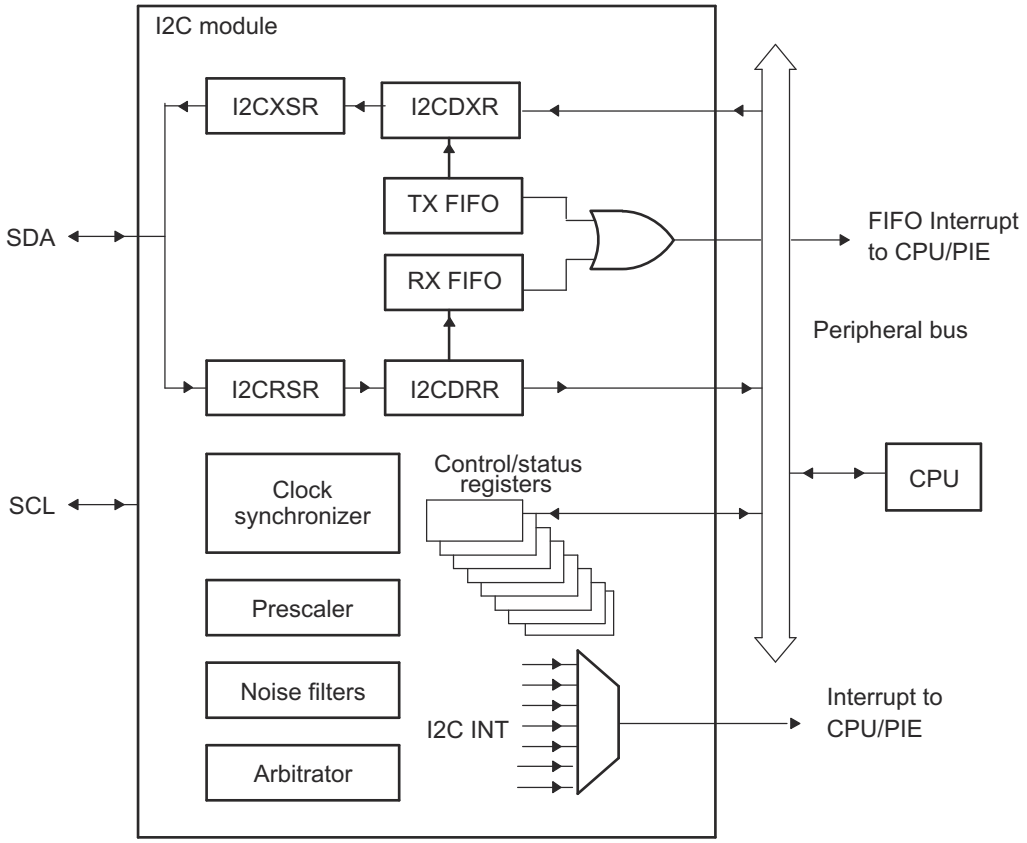


Figure 26-2. I2C Module Conceptual Block Diagram

26.1.5 Clock Generation

The I2C module clock determines the frequency at which the I2C module operates. A programmable prescaler in the I2C module divides down the SYSCLK to produce the I2C module clock and this I2C module clock is divided further to produce the I2C master clock on the SCL pin. Figure 26-3 shows the clock generation diagram for I2C module.

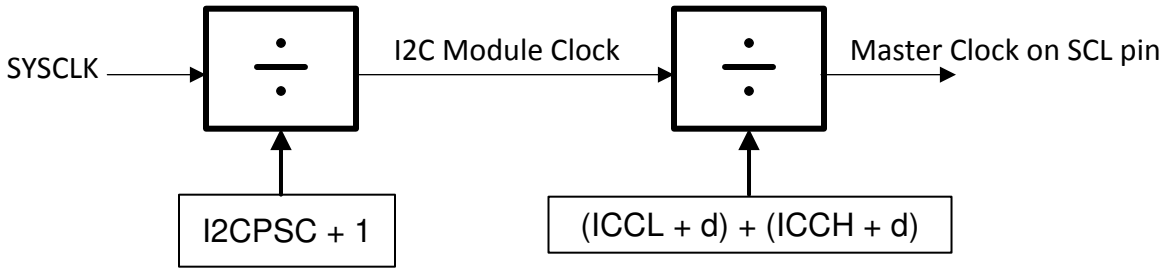


Figure 26-3. Clocking Diagram for the I2C Module

Note

To meet all of the I2C protocol timing specifications, the I2C module clock must be between 7-12 MHz.

To specify the divide-down value, initialize the IPSC field of the prescaler register, I2CPSC. The resulting frequency is:

$$\text{I2C Module Clock (Fmod)} = \frac{\text{SYSCLK}}{(\text{I2CPSC} + 1)} \quad (26)$$

The prescaler must be initialized only while the I2C module is in the reset state (IRS = 0 in I2CMDR). The prescaled frequency takes effect only when IRS is changed to 1. Changing the IPSC value while IRS = 1 has no effect.

The master clock appears on the SCL pin when the I2C module is configured to be a master on the I2C bus. This clock controls the timing of communication between the I2C module and a slave. As shown in Figure 26-3, a second clock divider in the I2C module divides down the module clock to produce the master clock. The clock divider uses the ICCL value of I2CCLKL to divide down the low portion of the module clock signal and uses the ICCH value of I2CCLKH to divide down the high portion of the module clock signal. See Section 26.1.6 for the master clock frequency equation.

26.1.6 I2C Clock Divider Registers (I2CCLKL and I2CCLKH)

As explained in Section 26.1.5, when the I2C module is a master, the I2C module clock is divided down further to use as the master clock on the SCL pin. As shown in Figure 26-4, the shape of the master clock depends on two divide-down values:

- ICCL in I2CCLKL. For each master clock cycle, ICCL determines the amount of time the signal is low.
- ICCH in I2CCKLH. For each master clock cycle, ICCH determines the amount of time the signal is high.

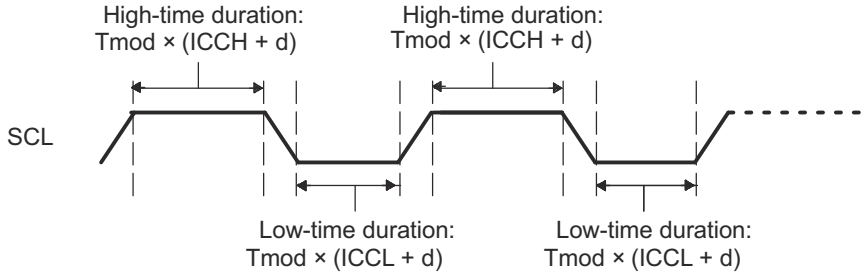


Figure 26-4. The Roles of the Clock Divide-Down Values (ICCL and ICCH)

26.1.6.1 Formula for the Master Clock Period

The master clock period (Tmst) is a multiple of the period of the I2C Module Clock (Tmod):

$$\text{Master Clock period (Tmst)} = \frac{[\,(\text{ICCH} + d) + (\text{ICCL} + d)\,]}{\text{I2C Module Clock (Fmod)}} \quad (27)$$

where d depends on the divide-down value IPSC, as shown in Table 26-1. IPSC is described in the I2CPSC register.

Table 26-1. Dependency of Delay d on the Divide-Down Value IPSC

IPSC	d
0	7
1	6
Greater than 1	5

26.2 Configuring Device Pins

The GPIO mux registers must be configured to connect this peripheral to the device pins. To avoid glitches on the pins, the GPyGMUX bits must be configured first (while keeping the corresponding GPyMUX bits at the default of zero), followed by writing the GPyMUX register to the desired value.

Some IO functionality is defined by GPIO register settings independent of this peripheral. For input signals, the GPIO input qualification should be set to asynchronous mode by setting the appropriate GPxQSELn register bits to 11b. The internal pullups can be configured in the GPyPUD register.

See the *GPIO* chapter for more details on GPIO mux and settings.

26.3 I2C Module Operational Details

This section provides an overview of the I2C bus protocol and how it is implemented.

26.3.1 Input and Output Voltage Levels

One clock pulse is generated by the master device for each data bit transferred. Due to a variety of different technology devices that can be connected to the I2C bus, the levels of logic 0 (low) and logic 1 (high) are not fixed and depend on the associated level of V_{DD}. For details, see the data manual for your particular device.

26.3.2 Data Validity

The data on SDA must be stable during the high period of the clock (see Figure 26-5). The high or low state of the data line, SDA, should change only when the clock signal on SCL is low.

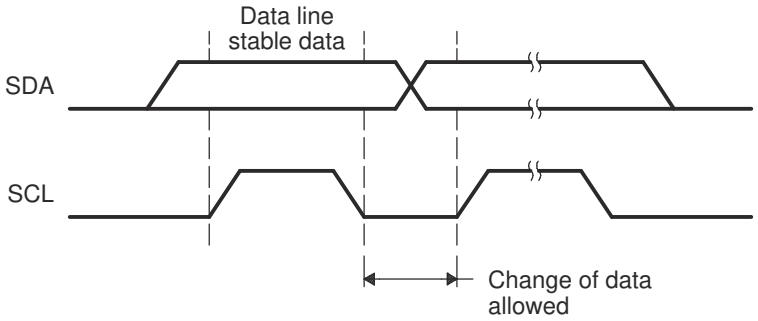


Figure 26-5. Bit Transfer on the I2C bus

26.3.3 Operating Modes

The I2C module has four basic operating modes to support data transfers as a master and as a slave. See Table 26-2 for the names and descriptions of the modes.

If the I2C module is a master, it begins as a master-transmitter and typically transmits an address for a particular slave. When giving data to the slave, the I2C module must remain a master-transmitter. To receive data from a slave, the I2C module must be changed to the master-receiver mode.

If the I2C module is a slave, it begins as a slave-receiver and typically sends acknowledgment when it recognizes its slave address from a master. If the master will be sending data to the I2C module, the module must remain a slave-receiver. If the master has requested data from the I2C module, the module must be changed to the slave-transmitter mode.

Table 26-2. Operating Modes of the I2C Module

Operating Mode	Description
Slave-receiver mode	The I2C module is a slave and receives data from a master. All slaves begin in this mode. In this mode, serial data bits received on SDA are shifted in with the clock pulses that are generated by the master. As a slave, the I2C module does not generate the clock signal, but it can hold SCL low while the intervention of the device is required (RSFULL = 1 in I2CSTR) after a byte has been received. See Section 26.3.7 for more details.
Slave-transmitter mode	The I2C module is a slave and transmits data to a master. This mode can be entered only from the slave-receiver mode; the I2C module must first receive a command from the master. When you are using any of the 7-bit/10-bit addressing formats, the I2C module enters its slave-transmitter mode if the slave address byte is the same as its own address (in I2COAR) and the master has transmitted R/ $\overline{W}$ = 1. As a slave-transmitter, the I2C module then shifts the serial data out on SDA with the clock pulses that are generated by the master. While a slave, the I2C module does not generate the clock signal, but it can hold SCL low while the intervention of the device is required (XSMT = 0 in I2CSTR) after a byte has been transmitted. See Section 26.3.7 for more details.
Master-receiver mode	The I2C module is a master and receives data from a slave. This mode can be entered only from the master-transmitter mode; the I2C module must first transmit a command to the slave. When you are using any of the 7-bit/10-bit addressing formats, the I2C module enters its master-receiver mode after transmitting the slave address byte and R/ $\overline{W}$ = 1. Serial data bits on SDA are shifted into the I2C module with the clock pulses generated by the I2C module on SCL. The clock pulses are inhibited and SCL is held low when the intervention of the device is required (RSFULL = 1 in I2CSTR) after a byte has been received.
Master-transmitter mode	The I2C module is a master and transmits control information and data to a slave. All masters begin in this mode. In this mode, data assembled in any of the 7-bit/10-bit addressing formats is shifted out on SDA. The bit shifting is synchronized with the clock pulses generated by the I2C module on SCL. The clock pulses are inhibited and SCL is held low when the intervention of the device is required (XSMT = 0 in I2CSTR) after a byte has been transmitted.

To summarize, SCL will be held low in the following conditions:

- When an overrun condition is detected (RSFULL = 1), in Slave-receiver mode.
- When an underflow condition is detected (XSMT = 0), in Slave-transmitter mode.

I2C slave nodes have to accept and provide data when the I2C master node requests it.

- To release SCL in slave-receiver mode, read data from I2CDRR.
- To release SCL in slave-transmitter mode, write data to I2CDXR.
- To force a release without handling the data, reset the module using the I2CMDR.IRS bit.

Table 26-3. Master-Transmitter/Receiver Bus Activity Defined by the RM, STT, and STP Bits of I2CMDR

RM	STT	STP	Bus Activity(1)	Description
0	0	0	None	No activity
0	0	1	P	STOP condition
0	1	0	S-A-D..(n)..D.	START condition, slave address, n data bytes (n = value in I2CCNT)
0	1	1	S-A-D..(n)..D-P	START condition, slave address, n data bytes, STOP condition (n = value in I2CCNT)
1	0	0	None	No activity
1	0	1	P	STOP condition
1	1	0	S-A-D-D-D.	Repeat mode transfer: START condition, slave address, continuous data transfers until STOP condition or next START condition
1	1	1	None	Reserved bit combination (No activity)

(1) S = START condition; A = Address; D = Data byte; P = STOP condition;

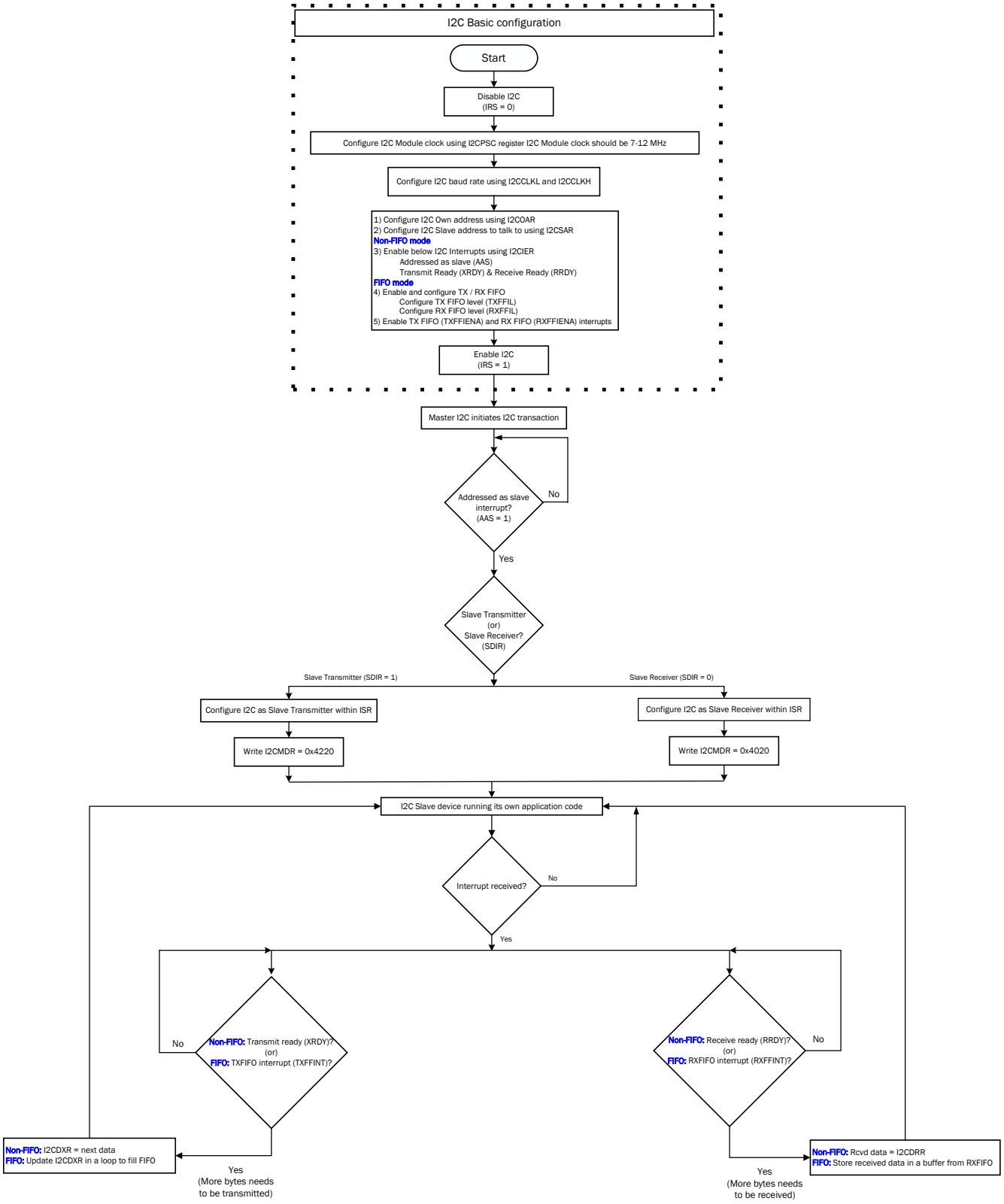


Figure 26-6. I2C Slave TX / RX Flowchart

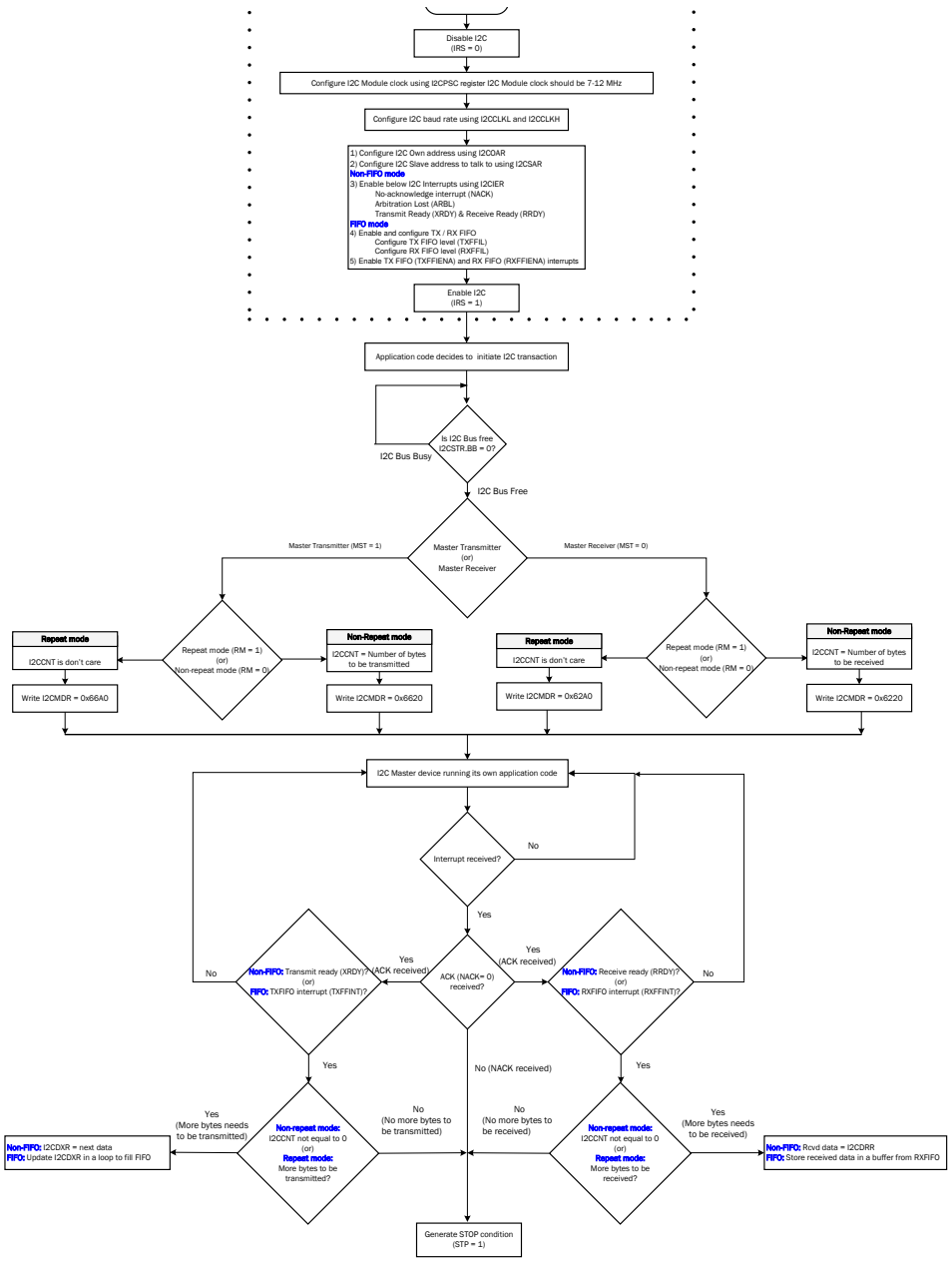


Figure 26-7. I2C Master TX / RX Flowchart

26.3.4 I2C Module START and STOP Conditions

START and STOP conditions can be generated by the I2C module when the module is configured to be a master on the I2C bus. As shown in Figure 26-8:

- The START condition is defined as a high-to-low transition on the SDA line while SCL is high. A master drives this condition to indicate the start of a data transfer.
- The STOP condition is defined as a low-to-high transition on the SDA line while SCL is high. A master drives this condition to indicate the end of a data transfer.

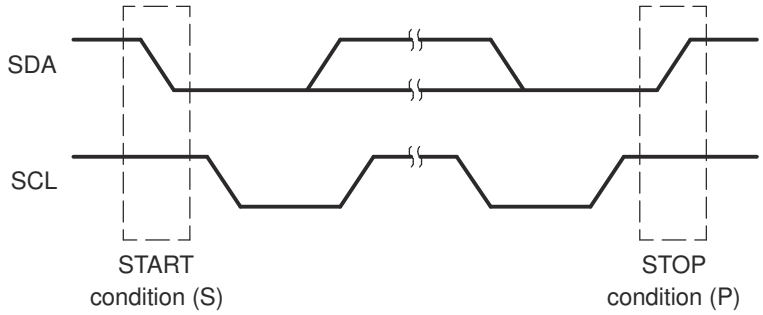


Figure 26-8. I2C Module START and STOP Conditions

After a START condition and before a subsequent STOP condition, the I2C bus is considered busy, and the bus busy (BB) bit of I2CSTR is 1. Between a STOP condition and the next START condition, the bus is considered free, and BB is 0.

For the I2C module to start a data transfer with a START condition, the master mode bit (MST) and the START condition bit (STT) in I2CMDR must both be 1. For the I2C module to end a data transfer with a STOP condition, the STOP condition bit (STP) must be set to 1. When the BB bit is set to 1 and the STT bit is set to 1, a repeated START condition is generated. For a description of I2CMDR and its bits (including MST, STT, and STP), see Section 26.7.

The I2C peripheral cannot detect a START or STOP condition while it is in reset (IRS = 0). The BB bit will remain in the cleared state (BB = 0) while the I2C peripheral is in reset (IRS = 0). When the I2C peripheral is taken out of reset (IRS set to 1) the BB bit will not correctly reflect the I2C bus status until a START or STOP condition is detected.

Follow these steps before initiating the first data transfer with I2C:

1. After taking the I2C peripheral out of reset by setting the IRS bit to 1, wait a period larger than the total time taken for the longest data transfer in the application. By waiting for a period of time after I2C comes out of reset, users can ensure that at least one START or STOP condition will have occurred on the I2C bus and been captured by the BB bit. After this period, the BB bit will correctly reflect the state of the I2C bus.
2. Check the BB bit and verify that BB = 0 (bus not busy) before proceeding.
3. Begin data transfers.

Not resetting the I2C peripheral in between transfers ensures that the BB bit reflects the actual bus status. If users must reset the I2C peripheral in between transfers, repeat steps 1 through 3 every time the I2C peripheral is taken out of reset.

26.3.5 Non-repeat Mode vs Repeat Mode

Non-repeat mode:

- When I2CMDR.RM = 0, I2C module is configured in non-repeat mode.
- I2CCNT register determines the number of bytes to be transmitted (or) received
- If STP = 0 in I2CMDR, the ARDY bit is set when the internal data counter counts down to 0.
- If STP = 1, ARDY bit doesn't get set and I2C module generates a STOP condition when the internal data counter counts down to 0.

Note

In non-repeat mode (RM = 0), if I2CCNT is set to 0, I2C statemachine expects to transmit (or) receive 65536 bytes and not 0 bytes.

Repeat mode:

- When I2CMDR.RM = 1, I2C module is configured in repeat mode.
- I2CCNT register contents don't determine the number of bytes to be transmitted (or) received.
- Number of bytes to be transmitted (or) received can be controlled by software.
- ARDY bit gets set at end of transmission and receiption of each byte

Note

Once you start I2C transaction in non-repeat mode (or) repeat mode, you cannot switch into another mode until the I2C transaction is completed with a STOP condition.

26.3.6 Serial Data Formats

Figure 26-9 shows an example of a data transfer on the I2C bus. The I2C module supports 1 to 8-bit data values. In Figure 26-9, 8-bit data is transferred. Each bit put on the SDA line equates to 1 pulse on the SCL line, and the values are always transferred with the most significant bit (MSB) first. The number of data values that can be transmitted or received is unrestricted. The serial data format used in Figure 26-9 is the 7-bit addressing format. The I2C module supports the formats shown in Figure 26-10 through Figure 26-12 and described in the paragraphs that follow the figures.

Note

In Figure 26-9 through Figure 26-12, n = the number of data bits (from 1 to 8) specified by the bit count (BC) field of I2CMDR.

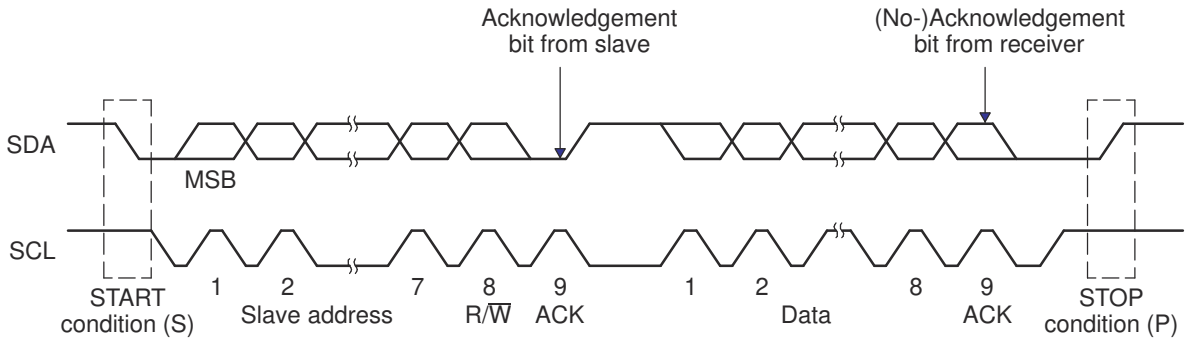


Figure 26-9. I2C Module Data Transfer (7-Bit Addressing with 8-bit Data Configuration Shown)

26.3.6.1 7-Bit Addressing Format

The 7-bit addressing format is the default format after reset. Disabling expanded address (I2CMDR.XA = 0) and free data format (I2CMDR.FDF = 0) enables 7-bit addressing format.

In this format (see Figure 26-10), the first byte after a START condition (S) consists of a 7-bit slave address followed by a R/ $\overline{W}$ bit. R/ $\overline{W}$ determines the direction of the data:

- R/ $\overline{W}$ = 0: The I2C master writes (transmits) data to the addressed slave. This can be achieved by setting I2CMDR.TRX = 1 (Transmitter mode)
- R/ $\overline{W}$ = 1: The I2C master reads (receives) data from the slave. This can be achieved by setting I2CMDR.TRX = 0 (Receiver mode)

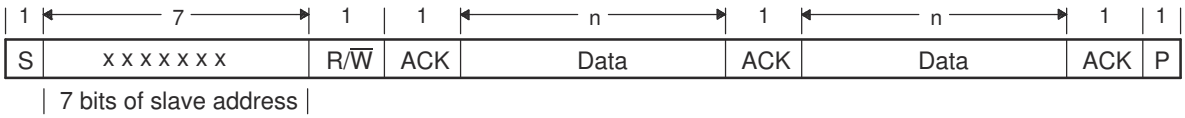


Figure 26-10. I2C Module 7-Bit Addressing Format (FDF = 0, XA = 0 in I2CMDR)

An extra clock cycle dedicated for acknowledgment (ACK) is inserted after each byte. If the ACK bit is inserted by the slave after the first byte from the master, it is followed by n bits of data from the transmitter (master or slave, depending on the R/ $\overline{W}$ bit). n is a number from 1 to 8 determined by the bit count (BC) field of I2CMDR. After the data bits have been transferred, the receiver inserts an ACK bit.

26.3.6.2 10-Bit Addressing Format

The 10-bit addressing format can be enabled by setting expanded address (I2CMDR.XA = 1) and disabling free data format (I2CMDR.FDF = 0).

The 10-bit addressing format (see Figure 26-11) is similar to the 7-bit addressing format, but the master sends the slave address in two separate byte transfers. The first byte consists of 11110b, the two MSBs of the 10-bit slave address, and R/ $\overline{W}$. The second byte is the remaining 8 bits of the 10-bit slave address. The slave must send acknowledgment after each of the two byte transfers. Once the master has written the second byte to the slave, the master can either write data or use a repeated START condition to change the data direction. For more details about using 10-bit addressing, see the NXP Semiconductors I2C bus specification.

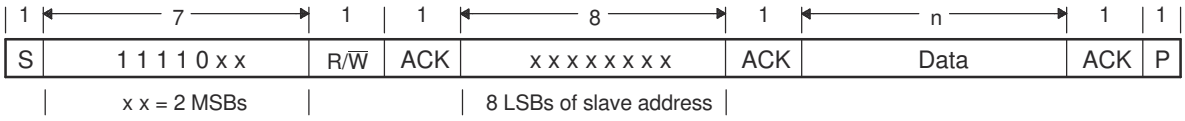


Figure 26-11. I2C Module 10-Bit Addressing Format (FDF = 0, XA = 1 in I2CMDR)

26.3.6.3 Free Data Format

The free data format can be enabled by setting I2CMDR. FDF = 1.

In this format (see Figure 26-12), the first byte after a START condition (S) is a data byte. An ACK bit is inserted after each data byte, which can be from 1 to 8 bits, depending on the BC field of I2CMDR. No address or data-direction bit is sent. Therefore, the transmitter and the receiver must both support the free data format, and the direction of the data must be constant throughout the transfer.

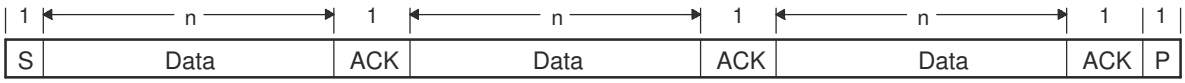


Figure 26-12. I2C Module Free Data Format (FDF = 1 in I2CMDR)

Note

The free data format is not supported in the digital loopback mode (I2CMDR.DLB = 1).

Table 26-4. How the MST and FDF Bits of I2CMDR Affect the Role of the TRX Bit of I2CMDR

MST	FDF	I2C Module State	Function of TRX
0	0	In slave mode but not free data format mode	TRX is a don't care. Depending on the command from the master, the I2C module responds as a receiver or a transmitter.
0	1	In slave mode and free data format mode	The free data format mode requires that the I2C module remains the transmitter or the receiver throughout the transfer. TRX identifies the role of the I2C module: TRX = 1: The I2C module is a transmitter. TRX = 0: The I2C module is a receiver.
1	0	In master mode but not free data format mode	TRX = 1: The I2C module is a transmitter. TRX = 0: The I2C module is a receiver.
1	1	In master mode and free data format mode	TRX = 0: The I2C module is a receiver. TRX = 1: The I2C module is a transmitter.

26.3.6.4 Using a Repeated START Condition

I2C master can communicate with multiple slave addresses without having to give up control of the I2C bus by driving a STOP condition. This can be achieved by driving another START condition at the end of each data type. The repeated START condition can be used with the 7-bit addressing, 10-bit addressing, and free data formats. Figure 26-13 shows a repeated START condition in the 7-bit addressing format.

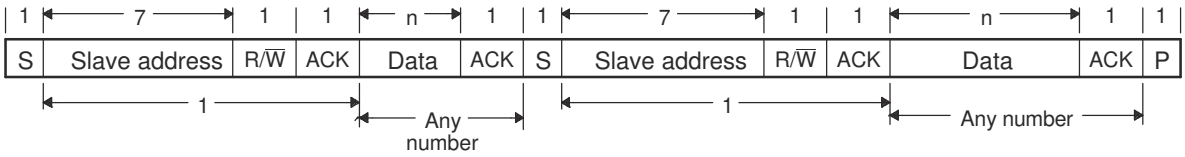


Figure 26-13. Repeated START Condition (in This Case, 7-Bit Addressing Format)

Note

In Figure 26-13, n = the number of data bits (from 1 to 8) specified by the bit count (BC) field of I2CMDR.

26.3.7 Clock Synchronization

Under normal conditions, only one master device generates the clock signal, SCL. During the arbitration procedure, however, there are two or more masters and the clock must be synchronized so that the data output can be compared. Figure 26-14 illustrates the clock synchronization. The wired-AND property of SCL means that a device that first generates a low period on SCL overrules the other devices. At this high-to-low transition, the clock generators of the other devices are forced to start their own low period. The SCL is held low by the device with the longest low period. The other devices that finish their low periods must wait for SCL to be released, before starting their high periods. A synchronized signal on SCL is obtained, where the slowest device determines the length of the low period and the fastest device determines the length of the high period.

If a device pulls down the clock line for a longer time, the result is that all clock generators must enter the wait state. In this way, a slave slows down a fast master and the slow device creates enough time to store a received byte or to prepare a byte to be transmitted.

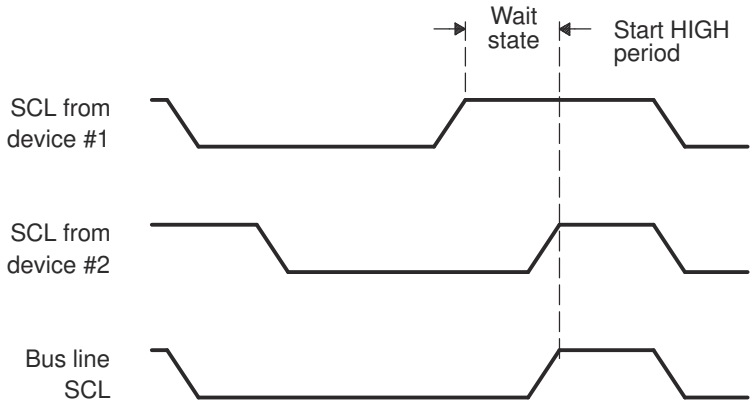


Figure 26-14. Synchronization of Two I2C Clock Generators During Arbitration

26.3.8 Arbitration

If two or more master-transmitters attempt to start a transmission on the same bus at approximately the same time, an arbitration procedure is invoked. The arbitration procedure uses the data presented on the serial data bus (SDA) by the competing transmitters. Figure 26-15 illustrates the arbitration procedure between two devices. The first master-transmitter that releases the SDA line high is overruled by another master-transmitter that drives the SDA low. The arbitration procedure gives priority to the device that transmits the serial data stream with the lowest binary value. Should two or more devices send identical first bytes, arbitration continues on the subsequent bytes.

If the I2C module is the losing master, it switches to the slave-receiver mode, sets the arbitration lost (ARBL) flag, and generates the arbitration-lost interrupt request.

If during a serial transfer the arbitration procedure is still in progress when a repeated START condition or a STOP condition is transmitted to SDA, the master-transmitters involved must send the repeated START condition or the STOP condition at the same position in the format frame. Arbitration is not allowed between:

- A repeated START condition and a data bit
- A STOP condition and a data bit
- A repeated START condition and a STOP condition

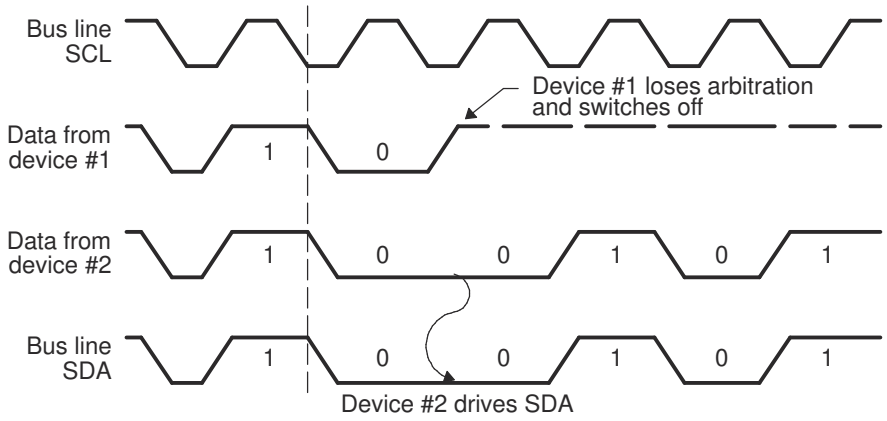


Figure 26-15. Arbitration Procedure Between Two Master-Transmitters

26.3.9 Digital Loopback Mode

The I2C module support a self-test mode called digital loopback, which is enabled by setting the DLB bit in the I2CMDR register. In this mode, data transmitted out of the I2CDXR register is received in the I2CDRR register. The data follows an internal path, and takes n cycles to reach I2CDRR, where:

n = 8 * (SYSCLK) / (I2C module clock (Fmod))

The transmit clock and the receive clock are the same. The address seen on the external SDA pin is the address in the I2COAR register. Figure 26-16 shows the signal routing in digital loopback mode.

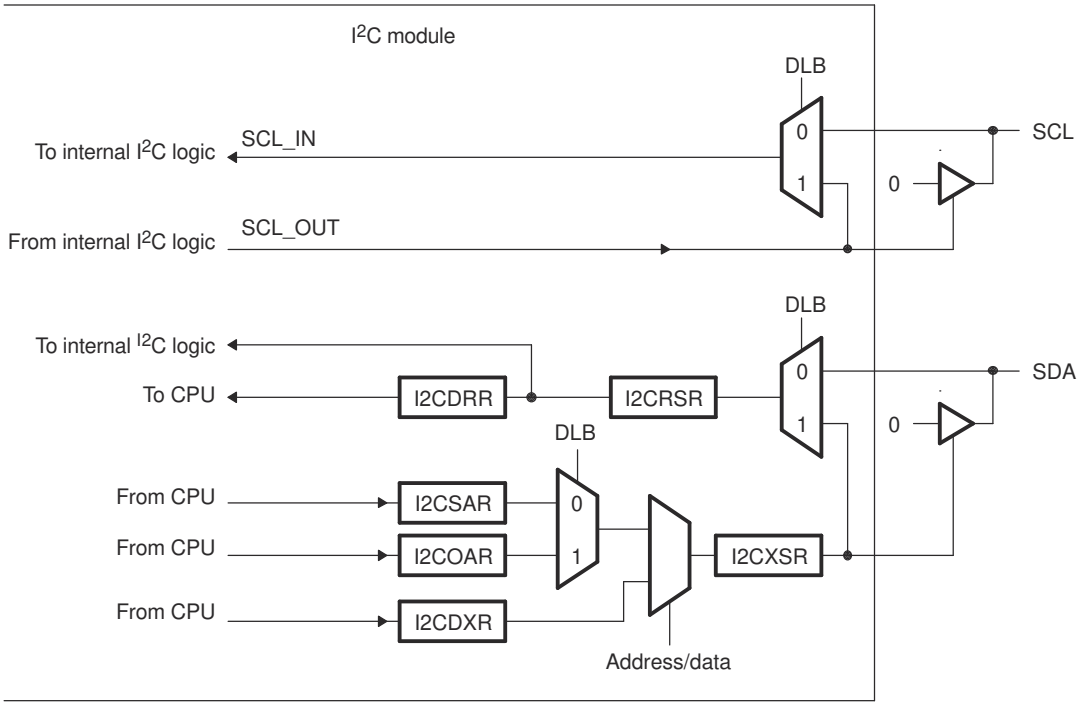


Figure 26-16. Pin Diagram Showing the Effects of the Digital Loopback Mode (DLB) Bit

Note

The free data format (I2CMDR.FDF = 1) is not supported in digital loopback mode.

26.3.10 NACK Bit Generation

When the I2C module is a receiver (master or slave), it can acknowledge or ignore bits sent by the transmitter. To ignore any new bits, the I2C module must send a no-acknowledge (NACK) bit during the acknowledge cycle on the bus. Table 26-5 summarizes the various ways you can tell the I2C module to send a NACK bit.

Table 26-5. Ways to Generate a NACK Bit

I2C Module Condition	NACK Bit Generation Options
Slave-receiver modes	• Allow an overrun condition (RSFULL = 1 in I2CSTR) • Reset the module (IRS = 0 in I2CMDR) • Set the NACKMOD bit of I2CMDR before the rising edge of the last data bit you intend to receive
Master-receiver mode AND Repeat mode (RM = 1 in I2CMDR)	• Generate a STOP condition (STP = 1 in I2CMDR) • Reset the module (IRS = 0 in I2CMDR) • Set the NACKMOD bit of I2CMDR before the rising edge of the last data bit you intend to receive
Master-receiver mode AND Nonrepeat mode (RM = 0 in I2CMDR)	• If STP = 1 in I2CMDR, allow the internal data counter to count down to 0 and thus force a STOP condition • If STP = 0, make STP = 1 to generate a STOP condition • Reset the module (IRS = 0 in I2CMDR). = 1 to generate a STOP condition • Set the NACKMOD bit of I2CMDR before the rising edge of the last data bit you intend to receive

26.4 Interrupt Requests Generated by the I2C Module

Each I2C module can generate two CPU interrupts.

1. Basic I2C interrupt: Possible basic I2C interrupt sources that can trigger this interrupt are described in Section 26.4.1.
2. I2C FIFO interrupt: Possible I2C FIFO interrupt sources that can trigger this interrupt are described in Section 26.4.2

26.4.1 Basic I2C Interrupt Requests

The I2C module generates the interrupt requests described in Table 26-6. As shown in Figure 26-17, all requests are multiplexed through an arbiter to a single I2C interrupt request to the CPU. Each interrupt request has a flag bit in the status register (I2CSTR) and an enable bit in the interrupt enable register (I2CIER). When one of the specified events occurs, its flag bit is set. If the corresponding enable bit is 0, the interrupt request is blocked. If the enable bit is 1, the request is forwarded to the CPU as an I2C interrupt.

The I2C interrupt is one of the maskable interrupts of the CPU. As with any maskable interrupt request, if it is properly enabled in the CPU, the CPU executes the corresponding interrupt service routine (I2CINT1A_ISR). The I2CINT1A_ISR for the I2C interrupt can determine the interrupt source by reading the interrupt source register, I2CISRC. Then the I2CINT1A_ISR can branch to the appropriate subroutine.

After the CPU reads I2CISRC, the following events occur:

1. The flag for the source interrupt is cleared in I2CSTR. Exception: The ARDY, RRDY, and XRDY bits in I2CSTR are not cleared when I2CISRC is read. To clear one of these bits, write a 1 to it.
2. The arbiter determines which of the remaining interrupt requests has the highest priority, writes the code for that interrupt to I2CISRC, and forwards the interrupt request to the CPU.

Table 26-6. Descriptions of the Basic I2C Interrupt Requests

I2C Interrupt Request	Interrupt Source
XRDYINT	Transmit ready condition: The data transmit register (I2CDXR) is ready to accept new data because the previous data has been copied from I2CDXR to the transmit shift register (I2CXSR). As an alternative to using XRDYINT, the CPU can poll the XRDY bit of the status register, I2CSTR. XRDYINT should not be used when in FIFO mode. Use the FIFO interrupts instead.
RRDYINT	Receive ready condition: The data receive register (I2CDRR) is ready to be read because data has been copied from the receive shift register (I2CRSR) to I2CDRR. As an alternative to using RRDYINT, the CPU can poll the RRDY bit of I2CSTR. RRDYINT should not be used when in FIFO mode. Use the FIFO interrupts instead.
ARDYINT	Register-access ready condition: The I2C module registers are ready to be accessed because the previously programmed address, data, and command values have been used. The specific events that generate ARDYINT are the same events that set the ARDY bit of I2CSTR. As an alternative to using ARDYINT, the CPU can poll the ARDY bit.
NACKINT	No-acknowledgment condition: The I2C module is configured as a master-transmitter and did not received acknowledgment from the slave-receiver. As an alternative to using NACKINT, the CPU can poll the NACK bit of I2CSTR.
ARBLINT	Arbitration-lost condition: The I2C module has lost an arbitration contest with another master-transmitter. As an alternative to using ARBLINT, the CPU can poll the ARBL bit of I2CSTR.
SCDINT	Stop condition detected: A STOP condition was detected on the I2C bus. As an alternative to using SCDINT, the CPU can poll the SCD bit of the status register, I2CSTR.
AASINT	Addressed as slave condition: The I2C has been addressed as a slave device by another master on the I2C bus. As an alternative to using AASINT, the CPU can poll the AAS bit of the status register, I2CSTR.

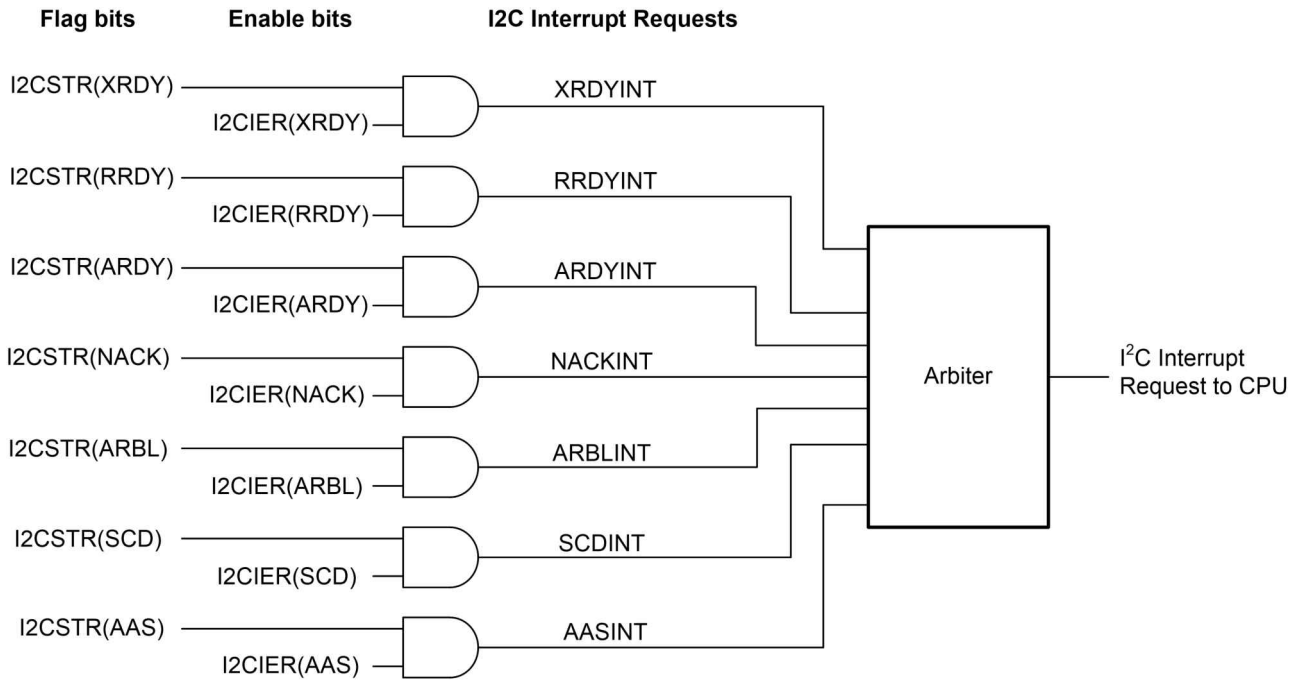


Figure 26-17. Enable Paths of the I2C Interrupt Requests

The priorities of the basic I2C interrupt requests are listed in order of highest priority to lowest priority:

1. ARBLINT
2. NACKINT
3. ARDYINT
4. RRDYINT
5. XRDYINT
6. SCDINT
7. AASINT

The normal transmit interrupt timing makes it possible for stale data to remain in the transmit buffer if a transaction is aborted in the middle of a byte. To avoid this, set the FCM bit in the I2CEMDR register. When this bit is set, the transmit data ready interrupt is generated only when data is required for a bus transaction. In master mode, the interrupt is first generated when the ACK of the address byte is received. In slave mode, the interrupt is first generated when the address is matched. Further interrupts are generated when the data is ACKed. In this mode XRDY is asserted at the same time as the transmit ready interrupt.

The I2C module has a backwards compatibility bit (BC) in the I2CEMDR register. The timing diagram in Figure 26-18 demonstrates the effect the backwards compatibility bit has on I2C module registers and interrupts when configured as a slave-transmitter.

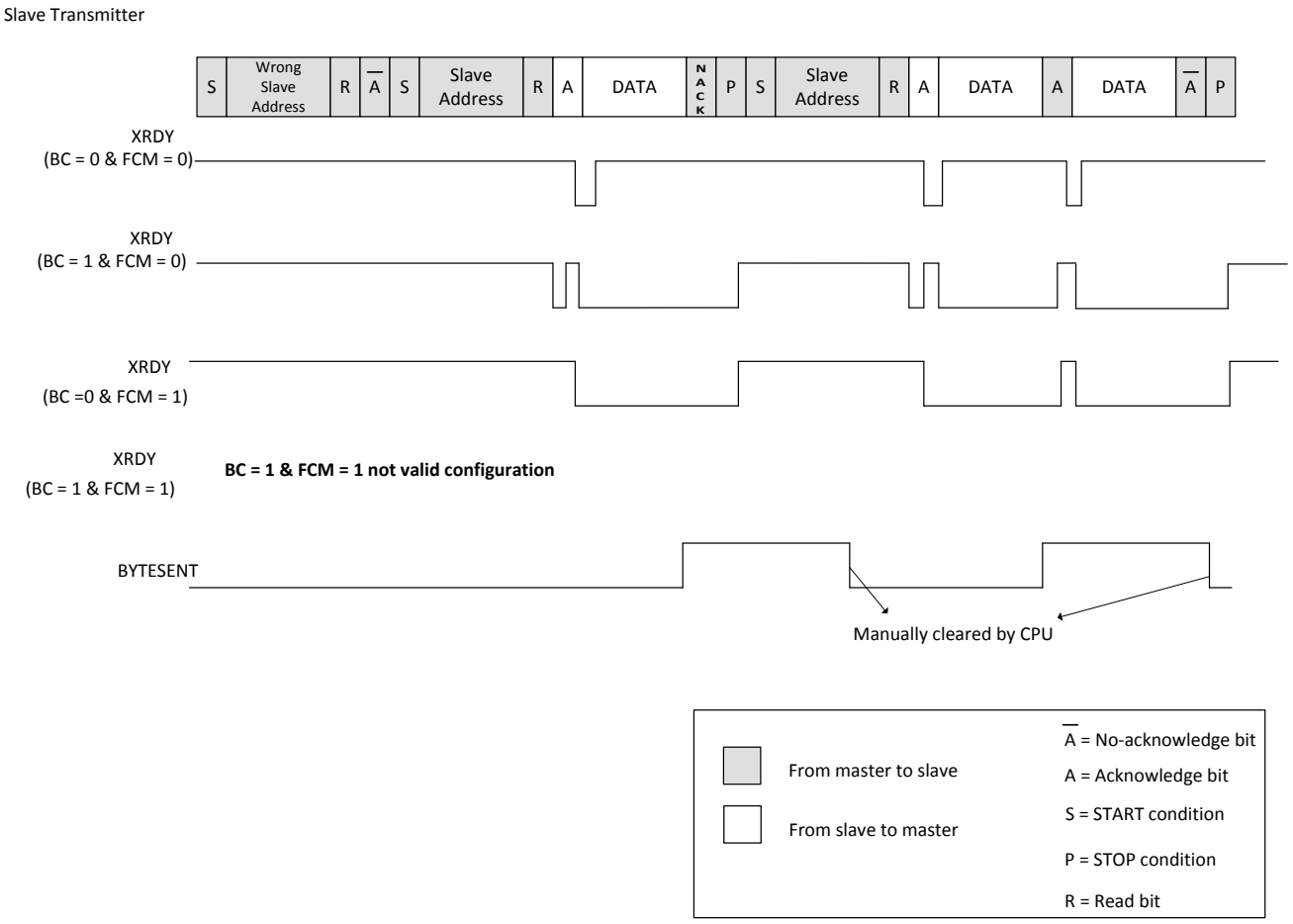


Figure 26-18. Backwards Compatibility Mode and Forward Compactibility bit, Slave Transmitter

26.4.2 I2C FIFO Interrupts

In addition to the seven basic I2C interrupts, the transmit and receive FIFOs each contain the ability to generate an interrupt (I2CINT2A). The transmit FIFO can be configured to generate an interrupt after transmitting a defined number of bytes, up to 16. The receive FIFO can be configured to generate an interrupt after receiving a defined number of bytes, up to 16. These two interrupt sources are ORed together into a single maskable CPU interrupt. Figure 26-19 shows the structure of I2C FIFO interrupt. The interrupt service routine can then read the FIFO interrupt status flags to determine from which source the interrupt came. See the I2C transmit FIFO register (I2CFFTX) and the I2C receive FIFO register (I2CFFRX) descriptions.

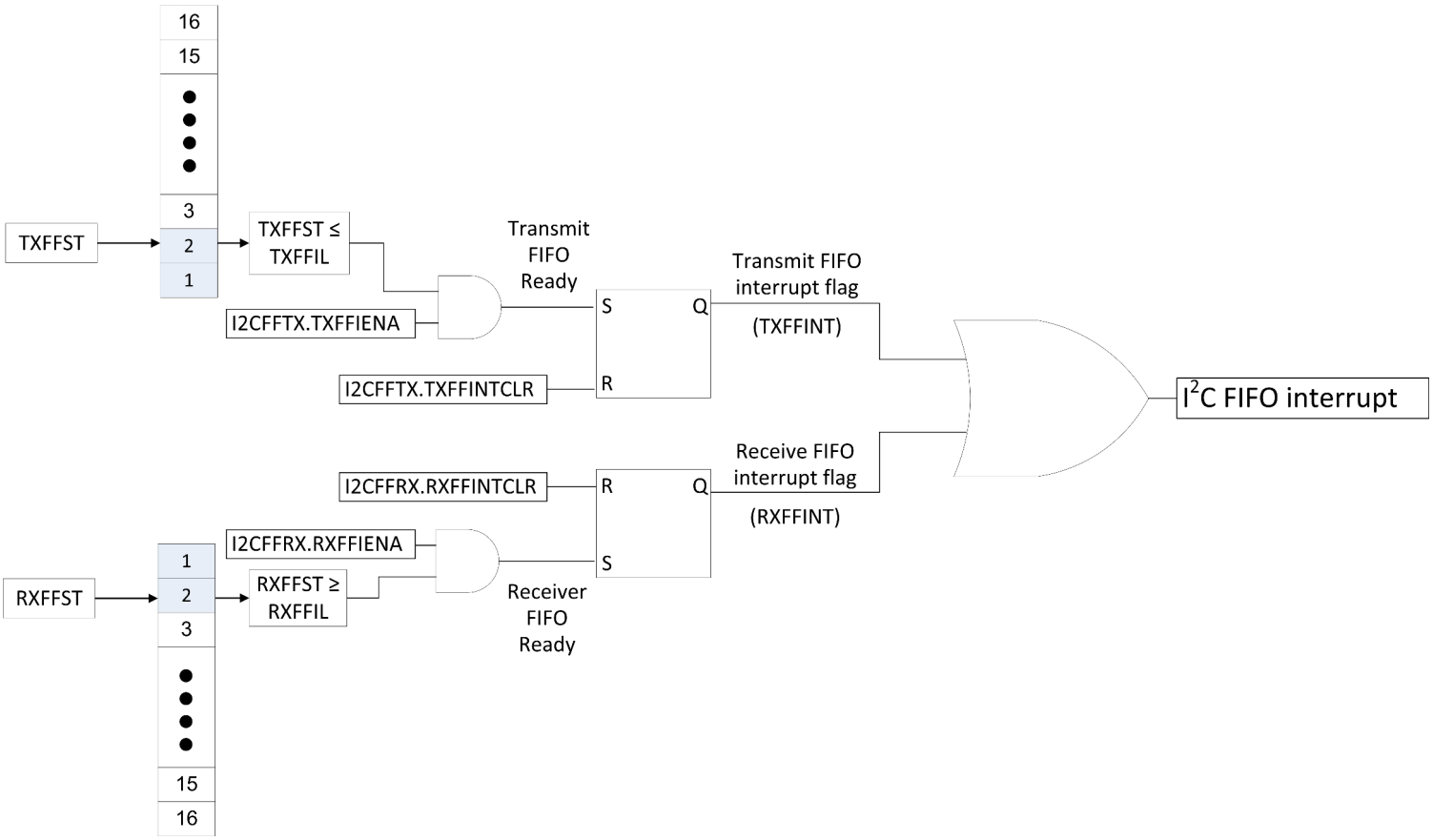


Figure 26-19. I2C_FIFO_interrupt

26.5 Resetting or Disabling the I2C Module

You can reset or disable the I2C module in two ways:

- Write 0 to the I2C reset bit (IRS) in the I2C mode register (I2CMDR). All status bits (in I2CSTR) are forced to their default values, and the I2C module remains disabled until IRS is changed to 1. The SDA and SCL pins are in the high-impedance state.
- Initiate a device reset by driving the $\overline{\text{XRS}}$ pin low. The entire device is reset and is held in the reset state until you drive the pin high. When the $\overline{\text{XRS}}$ pin is released, all I2C module registers are reset to their default values. The IRS bit is forced to 0, which resets the I2C module. The I2C module stays in the reset state until you write 1 to IRS.

The IRS must be 0 while you configure or reconfigure the I2C module. Forcing IRS to 0 can be used to save power and to clear error conditions.

26.6 Software

26.6.1 I2C Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/f28003x/examples/i2c

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

26.6.1.1 I2C Digital Loopback with FIFO Interrupts

FILE: i2c_ex1_loopback.c

This program uses the internal loopback test mode of the I2C module. Both the TX and RX I2C FIFOs and their interrupts are used. The pinmux and I2C initialization is done through the sysconfig file.

A stream of data is sent and then compared to the received stream. The sent data looks like this:
0000 0001
0001 0002
0002 0003
....
00FE 00FF
00FF 0000
etc..
This pattern is repeated forever.

External Connections

- None

Watch Variables

- *sData* - Data to send
- *rData* - Received data
- *rDataPoint* - Used to keep track of the last position in the receive stream for error checking

26.6.1.2 I2C EEPROM

FILE: i2c_ex2_eeprom.c

This program will write 1-14 words to EEPROM and read them back. The data written and the EEPROM address written to are contained in the message structure, i2cMsgOut. The data read back will be contained in the message structure i2cMsgIn.

External Connections

- Connect external I2C EEPROM at address 0x50
- Connect DEVICE_GPIO_PIN_SDAA on to external EEPROM SDA (serial data) pin
- Connect DEVICE_GPIO_PIN_SCLA on to external EEPROM SCL (serial clock) pin

Watch Variables

- *i2cMsgOut* - Message containing data to write to EEPROM
- *i2cMsgIn* - Message containing data read from EEPROM

26.6.1.3 I2C Digital External Loopback with FIFO Interrupts

FILE: i2c_ex3_external_loopback.c

This program uses the I2CA and I2CB modules for achieving external loopback. The I2CA TX FIFO and the I2CB RX FIFO are used along with their interrupts.

A stream of data is sent on I2CA and then compared to the received stream on I2CB. The sent data looks like this:
0000 0001
0001 0002
0002 0003
....
00FE 00FF
00FF 0000
etc..
This pattern is repeated forever.

External Connections

- Connect SCLA(DEVICE_GPIO_PIN_SCLA) to SCLB (DEVICE_GPIO_PIN_SCLB)
- Connect SDAA(DEVICE_GPIO_PIN_SDAA) to SDAB (DEVICE_GPIO_PIN_SDAB)
- Connect DEVICE_GPIO_PIN_LED1 to an LED used to depict data transfers.

Watch Variables

- *sData* - Data to send
- *rData* - Received data
- *rDataPoint* - Used to keep track of the last position in the receive stream for error checking

26.6.1.4 I2C EEPROM

FILE: i2c_ex4_eeprom_polling.c

This program will shows how to perform different EEPROM write and read commands using I2C polling method EEPROM used for this example is AT24C256

External Connections

- Connect external I2C EEPROM at address 0x50 Signal | I2CA | EEPROM

SCL | DEVICE_GPIO_PIN_SCLA | SCL SDA | DEVICE_GPIO_PIN_SDAA | SDA

Make sure to connect GND pins if EEPROM and C2000 device are in different board.

26.6.1.5 I2C Master Slave Communication using FIFO Interrupts

FILE: i2c_ex5_master_slave_interrupt.c

This program shows how to use I2CA and I2CB modules in both master and slave configuration This example uses I2C FIFO interrupts and doesn't using polling

Example1: I2CA as Master Transmitter and I2CB working Slave Receiver Example2: I2CA as Master Receiver and I2CB working Slave Transmitter Example3: I2CB as Master Transmitter and I2CA working Slave Receiver Example4: I2CB as Master Receiver and I2CA working Slave Transmitter

External Connections on launchpad should be made as:

- Signal | I2CA | I2CB
- SCL | DEVICE_GPIO_PIN_SCLA | DEVICE_GPIO_PIN_SCLB
- SDA | DEVICE_GPIO_PIN_SDAA | DEVICE_GPIO_PIN_SDAB

Watch Variables in memory window

- *I2CA_TXdata*
- *I2CA_RXdata*
- *I2CB_TXdata*
- *I2CB_RXdata* stream for error checking

26.6.1.6 I2C EEPROM

FILE: i2c_ex6_eeprom_interrupt.c

This program will shows how to perform different EEPROM write and read commands using I2C interrupts EEPROM used for this example is AT24C256

External Connections

- Connect external I2C EEPROM at address 0x50 Signal | I2CA | EEPROM

SCL | DEVICE_GPIO_PIN_SCLA | SCL SDA | DEVICE_GPIO_PIN_SDAA | SDA

Make sure to connect GND pins, if EEPROM and C2000 device are on different boards.

//Example 1: EEPROM Byte Write

//Example 2: EEPROM Byte Read

//Example 3: EEPROM word (16-bit) write

//Example 4: EEPROM word (16-bit) read

//Example 5: EEPROM Page write

//Example 6: EEPROM word Paged read

Watch Variables

- *TX_MsgBuffer* - Message buffer that stores the data to be transmitted
- *RX_MsgBuffer* - Message buffer that stores the data to be received

26.7 I2C Registers

This section describes the C28x I2C Module Registers.

26.7.1 I2C Base Address Table

Table 26-7. I2C Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
I2caRegs	I2C_REGS	I2CA_BASE	0x0000_7300	YES	-	YES	-	YES
I2cbRegs	I2C_REGS	I2CB_BASE	0x0000_7340	YES	-	YES	-	YES

26.7.2 I2C_REGS Registers

Table 26-8 lists the memory-mapped registers for the I2C_REGS registers. All register offset addresses not listed in Table 26-8 should be considered as reserved locations and the register contents should not be modified.

Table 26-8. I2C_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	I2COAR	I2C Own address		Go
1h	I2CIER	I2C Interrupt Enable		Go
2h	I2CSTR	I2C Status		Go
3h	I2CCLKL	I2C Clock low-time divider		Go
4h	I2CCLKH	I2C Clock high-time divider		Go
5h	I2CCNT	I2C Data count		Go
6h	I2CDRR	I2C Data receive		Go
7h	I2CSAR	I2C Slave address		Go
8h	I2CDXR	I2C Data Transmit		Go
9h	I2CMDR	I2C Mode		Go
Ah	I2CISRC	I2C Interrupt Source		Go
Bh	I2CEMDR	I2C Extended Mode		Go
Ch	I2CPSC	I2C Prescaler		Go
20h	I2CFFTX	I2C FIFO Transmit		Go
21h	I2CFFRX	I2C FIFO Receive		Go

Complex bit access types are encoded to fit into small table cells. Table 26-9 shows the codes that are used for access types in this section.

Table 26-9. I2C_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.

Table 26-9. I2C_REGS Access Type Codes (continued)

Access Type	Code	Description
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

26.7.2.1 I2COAR Register (Offset = 0h) [Reset = 0h]

I2COAR is shown in Figure 26-20 and described in Table 26-10.

Return to the Summary Table.

The I2C own address register (I2COAR) is a 16-bit register. The I2C module uses this register to specify its own slave address, which distinguishes it from other slaves connected to the I2C-bus. If the 7-bit addressing mode is selected (XA = 0 in I2CMDR), only bits 6-0 are used
write 0s to bits 9-7.

Figure 26-20. I2COAR Register

15	14	13	12	11	10	9	8
RESERVED						OAR	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
OAR							
R/W-0h							

Table 26-10. I2COAR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	OAR	R/W	0h	In 7-bit addressing mode (XA = 0 in I2CMDR): 00h-7Fh Bits 6-0 provide the 7-bit slave address of the I2C module. Write 0s to bits 9-7. In 10-bit addressing mode (XA = 1 in I2CMDR): 000h-3FFh Bits 9-0 provide the 10-bit slave address of the I2C module. Reset type: SYSRSn

26.7.2.2 I2CIER Register (Offset = 1h) [Reset = 0h]

I2CIER is shown in Figure 26-21 and described in Table 26-11.

Return to the Summary Table.

I2CIER is used by the CPU to individually enable or disable I2C interrupt requests.

Figure 26-21. I2CIER Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	AAS	SCD	XRDY	RRDY	ARDY	NACK	ARBL
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 26-11. I2CIER Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R	0h	Reserved
6	AAS	R/W	0h	Addressed as slave interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt request disabled 1h (R/W) = Interrupt request enabled
5	SCD	R/W	0h	Stop condition detected interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt request disabled 1h (R/W) = Interrupt request enabled
4	XRDY	R/W	0h	Transmit-data-ready interrupt enable bit. This bit should not be set when using FIFO mode. Reset type: SYSRSn 0h (R/W) = Interrupt request disabled 1h (R/W) = Interrupt request enabled
3	RRDY	R/W	0h	Receive-data-ready interrupt enable bit. This bit should not be set when using FIFO mode. Reset type: SYSRSn 0h (R/W) = Interrupt request disabled 1h (R/W) = Interrupt request enabled
2	ARDY	R/W	0h	Register-access-ready interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt request disabled 1h (R/W) = Interrupt request enabled
1	NACK	R/W	0h	No-acknowledgment interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt request disabled 1h (R/W) = Interrupt request enabled
0	ARBL	R/W	0h	Arbitration-lost interrupt enable Reset type: SYSRSn 0h (R/W) = Interrupt request disabled 1h (R/W) = Interrupt request enabled

26.7.2.3 I2CSTR Register (Offset = 2h) [Reset = 410h]

I2CSTR is shown in Figure 26-22 and described in Table 26-12.

Return to the Summary Table.

The I2C status register (I2CSTR) is a 16-bit register used to determine which interrupt has occurred and to read status information.

Figure 26-22. I2CSTR Register

15	14	13	12	11	10	9	8
RESERVED	SDIR	NACKSNT	BB	RSFULL	XSMT	AAS	AD0
R-0h	R/W1C-0h	R/W1C-0h	R-0h	R-0h	R-1h	R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED	BYTESENT	SCD	XRDY	RRDY	ARDY	NACK	ARBL
R-0h	R/W1C-0h	R/W1C-0h	R-1h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

Table 26-12. I2CSTR Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	SDIR	R/W1C	0h	Slave direction bit Reset type: SYSRSn 0h (R/W) = I2C is not addressed as a slave transmitter. SDIR is cleared by one of the following events: - It is manually cleared. To clear this bit, write a 1 to it. - Digital loopback mode is enabled. - A START or STOP condition occurs on the I2C bus. 1h (R/W) = I2C is addressed as a slave transmitter.
13	NACKSNT	R/W1C	0h	NACK sent bit. This bit is used when the I2C module is in the receiver mode. One instance in which NACKSNT is affected is when the NACK mode is used (see the description for NACKMOD in Reset type: SYSRSn 0h (R/W) = NACK not sent. NACKSNT bit is cleared by any one of the following events: - It is manually cleared. To clear this bit, write a 1 to it. - The I2C module is reset (either when 0 is written to the IRS bit of I2CMDR or when the whole device is reset). 1h (R/W) = NACK sent: A no-acknowledge bit was sent during the acknowledge cycle on the I2C-bus.
12	BB	R	0h	Bus busy bit. BB indicates whether the I2C-bus is busy or is free for another data transfer. See the paragraph following the table for more information Reset type: SYSRSn 0h (R/W) = Bus free. BB is cleared by any one of the following events: - The I2C module receives or transmits a STOP bit (bus free). - The I2C module is reset. 1h (R/W) = Bus busy: The I2C module has received or transmitted a START bit on the bus.

Table 26-12. I2CSTR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	RSFULL	R	0h	Receive shift register full bit. RSFULL indicates an overrun condition during reception. Overrun occurs when new data is received into the shift register (I2CRSR) and the old data has not been read from the receive register (I2CDRR). As new bits arrive from the SDA pin, they overwrite the bits in I2CRSR. The new data will not be copied to ICDRR until the previous data is read. Reset type: SYSRSn 0h (R/W) = No overrun detected. RSFULL is cleared by any one of the following events: - I2CDRR is read is read by the CPU. Emulator reads of the I2CDRR do not affect this bit. - The I2C module is reset. 1h (R/W) = Overrun detected
10	XSMT	R	1h	Transmit shift register empty bit. XSMT = 0 indicates that the transmitter has experienced underflow. Underflow occurs when the transmit shift register (I2CXSR) is empty but the data transmit register (I2CDXR) has not been loaded since the last I2CDXR-to-I2CXSR transfer. The next I2CDXR-to-I2CXSR transfer will not occur until new data is in I2CDXR. If new data is not transferred in time, the previous data may be re-transmitted on the SDA pin. Reset type: SYSRSn 0h (R/W) = Underflow detected (empty) 1h (R/W) = No underflow detected (not empty). XSMT is set by one of the following events: - Data is written to I2CDXR. - The I2C module is reset
9	AAS	R	0h	Addressed-as-slave bit Reset type: SYSRSn 0h (R/W) = In the 7-bit addressing mode, the AAS bit is cleared when receiving a NACK, a STOP condition, or a repeated START condition. In the 10-bit addressing mode, the AAS bit is cleared when receiving a NACK, a STOP condition, or by a slave address different from the I2C peripheral's own slave address. 1h (R/W) = The I2C module has recognized its own slave address or an address of all zeros (general call).
8	AD0	R	0h	Address 0 bits Reset type: SYSRSn 0h (R/W) = AD0 has been cleared by a START or STOP condition. 1h (R/W) = An address of all zeros (general call) is detected.
7	RESERVED	R	0h	Reserved
6	BYTESENT	R/W1C	0h	Byte Transmit over indication. BYTESENT is set when the master/slave has successfully sent the byte on SCL/SDA lines. This is diagnostic register which needs to be explicitly cleared by Software. In case not cleared the stale status would keep reflecting as no automated clear incorporated to avoid corner conditions. Reset type: SYSRSn 0h (R/W) = The I2C module has not finished transmitting the next data byte. BYTESENT is cleared by any one of the following events: - It is manually cleared. To clear this bit, write a 1 to it. - The I2C module is reset. 1h (R/W) = The I2C module has completed the transmission of a byte.

Table 26-12. I2CSTR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5	SCD	R/W1C	0h	Stop condition detected bit. SCD is set when the I2C sends or receives a STOP condition. The I2C module delays clearing of the I2CMDR[STP] bit until the SCD bit is set. Reset type: SYSRSn 0h (R/W) = STOP condition not detected since SCD was last cleared. SCD is cleared by any one of the following events: - I2CISRC is read by the CPU when it contains the value 110b (stop condition detected). Emulator reads of the I2CISRC do not affect this bit. - SCD is manually cleared. To clear this bit, write a 1 to it. - The I2C module is reset. 1h (R/W) = A STOP condition has been detected on the I2C bus.
4	XRDY	R	1h	Transmit-data-ready interrupt flag bit. When not in FIFO mode, XRDY indicates that the data transmit register (I2CDXR) is ready to accept new data. FCM=0 : When the previous data has been copied from I2CDXR to the transmit shift register (I2CXSR). The CPU can poll XRDY or use the XRDY interrupt request When in FIFO mode, use TXFFINT instead. FCM=1: XRDY is asserted only when next data is required it gets de asserted with write to I2CDXR. Both Polling and interrupt based data transfers are allowed in the FCM mode. Reset type: SYSRSn 0h (R/W) = I2CDXR not ready. XRDY is cleared when data is written to I2CDXR. 1h (R/W) = I2CDXR ready: Data has been copied from I2CDXR to I2CXSR. XRDY is also forced to 1 when the I2C module is reset.
3	RRDY	R/W1C	0h	Receive-data-ready interrupt flag bit. When not in FIFO mode, RRDY indicates that the data receive register (I2CDRR) is ready to be read because data has been copied from the receive shift register (I2CRSR) to I2CDRR. The CPU can poll RRDY or use the RRDY interrupt request When in FIFO mode, use RXFFINT instead. Reset type: SYSRSn 0h (R/W) = I2CDRR not ready. RRDY is cleared by any one of the following events: - I2CDRR is read by the CPU. Emulator reads of the I2CDRR do not affect this bit. - RRDY is manually cleared. To clear this bit, write a 1 to it. - The I2C module is reset. 1h (R/W) = I2CDRR ready: Data has been copied from I2CRSR to I2CDRR.
2	ARDY	R/W1C	0h	Register-access-ready interrupt flag bit (only Applicable when the I2C module is in the master mode). ARDY indicates that the I2C module registers are ready to be accessed because the previously programmed address, data, and command values have been used. The CPU can poll ARDY or use the ARDY interrupt request Reset type: SYSRSn 0h (R/W) = The registers are not ready to be accessed. ARDY is cleared by any one of the following events: - The I2C module starts using the current register contents. - ARDY is manually cleared. To clear this bit, write a 1 to it. - The I2C module is reset. 1h (R/W) = The registers are ready to be accessed. In the nonrepeat mode (RM = 0 in I2CMDR): If STP = 0 in I2CMDR, the ARDY bit is set when the internal data counter counts down to 0. If STP = 1, ARDY is not affected (instead, the I2C module generates a STOP condition when the counter reaches 0). In the repeat mode (RM = 1): ARDY is set at the end of each byte transmitted from I2CDXR.

Table 26-12. I2CSTR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	NACK	R/W1C	0h	No-acknowledgment interrupt flag bit. NACK applies when the I2C module is a master transmitter. NACK indicates whether the I2C module has detected an acknowledge bit (ACK) or a noacknowledge bit (NACK) from the slave receiver. The CPU can poll NACK or use the NACK interrupt request. Reset type: SYSRSn 0h (R/W) = ACK received/NACK not received. This bit is cleared by any one of the following events: - An acknowledge bit (ACK) has been sent by the slave receiver. - NACK is manually cleared. To clear this bit, write a 1 to it. - The CPU reads the interrupt source register (I2CISRC) and the register contains the code for a NACK interrupt. Emulator reads of the I2CISRC do not affect this bit. - The I2C module is reset. 1h (R/W) = NACK bit received. The hardware detects that a no-acknowledge (NACK) bit has been received. Note: While the I2C module performs a general call transfer, NACK is 1, even if one or more slaves send acknowledgment.
0	ARBL	R/W1C	0h	Arbitration-lost interrupt flag bit (only applicable when the I2C module is a master-transmitter). ARBL primarily indicates when the I2C module has lost an arbitration contest with another mastertransmitter. The CPU can poll ARBL or use the ARBL interrupt request. Reset type: SYSRSn 0h (R/W) = Arbitration not lost. AL is cleared by any one of the following events: - AL is manually cleared. To clear this bit, write a 1 to it. - The CPU reads the interrupt source register (I2CISRC) and the register contains the code for an AL interrupt. Emulator reads of the I2CISRC do not affect this bit. - The I2C module is reset. 1h (R/W) = Arbitration lost. AL is set by any one of the following events: - The I2C module senses that it has lost an arbitration with two or more competing transmitters that started a transmission almost simultaneously. - The I2C module attempts to start a transfer while the BB (bus busy) bit is set to 1. When AL becomes 1, the MST and STP bits of I2CMDR are cleared, and the I2C module becomes a slave-receiver.

26.7.2.4 I2CCLKL Register (Offset = 3h) [Reset = 0h]

I2CCLKL is shown in Figure 26-23 and described in Table 26-13.

Return to the Summary Table.

I2C Clock low-time divider

Figure 26-23. I2CCLKL Register

15	14	13	12	11	10	9	8
I2CCLKL							
R/W-0h							
7	6	5	4	3	2	1	0
I2CCLKL							
R/W-0h							

Table 26-13. I2CCLKL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	I2CCLKL	R/W	0h	Clock low-time divide-down value. To produce the low time duration of the master clock, the period of the module clock is multiplied by (ICCL + d). d is an adjustment factor based on the prescaler. See the Clock Divider Registers section of the Introduction for details. Note: These bits must be set to a non-zero value for proper I2C clock generation. Reset type: SYSRSn

26.7.2.5 I2CCLKH Register (Offset = 4h) [Reset = 0h]

I2CCLKH is shown in Figure 26-24 and described in Table 26-14.

Return to the Summary Table.

I2C Clock high-time divider

Figure 26-24. I2CCLKH Register

15	14	13	12	11	10	9	8
I2CCLKH							
R/W-0h							
7	6	5	4	3	2	1	0
I2CCLKH							
R/W-0h							

Table 26-14. I2CCLKH Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	I2CCLKH	R/W	0h	Clock high-time divide-down value. To produce the high time duration of the master clock, the period of the module clock is multiplied by (ICCL + d). d is an adjustment factor based on the prescaler. See the Clock Divider Registers section of the Introduction for details. Note: These bits must be set to a non-zero value for proper I2C clock generation. Reset type: SYSRSn

26.7.2.6 I2CCNT Register (Offset = 5h) [Reset = 0h]

I2CCNT is shown in Figure 26-25 and described in Table 26-15.

Return to the Summary Table.

I2CCNT is a 16-bit register used to indicate how many data bytes to transfer when the I2C module is configured as a transmitter, or to receive when configured as a master receiver. In the repeat mode (RM = 1), I2CCNT is not used.
The value written to I2CCNT is copied to an internal data counter. The internal data counter is decremented by 1 for each byte transferred (I2CCNT remains unchanged). If a STOP condition is requested in the master mode (STP = 1 in I2CMDR), the I2C module terminates the transfer with a STOP condition when the countdown is complete (that is, when the last byte has been transferred).

Figure 26-25. I2CCNT Register

15	14	13	12	11	10	9	8
I2CCNT							
R/W-0h							
7	6	5	4	3	2	1	0
I2CCNT							
R/W-0h							

Table 26-15. I2CCNT Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	I2CCNT	R/W	0h	Data count value. I2CCNT indicates the number of data bytes to transfer or receive. If a STOP condition is specified (STP=1) then I2CCNT will decrease after each byte is sent until it reaches zero, which in turn will generate a STOP condition. The value in I2CCNT is a don't care when the RM bit in I2CMDR is set to 1. The start value loaded to the internal data counter is 65536. The start value loaded to internal data counter is 1-65535. Reset type: SYSRSn 0h (R/W) = data count value is 65536 1h (R/W) = data count value is 1 2h (R/W) = data count value is 2 FFFFh (R/W) = data count value is 65535

26.7.2.7 I2CDRR Register (Offset = 6h) [Reset = 0h]

I2CDRR is shown in Figure 26-26 and described in Table 26-16.

Return to the Summary Table.

I2CDRR is a 16-bit register used by the CPU to read received data. The I2C module can receive a data byte with 1 to 8 bits. The number of bits is selected with the bit count (BC) bits in I2CMDR. One bit at a time is shifted in from the SDA pin to the receive shift register (I2CRSR). When a complete data byte has been received, the I2C module copies the data byte from I2CRSR to I2CDRR. The CPU cannot access I2CRSR directly.
If a data byte with fewer than 8 bits is in I2CDRR, the data value is right-justified, and the other bits of I2CDRR(7-0) are undefined. For example, if BC = 011 (3-bit data size), the receive data is in I2CDRR(2-0), and the content of I2CDRR(7-3) is undefined.
When in the receive FIFO mode, the I2CDRR register acts as the receive FIFO buffer.

Figure 26-26. I2CDRR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
DATA							
R-0h							

Table 26-16. I2CDRR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-0	DATA	R	0h	Receive data Reset type: SYSRSn

26.7.2.8 I2CSAR Register (Offset = 7h) [Reset = 3FFh]

I2CSAR is shown in Figure 26-27 and described in Table 26-17.

Return to the Summary Table.

The I2C slave address register (I2CSAR) is a 16-bit register for storing the next slave address that will be transmitted by the I2C module when it is a master. The SAR field of I2CSAR contains a 7-bit or 10-bit slave address. When the I2C module is not using the free data format (FDF = 0 in I2CMDR), it uses this address to initiate data transfers with a slave, or slaves. When the address is nonzero, the address is for a particular slave. When the address is 0, the address is a general call to all slaves. If the 7-bit addressing mode is selected (XA = 0 in I2CMDR), only bits 6-0 of I2CSAR are used
write 0s to bits 9-7.

Figure 26-27. I2CSAR Register

15	14	13	12	11	10	9	8
RESERVED						SAR	
R-0h						R/W-3FFh	
7	6	5	4	3	2	1	0
SAR							
R/W-3FFh							

Table 26-17. I2CSAR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-0	SAR	R/W	3FFh	In 7-bit addressing mode (XA = 0 in I2CMDR): 00h-7Fh Bits 6-0 provide the 7-bit slave address that the I2C module transmits when it is in the master-transmitter mode. Write 0s to bits 9-7. In 10-bit addressing mode (XA = 1 in I2CMDR): 000h-3FFh Bits 9-0 provide the 10-bit slave address that the I2C module transmits when it is in the master transmitter mode. Reset type: SYSRSn

26.7.2.9 I2CDXR Register (Offset = 8h) [Reset = 0h]

I2CDXR is shown in Figure 26-28 and described in Table 26-18.

Return to the Summary Table.

The CPU writes transmit data to I2CDXR. This 16-bit register accepts a data byte with 1 to 8 bits. Before writing to I2CDXR, specify how many bits are in a data byte by loading the appropriate value into the bit count (BC) bits of I2CMDR. When writing a data byte with fewer than 8 bits, make sure the value is right-aligned in I2CDXR. After a data byte is written to I2CDXR, the I2C module copies the data byte to the transmit shift register (I2CXSR). The CPU cannot access I2CXSR directly. From I2CXSR, the I2C module shifts the data byte out on the SDA pin, one bit at a time.
When in the transmit FIFO mode, the I2CDXR register acts as the transmit FIFO buffer.

Figure 26-28. I2CDXR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
DATA							
R/W-0h							

Table 26-18. I2CDXR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-0	DATA	R/W	0h	Transmit data Reset type: SYSRSn

26.7.2.10 I2CMDR Register (Offset = 9h) [Reset = 0h]

I2CMDR is shown in Figure 26-29 and described in Table 26-19.

Return to the Summary Table.

The I2C mode register (I2CMDR) is a 16-bit register that contains the control bits of the I2C module.

Figure 26-29. I2CMDR Register

15	14	13	12	11	10	9	8
NACKMOD	FREE	STT	RESERVED	STP	MST	TRX	XA
R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RM	DLB	IRS	STB	FDF	BC		
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h		

Table 26-19. I2CMDR Register Field Descriptions

Bit	Field	Type	Reset	Description
15	NACKMOD	R/W	0h	NACK mode bit. This bit is only applicable when the I2C module is acting as a receiver. Reset type: SYSRSn 0h (R/W) = In the slave-receiver mode: The I2C module sends an acknowledge (ACK) bit to the transmitter during each acknowledge cycle on the bus. The I2C module only sends a no-acknowledge (NACK) bit if you set the NACKMOD bit. In the master-receiver mode: The I2C module sends an ACK bit during each acknowledge cycle until the internal data counter counts down to 0. At that point, the I2C module sends a NACK bit to the transmitter. To have a NACK bit sent earlier, you must set the NACKMOD bit 1h (R/W) = In either slave-receiver or master-receiver mode: The I2C module sends a NACK bit to the transmitter during the next acknowledge cycle on the bus. Once the NACK bit has been sent, NACKMOD is cleared. Important: To send a NACK bit in the next acknowledge cycle, you must set NACKMOD before the rising edge of the last data bit.
14	FREE	R/W	0h	This bit controls the action taken by the I2C module when a debugger breakpoint is encountered. Reset type: SYSRSn 0h (R/W) = When I2C module is master: If SCL is low when the breakpoint occurs, the I2C module stops immediately and keeps driving SCL low, whether the I2C module is the transmitter or the receiver. If SCL is high, the I2C module waits until SCL becomes low and then stops. When I2C module is slave: A breakpoint forces the I2C module to stop when the current transmission/reception is complete. 1h (R/W) = The I2C module runs free that is, it continues to operate when a breakpoint occurs.
13	STT	R/W	0h	START condition bit (only applicable when the I2C module is a master). The RM, STT, and STP bits determine when the I2C module starts and stops data transmissions (see Table 9-6). Note that the STT and STP bits can be used to terminate the repeat mode, and that this bit is not writable when IRS = 0. Reset type: SYSRSn 0h (R/W) = In the master mode, STT is automatically cleared after the START condition has been generated. 1h (R/W) = In the master mode, setting STT to 1 causes the I2C module to generate a START condition on the I2C-bus
12	RESERVED	R	0h	Reserved

Table 26-19. I2CMDR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	STP	R/W	0h	STOP condition bit (only applicable when the I2C module is a master). In the master mode, the RM,STT, and STP bits determine when the I2C module starts and stops data transmissions. Note that the STT and STP bits can be used to terminate the repeat mode, and that this bit is not writable when IRS=0. When in non-repeat mode, at least one byte must be transferred before a stop condition can be generated. The I2C module delays clearing of this bit until after the I2CSTR[SCD] bit is set. To avoid disrupting the I2C state machine, the user must wait until this bit is clear before initiating a new message. Reset type: SYSRSn 0h (R/W) = STP is automatically cleared after the STOP condition has been generated 1h (R/W) = STP has been set by the device to generate a STOP condition when the internal data counter of the I2C module counts down to 0.
10	MST	R/W	0h	Master mode bit. MST determines whether the I2C module is in the slave mode or the master mode. MST is automatically changed from 1 to 0 when the I2C master generates a STOP condition Reset type: SYSRSn 0h (R/W) = Slave mode. The I2C module is a slave and receives the serial clock from the master. 1h (R/W) = Master mode. The I2C module is a master and generates the serial clock on the SCL pin.
9	TRX	R/W	0h	Transmitter mode bit. When relevant, TRX selects whether the I2C module is in the transmitter mode or the receiver mode. Reset type: SYSRSn 0h (R/W) = Receiver mode. The I2C module is a receiver and receives data on the SDA pin. 1h (R/W) = Transmitter mode. The I2C module is a transmitter and transmits data on the SDA pin.
8	XA	R/W	0h	Expanded address enable bit. Reset type: SYSRSn 0h (R/W) = 7-bit addressing mode (normal address mode). The I2C module transmits 7-bit slave addresses (from bits 6-0 of I2CSAR), and its own slave address has 7 bits (bits 6-0 of I2COAR). 1h (R/W) = 10-bit addressing mode (expanded address mode). The I2C module transmits 10-bit slave addresses (from bits 9-0 of I2CSAR), and its own slave address has 10 bits (bits 9-0 of I2COAR).
7	RM	R/W	0h	Repeat mode bit (only applicable when the I2C module is a master-transmitter). The RM, STT, and STP bits determine when the I2C module starts and stops data transmissions Reset type: SYSRSn 0h (R/W) = Nonrepeat mode. The value in the data count register (I2CCNT) determines how many bytes are received/transmitted by the I2C module. 1h (R/W) = Repeat mode. A data byte is transmitted each time the I2CDXR register is written to (or until the transmit FIFO is empty when in FIFO mode) until the STP bit is manually set. The value of I2CCNT is ignored. The ARDY bit/interrupt can be used to determine when the I2CDXR (or FIFO) is ready for more data, or when the data has all been sent and the CPU is allowed to write to the STP bit.

Table 26-19. I2CMDR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	DLB	R/W	0h	Digital loopback mode bit. Reset type: SYSRSn 0h (R/W) = Digital loopback mode is disabled. 1h (R/W) = Digital loopback mode is enabled. For proper operation in this mode, the MST bit must be 1. In the digital loopback mode, data transmitted out of I2CDXR is received in I2CDRR after n device cycles by an internal path, where: n = ((I2C input clock frequency/module clock frequency) x 8) The transmit clock is also the receive clock. The address transmitted on the SDA pin is the address in I2COAR. Note: The free data format (FDF = 1) is not supported in the digital loopback mode.
5	IRS	R/W	0h	I2C module reset bit. Reset type: SYSRSn 0h (R/W) = The I2C module is in reset/disabled. When this bit is cleared to 0, all status bits (in I2CSTR) are set to their default values. 1h (R/W) = The I2C module is enabled. This has the effect of releasing the I2C bus if the I2C peripheral is holding it.
4	STB	R/W	0h	START byte mode bit. This bit is only applicable when the I2C module is a master. As described in version 2.1 of the Philips Semiconductors I2C-bus specification, the START byte can be used to help a slave that needs extra time to detect a START condition. When the I2C module is a slave, it ignores a START byte from a master, regardless of the value of the STB bit. Reset type: SYSRSn 0h (R/W) = The I2C module is not in the START byte mode. 1h (R/W) = The I2C module is in the START byte mode. When you set the START condition bit (STT), the I2C module begins the transfer with more than just a START condition. Specifically, it generates: 1. A START condition 2. A START byte (0000 0001b) 3. A dummy acknowledge clock pulse 4. A repeated START condition Then, as normal, the I2C module sends the slave address that is in I2CSAR.
3	FDF	R/W	0h	Free data format mode bit. Reset type: SYSRSn 0h (R/W) = Free data format mode is disabled. Transfers use the 7-/10-bit addressing format selected by the XA bit. 1h (R/W) = Free data format mode is enabled. Transfers have the free data (no address) format described in Section 9.2.5. The free data format is not supported in the digital loopback mode (DLB=1).
2-0	BC	R/W	0h	Bit count bits. BC defines the number of bits (1 to 8) in the next data byte that is to be received or transmitted by the I2C module. The number of bits selected with BC must match the data size of the other device. Notice that when BC = 000b, a data byte has 8 bits. BC does not affect address bytes, which always have 8 bits. Note: If the bit count is less than 8, receive data is right-justified in I2CDRR(7-0), and the other bits of I2CDRR(7-0) are undefined. Also, transmit data written to I2CDXR must be right-justified Reset type: SYSRSn 0h (R/W) = 8 bits per data byte 1h (R/W) = 1 bit per data byte 2h (R/W) = 2 bits per data byte 3h (R/W) = 3 bits per data byte 4h (R/W) = 4 bits per data byte 5h (R/W) = 5 bits per data byte 6h (R/W) = 6 bits per data byte 7h (R/W) = 7 bits per data byte

26.7.2.11 I2CISRC Register (Offset = Ah) [Reset = 0h]

I2CISRC is shown in Figure 26-30 and described in Table 26-20.

Return to the Summary Table.

The I2C interrupt source register (I2CISRC) is a 16-bit register used by the CPU to determine which event generated the I2C interrupt.

Figure 26-30. I2CISRC Register

15	14	13	12	11	10	9	8
RESERVED				WRITE_ZEROS			
R-0h				R/W-0h			
7	6	5	4	3	2	1	0
RESERVED						INTCODE	
R-0h						R-0h	

Table 26-20. I2CISRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-8	WRITE_ZEROS	R/W	0h	TI internal testing bits These reserved bit locations should always be written as zeros. Reset type: SYSRSn
7-3	RESERVED	R	0h	Reserved
2-0	INTCODE	R	0h	Interrupt code bits. The binary code in INTCODE indicates the event that generated an I2C interrupt. A CPU read will clear this field. If another lower priority interrupt is pending and enabled, the value corresponding to that interrupt will then be loaded. Otherwise, the value will stay cleared. The interrupt events below are listed in descending order of priority. That is INTCODE 1 (Arbitration lost) has the highest priority and INTCODE 7 (Addressed as slave) has the lowest priority. In the case of an arbitration lost, a no-acknowledgment condition detected, or a stop condition detected, a CPU read will also clear the associated interrupt flag bit in the I2CSTR register. Emulator reads will not affect the state of this field or of the status bits in the I2CSTR register. Reset type: SYSRSn 0h (R/W) = None 1h (R/W) = Arbitration lost 2h (R/W) = No-acknowledgment condition detected 3h (R/W) = Registers ready to be accessed 4h (R/W) = Receive data ready 5h (R/W) = Transmit data ready 6h (R/W) = Stop condition detected 7h (R/W) = Addressed as slave

26.7.2.12 I2CEMDR Register (Offset = Bh) [Reset = 1h]

I2CEMDR is shown in Figure 26-31 and described in Table 26-21.

Return to the Summary Table.

I2C Extended Mode

Figure 26-31. I2CEMDR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						FCM	BC
R-0h						R/W-0h	R/W-1h

Table 26-21. I2CEMDR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-2	RESERVED	R	0h	Reserved
1	FCM	R/W	0h	Forward Compatibility mode. This bit when programmed brings the functionality of Tx request only when Tx data required regardless of data status in Tx buffer for non-FIFO mode. This register affects the XRDY behavior hence needs to be set after releasing the IRS (I2CMDR[5]). Reset type: SYSRSn 0h (R/W) = Legacy functionality of requesting Tx data upon buffer copy to shift register or upon start condition is active. Stale data is reused after illegal start, ARB Lost, NACK conditions. 1h (R/W) = New functionality of requesting data only upon ACK (address/data) is active.
0	BC	R/W	1h	Backwards compatibility mode. This bit affects the timing of the transmit status bits (XRDY and XSMT) in the I2CSTR register when in slave transmitter mode. Reset type: SYSRSn 0h (R/W) = See the "Backwards Compatibility Mode Bit, Slave Transmitter" Figure for details. 1h (R/W) = See the "Backwards Compatibility Mode Bit, Slave Transmitter" Figure for details.

26.7.2.13 I2CPSC Register (Offset = Ch) [Reset = 0h]

I2CPSC is shown in Figure 26-32 and described in Table 26-22.

Return to the Summary Table.

The I2C prescaler register (I2CPSC) is a 16-bit register (see Figure 14-21) used for dividing down the I2C input clock to obtain the desired module clock for the operation of the I2C module. See the device-specific data manual for the supported range of values for the module clock frequency.
IPSC must be initialized while the I2C module is in reset (IRS = 0 in I2CMDR). The prescaled frequency takes effect only when IRS is changed to 1. Changing the IPSC value while IRS = 1 has no effect.

Figure 26-32. I2CPSC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
IPSC							
R/W-0h							

Table 26-22. I2CPSC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-0	IPSC	R/W	0h	I2C prescaler divide-down value. IPSC determines how much the CPU clock is divided to create the module clock of the I2C module: module clock frequency = I2C input clock frequency/(IPSC + 1) Note: IPSC must be initialized while the I2C module is in reset (IRS = 0 in I2CMDR). Reset type: SYSRSn

26.7.2.14 I2CFFTX Register (Offset = 20h) [Reset = 0h]

I2CFFTX is shown in Figure 26-33 and described in Table 26-23.

Return to the Summary Table.

The I2C transmit FIFO register (I2CFFTX) is a 16-bit register that contains the I2C FIFO mode enable bit as well as the control and status bits for the transmit FIFO mode of operation on the I2C peripheral.

Figure 26-33. I2CFFTX Register

15	14	13	12	11	10	9	8
RESERVED	I2CFFEN	TXFFRST	TXFFST				
R-0h	R/W-0h	R/W-0h	R-0h				
7	6	5	4	3	2	1	0
TXFFINT	TXFFINTCLR	TXFFIENA	TXFFIL				
R-0h	R-0/W1S-0h	R/W-0h	R/W-0h				

Table 26-23. I2CFFTX Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	I2CFFEN	R/W	0h	I2C FIFO mode enable bit. This bit must be enabled for either the transmit or the receive FIFO to operate correctly. Reset type: SYSRSn 0h (R/W) = Disable the I2C FIFO mode. 1h (R/W) = Enable the I2C FIFO mode.
13	TXFFRST	R/W	0h	Transmit FIFO Reset Reset type: SYSRSn 0h (R/W) = Reset the transmit FIFO pointer to 0000 and hold the transmit FIFO in the reset state. 1h (R/W) = Enable the transmit FIFO operation.
12-8	TXFFST	R	0h	Contains the status of the transmit FIFO: xxxxx Transmit FIFO contains xxxxx bytes. 00000 Transmit FIFO is empty. Note: Since these bits are reset to zero, the transmit FIFO interrupt flag will be set when the transmit FIFO operation is enabled and the I2C is taken out of reset. This will generate a transmit FIFO interrupt if enabled. To avoid any detrimental effects from this, write a one to the TXFFINTCLR once the transmit FIFO operation is enabled and the I2C is taken out of reset. Reset type: SYSRSn
7	TXFFINT	R	0h	Transmit FIFO interrupt flag. This bit cleared by a CPU write of a 1 to the TXFFINTCLR bit. If the TXFFIENA bit is set, this bit will generate an interrupt when it is set. Reset type: SYSRSn 0h (R/W) = Transmit FIFO interrupt condition has not occurred. 1h (R/W) = Transmit FIFO interrupt condition has occurred.
6	TXFFINTCLR	R-0/W1S	0h	Transmit FIFO Interrupt Flag Clear Reset type: SYSRSn 0h (R/W) = Writes of zeros have no effect. Reads return a 0. 1h (R/W) = Writing a 1 to this bit clears the TXFFINT flag.
5	TXFFIENA	R/W	0h	Transmit FIFO Interrupt Enable Reset type: SYSRSn 0h (R/W) = Disabled. TXFFINT flag does not generate an interrupt when set. 1h (R/W) = Enabled. TXFFINT flag does generate an interrupt when set.

Table 26-23. I2CFFTX Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-0	TXFFIL	R/W	0h	Transmit FIFO interrupt level. These bits set the status level that will set the transmit interrupt flag. When the TXFFST4-0 bits reach a value equal to or less than these bits, the TXFFINT flag will be set. This will generate an interrupt if the TXFFIENA bit is set. Because the I2C on this device has a 16-level transmit FIFO, these bits cannot be configured for an interrupt of more than 16 FIFO levels. Reset type: SYSRSn

26.7.2.15 I2CFFRX Register (Offset = 21h) [Reset = 0h]

I2CFFRX is shown in Figure 26-34 and described in Table 26-24.

Return to the Summary Table.

The I2C receive FIFO register (I2CFFRX) is a 16-bit register that contains the control and status bits for the receive FIFO mode of operation on the I2C peripheral.

Figure 26-34. I2CFFRX Register

15	14	13	12	11	10	9	8
RESERVED		RXFFRST	RXFFST				
R-0h		R/W-0h	R-0h				
7	6	5	4	3	2	1	0
RXFFINT	RXFFINTCLR	RXFFIENA	RXFFIL				
R-0h	R-0/W1S-0h	R/W-0h	R/W-0h				

Table 26-24. I2CFFRX Register Field Descriptions

Bit	Field	Type	Reset	Description
15-14	RESERVED	R	0h	Reserved
13	RXFFRST	R/W	0h	I2C receive FIFO reset bit Reset type: SYSRSn 0h (R/W) = Reset the receive FIFO pointer to 0000 and hold the receive FIFO in the reset state. 1h (R/W) = Enable the receive FIFO operation.
12-8	RXFFST	R	0h	Contains the status of the receive FIFO: xxxxx Receive FIFO contains xxxxx bytes 00000 Receive FIFO is empty. Reset type: SYSRSn
7	RXFFINT	R	0h	Receive FIFO interrupt flag. This bit cleared by a CPU write of a 1 to the RXFFINTCLR bit. If the RXFFIENA bit is set, this bit will generate an interrupt when it is set Reset type: SYSRSn 0h (R/W) = Receive FIFO interrupt condition has not occurred. 1h (R/W) = Receive FIFO interrupt condition has occurred.
6	RXFFINTCLR	R-0/W1S	0h	Receive FIFO interrupt flag clear bit. Reset type: SYSRSn 0h (R/W) = Writes of zeros have no effect. Reads return a zero. 1h (R/W) = Writing a 1 to this bit clears the RXFFINT flag.
5	RXFFIENA	R/W	0h	Receive FIFO interrupt enable bit. Reset type: SYSRSn 0h (R/W) = Disabled. RXFFINT flag does not generate an interrupt when set. 1h (R/W) = Enabled. RXFFINT flag does generate an interrupt when set.
4-0	RXFFIL	R/W	0h	Receive FIFO interrupt level. These bits set the status level that will set the receive interrupt flag. When the RXFFST4-0 bits reach a value equal to or greater than these bits, the RXFFINT flag is set. This will generate an interrupt if the RXFFIENA bit is set. Note: Since these bits are reset to zero, the receive FIFO interrupt flag will be set if the receive FIFO operation is enabled and the I2C is taken out of reset. This will generate a receive FIFO interrupt if enabled. To avoid this, modify these bits on the same instruction as or prior to setting the RXFFRST bit. Because the I2C on this device has a 16-level receive FIFO, these bits cannot be configured for an interrupt of more than 16 FIFO levels. Reset type: SYSRSn

26.7.3 I2C Registers to Driverlib Functions

Table 26-25. I2C Registers to Driverlib Functions

File	Driverlib Function
OAR	
i2c.h	I2C_setOwnSlaveAddress
IER	
i2c.c	I2C_enableInterrupt
i2c.c	I2C_disableInterrupt
STR	
i2c.c	I2C_getInterruptStatus
i2c.c	I2C_clearInterruptStatus
i2c.h	I2C_isBusBusy
i2c.h	I2C_getStatus
i2c.h	I2C_clearStatus
CLKL	
i2c.c	I2C_initMaster
CLKH	
i2c.c	I2C_initMaster
CNT	
i2c.h	I2C_setDataCount
DRR	
i2c.h	I2C_getData
SAR	
i2c.h	I2C_setSlaveAddress
DXR	
i2c.h	I2C_putData
MDR	
i2c.h	I2C_enableModule
i2c.h	I2C_disableModule
i2c.h	I2C_setConfig
i2c.h	I2C_setBitCount
i2c.h	I2C_sendStartCondition
i2c.h	I2C_sendStopCondition
i2c.h	I2C_sendNACK
i2c.h	I2C_getStopConditionStatus
i2c.h	I2C_setAddressMode
i2c.h	I2C_setEmulationMode
i2c.h	I2C_enableLoopback
i2c.h	I2C_disableLoopback
ISRC	
i2c.h	I2C_getInterruptSource
EMDR	
i2c.h	I2C_setExtendedMode
PSC	
i2c.c	I2C_initMaster
i2c.c	I2C_configureModuleFrequency
i2c.h	I2C_getPreScaler

Table 26-25. I2C Registers to Driverlib Functions (continued)

File	Driverlib Function
FFTX	
i2c.c	I2C_enableInterrupt
i2c.c	I2C_disableInterrupt
i2c.c	I2C_getInterruptStatus
i2c.c	I2C_clearInterruptStatus
i2c.h	I2C_enableFIFO
i2c.h	I2C_disableFIFO
i2c.h	I2C_setFIFOInterruptLevel
i2c.h	I2C_getFIFOInterruptLevel
i2c.h	I2C_getTxFIFOStatus
FFRX	
i2c.c	I2C_enableInterrupt
i2c.c	I2C_disableInterrupt
i2c.c	I2C_getInterruptStatus
i2c.c	I2C_clearInterruptStatus
i2c.h	I2C_enableFIFO
i2c.h	I2C_disableFIFO
i2c.h	I2C_setFIFOInterruptLevel
i2c.h	I2C_getFIFOInterruptLevel
i2c.h	I2C_getRxFIFOStatus

This page intentionally left blank.

Chapter 27

Power Management Bus Module (PMBus)

This chapter describes the features and operation of the Power Management Bus (PMBus) module.

27.1 Introduction

The PMBus module provides an interface between the microcontroller and devices compliant with the SMI Forum PMBus Specification Part I version 1.0 and Part II version 1.1. The PMBus is based on SMBus, which uses a similar physical layer to the I2C. This chapter assumes you are familiar with the PMBus, SMBus, and I2C bus specifications.

27.1.1 PMBUS Related Collateral

Foundational Materials

- C2000 Academy - Communications
- Seven things to know about PMBus (Video)

Getting Started Materials

- C28x PMBus Communications Stack User's Guide Application Report
- *Software Implementation of PMBus over I2C for TMS320F2803x Application Report*

Expert Materials

- 9 things you need to know about PMBus Point-of-Load Power (Video)

27.1.2 Features

The PMBus module has the following features:

- Compliance with the SMI Forum PMBus Specification (Part I v1.0 and Part II v1.1)
- Support for master and slave modes
- Support for I2C modes
- Support for three speeds:
  - Standard Mode: Up to 100 kHz
  - Fast Mode: Up to 400 kHz
- Packet error checking
- CONTROL and ALERT signals
- Clock high and low time-outs
- Four-byte transmit and receive buffers
- One maskable interrupt, which can be generated by several conditions:
  - Receive data ready
  - Transmit buffer empty
  - Slave address received
  - End of message
  - ALERT input asserted
  - Clock low time-out
  - Clock high time-out
  - Bus free

27.1.3 Block Diagram

Figure 27-1 shows the block diagram for PMBus. The PMBus module handles the lower levels of the PMBus protocol. In addition to controlling signal levels and timing, parsing addresses, and buffering data, it also directly supports complex transactions such as Read Word and Process Call.

There are four PMBus signals:

- **SCL** is the bus clock. It is normally controlled by the master, but may be held low by a slave to delay a transaction and allow more time for processing.
- **SDA** is the bidirectional data line.
- **CONTROL** is a slave input that can trigger an interrupt. It can be used to tell a slave device to shut down.

- **ALERT** is a slave output/master input that allows a slave to request attention from the master.

The SDA and SCL timings produced by the module are derived from SYSCLK. To comply with the PMBus timing specs, the bit clock divider must be set by way of the PMBTIMCLK register to provide a bit clock of 10 MHz or less.

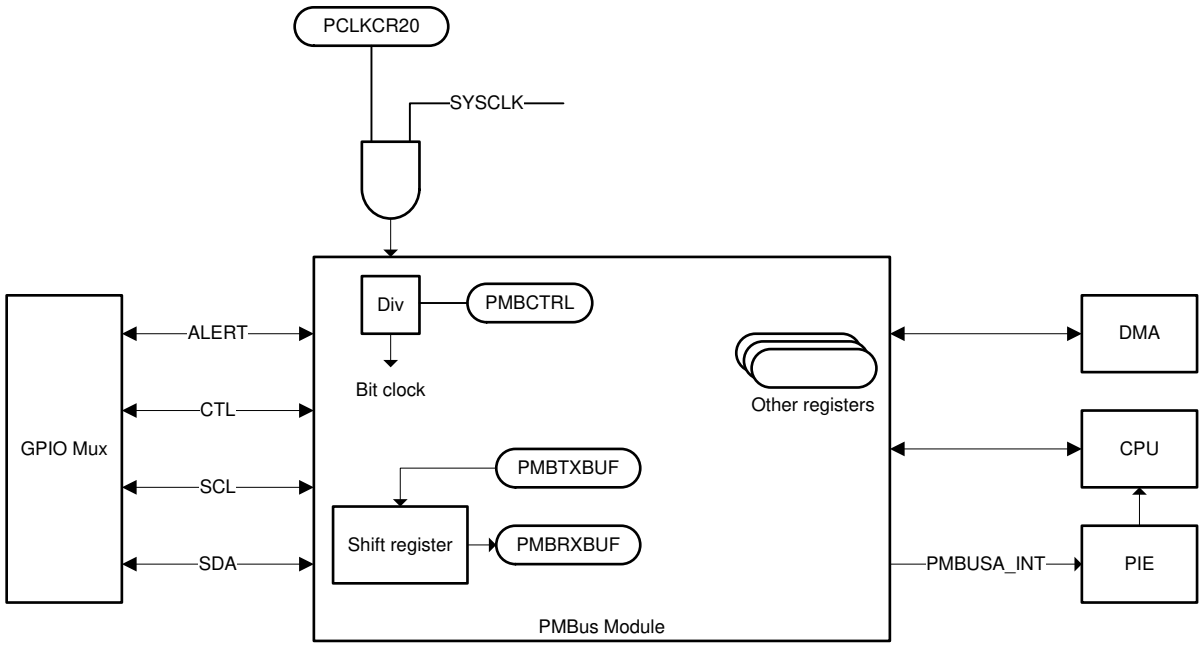


Figure 27-1. PMBus Module Block Diagram

27.2 Configuring Device Pins

The GPIO mux registers must be configured to connect this peripheral to the device pins. To avoid glitches on the pins, the GPyGMUX bits must be configured first (while keeping the corresponding GPyMUX bits at the default of zero), followed by writing the GPyMUX register to the desired value.

Some IO functionality is defined by GPIO register settings independent of this peripheral. For input signals, the GPIO input qualification is set to asynchronous mode by setting the appropriate GPxQSELn register bits to 11b. The internal pullups are configured in the GPyPUD register.

Note

The GPIO configuration register GPyODR must be set to normal mode when the PMBus is used. The open-drain operation for PMBus is managed by the PMBus module

See the *General-Purpose Input/Output (GPIO)* chapter for more details on GPIO mux and settings.

27.3 Slave Mode Operation

This section describes the configuration and operation of the PMBus module in slave mode.

27.3.1 Configuration

To configure the module, write a suitable clock divider to the PMBCTRL register's CLKDIV field to produce a bit clock frequency of less than 10 MHz. To activate slave mode, set the SLAVE_EN bit in the PMBCTRL register. Next, set up the PMBSC register. The following options are configurable:

- Slave address and mask (SLAVE_ADDR and SLAVE_MASK): Sets the slave address and mask for message acceptance.

- Manual slave address acknowledgement (MAN_SLAVE_ACK): When enabled, allows software to decide whether to acknowledge (ACK) an address. When disabled, the decision to ACK is made automatically based on the slave address and mask.
- PEC enable (PEC_ENA): Set this bit if Packet Error Checking (PEC) is used on the bus.
- Manual command byte acknowledgement (MAN_CMD): Similar to manual slave acknowledgement, setting this bit allows software to decide whether to acknowledge (ACK) a command byte.
- Number of bytes to acknowledge automatically (RX_BYTE_ACK_CNT): This is normally set to the max value, which allows the entire receive buffer to be used. However, smaller values may be used if the application requires that erroneous messages be detected and not acknowledged (NACKed) as soon as possible.

Manual acknowledgement is done by writing a one to the PMBACK register. Even with automatic acknowledgement, some writes to PMBACK are required. If the message (not including the address) is longer than 4 bytes, each packet of 4 bytes much be acknowledged. The PMBus module will stretch the clock (hold it low) until an ACK is issued. The module will then pull the data line low and release the clock, providing the ACK signal to the master.

If the complete message or the last part of the message is less than 4 bytes (or the RX_BYTE_ACK_CNT limit), do not write to PMBACK.

Writing a zero to the PMBACK bit will send a NACK. This may only be done when the module is waiting for an acknowledgement. If a zero is written at any other time, the NACK will be issued during the next message.

27.3.2 Message Handling

This section describes some of the message types for PMBus and how to determine which message type is being received in slave mode. It is oriented toward the most efficient mode of operation – with automatic address and command acknowledgment. It is also oriented toward having Packet Error Checking (PEC) enabled.

If automatic address acknowledgement is disabled, all messages will start by setting the SLAVE_ADDR_READY register bit high. Read commands will have two instructions one for the read, and one for the write. If automatic command acknowledgement is enabled, the DATA READY bit will be set high, as well. If the message has no PEC, the number of bytes available will be n-1. For example, with PEC, a QUICK COMMAND will have one byte. With no PEC, a QUICK COMMAND will have zero bytes.

Note that the byte count does not increment as bytes arrive. No bits are set in the PMBST register until a stop message is received, the receive buffer is full, or a fault occurs. Then, all appropriate bit values are placed in the register together. All that is necessary to receive a quick command is to ACK the message by writing a one to the PMBACK register.

27.3.2.1 Quick Command

Quick Commands (Figure 27-2) received by the PMBus module in slave mode require a simple acknowledgment of the received device address. In automatic address acknowledge mode, the module processes the quick command without firmware interaction. Upon receipt of the end of message, the firmware has the option to read the received address in the PMBHSA register. In manual address acknowledge mode, the address is acknowledged by writing to the PMBACK register.

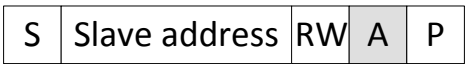


Figure 27-2. Quick Command Message

27.3.2.2 Send Byte

A Send Byte message (Figure 27-3) consists of the device address, a single data byte, and an optional PEC byte. To process the PEC byte correctly, PEC processing must be enabled in the PMBSC register. In automatic address acknowledge mode, the data and optional PEC byte are acknowledged without firmware interaction. The module generates an End of Message interrupt, reads the status register and finds the data ready indication bit set. In manual mode, the address is acknowledged by the firmware, while the remaining data and PEC bytes are acknowledged by the module.

The PMBus module stores Data Byte #0 into the PMBRXBUF register. The data byte will be stored into bits 7-0. In non-PEC mode, the RX Byte Count in the PMBSTS register will indicate one byte received. If PEC processing is enabled, the PEC byte is also stored into the PMBRXBUF register, with the PEC byte residing in bits 15-8. The RX Byte Count in the PMBSTS register will indicate two bytes received. The PEC Valid bit in the PMBSTS register indicates the validity of the received PEC byte.

When a Send Byte message is received, the Data Ready bit is set along with the EOM and, assuming that the PEC is valid, the PEC valid bit. The read byte count (RD_BYTE_COUNT) register will contain a 2. All that is necessary to receive a send byte command is to ACK the message by writing a 1 to the PMBACK register. Before doing the ACK, read the byte from the lowest byte of the PMBRXBUF register.

S	Slave address	Wr	A	Byte #0	A	P

S	Slave address	Wr	A	Byte #0	A	PEC	A	P

Figure 27-3. Send Byte Message With and Without PEC

27.3.2.3 Receive Byte

A Receive Byte message (Figure 27-4) consists of the device address, a single data byte, and an optional PEC byte. In automatic address acknowledge mode, the firmware receives a data request interrupt following reception of the slave address. The data byte to be sent to the master is stored into bits 7-0 of the PMBTXBUF register and Transmit Byte Count bits within the PMBSC register are set to a value of 1. If PEC processing is enabled, the Transmit PEC bit (bit 19) within the PMBSC register is set to 1, along with the Enable PEC bit (bit 15). The module automatically appends the calculated PEC byte at the completion of the message.

S	Slave address	Rd	A	Byte #0	NA	P

S	Slave address	Rd	A	Byte #0	A	PEC	NA	P

Figure 27-4. Receive Byte Message With and Without PEC

27.3.2.4 Write Byte/Word

The Write Byte and Write Word messages (Figure 27-5) consist of a slave address, a command word, transmitted data bytes and an optional PEC byte. In automatic address acknowledge mode, the data bytes and optional PEC byte are acknowledged without firmware interaction. The acknowledgment of the command word is configured through the PMBSC register. The firmware receives an End of Message interrupt in all cases except for Write Word with PEC message, reads the status register and finds the data ready indication bit set.

In the case of a Write Word with PEC byte message, the data ready interrupt is enabled after receiving 4 bytes (command byte, the 2 data bytes and the PEC byte). The firmware reads the data from the PMBRXBUF register and must write the PMBACK register to acknowledge back to the master. The PMBus module holds SCL low until the firmware responds to the received data.

In all other cases, the EOM interrupt is received and data can be read from the PMBRXBUF register. The firmware is not required to send an acknowledgement back to the master.

The Write Byte message will look exactly the same as the Send Byte, except the RD_BYTE_COUNT register will contain a 3. The Write Word message will have a RD_BYTE_COUNT of 4.

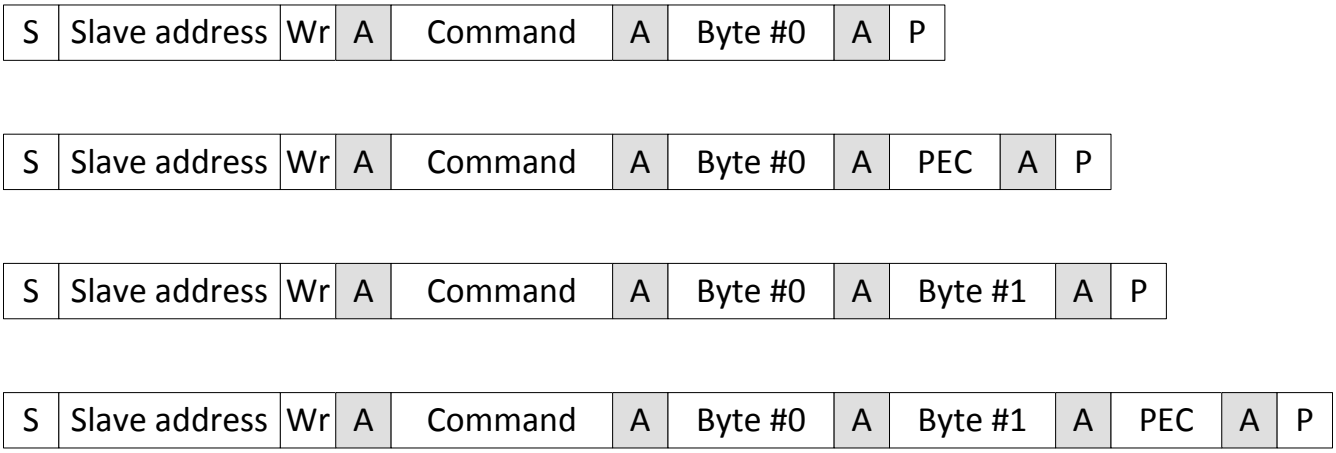


Figure 27-5. Write Byte and Write Word Messages With and Without PEC

27.3.2.5 Read Byte/Word

The Read Byte and Read Word messages (Figure 27-6) consist of a slave address, a command word, received data bytes from a slave, and an optional PEC byte. Address and command acknowledgment is configured through the PMBSC register. In automatic mode, the PMBus module provides a data ready and data request interrupt following receipt of a repeated start and slave address. The received command byte is found in bits 7-0 of the PMBRXBUF register. The firmware responds to the data request by programming the data bytes into the PMBTXBUF register and the TX Byte Count bits in the PMBSC register. If PEC processing is enabled, the Transmit PEC bit should also be asserted. An EOM interrupt indicates completion of the message to the Master.

When the repeated start (Sr) signal is received, the Data Ready bit will be asserted with a RD_BYTE_COUNT of 1. At this point, the operation cannot be distinguished from a group command Send Byte message. When the same device address is sent out with a read, the Data Request bit will be asserted. If data has already been written to the PMTXBUF register before the Device Addr is received, the Data Request bit will not be asserted. So if group commands are also expected, it is necessary to read the Data Ready with a RD_BYTE_COUNT of 1, and then wait and see whether the next event is an EOM or a Data Request. If it is an EOM, the command should be processed as a group send byte. If it is a Data Request, the command should be processed as a read. Depending on the command, it could be a read byte, word, or block. If the PMBus module is polled, both the Data Ready and the Data Request bits could possibly be set between polling intervals. This should be considered in the design of the firmware.

Once the read command is recognized, you must respond by writing data to the PMBTXBUF register. It is also necessary to make sure that the values in the PMBSC register are correct. The transmit byte count and PEC bit must be set appropriately. For a read byte, the transmit byte count can be loaded with a 1. If the transmission of a PEC byte is desired, the TX_PEC bit must be set. After this, the data can be written to PMBTXBUF, which starts the transmission. All bytes should be written to PMBTXBUF at the same time. After the master receives the message, it will NACK the last byte to indicate that the correct number of bytes have been received. This will cause the EOM bit to be set in the PMBSTS register, indicating to the firmware that the Read Byte message is complete.

Figure 27-6. Read Byte and Read Word Messages With and Without PEC

27.3.2.6 Process Call

The Process Call (Figure 27-7) protocol consists of a Write Word message, followed by a Read Word message, without a stop condition between the two messages. Address and command acknowledgement is configured through the PMBSC register. In automatic mode, following receipt of the repeated start and slave address, the PMBus module provides a data ready and a data request interrupt. The repeated start bit is set in the PMBSTS register to indicate the receipt of the first part of the Process Call message. The received command byte is found in bits 7-0 of the PMBRXBUF register, while the two data bytes received from the master can be found in bits 23-8. Upon receipt of the repeated start and a data request from the module, the firmware programs the PMBTXBUF with the 2 data bytes to be sent to the master. If PEC processing is enabled, the Transmit PEC bit within the PMBSC register is asserted. The EOM interrupt will indicate the read word portion of the Process Call message has been completed by the module.

S	Slave address	Wr	A	Command	A	Byte #0	A	Byte #1	A
Sr	Slave address	Rd	A	Byte #0	A	Byte #1	NA	P	

S	Slave address	Wr	A	Command	A	Byte #0	A	Byte #1	A
Sr	Slave address	Rd	A	Byte #0	A	Byte #1	A	PEC	NA	P

Figure 27-7. Process Call Message With and Without PEC

27.3.2.7 Block Write

The Block Write (Figure 27-8) protocol is similar to Write Word in its structure, except that there are more than 2 data bytes in the message. Following the receipt of the command byte, the block length and 2 data bytes, the PMBus module provides a data ready interrupt. The module waits for the firmware to read the received data and program the acknowledge register. While waiting for an ACK from the firmware, the module will drive the clock line low, stalling the bus. The data ready interrupts will continue for the duration of the message at a frequency of every 4 data bytes. The number of bytes received can be found within the PMBSTS register. At the end of the message, less than 4 bytes may be stored in the PMBRXBUF register. The PEC Valid bit can be checked to determine if the received PEC value is accurate.

S	Slave address	Wr	A	Command	A	Byte count = N	A	Byte #0	A
Byte #1	A	• • •	A	Byte #(N-1)	A	P			

S	Slave address	Wr	A	Command	A	Byte count = N	A	Byte #0	A
Byte #1	A	• • •	A	Byte #(N-1)	A	PEC	A	P	

Figure 27-8. Block Write Message With and Without PEC

27.3.2.8 Block Read

The Block Read (Figure 27-9) protocol is similar to a Read Word in its structure, except that there are more than 2 data bytes in the message. Following the receipt of the repeated slave address, a data ready and data request interrupt is generated by the PMBus module. The command byte received from the master can be found in bits 7-0 of the PMBRXBUF register. The SCL line is held low until the firmware programs data bytes into the PMBTXBUF register. The firmware is required to load the block length into bits 7-0 of the PMBTXBUF register during the initial programming of the register. After 4 bytes have been transmitted, the module will issue a data request interrupt and hold SCL low again until the firmware has programmed additional data into the PMBTXBUF register.

Block read starts the same as Read Word or Read Byte, but TX_COUNT is loaded with a 4 the first time, and TX_PEC is not set. Instead of waiting for an EOM after the first transmission, the firmware instead waits for a Data Request, indicating that the master is ready for more data. Until the last 4 or less bytes, the firmware simply writes a 4 to TX_COUNT and then writes the 4 bytes to PMBTXBUF. TX_PEC is left cleared. Then when the last 4 or fewer bytes are to be transmitted, the firmware writes out the appropriate byte count, sets the TX_PEC bit, and writes the data to PMBTXBUF. The PMBus module will write out the data, followed by the PEC, and then the EOM bit will be set when the master NACKs the PEC.

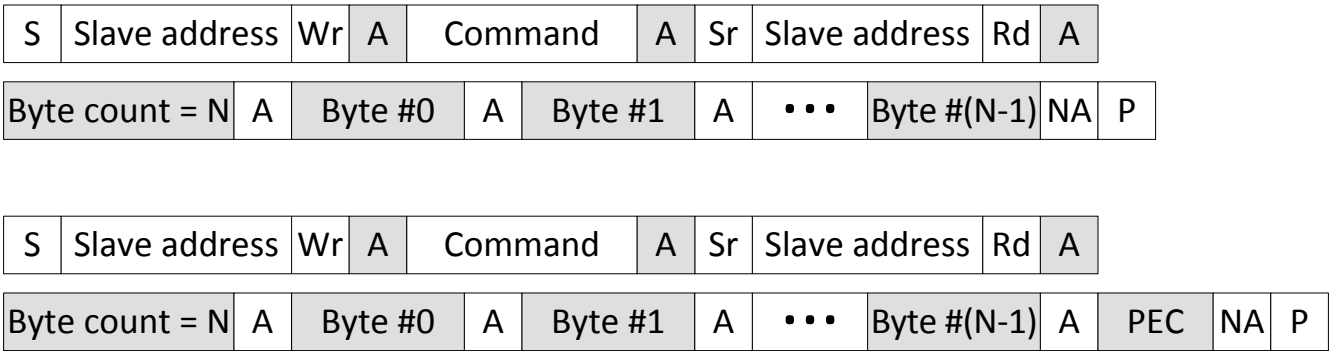


Figure 27-9. Block Read Message With and Without PEC

27.3.2.9 Block Write-Block Read Process Call

The Block Write-Block Read Process Call (Figure 27-10) protocol combines the Block Write and Block Read protocols, removing the stop condition between the two messages. The processing of the Block Read-Block Write Process Call message is similar to the mode of operation for the Process Call message. After acknowledgement of the address and command bytes, the PMBus module generates a data ready interrupt upon detection of 4 data bytes or a repeated start condition. After receiving the repeated start, the firmware will be required to load transmit data to send to the master. Bits 7-0 of the initial programming of the PMBTXBUF register must represent the byte count of the block data sent to the master.

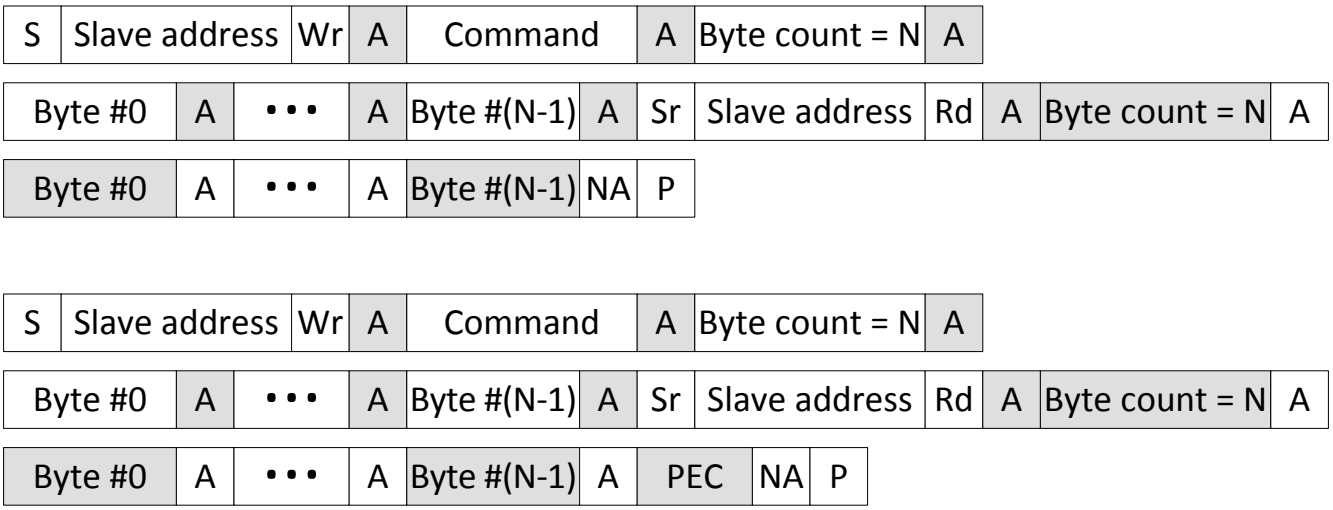


Figure 27-10. Block Write-Block Read Process Call Message With and Without PEC

27.3.2.10 Alert Response

The Alert Response Message (Figure 27-11) is utilized when the master detects an alert condition from a slave on the PMBus. In automatic address acknowledge mode, upon detection of the Alert Response Address, the PMBus module provides an acknowledgement to the master and sends the programmed slave address within the PMBSC register. The module only responds to the message if the Alert En bit within PMBCTRL register has been previously set. After receiving the Alert Response message, the module will clear the alert condition and enable bit within the PMBCTRL register.

In manual address acknowledge mode, the firmware must read the received address from the PMBRXBUF register and transmit the desired slave address back to the master. The PMBCTRL register must be reprogrammed to disable the Alert En bit used to initiate the Alert Response message from the master.

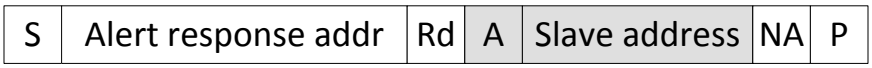


Figure 27-11. Alert Response Message

27.3.2.11 Extended Command

The PMBus module provides support for extended commands that allow for an extra 256 command codes. Both command bytes are stored in the PMBRXBUF register along with the data bytes. In recognizing the extended command messages, the Repeated Start bit and the Rd Byte Count Bits within the PMBSTS register are utilized. For Extended Command Write Byte/Write Word messages (Figure 27-12), the two command bytes are stored in bits 15-0 of the PMBRXBUF register. The initial command byte should hold the command extension code, representing utilization of the extended command protocol. The Repeated Start bit is also set, received after the retransmission of the device address. The Rd Byte Count equals 3 for an Ext Cmd Write Byte message and 4 for an Ext Cmd Write Word message.

For the Extended Command Read Byte/Read Word messages (Figure 27-13), the module will generate a data ready and data request interrupt following reception of the repeated device address. The two command bytes will be found in bits 15-0 of the PMBRXBUF register, with the initial command byte matching the command extension code. The firmware will be required to load transmit data to complete the message back to the master.

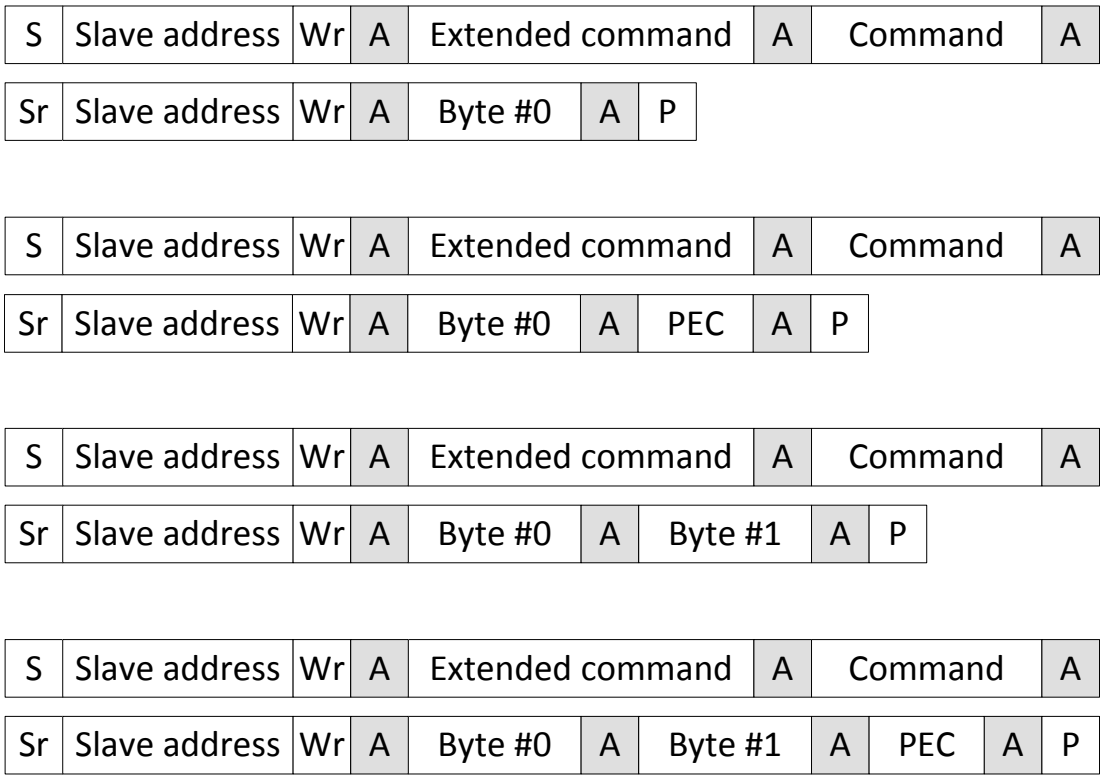


Figure 27-12. Extended Command Write Byte and Write Word Messages With and Without PEC

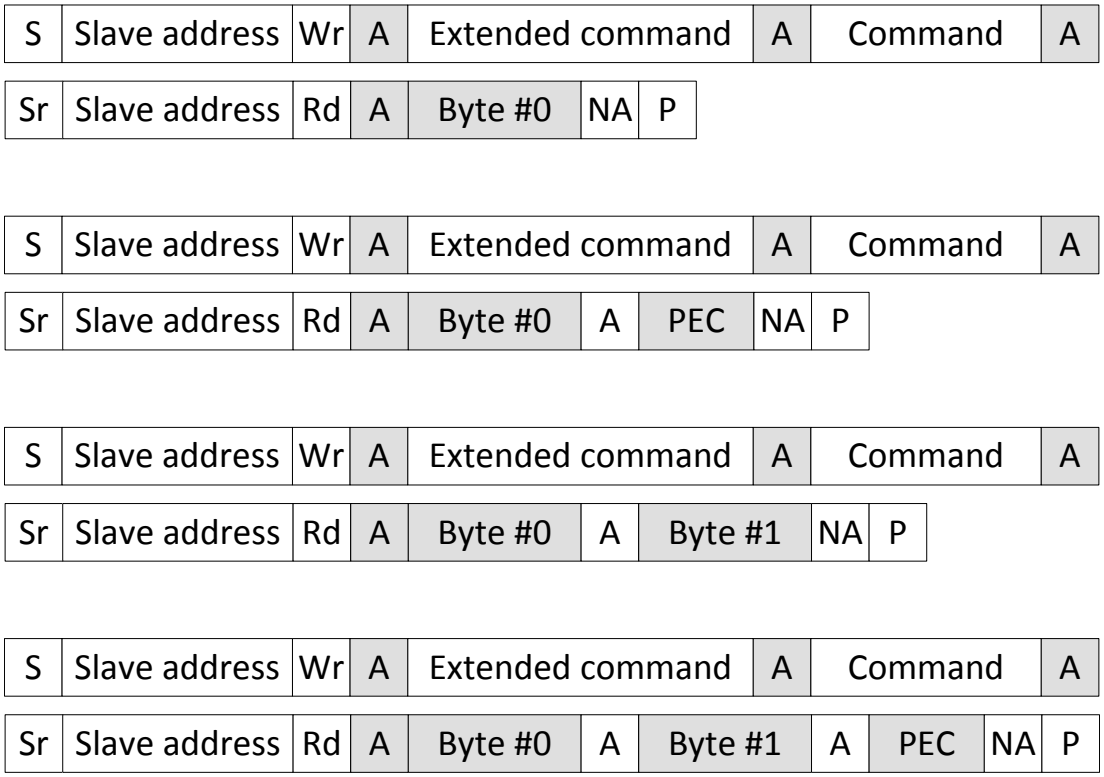


Figure 27-13. Extended Command Read Byte and Read Word Messages With and Without PEC

27.3.2.12 Group Command

The PMBus module supports the Group Command protocol. The Group Command (Figure 27-14) protocol is used to send commands to more than one device within the same message. When devices on the bus detect the stop condition at the conclusion of the Group Command message, the received commands are executed concurrently. Following address and command acknowledgment, the module provides a data ready interrupt upon detection of 4 data bytes or the transmission of a repeated start on the bus. The firmware should wait for the EOM interrupt before processing the received command, as required by the use of the Group Command message.

For Group Commands, the data ready bit will be set as soon as the repeated start is received. The data can then be read into memory. But the data should not be acted upon until the EOM bit is set, which will occur when all of the messages have been received. Other than this delayed EOM, there is no difference for the slave firmware in receiving a Group Command than any other write message.

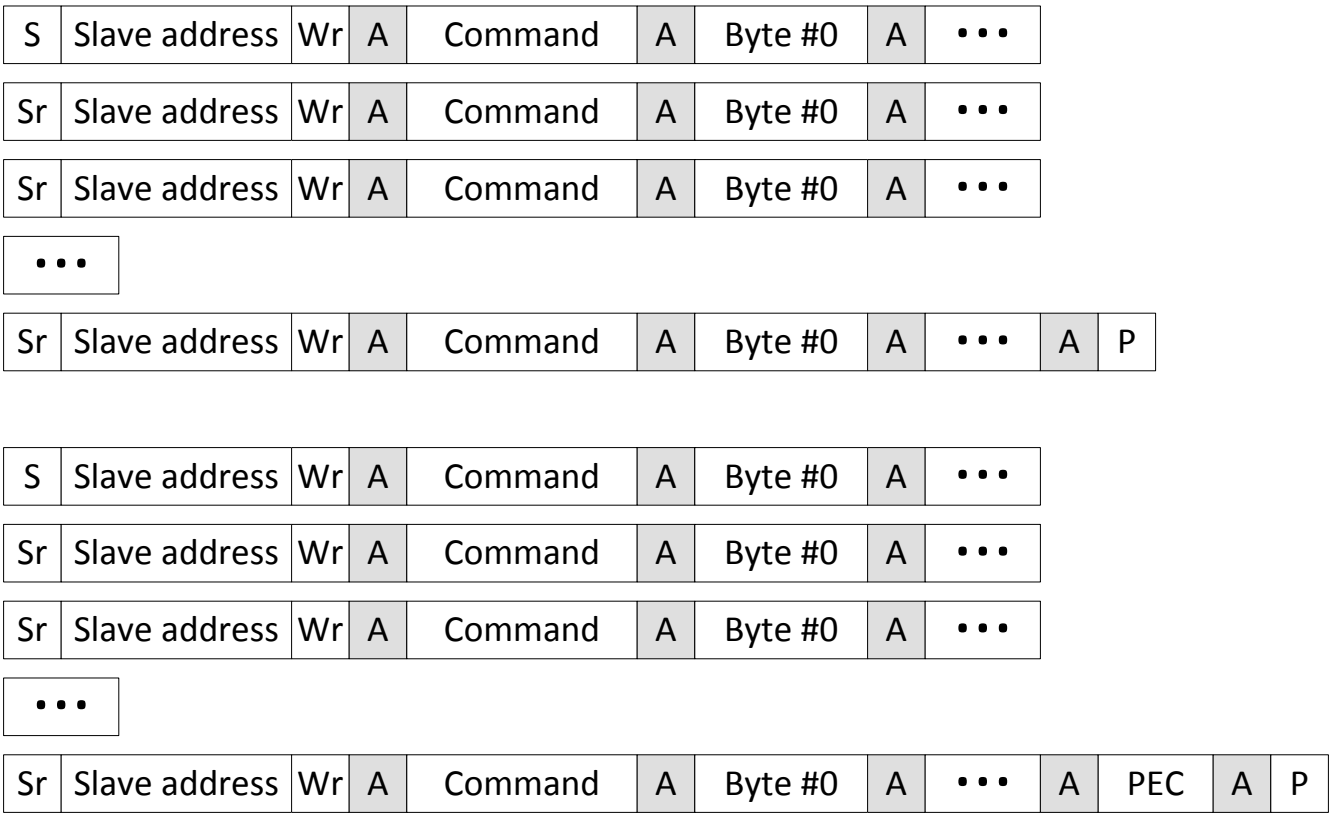


Figure 27-14. Group Command Message With and Without PEC

27.4 Master Mode Operation

This section describes the configuration and operation of the PMBus module in master mode.

27.4.1 Configuration

First, write a suitable clock divider to the PMBCTRL register's CLKDIV field to produce a bit clock frequency of less than 10 MHz. To activate master mode, set the MASTER_EN bit and clear the SLAVE_EN bit in the PMBCTRL register. For each transaction, set up the PMBMC register. The following options are configurable:

- Slave address (SLAVE_ADDR): Sets the slave address for the next transaction.
- PEC enable (PEC_ENA): If Packet Error Checking (PEC) is used on the bus, set this bit.
- Extended command code enable (EXT_CMD): When set, uses two bytes for commands.
- Command code enable (CMD_ENA): When set, sends a command byte at the start of the transaction.
- Byte count (BYTE_COUNT): Determines the number of data bytes to transfer. This does not include the block length byte, which is generated automatically when needed.
- Special command enables (GRP_CMD and PRC_CALL): Enables special behavior for group commands and process calls.

Writing to the PMBMC register will start a transfer.

Manual acknowledgement of received data is not needed.

27.4.2 Message Handling

This section describes the behavior and required configuration for each command type.

27.4.2.1 Quick Command

Quick commands (Figure 27-15) are initiated in master mode by simply programming the desired slave device address into the PMBMC. The byte count within the PMBMC register is configured to 0 bytes by writing all zeros to bits 15-8. Upon transmission of the device address, the PMBus module will monitor the slave acknowledgement of the address. If the address is not acknowledged, the NACK bit within the status register is enabled and the PMBus module automatically sends a stop condition on the bus to terminate the message. If the address is acknowledged, a data request is issued to the processor. The firmware writes a zero to the PMBACK to terminate the message, forcing the PMBus modules to write a stop condition onto the bus.

Figure 27-15. Quick Command Message

27.4.2.2 Send Byte

A Send Byte message (Figure 27-16) consists of the device address, a single data byte, and an optional PEC byte. To initiate a Send Byte message, the data byte to be transmitted to the slave is loaded into bits 7-0 of the PMBTXBUF register. The PMBMC register is configured with the device address. To transmit a PEC byte with the message, the PEC_EN bit within the PMBMC register is asserted high when the address is programmed.

After programming the PMBMC register, the PMBus module transmits the Send Byte message. The firmware can wait for an End of Message interrupt from the PMBus module. Upon receipt of the EOM interrupt, the PMBSTS register is read to verify the slave properly acknowledged the transmitted data.

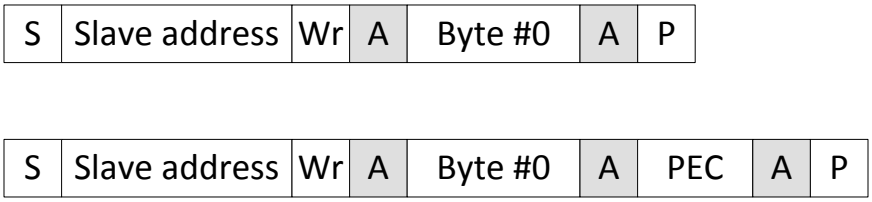


Figure 27-16. Send Byte Message With and Without PEC

27.4.2.3 Receive Byte

A Receive Byte message (Figure 27-17) consists of the device address, a single data byte, and an optional PEC byte. Data is being read from the slave in a Receive Byte message. To initiate a Receive Byte message, the firmware programs the device address, the R/W bit and the optional PEC_EN into the PMBMC register. The R/W bit is enabled high to indicate a read message type (data transmitted from slave to master).

After programming the PMBMC register, the PMBus module transmits the Receive Byte message. The firmware can wait for an End of Message interrupt from the PMBus module to verify the accuracy of the message transmission. Upon receipt of the EOM interrupt, the PMBSTS register is read to verify proper slave acknowledgement of the device address and to determine if any data is available for reading in the PMBRXBUF register. If PEC_EN was asserted in the PMBMC register, the PEC_VALID bit in the PMBSTS register is also checked to ensure a proper PEC byte was received from the slave with the received data.

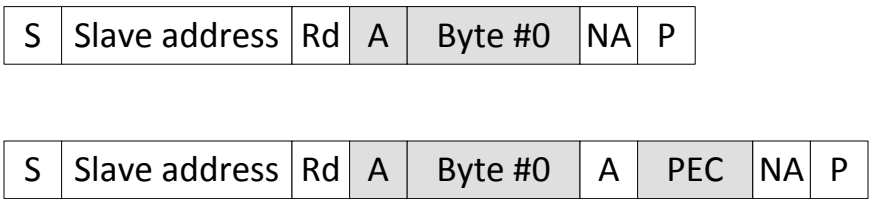


Figure 27-17. Receive Byte Message With and Without PEC

27.4.2.4 Write Byte/Word

The Write Byte and Write Word messages (Figure 27-18) consist of a device address, a command byte, transmitted data bytes, and an optional PEC byte. Write Byte messages include a single byte, while the Write Word messages support transmission of 2 bytes to the corresponding slave module. Similar to the Send Byte protocol, the PMBMC register is configured to send 1 or 2 bytes, the CMD_EN bit is set to enable command byte transmission and the optional PEC_EN bit is set.

With the command byte transmission enabled, the format of the PMBTXBUF register differs from the Send Byte protocol. In bits 7-0, the firmware must program the command byte to be sent to the slave. The data byte(s) are programmed into bits 15-8 and bits 23-16.

After programming the PMBMC register, the PMBus module transmits the Write Byte/Word message. The firmware can wait for an End of Message interrupt from the module to verify the accuracy of the message transmission. The PMBSTS register indicates if the slave acknowledged the message properly.

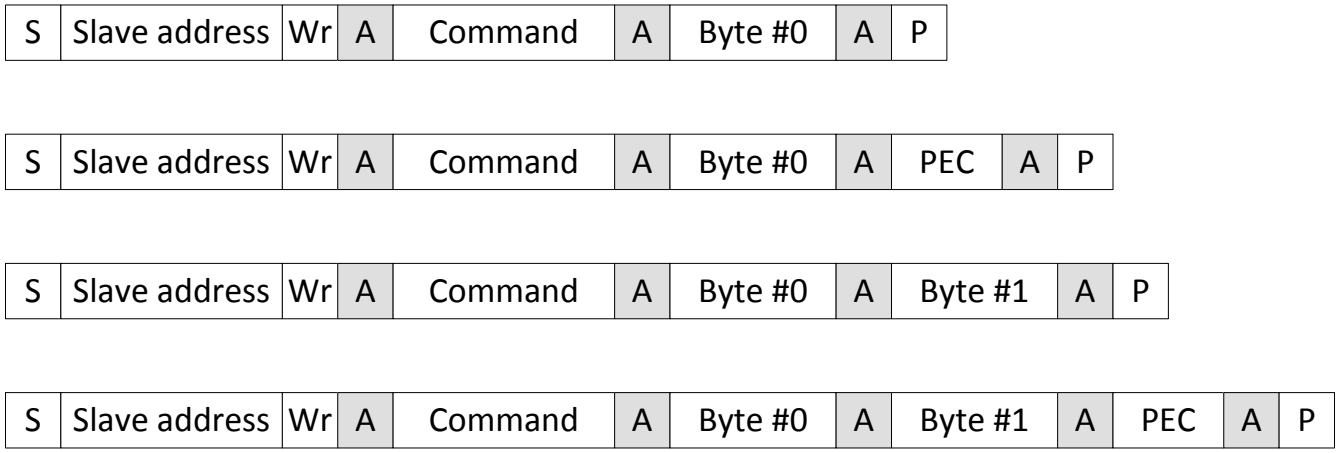


Figure 27-18. Write Byte and Write Word Messages With and Without PEC

27.4.2.5 Read Byte/Word

The Read Byte and Read Word messages (Figure 27-19) consist of a device address, a command byte, received data bytes from a slave, and an optional PEC byte. Read Byte messages include a single byte, while the Read Word message protocol supports receipt of 2 bytes from the slave. Similar to the Receive Byte Protocol, the PMBMC register is configured to receive 1 or 2 bytes, the CMD_EN bit is set and the PEC_EN is configured to expect or not expect a PEC byte appended to the message. The PMBus module will automatically terminate the message after the expected number of bytes is received from the slave or if the slave does not properly acknowledge any portion of the message.

In addition to programming the PMBMC register, the firmware is expected to load the command byte into bits 7-0 of the PMBTXBUF register. Any data received from the slave will be found in the PMBRXBUF register.

Figure 27-19. Read Byte and Read Word Messages With and Without PEC

27.4.2.6 Process Call

The Process Call (Figure 27-20) protocol consists of a Write Word message, followed by a Read Word message, without a stop condition between the two messages. A PEC byte can be appended to the read data from the slave as an option to the message protocol. The PMBMC register includes a PRC_CALL bit, which enables the transmission of a Process Call message onto the PMBus. The PMBus module will automatically generate a repeated start condition and initiate the Read Word portion of the message when the process call bit is enabled.

To complete the Write Word portion of the Process Call, the PMBTXBUF register is loaded with the command byte in bits 7-0 and the data bytes are loaded into bits 23-8 of the register.

After programming the PMBMC register, the PMBus module transmits the Process Call Message. The firmware can wait for an End of Message interrupt from the module to determine the validity of the message. Upon the receipt of the EOM, the PMBSTS register should indicate the receipt of 2 bytes from the Read Word portion of the Process Call message and the status of the Slave acknowledgment of the transmit data. If PEC processing is enabled, the PEC_VAL bit within the PMBSTS register indicates the accuracy of the PEC byte received from the slave during the Read Word part of the message.

The PRC_CALL bit within the PMBMC register must be disabled for the next non-Process Call message. Please note that any write to the PMBMC register initiates a message, so reconfiguration of the master is not recommended until the firmware requires a new message to be transmitted.

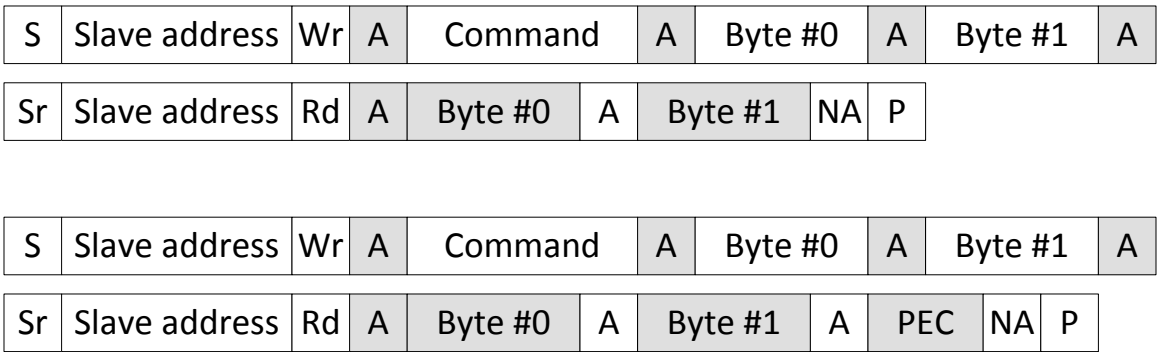


Figure 27-20. Process Call Message With and Without PEC

27.4.2.7 Block Write

The Block Write (Figure 27-21) protocol is similar to a Write Word in its structure, with the exception of transmission of more than 2 data bytes in the message. Additionally, the first data byte following the command byte specifies the length of the block of data bytes. As with a majority of the message protocols, the PEC byte can be appended to the end of the write data to the slave.

To initiate a Block Write message on the bus, the PMBMC register is programmed with the block length in the Byte Count bits. The block length is the number of data bytes, excluding the command byte and the first data byte that contains the block length. The PMBus module will automatically insert the block length into the message if the number of data bytes specified by the firmware exceeds 2. The initial write data is loaded into the PMBTXBUF register. With bits 7-0 representing the command byte, the remaining 3 bytes represent the first three data bytes following the block length.

Following programming of the PMBMC register, the Block Write message is transmitted. If the block length exceeds three bytes, the PMBus module will provide a data request interrupt, indicating the need for additional data bytes in the PMBTXBUF register. The PMBus module assumes that if more than 4 bytes are needed to complete the message, the firmware will utilize all 4 bytes when programming the PMBTXBUF register. If less than 4 bytes are needed to finish the Block Write message, the firmware only needs to program the appropriate bits of the PMBTXBUF register.

Upon completion of the message, the PMBus module issues an EOM interrupt. The PMBSTS register can be checked to verify the slave accepted the block of write data.

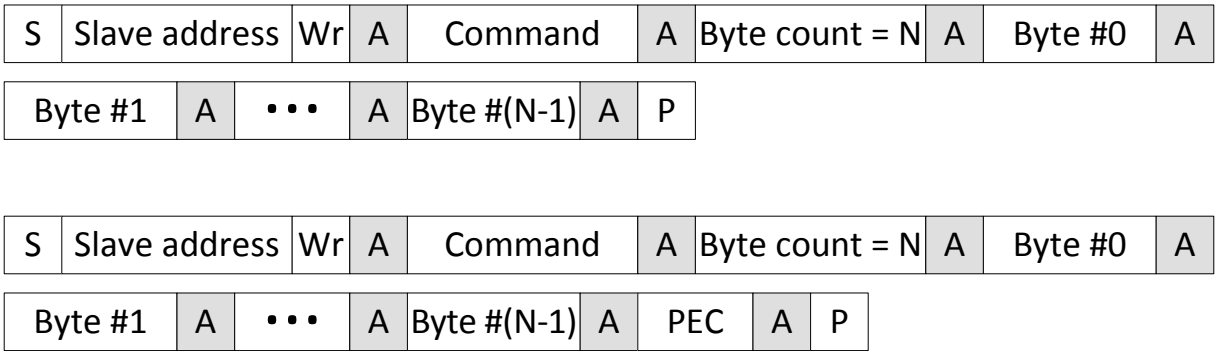


Figure 27-21. Block Write Message With and Without PEC

27.4.2.8 Block Read

The Block Read (Figure 27-22) protocol is similar to a Read Word in its structure, with the exception that there are more than 2 data bytes received from the slave. The first data byte transmitted by the slave represents the block length of the data being written by the slave. If PEC processing is enabled, the slave appends a PEC byte to the end of the message.

To initiate a Block Read message on the PMBus, the PMBMC register is programmed with the block length in the Byte Count bits. This count excludes the command byte, any slave address and the block length bytes in the message. The command byte to be transmitted to the slave is written into bits 7-0 of the PMBTXBUF register prior to the programming of the PMBMC register.

After configuring the PMBMC register, the Block Read message is transmitted. The module interrupts the firmware upon receipt of 4 data bytes from the slave. If the block length is 3, the EOM interrupt will be received concurrently with the data ready interrupt. Otherwise, only a data ready interrupt is asserted, indicating 4 bytes are ready for reading by the firmware. At the end of the message, less than 4 bytes may be stored in the PMBRXBUF register. The RX Byte Count bits in the PMBSTS register indicate the number of bytes available in the final data transfer. The firmware may verify the received PEC upon detection of the End of Message interrupt.

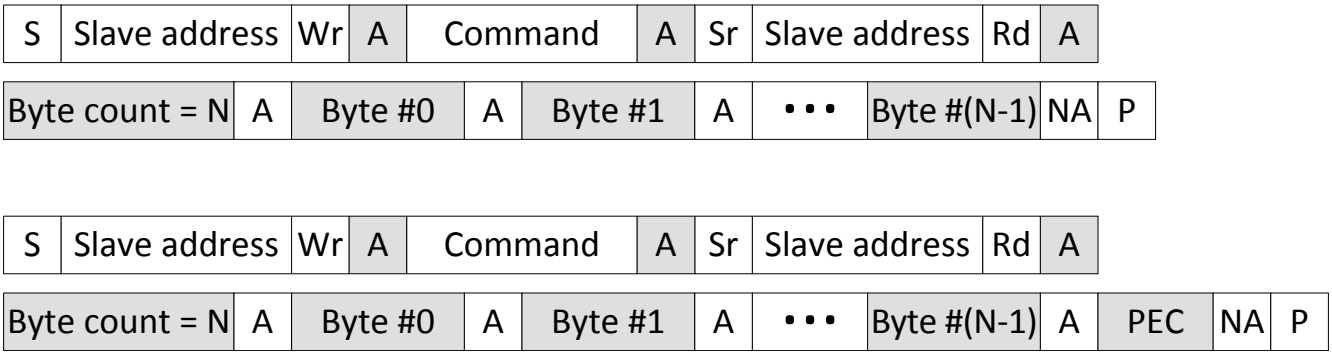


Figure 27-22. Block Read Message With and Without PEC

27.4.2.9 Block Write-Block Read Process Call

The Block Write-Block Read Process Call (Figure 27-23) protocol combines the Block Write and Block Read protocols, removing the stop condition between the two messages. The operation of the master is similar to a Block Write operation. Loading the block length into the byte count bits of the PMBMC register provides the length of the Block Write portion of the message. In addition, the PRC_CALL bit within the PMBMC register must be enabled. Upon completion of the Block Write part of the message, the PMBus module will automatically issue a Repeated Start condition on the PMBus and start transmission of the Block Read portion of the message. Operation of the PMBus module after the Repeated Start condition is the same as it would be in a simple Block Read Message.

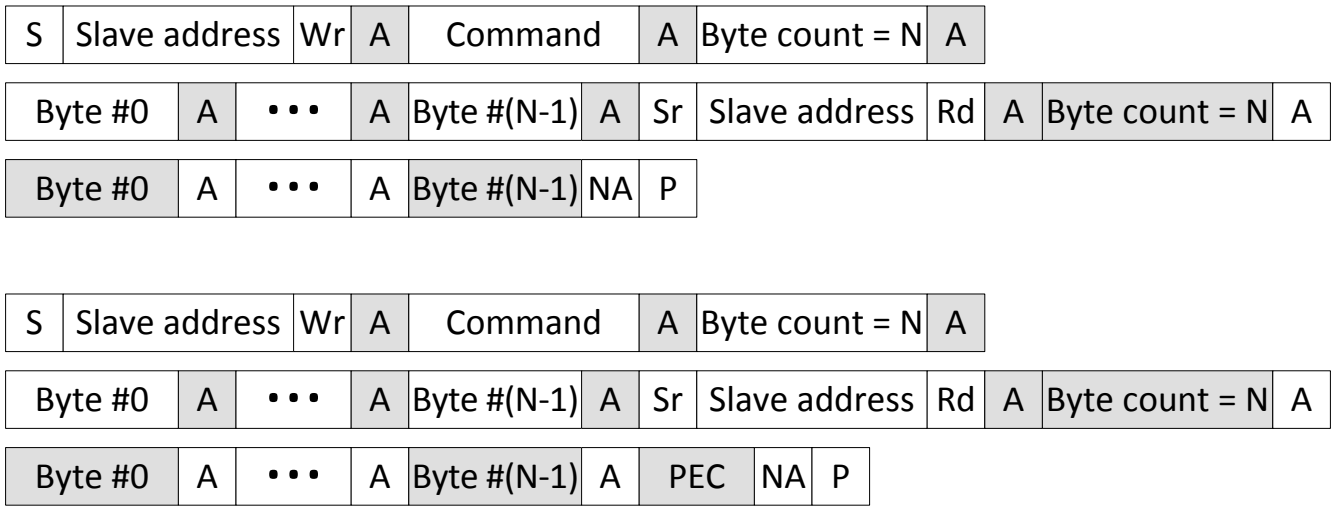


Figure 27-23. Block Write-Block Read Process Call Message With and Without PEC

27.4.2.10 Alert Response

The Alert Response Message (Figure 27-24) is utilized when the master detects an alert condition from a slave. In master mode, the Alert Response Message is simply a Receive Byte message with PEC disabled and the slave address set to 0xC (Alert Response address). The PMBus module detects the alert condition on an input and interrupts the firmware indicating the assertion of an alert condition (slave desires to communicate with master). Programming the PMBMC register with the Alert Response address initiates the Alert Response message and provides the device address of the slave requesting service. The device address will be found in the PMBRXBUF register following receipt of the EOM interrupt.

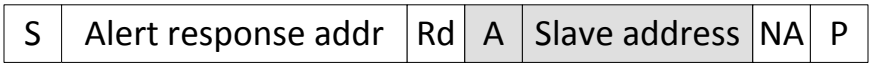


Figure 27-24. Alert Response Message

27.4.2.11 Extended Command

The PMBus module provides support for extended commands which allow for an extra 256 command codes. By asserting the EXT_CMD bit within the PMBMC register, two command bytes are transmitted on the message protocol. Extended commands can be added to the Write Byte and Write Word (Figure 27-25) and the Read Byte and Read Word (Figure 27-26) protocols. Operation of the PMBus module in extended command mode is similar to these formats. In programming the write data or first part of the read message, the second command byte is loaded into bits 15-8 of the PMBTXBUF register with the remaining data bytes. The remaining operation of the module is identical to the previous protocols, except for the inclusion of a Repeated Start condition and slave address in the write messages. No support is required by firmware for these additional bytes in the write messages. The module will interpret the EXT_CMD bit and make the appropriate format changes.

S	Slave address	Wr	A	Extended command	A	Command	A		
Sr	Slave address	Wr	A	Byte #0	A	P			

S	Slave address	Wr	A	Extended command	A	Command	A		
Sr	Slave address	Wr	A	Byte #0	A	PEC	A	P	

S	Slave address	Wr	A	Extended command	A	Command	A		
Sr	Slave address	Wr	A	Byte #0	A	Byte #1	A	P	

S	Slave address	Wr	A	Extended command	A	Command	A		
Sr	Slave address	Wr	A	Byte #0	A	Byte #1	A	PEC	A	P

Figure 27-25. Extended Command Write Byte and Write Word Messages With and Without PEC

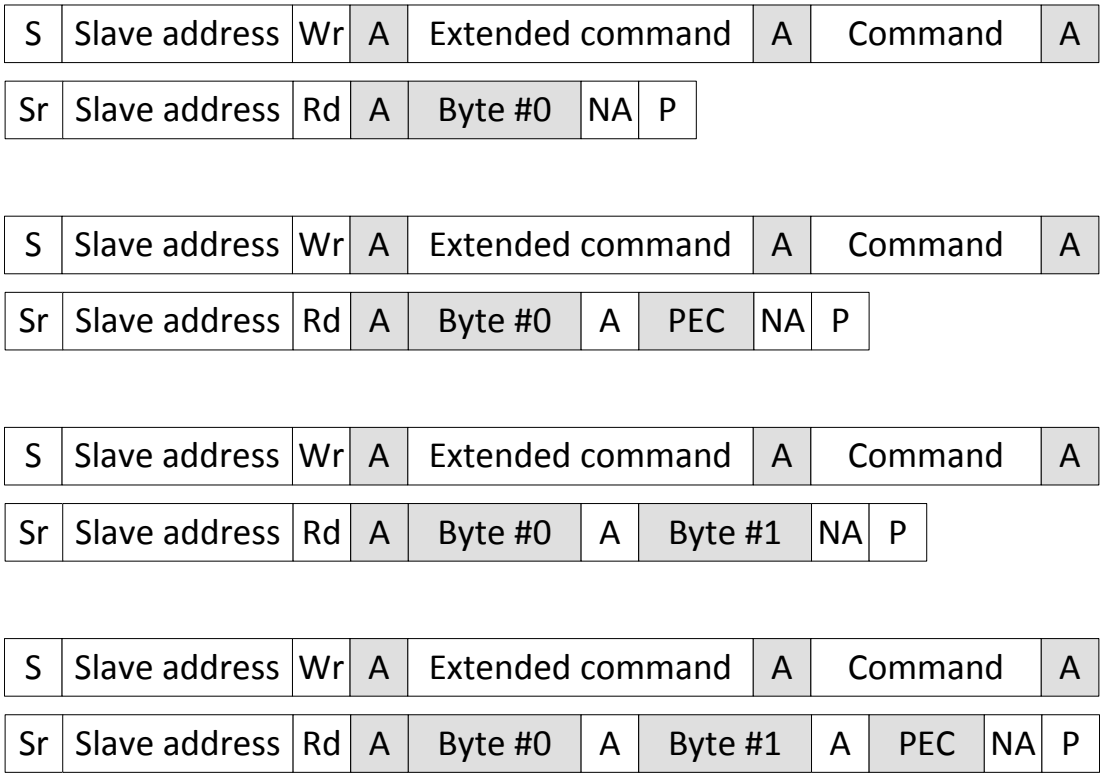


Figure 27-26. Extended Command Read Byte and Read Word Messages With and Without PEC

27.4.2.12 Group Command

The Group Command (Figure 27-27) protocol is used to send commands to more than one device within the same message. When devices on the bus detect the stop condition at the conclusion of the Group Command message, the received commands are executed concurrently. To initiate a Group Command, the GRP_CMD bit within the PMBMC register must be set when programming the slave address for the first device in the message. The rest of the message is processed as a write byte/word message. At the conclusion of the first part of the Group Command message, the firmware programs the next device address in the PMBMC register. The PMBus module will send a repeated start on the bus and begin the next part of the message. When programming the last device address of the Group Command message, the firmware must disable the GRP_CMD bit when programming the PMBMC register.

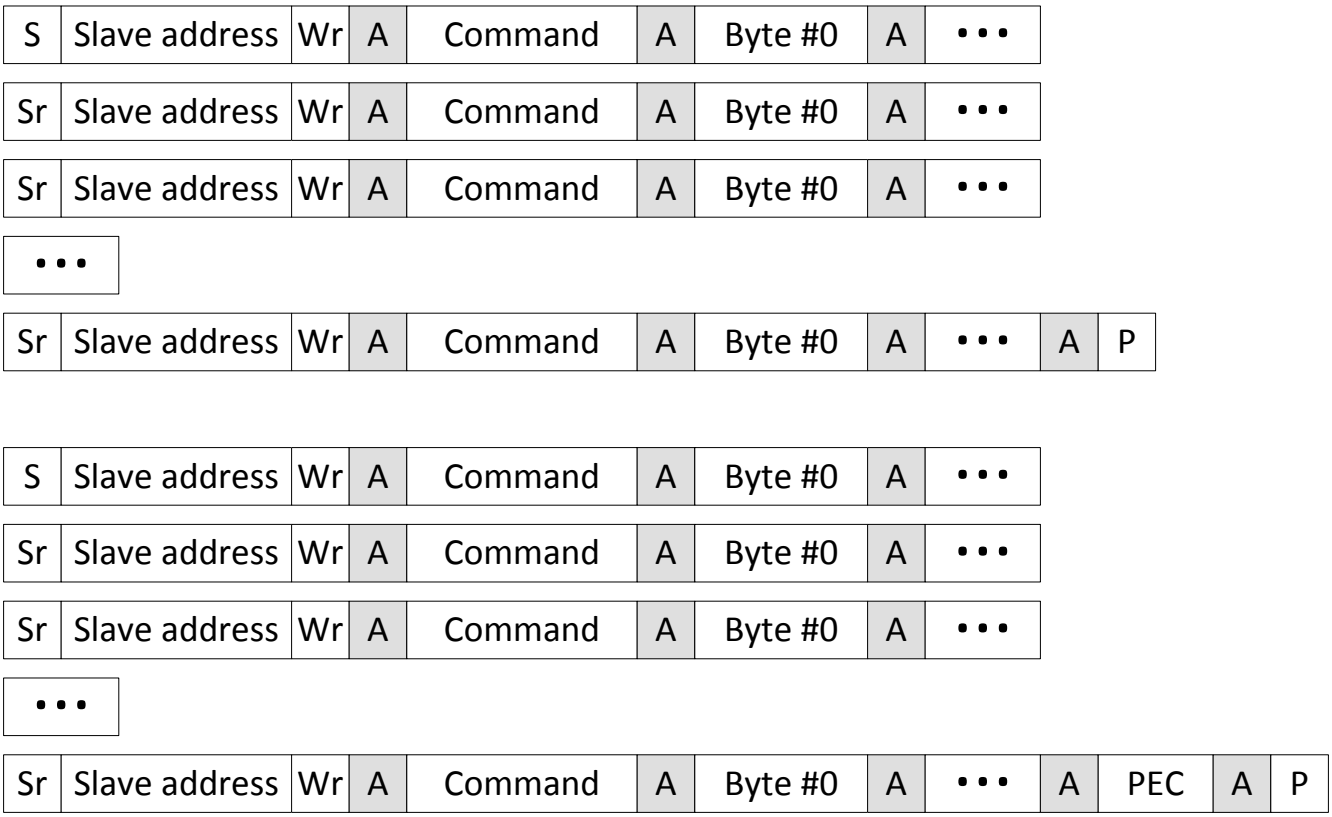


Figure 27-27. Group Command Message With and Without PEC

27.5 PMBus Registers

This section describes the Power-Management Bus module Registers.

27.5.1 PMBUS Base Address Table

Table 27-1. PMBUS Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
PmbusaRegs	PMBUS_REGS	PMBUSA_BASE	0x0000_6400	YES	YES	YES	YES	YES

27.5.2 PMBUS_REGS Registers

Table 27-2 lists the memory-mapped registers for the PMBUS_REGS registers. All register offset addresses not listed in Table 27-2 should be considered as reserved locations and the register contents should not be modified.

Table 27-2. PMBUS_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	PMBMC	PMBUS Master Mode Control Register	EALLOW	Go
2h	PMBTXBUF	PMBUS Transmit Buffer		Go
4h	PMBRXBUF	PMBUS Receive buffer		Go
6h	PMBACK	PMBUS Acknowledge Register		Go
8h	PMBSTS	PMBUS Status Register		Go
Ah	PMBINTM	PMBUS Interrupt Mask Register	EALLOW	Go
Ch	PMBSC	PMBUS Slave Mode Configuration Register	EALLOW	Go
Eh	PMBHSA	PMBUS Hold Slave Address Register		Go
10h	PMBCTRL	PMBUS Control Register	EALLOW	Go
12h	PMBTIMCTL	PMBUS Timing Control Register	EALLOW	Go
14h	PMBTIMCLK	PMBUS Clock Timing Register	EALLOW	Go
16h	PMBTIMSTSETUP	PMBUS Start Setup Time Register	EALLOW	Go
18h	PMBTIMBIDLE	PMBUS Bus Idle Time Register	EALLOW	Go
1Ah	PMBTIMLOWTIMOUT	PMBUS Clock Low Timeout Value Register	EALLOW	Go
1Ch	PMBTIMHIGHTIMOUT	PMBUS Clock High Timeout Value Register	EALLOW	Go

Complex bit access types are encoded to fit into small table cells. Table 27-3 shows the codes that are used for access types in this section.

Table 27-3. PMBUS_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
RC	R C	Read to Clear
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

27.5.2.1 PMBMC Register (Offset = 0h) [Reset = 0h]

PMBMC is shown in Figure 27-28 and described in Table 27-4.

Return to the Summary Table.

PMBUS Master Mode Control Register

Figure 27-28. PMBMC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED			PRC_CALL	GRP_CMD	PEC_ENA	EXT_CMD	CMD_ENA
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
BYTE_COUNT							
R/W-0h							
7	6	5	4	3	2	1	0
SLAVE_ADDR							RW
R/W-0h							R/W-0h

Table 27-4. PMBMC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-21	RESERVED	R	0h	Reserved
20	PRC_CALL	R/W	0h	0 = Default state for all messages besides Process Call message 1 = Enables transmission of Process Call message Reset type: SYSRSn
19	GRP_CMD	R/W	0h	0 = Default state for all messages besides Group Command message 1 = Enables transmission of Group Command message Reset type: SYSRSn
18	PEC_ENA	R/W	0h	0 = Disables PEC processing 1 = Enables PEC byte transmission/reception Reset type: SYSRSn
17	EXT_CMD	R/W	0h	0 = Use 1 byte for Command Code 1 = Use 2 bytes for Command Code Reset type: SYSRSn
16	CMD_ENA	R/W	0h	0 = Disables use of command code on Master initiated messages (1 = Enables use of command code on Master initiated messages Reset type: SYSRSn
15-8	BYTE_COUNT	R/W	0h	Indicates number of data bytes transmitted in current message. Byte count does not include any device addresses, command words or block lengths in block messages. In block messages, the PMBus Interface automatically inserts the block length into the message based on the byte count setting. The firmware only needs to load the address, command words and data to be transmitted. PMBus Interface supports byte writes up to 255 bytes. Reset type: SYSRSn
7-1	SLAVE_ADDR	R/W	0h	Specifies the address of the slave to which the current message is directed towards. Reset type: SYSRSn
0	RW	R/W	0h	0 = Message is a write transaction (data from Master to Slave) 1 = Message is a read transaction (data from Slave to Master) Reset type: SYSRSn

27.5.2.2 PMBTXBUF Register (Offset = 2h) [Reset = 0h]

PMBTXBUF is shown in Figure 27-29 and described in Table 27-5.

Return to the Summary Table.

PMBUS Transmit Buffer

Figure 27-29. PMBTXBUF Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TXDATA																															
R/W-0h																															

Table 27-5. PMBTXBUF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TXDATA	R/W	0h	Bits 31-24: BYTE3 - Last data byte transmitted from Transmit Data Buffer Bits 23-16: BYTE2 - Third data byte transmitted from Transmit Data Buffer Bits 15-8: BYTE1 - Second data byte transmitted from Transmit Data Buffer Bits 7-0: BYTE0 - First data byte transmitted from Transmit Data Buffer Reset type: SYSRSn

27.5.2.3 PMBRXBUF Register (Offset = 4h) [Reset = 0h]

PMBRXBUF is shown in Figure 27-30 and described in Table 27-6.

Return to the Summary Table.

PMBUS Receive buffer

Figure 27-30. PMBRXBUF Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RXDATA																															
R-0h																															

Table 27-6. PMBRXBUF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	RXDATA	R	0h	Bits 31-24: BYTE3 - Last data byte received in Receive Data Buffer Bits 23-16: BYTE2 - Third data byte received in Receive Data Buffer Bits 15-8: BYTE1 - Second data byte received in Receive Data Buffer Bits 7-0: BYTE0 - First data byte received in Receive Data Buffer Reset type: SYSRSn

27.5.2.4 PMBACK Register (Offset = 6h) [Reset = 0h]

PMBACK is shown in Figure 27-31 and described in Table 27-7.

Return to the Summary Table.

PMBUS Acknowledge Register

Figure 27-31. PMBACK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED															ACK
R-0h															R/W-0h

Table 27-7. PMBACK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	ACK	R/W	0h	0 = NACK received data 1 = Acknowledge received data, bit clears upon issue of ACK on PMBus Reset type: SYSRSn

27.5.2.5 PMBSTS Register (Offset = 8h) [Reset = 00340000h]

PMBSTS is shown in Figure 27-32 and described in Table 27-8.

Return to the Summary Table.

PMBUS Status Register

Figure 27-32. PMBSTS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED		SCL_RAW	SDA_RAW	CONTROL_RAW	ALERT_RAW	CONTROL_EDGE	ALERT_EDGE
R-0h		R-1h	R-1h	R-0h	R-1h	RC-0h	RC-0h
15	14	13	12	11	10	9	8
MASTER	LOST_ARB	BUS_FREE	UNIT_BUSY	RPT_START	SLAVE_ADDR_READY	CLK_HIGH_DETECTED	CLK_LOW_TIMEOUT
RC-0h	RC-0h	RC-0h	RC-0h	RC-0h	RC-0h	RC-0h	RC-0h
7	6	5	4	3	2	1	0
PEC_VALID	NACK	EOM	DATA_REQUEST	DATA_READY	RD_BYTE_COUNT		
RC-0h	RC-0h	RC-0h	RC-0h	RC-0h	RC-0h		

Table 27-8. PMBSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R	0h	Reserved
21	SCL_RAW	R	1h	0 = PMBus clock pin observed at logic level low 1 = PMBus clock pin observed at logic level high Reset type: SYSRSn
20	SDA_RAW	R	1h	0 = PMBus data pin observed at logic level low 1 = PMBus data pin observed at logic level high Reset type: SYSRSn
19	CONTROL_RAW	R	0h	0 = Control pin observed at logic level low 1 = Control pin observed at logic level high Reset type: SYSRSn
18	ALERT_RAW	R	1h	0 = Alert pin observed at logic level low 1 = Alert pin observed at logic level high Reset type: SYSRSn
17	CONTROL_EDGE	RC	0h	0 = Control pin has not transitioned 1 = Control pin has been asserted by another device on PMBus Reset type: SYSRSn
16	ALERT_EDGE	RC	0h	0 = Alert pin has not transitioned 1 = Alert pin has been asserted by another device on PMBus Reset type: SYSRSn
15	MASTER	RC	0h	0 = PMBus Interface in Slave Mode or Idle Mode 1 = PMBus Interface in Master Mode Reset type: SYSRSn
14	LOST_ARB	RC	0h	0 = Master has attained control of PMBus 1 = Master has lost arbitration and control of PMBus Reset type: SYSRSn
13	BUS_FREE	RC	0h	0 = PMBus processing current message 1 = PMBus available for new message Reset type: SYSRSn

Table 27-8. PMBSTS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	UNIT_BUSY	RC	0h	0 = PMBus Interface is idle, ready to transmit/receive message 1 = PMBus Interface is busy, processing current message Reset type: SYSRSn
11	RPT_START	RC	0h	0 = No Repeated Start received by interface 1 = Repeated Start condition received by interface Reset type: SYSRSn
10	SLAVE_ADDR_READY	RC	0h	0 = Indicates no slave address is available for reading 1 = Slave address ready to be read from Receive Data Register (Bits 6:0) Reset type: SYSRSn
9	CLK_HIGH_DETECTED	RC	0h	0 = No Clock High condition detected 1 = Clock High exceeded 50us during message Reset type: SYSRSn
8	CLK_LOW_TIMEOUT	RC	0h	0 = No clock low timeout detected 1 = Clock low timeout detected, clock held low for greater than 35ms Reset type: SYSRSn
7	PEC_VALID	RC	0h	0 = Received PEC not valid (if EOM is asserted) 1 = Received PEC is valid Note: PEC_VALID status is don't care during the message. This will have a valid value only after EOM. Reset type: SYSRSn
6	NACK	RC	0h	0 = Data transmitted has been accepted by receiver 1 = Receiver has not accepted transmitted data Reset type: SYSRSn
5	EOM	RC	0h	0 = Message still in progress or PMBus in idle state. 1 = End of current message detected Reset type: SYSRSn
4	DATA_REQUEST	RC	0h	0 = No data needed by PMBus Interface 1 = PMBus Interface request additional data. PMBus clock stretching enabled to stall bus Reset type: SYSRSn
3	DATA_READY	RC	0h	0 = No data available for reading by processor 1 = PMBus Interface read buffer full, firmware required to read data prior to further bus activity. PMBus clock stretching enabled to stall bus until data is read by firmware. Reset type: SYSRSn
2-0	RD_BYTE_COUNT	RC	0h	0 = No received data 1 = 1 byte received. Data located in Receive Data Register, Bits 7-0 2 = 2 bytes received. Data located in Receive Data Register, Bits 15-0 3 = 3 bytes received. Data located in Receive Data Register, Bits 23-0 4 = 4 bytes received. Data located in Receive Data Register, Bits 31-0 Reset type: SYSRSn

27.5.2.6 PMBINTM Register (Offset = Ah) [Reset = 3FFh]

PMBINTM is shown in Figure 27-33 and described in Table 27-9.

Return to the Summary Table.

PMBUS Interrupt Mask Register

Figure 27-33. PMBINTM Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						CLK_HIGH_DETECT	LOST_ARB
R-0h						R/W-1h	R/W-1h
7	6	5	4	3	2	1	0
CONTROL	ALERT	EOM	SLAVE_ADDR_READY	DATA_REQUEST	DATA_READY	BUS_LOW_TIMEOUT	BUS_FREE
R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h	R/W-1h

Table 27-9. PMBINTM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9	CLK_HIGH_DETECT	R/W	1h	0 = Generates interrupt if clock high exceeds 50us during message 1 = Disables interrupt generation for Clock High detection Reset type: SYSRSn
8	LOST_ARB	R/W	1h	0 = Generates interrupt upon assertion of Lost Arbitration flag 1 = Disables interrupt generation upon assertion of Lost Arbitration flag Reset type: SYSRSn
7	CONTROL	R/W	1h	0 = Generates interrupt upon assertion of Control flag 1 = Disables interrupt generation upon assertion of Control flag Reset type: SYSRSn
6	ALERT	R/W	1h	0 = Generates interrupt upon assertion of Alert flag 1 = Disables interrupt generation upon assertion of Alert flag Reset type: SYSRSn
5	EOM	R/W	1h	0 = Generates interrupt upon assertion of End of Message flag 1 = Disables interrupt generation upon assertion of End of Message flag Reset type: SYSRSn
4	SLAVE_ADDR_READY	R/W	1h	0 = Generates interrupt upon assertion of Slave Address Ready flag 1 = Disables interrupt generation upon assertion of Slave Address Ready flag Reset type: SYSRSn
3	DATA_REQUEST	R/W	1h	0 = Generates interrupt upon assertion of Data Request flag 1 = Disables interrupt generation upon assertion of Data Request flag Reset type: SYSRSn
2	DATA_READY	R/W	1h	0 = Generates interrupt upon assertion of Data Ready flag 1 = Disables interrupt generation upon assertion of Data Ready flag Reset type: SYSRSn

Table 27-9. PMBINTM Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	BUS_LOW_TIMEOUT	R/W	1h	0 = Generates interrupt upon assertion of Clock Low Timeout flag 1 = Disables interrupt generation upon assertion of Clock Low Timeout flag Reset type: SYSRSn
0	BUS_FREE	R/W	1h	0 = Generates interrupt upon assertion of Bus Free flag 1 = Disables interrupt generation upon assertion of Bus Free flag Reset type: SYSRSn

27.5.2.7 PMBSC Register (Offset = Ch) [Reset = 00607F7Ch]

PMBSC is shown in Figure 27-34 and described in Table 27-10.

Return to the Summary Table.

PMBUS Slave Mode Configuration Register

Figure 27-34. PMBSC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED	RX_BYTE_ACK_CNT		MAN_CMD	TX_PEC	TX_COUNT		
R-0h	R/W-3h		R/W-0h	R/W-0h	R/W-0h		
15	14	13	12	11	10	9	8
PEC_ENA	SLAVE_MASK						
R/W-0h	R/W-7Fh						
7	6	5	4	3	2	1	0
MAN_SLAVE_ACK	SLAVE_ADDR						
R/W-0h	R/W-7Ch						

Table 27-10. PMBSC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R	0h	Reserved
22-21	RX_BYTE_ACK_CNT	R/W	3h	Configures number of data bytes to automatically acknowledge when receiving data in slave mode. 00 = 1 byte received by slave. Firmware is required to manually acknowledge every received byte. 01 = 2 bytes received by slave. Hardware automatically acknowledges the first received byte. Firmware is required to manually acknowledge after the second received byte. 10 = 3 bytes received by slave. Hardware automatically acknowledges the first 2 received bytes. Firmware is required to manually acknowledge after the third received byte. 11 = 4 bytes received by slave. Hardware automatically acknowledges the first 3 received bytes. Firmware is required to manually acknowledge after the fourth received byte Reset type: SYSRSn
20	MAN_CMD	R/W	0h	0 = Slave automatically acknowledges received command code 1 = Data Request flag generated after receipt of command code, firmware required to issue ACK to continue message Reset type: SYSRSn
19	TX_PEC	R/W	0h	Asserted when the slave needs to send a PEC byte at end of message. PMBus Interface will transmit the calculated PEC byte after transmitting the number of data bytes indicated by TX Byte Cnt(Bits 18:16). Reset type: SYSRSn

Table 27-10. PMBSC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-16	TX_COUNT	R/W	0h	0 = No bytes valid 1 = One byte valid, Byte #0 (Bits 7:0 of Transmit Data Register) 2 = Two bytes valid, Bytes #0 and #1 (Bits 15:0 of Transmit Data Register) 3 = Three bytes valid, Bytes #0-2 (Bits 23:0 of Transmit Data Register) 4 = Four bytes valid, Bytes #0-3 (Bits 31:0 of Transmit Data Register) Reset type: SYSRSn
15	PEC_ENA	R/W	0h	0 = PEC processing disabled 1 = PEC processing enabled Reset type: SYSRSn
14-8	SLAVE_MASK	R/W	7Fh	Used in address detection, the slave mask enables acknowledgement of multiple device addresses by the slave. Writing a '0' to a bit within the slave mask enables the corresponding bit in the slave address to be either '1' or '0' and still allow for a match. Writing a '0' to all bits in the mask enables the PMBus Interface to acknowledge any device address. Upon power-up, the slave mask defaults to 7Fh, indicating the slave will only acknowledge the address programmed into the Slave Address (Bits 6-0). Reset type: SYSRSn
7	MAN_SLAVE_ACK	R/W	0h	0 = Slave automatically acknowledges device address specified in SLAVE_ADDR, Bits 6:0 1 = Enables the Manual Slave Address Acknowledgement Mode. Firmware is required to read received address and acknowledge on every message Note: When bit 31 (I2C_mode) of PMBCTRL register is set it is recommended to use manual acknowledging of slave address only (MAN_SLAVE_ACK =1). Reset type: SYSRSn
6-0	SLAVE_ADDR	R/W	7Ch	Configures the current device address of the slave. Used in automatic slave address acknowledge mode (default mode). The PMBus Interface will compare the received device address with the value stored in the Slave Address bits and the mask configured in the Slave Mask bits. If matching, the slave will acknowledge the device address. Reset type: SYSRSn

27.5.2.8 PMBHSA Register (Offset = Eh) [Reset = 0h]

PMBHSA is shown in Figure 27-35 and described in Table 27-11.

Return to the Summary Table.

PMBUS Hold Slave Address Register

Figure 27-35. PMBHSA Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
SLAVE_ADDR							SLAVE_RW
R-0h							R-0h

Table 27-11. PMBHSA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-1	SLAVE_ADDR	R	0h	Stored device address acknowledged by the slave Reset type: SYSRSn
0	SLAVE_RW	R	0h	Stored R/W bit from address acknowledged by the slave 0 = Write Access 1 = Read Access Reset type: SYSRSn

27.5.2.9 PMBCTRL Register (Offset = 10h) [Reset = 00200000h]

PMBCTRL is shown in Figure 27-36 and described in Table 27-12.

Return to the Summary Table.

PMBUS Control Register

Figure 27-36. PMBCTRL Register

31	30	29	28	27	26	25	24
I2CMODE	RESERVED			CLKDIV			
R/W-0h	R-0h			R/W-0h			
23	22	21	20	19	18	17	16
CLKDIV	MASTER_EN	SLAVE_EN	CLK_LO_DIS	IBIAS_B_EN	IBIAS_A_EN	SCL_DIR	SCL_VALUE
R/W-0h	R/W-0h	R/W-1h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
SCL_MODE	SDA_DIR	SDA_VALUE	SDA_MODE	CNTL_DIR	CNTL_VALUE	CNTL_MODE	ALERT_DIR
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
ALERT_VALUE	ALERT_MODE	CNTL_INT_EDGE	RESERVED	FAST_MODE	BUS_LO_INT_EDGE	ALERT_EN	RESET
R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 27-12. PMBCTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	I2CMODE	R/W	0h	0 = PMBUS mode 1 = I2C mode Reset type: SYSRSn
30-28	RESERVED	R	0h	Reserved
27-23	CLKDIV	R/W	0h	The clock to the PMBUS transmit/receive FSMs (FSM_CLK) is divided version of the SYSCLK clock. Frequency(FSM_CLK) = Frequency(SYSCLK)/(CLKDIV+1) Note: FSM_CLK should be less than (or) equal to 10MHz. Reset type: SYSRSn
22	MASTER_EN	R/W	0h	0 = Disables PMBus Master capability 1 = Enables PMBus Master capability Reset type: SYSRSn
21	SLAVE_EN	R/W	1h	0 = Disables PMBus Slave capability 1 = Enables PMBus Slave capability Reset type: SYSRSn
20	CLK_LO_DIS	R/W	0h	0 = Clock Low Timeout Enabled 1 = Clock Low Timeout Disabled Reset type: SYSRSn
19	IBIAS_B_EN	R/W	0h	0 = Disables Current Source for PMBUS address detection thru ADC 1 = Enables Current Source for PMBUS address detection thru ADC Reset type: SYSRSn
18	IBIAS_A_EN	R/W	0h	0 = Disables Current Source for PMBUS address detection thru ADC 1 = Enables Current Source for PMBUS address detection thru ADC Reset type: SYSRSn
17	SCL_DIR	R/W	0h	0 = PMBus clock pin configured as output 1 = PMBus clock pin configured as input Reset type: SYSRSn
16	SCL_VALUE	R/W	0h	0 = PMBus clock pin driven low in GPIO Mode 1 = PMBus clock pin driven high in GPIO Mode Reset type: SYSRSn

Table 27-12. PMBCTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15	SCL_MODE	R/W	0h	0 = PMBus clock pin configured in functional mode 1 = PMBus clock pin configured as GPIO Reset type: SYSRSn
14	SDA_DIR	R/W	0h	0 = PMBus data pin configured as output 1 = PMBus data pin configured as input Reset type: SYSRSn
13	SDA_VALUE	R/W	0h	0 = PMBus data pin driven low in GPIO Mode 1 = PMBus data pin driven high in GPIO Mode Reset type: SYSRSn
12	SDA_MODE	R/W	0h	0 = PMBus data pin driven low in GPIO Mode 1 = PMBus data pin driven high in GPIO Mode Reset type: SYSRSn
11	CNTL_DIR	R/W	0h	0 = Control pin configured as output 1 = Control pin configured as input Reset type: SYSRSn
10	CNTL_VALUE	R/W	0h	0 = Control pin driven low in GPIO Mode 1 = Control pin driven high in GPIO Mode Reset type: SYSRSn
9	CNTL_MODE	R/W	0h	0 = Control pin configured in functional mode (Default) 1 = Control pin configured as GPIO Reset type: SYSRSn
8	ALERT_DIR	R/W	0h	0 = Alert pin configured as output 1 = Alert pin configured as input Reset type: SYSRSn
7	ALERT_VALUE	R/W	0h	0 = Alert pin driven low in GPIO Mode 1 = Alert pin driven high in GPIO Mode Reset type: SYSRSn
6	ALERT_MODE	R/W	0h	0 = Alert pin configured in functional mode 1 = Aler3 pin configured as GPIO Reset type: SYSRSn
5	CNTL_INT_EDGE	R/W	0h	0 = Interrupt generated on falling edge of Control 1 = Interrupt generated on rising edge of Control Reset type: SYSRSn
4	RESERVED	R	0h	Reserved
3	FAST_MODE	R/W	0h	0 = Standard 100 KHz mode enabled 1 = Fast Mode enabled (400KHz operation on PMBus) Reset type: SYSRSn
2	BUS_LO_INT_EDGE	R/W	0h	0 = Interrupt generated on rising edge of clock low timeout 1 = Interrupt generated on falling edge of clock low timeout Reset type: SYSRSn
1	ALERT_EN	R/W	0h	0 = PMBus Alert is not driven by slave, pulled up high on PMBus 1 = PMBus Alert driven low by slave Reset type: SYSRSn
0	RESET	R/W	0h	0 = No reset of internal state machines (Default) 1 = Control state machines are reset to initial states Note: Status register PMBSTS should be explicitly cleared by reading the register after softrest as this will not be cleared by Software Reset. Reset type: SYSRSn

27.5.2.10 PMBTIMCTL Register (Offset = 12h) [Reset = 0h]

PMBTIMCTL is shown in Figure 27-37 and described in Table 27-13.

Return to the Summary Table.

PMBUS Timing Control Register

Figure 27-37. PMBTIMCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							TIM_OVERRIDE
R-0h							R/W-0h

Table 27-13. PMBTIMCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	TIM_OVERRIDE	R/W	0h	0 PMBUS FSMs uses the default settings of the timing parameters. 1 PMBUS FSMs would use the settings in following registers: * PMBTIMCLK * PMBTIMSTSETUP * PMBTIMBIDLE * PMBTIMLOWTIMOUT * PMBTIMHIGHTIMOUT Reset type: SYSRSn

27.5.2.11 PMBTIMCLK Register (Offset = 14h) [Reset = 0060002Fh]

PMBTIMCLK is shown in Figure 27-38 and described in Table 27-14.

Return to the Summary Table.

PMBUS Clock Timing Register

Figure 27-38. PMBTIMCLK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED								CLK_FREQ							
R-0h								R/W-60h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								CLK_HIGH_LIMIT							
R-0h								R/W-2Fh							

Table 27-14. PMBTIMCLK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	CLK_FREQ	R/W	60h	Defines the number of PMBUS FSM input clock in the PMBUS master clock period. Number of FSM clocks in the one clock period = (CLK_HIGH_LIMIT+4) Reset type: SYSRSn
15-8	RESERVED	R	0h	Reserved
7-0	CLK_HIGH_LIMIT	R/W	2Fh	Defines the number of PMBUS FSM input clock in the PMBUS master clock high pulse. Number of FSM clocks in the one clock high pulse = (CLK_HIGH_LIMIT+3) Reset type: SYSRSn

27.5.2.12 PMBTIMSTSETUP Register (Offset = 16h) [Reset = 2Fh]

PMBTIMSTSETUP is shown in Figure 27-39 and described in Table 27-15.

Return to the Summary Table.

PMBUS Start Setup Time Register

Figure 27-39. PMBTIMSTSETUP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								TSU_STA							
R-0h																								R/W-2Fh							

Table 27-15. PMBTIMSTSETUP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	TSU_STA	R/W	2Fh	Determines the Setup time between last rise edge of the PMBUS master clock and the next start edge, TSU_STA value defines the setup time in terms of PMBUS FSM clock cycles. Reset type: SYSRSn

27.5.2.13 PMBTIMBIDLE Register (Offset = 18h) [Reset = 1F3h]

PMBTIMBIDLE is shown in Figure 27-40 and described in Table 27-16.

Return to the Summary Table.

PMBUS Bus Idle Time Register

Figure 27-40. PMBTIMBIDLE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																						BUSIDLE									
R-0h																						R/W-1F3h									

Table 27-16. PMBTIMBIDLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9-0	BUSIDLE	R/W	1F3h	Determines the duration for which PMBUS clock and Data are 1 , to conclude that the bus is IDLE. BUSIDLE value is in terms of number of PMBUS FSM clock cycles. Reset type: SYSRSn

27.5.2.14 PMBTIMLOWTIMOUT Register (Offset = 1Ah) [Reset = 0005572Fh]

PMBTIMLOWTIMOUT is shown in Figure 27-41 and described in Table 27-17.

Return to the Summary Table.

PMBUS Clock Low Timeout Value Register

Figure 27-41. PMBTIMLOWTIMOUT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												CLKLOWTIMOUT																			
R-0h												R/W-0005572Fh																			

Table 27-17. PMBTIMLOWTIMOUT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19-0	CLKLOWTIMOUT	R/W	0005572Fh	Determines the duration for which PMBUS clock if low , will result in a clock low timeout condition. CLKLOWTIMOUT value is in terms of number of PMBUS FSM clock cycles. Reset type: SYSRSn

27.5.2.15 PMBTIMHIGHTIMOUT Register (Offset = 1Ch) [Reset = 1F3h]

PMBTIMHIGHTIMOUT is shown in Figure 27-42 and described in Table 27-18.

Return to the Summary Table.

PMBUS Clock High Timeout Value Register

Figure 27-42. PMBTIMHIGHTIMOUT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED						CLKHIGHTIMOUT									
R-0h						R/W-1F3h									

Table 27-18. PMBTIMHIGHTIMOUT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9-0	CLKHIGHTIMOUT	R/W	1F3h	Determines the duration for which PMBUS clock if high , will result in a clock high timeout condition. CLKHIGHTIMOUT value is in terms of number of PMBUS FSM clock cycles. Reset type: SYSRSn

27.5.3 PMBUS Registers to Driverlib Functions

Table 27-19. PMBUS Registers to Driverlib Functions

File	Driverlib Function
PMBMC	
pmbus.h	PMBus_configMaster
pmbus.h	PMBus_setSlaveAddress
PMBTXBUF	
pmbus.c	PMBus_putSlaveData
pmbus.c	PMBus_putMasterData
PMBRXBUF	
pmbus.c	PMBus_getData
PMBACK	
pmbus.c	PMBus_ackAddress
pmbus.c	PMBus_ackCommand
pmbus.h	PMBus_ackTransaction
pmbus.h	PMBus_nackTransaction
PMBSTS	
pmbus.c	PMBus_getInterruptStatus
pmbus.h	PMBus_getStatus
PMBINTM	
pmbus.c	PMBus_initSlaveMode
pmbus.c	PMBus_configSlave
pmbus.c	PMBus_initMasterMode
pmbus.c	PMBus_configModuleClock
pmbus.c	PMBus_configBusClock
pmbus.h	PMBus_enableInterrupt
pmbus.h	PMBus_disableInterrupt

Table 27-19. PMBUS Registers to Driverlib Functions (continued)

File	Driverlib Function
pmbus.h	PMBus_enableI2CMode
pmbus.h	PMBus_disableI2CMode
PMBSC	
pmbus.c	PMBus_initSlaveMode
pmbus.c	PMBus_configSlave
pmbus.c	PMBus_putSlaveData
pmbus.c	PMBus_ackAddress
pmbus.c	PMBus_ackCommand
pmbus.h	PMBus_setOwnAddress
PMBHSA	
pmbus.c	PMBus_verifyPEC
pmbus.h	PMBus_getOwnAddress
pmbus.h	PMBus_getCurrentAccessType
PMBCTRL	
pmbus.c	PMBus_initSlaveMode
pmbus.c	PMBus_initMasterMode
pmbus.c	PMBus_configModuleClock
pmbus.c	PMBus_configBusClock
pmbus.h	PMBus_disableModule
pmbus.h	PMBus_enableModule
pmbus.h	PMBus_enableI2CMode
pmbus.h	PMBus_disableI2CMode
pmbus.h	PMBus_assertAlertLine
pmbus.h	PMBus_deassertAlertLine
pmbus.h	PMBus_setCtrlIntEdge
pmbus.h	PMBus_setClkLowTimeoutIntEdge
PMBTIMCTL	
pmbus.c	PMBus_configBusClock
PMBTIMCLK	
pmbus.c	PMBus_configBusClock
PMBTIMSTSETUP	
pmbus.c	PMBus_configBusClock
PMBTIMBIDLE	
pmbus.c	PMBus_configBusClock
PMBTIMLOWTIMOUT	
pmbus.c	PMBus_configBusClock
PMBTIMHIGHTIMOUT	
pmbus.c	PMBus_configBusClock

This page intentionally left blank.

Chapter 28

Controller Area Network (CAN)

This chapter describes the Controller Area Network (CAN) module. CAN is a serial communications protocol that efficiently supports distributed real-time control with a high level of reliability. The CAN module supports bit rates up to 1 Mbit/s and is compliant with the ISO11898-1 (CAN 2.0B) protocol specification.

Further information can be found in:

- *Calculator for CAN Bit Timing Parameters Application Report*
- *Programming Examples and Debug Strategies for the DCAN Module Application Report*
- *Configurable Error Generator for Controller Area Network Application Report*

28.1 Introduction

This device uses the CAN IP known as DCAN.

28.1.1 DCAN Related Collateral

Foundational Materials

- Automotive CAN Overview and Training (Video)
- C2000 Academy - Communications
  - Refer to the DCAN section
- CAN Physical layer (Video)
- CAN and CAN FD Overview (Video)
- CAN and CAN FD Protocol (Video)

Expert Materials

- *Calculator for CAN Bit Timing Parameters Application Report*

28.1.2 Features

The CAN module implements the following features:

- Complies with ISO11898-1 (Bosch® CAN protocol specification 2.0 A and B)
- Bit rates up to 1 Mbps
- Multiple clock sources
- 32 message objects (“message objects” are also referred to as “mailboxes” in this document; the two terms are used interchangeably), each with the following properties:
  - Configurable as receive or transmit
  - Configurable with standard (11-bit) or extended (29-bit) identifier
  - Supports programmable identifier receive mask
  - Supports data and remote frames
  - Holds 0 to 8 bytes of data
  - Parity-checked configuration and data RAM
- Individual identifier mask for each message object
- Programmable FIFO mode for message objects
- Programmable loop-back modes for self-test operation
- Suspend mode for debug support
- Software module reset
- Automatic bus-on, after bus-off state by a programmable 32-bit timer
- Message-RAM parity-check mechanism
- Two interrupt lines
- DMA support

Note

For a CAN bit clock of 120 MHz, the smallest bit rate possible is 4.6875 kbps.

Depending on the timing settings used, the accuracy of the on-chip zero-pin oscillator (specified in the data manual) may not meet the requirements of the CAN protocol. In this situation, an external clock source must be used.

28.1.3 Block Diagram

Figure 28-1 shows a block diagram of the CAN module. Following is a description of some of the main blocks.

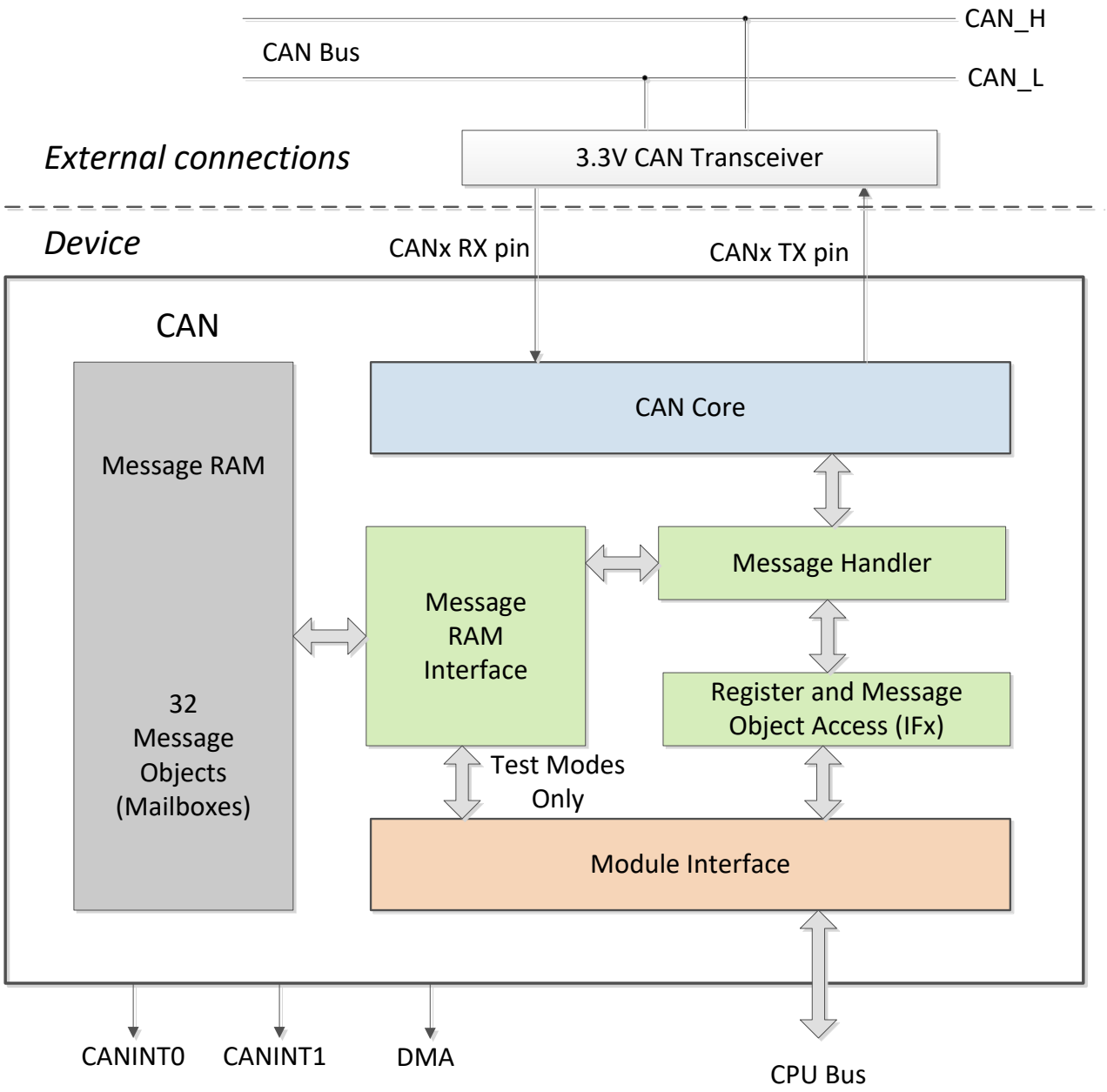


Figure 28-1. CAN Block Diagram

28.1.3.1 CAN Core

The CAN core consists of the CAN Protocol Controller and the Rx/Tx Shift register. It handles all ISO 118981 protocol functions.

28.1.3.2 Message Handler

The message handler is a state machine which controls the data transfer between the single-port Message RAM and the CAN Core's Rx/Tx Shift register. It also handles acceptance filtering and the interrupt request generation as programmed in the control registers.

28.1.3.3 Message RAM

The CAN message RAM enables the storage of 32 CAN messages.

28.1.3.4 Registers and Message Object Access (IFx)

Data consistency is ensured by indirect accesses to the message objects. During normal operation, all CPU and DMA accesses to the message RAM are done through Interface registers.

Three Interface register sets control the CPU read and write accesses to the Message RAM. There are two Interface register sets for read/write access (IF1 and IF2) and one Interface register set for read access only (IF3). See also Section 28.13. The Interface registers have the same word length as the message RAM.

In a dedicated test mode, the message RAM is memory-mapped and can be directly accessed.

28.2 Functional Description

The CAN module performs CAN protocol communication according to ISO 11898-1. The bit rate can be programmed to values up to 1 Mbps. A CAN transceiver chip is required for the connection to the physical layer (CAN bus).

For communication on a CAN network, individual message objects can be configured. The message objects and identifier masks are stored in the Message RAM.

All functions concerning the handling of messages are implemented in the message handler. These functions are: acceptance filtering; the transfer of messages between the CAN Core and the Message RAM; and the handling of transmission requests as well as the generation of interrupts or DMA requests.

The register set of the CAN may be accessed directly by the CPU through the module interface. These registers are used to control and configure the CAN core and the message handler, and to access the message RAM.

28.2.1 Configuring Device Pins

The GPIO mux registers must be configured to connect this peripheral to the device pins. To avoid glitches on the pins, the GPyGMUX bits must be configured first (while keeping the corresponding GPyMUX bits at the default of zero), followed by writing the GPyMUX register to the desired value.

Some I/O functionality is defined by GPIO register settings independent of this peripheral. For input signals, the GPIO input qualification should be set to asynchronous mode by setting the appropriate GPxQSELn register bits to 11b. The internal pull-ups can be configured in the GPyPUD register.

See the *General-Purpose Input/Output (GPIO)* chapter for more details on GPIO mux and settings.

28.2.2 Address/Data Bus Bridge

The CAN module uses a special addressing scheme to support byte accesses. It is recommended to only use 32-bit accesses to the CAN registers using the *HWREG_BP()* macro that uses the *__byte_peripheral_32()* intrinsic. If 16-bit accesses are to be used, the lower 16 bits should be written to the register's address, and the upper 16 bits should be written to the register's address, plus 2.

Because of the bus bridge, the view of the CAN module's register space through the Code Composer Studio™ (CCS) IDE memory window does not always match the actual addressing. When the view mode is 32-bit or 16-bit, even addresses are effectively duplicated; odd addresses should be ignored. When the view mode is 8-bit, even addresses from within the CAN module are duplicated into the odd addresses in the CCS memory view; odd addresses from the module are not displayed.

Table 28-1. CAN Register Access from Software

CAN Register Space			C28x 8-Bit		C28x 16-Bit		C28x 32-Bit	
Address	Name	Data	Access	Data	Access	Data	Address	Data
0x00	CAN_CTL	0x33221100	__byte((int *)0x00,0)	0x0000	(*((short *)(0x00)))	0x1100	(*((long *)(0x00)))	0x33221100
0x04	CAN_ES	0x77665544	__byte((int *)0x01,0)	0x0011	(*((short *)(0x01)))	0x1100	(*((long *)(0x01)))	0x33221100
0x08	CAN_ERRC	0xBBAA9988	__byte((int *)0x02,0)	0x0022	(*((short *)(0x02)))	0x3322	(*((long *)(0x02)))	0x33221100
0x0C	CAN_BTR	0xFFEEDDCC	__byte((int *)0x03,0)	0x0033	(*((short *)(0x03)))	0x3322	(*((long *)(0x03)))	0x33221100
			__byte((int *)0x04,0)	0x0044	(*((short *)(0x04)))	0x5544	(*((long *)(0x04)))	0x77665544
			__byte((int *)0x05,0)	0x0055	(*((short *)(0x05)))	0x5544	(*((long *)(0x05)))	0x77665544
			__byte((int *)0x06,0)	0x0066	(*((short *)(0x06)))	0x7766	(*((long *)(0x06)))	0x77665544
			__byte((int *)0x07,0)	0x0077	(*((short *)(0x07)))	0x7766	(*((long *)(0x07)))	0x77665544
			__byte((int *)0x08,0)	0x0088	(*((short *)(0x08)))	0x9988	(*((long *)(0x08)))	0xBBAA9988
			__byte((int *)0x09,0)	0x0099	(*((short *)(0x09)))	0x9988	(*((long *)(0x09)))	0xBBAA9988
			__byte((int *)0x0A,0)	0x00AA	(*((short *)(0x0A)))	0xBBAA	(*((long *)(0x0A)))	0xBBAA9988
			__byte((int *)0x0B,0)	0x00BB	(*((short *)(0x0B)))	0xBBAA	(*((long *)(0x0B)))	0xBBAA9988
			__byte((int *)0x0C,0)	0x00CC	(*((short *)(0x0C)))	0xDDCC	(*((long *)(0x0C)))	0xFFEEDDCC
			__byte((int *)0x0D,0)	0x00DD	(*((short *)(0x0D)))	0xDDCC	(*((long *)(0x0D)))	0xFFEEDDCC
			__byte((int *)0x0E,0)	0x00EE	(*((short *)(0x0E)))	0xFFEE	(*((long *)(0x0E)))	0xFFEEDDCC
			__byte((int *)0x0F,0)	0x00FF	(*((short *)(0x0F)))	0xFFEE	(*((long *)(0x0F)))	0xFFEEDDCC

Table 28-2. CAN Register Access from Code Composer Studio™ IDE

CCS 8-Bit			CCS 16-Bit			CCS 32-Bit	
Address	Displayed Data		Address	Displayed Data		Address	Displayed Data
0x00	0x00		0x00	0x1100		0x00	0x11001100
0x01	0x00		0x01	0x1100		0x02	0x33223322
0x02	0x22		0x02	0x3322		0x04	0x55445544
0x03	0x22		0x03	0x3322		0x06	0x77667766
0x04	0x44		0x04	0x5544		0x08	0x99889988
0x05	0x44		0x05	0x5544		0x0A	0xBBAABBAA
0x06	0x66		0x06	0x7766		0x0C	0xDDCCDDCC
0x07	0x66		0x07	0x7766		0x0E	0xFFEEFFEE
0x08	0x88		0x08	0x9988			
0x09	0x88		0x09	0x9988			
0x0A	0xAA		0x0A	0xBBAA			
0x0B	0xAA		0x0B	0xBBAA			
0x0C	0xCC		0x0C	0xDDCC			
0x0D	0xCC		0x0D	0xDDCC			
0x0E	0xEE		0x0E	0xFFEE			
0x0F	0xEE		0x0F	0xFFEE			

28.3 Operating Modes

28.3.1 Initialization

The initialization mode is entered either by software (by setting the **Init** bit in the CAN_CTL register), by hardware reset, or by going bus-off. While the Init bit is set, the message transfer from and to the CAN bus is stopped, and the status of the CAN_TX output is recessive (high). The CAN error counters are not updated. Setting the Init bit does not change any other configuration register.

To initialize the CAN Controller, the CPU has to configure the CAN Bit timing and those message objects which have to be used for CAN communication. Message objects which are not needed, can be deactivated with their MsgVal bits cleared.

The access to the Bit Timing Register for the configuration of the bit timing is enabled when both **Init** and CCE bits in the CAN Control register are set.

Clearing the Init bit finishes the software initialization. Afterwards, the bit stream processor (BSP) synchronizes itself to the data transfer on the CAN bus by waiting for the occurrence of a sequence of 11 consecutive recessive bits (= Bus Idle) before it can take part in bus activities and start the message transfer. For more details, see Section 28.12.

The initialization of the message objects is independent of the Init bit; however, all message objects should be configured to particular identifiers or set to "not-valid" before the message transfer is started.

It is possible to change the configuration of message objects during normal operation by the CPU. After setup and subsequent transfer of message object from interface registers to message RAM, the acceptance filtering will be applied to it when the modified message object number is same or smaller than the previously found message object. This assures data consistency even when changing message objects; for example, while there is a pending CAN frame reception.

28.3.2 CAN Message Transfer (Normal Operation)

Once the CAN is initialized and the Init bit is reset to zero, the CAN Core synchronizes itself to the CAN bus and is ready for communication.

Received messages are stored into their appropriate message objects if they pass acceptance filtering. The whole message (MSGID, DLC, and up to eight data bytes) is stored into the message object. As a consequence, for example, if the identifier mask is used, the MSGID bits which are masked to "don't care" may change in the message object when a received message is stored.

The CPU may read or write each message at any time via the Interface registers, as the message handler guarantees data consistency in case of concurrent accesses.

Messages to be transmitted can be updated by the CPU. If a permanent message object (MSGID, control bits set up during configuration, and setup for multiple CAN transfers) exists for the message, it is possible to only update the data bytes. If several transmit messages should be assigned to one message object, the whole message object has to be configured before the transmission of this message is requested.

The transmission of multiple message objects may be requested at the same time. They are subsequently transmitted, according to their internal priority. Messages may be updated or set to 'not valid' at any time, even if a requested transmission is still pending. However, the data bytes will be discarded if a message is updated before a pending transmission has started.

Depending on the configuration of the message object, a transmission may be automatically requested by the reception of a remote frame with a matching identifier.

28.3.2.1 Disabled Automatic Retransmission

According to the CAN Specification (see ISO11898, 6.3.3 Recovery Management), the CAN provides a mechanism to automatically retransmit frames which have lost arbitration or have been disturbed by errors during transmission. The frame transmission service will not be confirmed to the user before the transmission is successfully completed

By default, this automatic retransmission is enabled. It can be disabled by setting the DAR bit in the CAN Control register. Further details to this mode are provided in Section 28.11.3.

28.3.2.2 Auto-Bus-On

After the CAN has entered the bus-off state, the CPU can start a bus-off-recovery sequence by resetting the *Init* bit. If this is not done, the module will stay in bus-off state.

The CAN provides an automatic auto-bus-on feature which is enabled by the ABO bit. If set, the CAN will automatically start the bus-off-recovery sequence. The sequence can be delayed by a user-defined number of clock cycles.

Note

If the CAN module goes Bus-Off due to multiple CAN bus errors, it stops all bus activities and automatically sets the Init bit. Once the Init bit is cleared by the application (or due to the auto-bus-on feature), the device will wait for 129 occurrences of Bus Idle (equal to 129 * 11 consecutive recessive bits) before resuming normal operation. The Bus-Off recovery sequence cannot be shortened by setting or resetting the Init bit. At the end of the bus-off recovery sequence, the error counters will be reset. After the Init bit is reset, each time when a sequence of 11 recessive bits is monitored, a Bit0 Error code is written to the Error and Status Register. This enables the CPU to check whether the CAN bus is stuck at dominant or continuously disturbed, and to monitor the proceeding of the Bus-Off recovery sequence.

28.3.3 Test Modes

The CAN module provides several test modes which are mainly intended for self-test purposes. Figure 28-2 aids in understanding the various test modes. It should be viewed as representative of the module behavior, and not as a gate-accurate implementation of the module. For the sake of brevity, this diagram does not include the GPIO muxing or the I/O buffers.

For all test modes, the Test bit in the CAN Control register needs to be set to 1 to enable write access to the CAN_TEST register.

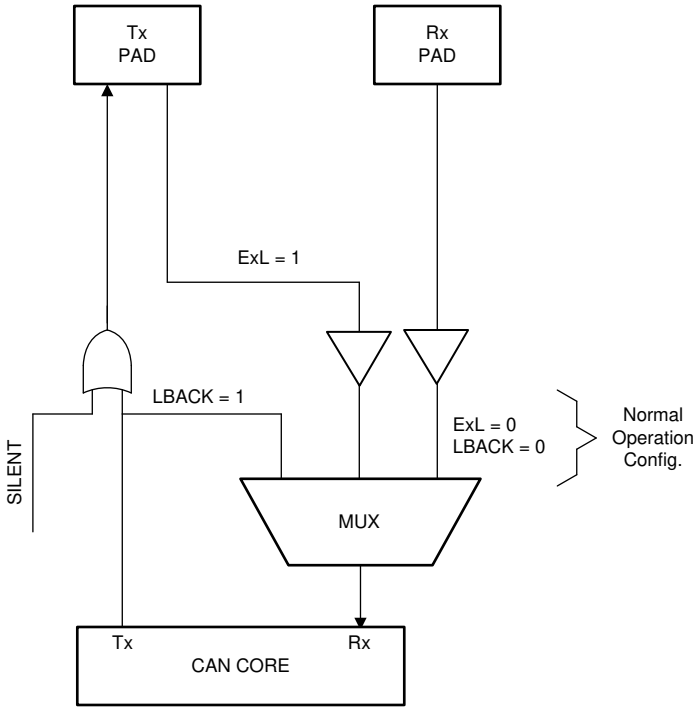


Figure 28-2. CAN_MUX

28.3.3.1 Silent Mode

The silent mode may be used to analyze the traffic on the CAN bus without affecting it by sending dominant bits (for example, acknowledge bit, overload flag, active error flag). The CAN is still able to receive valid data frames and valid remote frames, but it will not send any dominant bits. However, the received frames are internally routed to the CAN Core.

Figure 28-3 shows the connection of signals CAN_TX and CAN_RX to the CAN core in silent mode. Silent mode can be activated by setting the Silent bit in test register (CAN_TEST), to 1. In ISO 11898-1, the silent mode is called the bus monitoring mode.

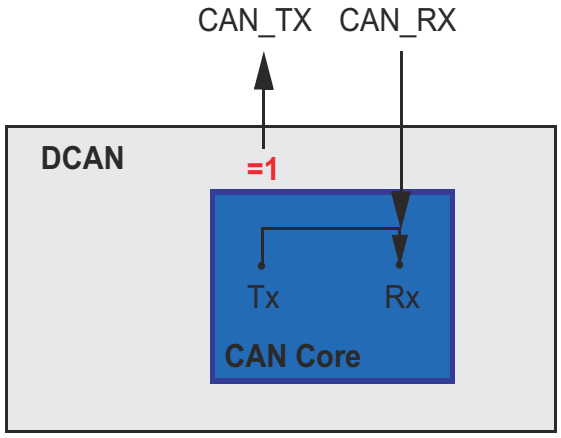


Figure 28-3. CAN Core in Silent Mode

28.3.3.2 Loopback Mode

The loopback mode is mainly intended for hardware self-test functions. In this mode, the CAN core uses internal feedback from Tx output to Rx input. Transmitted messages are treated as received messages, and can be stored into message objects if they pass acceptance filtering. The actual value of the CAN_RX input pin is disregarded by the CAN core. Transmitted messages still can be monitored at the CAN_TX pin.

In order to be independent from external stimulation, the CAN core ignores acknowledge errors (recessive bit sampled in the acknowledge slot of a data/remote frame) in loopback mode.

Figure 28-4 shows the connection of signals CAN_TX and CAN_RX to the CAN core in loopback mode. Loopback mode can be activated by setting the LBack bit in the CAN_TEST register to 1.

Note

In loopback mode, the signal path from CAN core to Tx pin, the Tx pin itself, and the signal path from Tx pin back to CAN Core are disregarded. For including these into the testing, see Section 28.3.3.3.

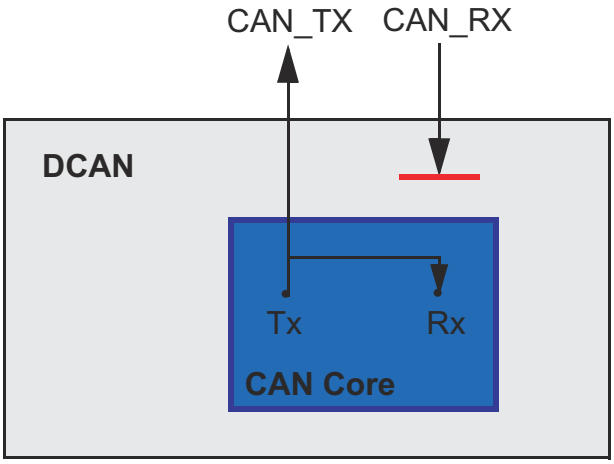


Figure 28-4. CAN Core in Loopback Mode

28.3.3.3 External Loopback Mode

The external loopback mode is similar to the loopback mode; however, it includes the signal path from CAN Core to Tx pin, the Tx pin itself, and the signal path from Tx pin back to the CAN Core. When the external loopback mode is selected, the CAN core is connected to the input buffer of the Tx pin. With this configuration, the Tx pin IO circuit can be tested. External loopback mode can be activated by setting the ExL bit in Test Register to 1.

Figure 28-5 shows the connection of signals CAN_TX and CAN_RX to the CAN Core in external loopback mode.

Note

When loopback mode is active (LBack bit set), the ExL bit will be ignored.

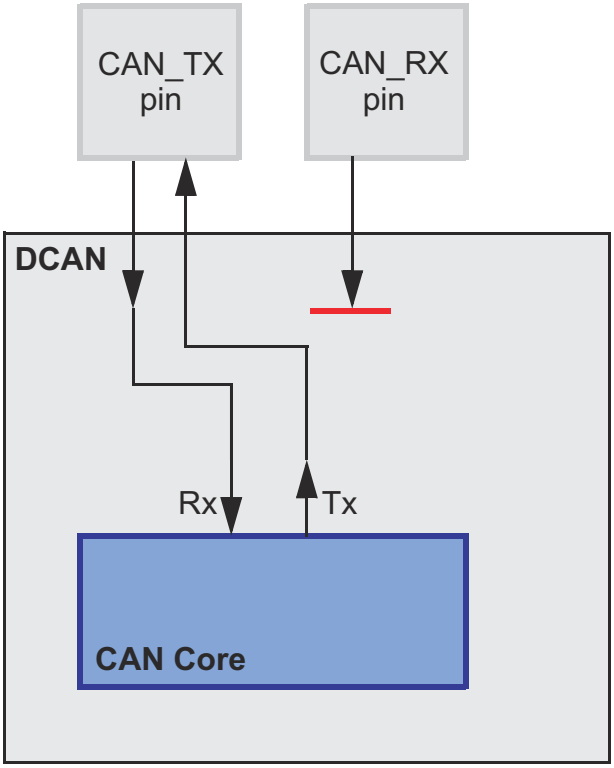


Figure 28-5. CAN Core in External Loopback Mode

28.3.3.4 Loopback Combined with Silent Mode

It is also possible to combine loopback mode and silent mode by setting bits LBack and Silent at the same time. The CAN hardware can be tested without affecting the CAN network. In this mode, the CAN_RX pin is disconnected from the CAN Core and no dominant bits will be sent on the CAN_TX pin.

Figure 28-6 shows the connection of the signals CAN_TX and CAN_RX to the CAN Core in case of the combination of loopback mode with silent mode.

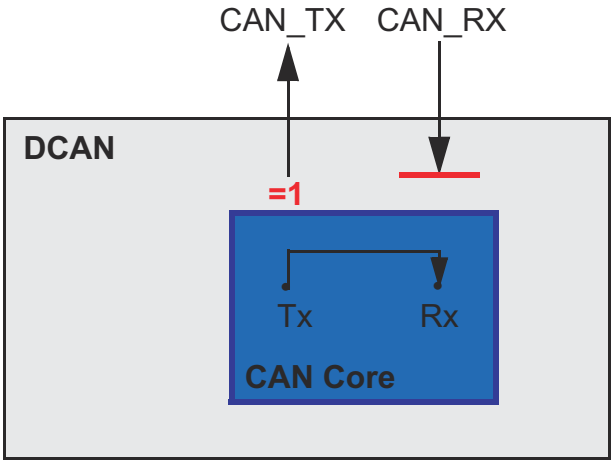


Figure 28-6. CAN Core in Loopback Combined with Silent Mode

28.4 Multiple Clock Source

The *System Control* chapter and the device data sheet provide for more information on how to configure the relevant clock source registers in the system module.

Note

The CAN core has to be programmed to at least eight clock cycles per bit time. To achieve a transfer rate of 1 Mbps an oscillator frequency of 8 MHz or higher has to be used.

28.5 Interrupt Functionality

Interrupts can be generated on two interrupt lines: CAN0INT and CAN1INT. These lines can be enabled by setting the IE0 and IE1 bits, respectively, in the CAN Control register.

The CAN provides three groups of interrupt sources: message object interrupts, status change interrupts and error interrupts. The source of an interrupt can be determined by the interrupt identifiers Int0ID / Int1ID in the Interrupt register. When no interrupt is pending, the register will hold the value zero. Each interrupt line remains active until the dedicated field in the Interrupt register (Int0ID / Int1ID) again reach zero (this means the cause of the interrupt is reset), or until IE0 / IE1 are reset. The value 0x8000 in the Int0ID field indicates that an interrupt is pending because the CAN Core has updated (not necessarily changed) the Error and Status Register (Error Interrupt or Status Interrupt). This interrupt has the highest priority. The CPU can update (reset) the status bits RxOk, TxOk and LEC by reading the Error and Status Register, but a write access of the CPU will never generate or reset an interrupt.

Values between 1 and the number of the last message object indicates that the source of the interrupt is one of the message objects. INT0ID / INT1ID will point to the pending message interrupt with the highest priority. The Message Object 1 has the highest priority, the last message object has the lowest priority.

An interrupt service routine which reads the message from the interrupt source, may also read the message and reset the message object's IntPnd at the same time (ClrIntPnd bit in the IF1/IF2 Command register). When IntPnd is cleared, the Interrupt register will point to the next message object with a pending interrupt.

The CAN module features a module-level interrupt enable and acknowledge mechanism. To enable the CAN0 and CAN1 interrupts, you must set the appropriate bits in the CAN_GLB_INT_EN register. When handling an interrupt, the individual message or status change flag must be cleared prior to acknowledging the interrupt via CAN_GLB_INT_CLR and PIEACK.

28.5.1 Message Object Interrupts

Message object interrupts are generated by events from the message objects. They are controlled by the flags IntPND, TxIE and RxIE which are described in Section 28.14.1. Message object interrupts can be routed to the CAN0INT or CAN1INT line, which is controlled by the Interrupt Multiplexer register.

Note that writing to the IntPnd bit in the CAN_IFnMCTL registers can force an interrupt.

28.5.2 Status Change Interrupts

The events RxOk, TxOk, and LEC in the Error and Status register belong to the status change interrupts. The status change interrupt group can be enabled by the SIE bit in the CAN Control Register. If SIE is set, a status change interrupt will be generated at each CAN frame, independent of bus errors or valid CAN communication, and also independent of the Message RAM configuration. Status Change interrupts can only be routed to interrupt line CAN0INT which has to be enabled by setting IE0 in the CAN_CTL Register.

28.5.3 Error Interrupts

The events PER, BOff and EWarn, belong to the error interrupts. The error interrupt group can be enabled by setting bit EIE. Also, error interrupts can only be routed to interrupt line CAN0INT, which has to be enabled by setting IE0 in the CAN_CTL register.

28.5.4 PIE Nomenclature for DCAN Interrupts

Table 28-3 shows the PIE nomenclature for the interrupts.

Table 28-3. PIE Nomenclature for Interrupts

Interrupt	CANA	CANB
CANINT0	CANA_0	CANB_0
CANINT1	CANA_1	CANB_1

28.5.5 Interrupt Topologies

Interrupt topologies for CAN are illustrated in Figure 28-7 and Figure 28-8. Mailbox interrupts for transmit and receive operations can be routed to both CANINT0 and CANINT1. However, error and status interrupts can only be routed to CANINT0.

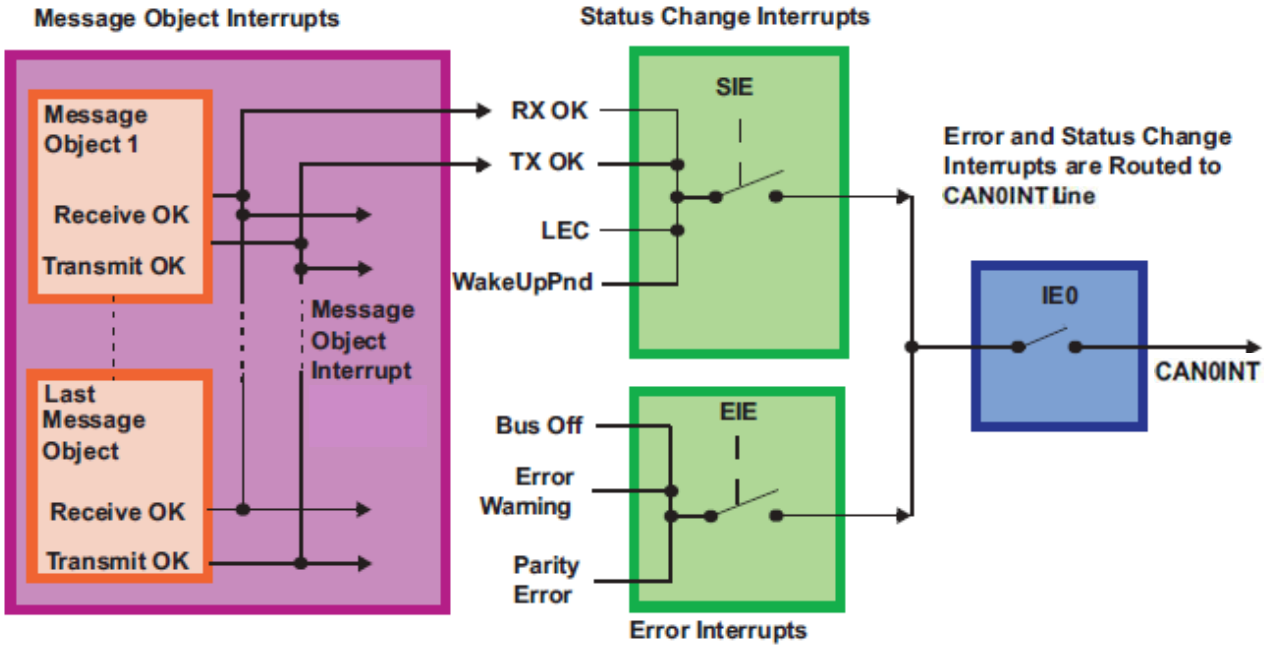


Figure 28-7. CAN Interrupt Topology 1

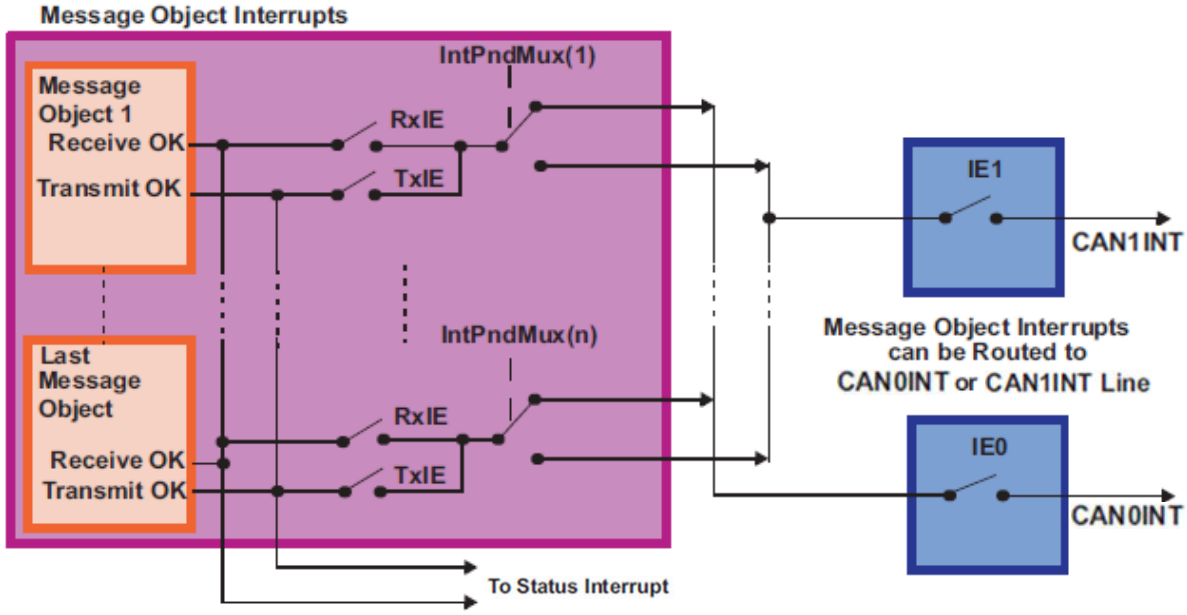


Figure 28-8. CAN Interrupt Topology 2

28.6 DMA Functionality

The CAN module provides three DMA trigger outputs, one for each of the three Interface Registers IF1, IF2 and IF3. These can be enabled using the DE1, DE2, and DE3 bits in the CAN_CTL register.

The Update of IF1 and IF2 registers will be initiated by a write access to the IF1 and IF2 Command registers, respectively. Once enabled, setting the DMAActive bit in the IF1CMD or IF2CMD registers will cause a DMA request the next time the corresponding interface becomes available.

The IF3 registers content can be automatically updated on reception of CAN messages in message objects which are programmed for automatic IF3 update. That is, when IF3 DMA requests are enabled, all IF3 updates will trigger a DMA request.

When a DCAN internal IFx update is complete, a DMA request will be activated and stays active until the first access to one of the relevant IFx registers; that is, DMA requests are cleared after the first read or write access to an IF register set.

28.7 Parity Check Mechanism

The CAN provides a parity check mechanism to ensure data integrity of message RAM data. For each word (32 bits) in Message RAM, one parity bit will be calculated.

Parity information is stored in the Message RAM on write accesses and will be checked against the stored parity bit from Message RAM on read accesses.

The parity check functionality can be enabled or disabled by the PMD bit field in the CAN Control register. In case of a disabled parity check, the parity bits in message RAM will be left unchanged on write access to data area and no check will be done on read access.

If parity checking is enabled, parity bits will be automatically generated and checked by the CAN. A parity bit will be set if the modulo-2-sum of the data bits is 1. This means that if the parity bit is set, then there are an odd number of 1 bits in the data.

28.7.1 Behavior on Parity Error

On any read access to Message RAM, for example, during the start of a CAN frame transmission, the parity of the message object will be checked. If a parity error is detected, the PER bit in the Error and Status register will be set. If error interrupts are enabled, an interrupt would also be generated. In order to avoid the transmission of invalid data over the CAN bus, the MsgVal bit of the message object will be reset.

The message object data can be read by the CPU, independently of parity errors. Thus, the application has to ensure that the read data is valid, for example, by immediately checking the Parity Error Code register on parity error interrupt.

28.8 Debug Mode

The module supports the usage of an external debug unit by providing functions like pausing CAN activities and making message RAM content accessible from the debugger. Debug mode is entered automatically when an external debugger is connected and the core is halted.

Before entering Debug mode, the circuit will wait until a transmission is started, a reception is finished, or the Bus idle state is recognized. If the IDS bit is set, the debugger immediately interrupts the current transmission or reception. Afterwards, the CAN enters Debug mode, indicated by the InitDbg flag, in the CAN Control register. During debug mode, all CAN registers can be accessed. Reading reserved bits will return '0'. Writing to reserved bits will have no effect. Also, the message RAM will be memory mapped. This allows the external debug unit to read the message RAM. For the memory organization (see Section 28.14.3).

Note

During debug mode, the Message RAM cannot be accessed via the IFx register sets.

Writing to control registers in Debug mode may influence the CAN state machine and further message handling.

For debug support, the auto clear functionality of the following CAN registers is disabled:

- Error and Status register (clear of status flags by read)
- IF1/IF2 Command registers (clear of DMA Active flag by read/write)

28.9 Module Initialization

After hardware reset, the Init bit in the CAN Control register is set and all CAN protocol functions are disabled. The configuration of the bit timing and of the message objects should be completed before the CAN protocol functions are enabled.

For the configuration of the message objects, see Section 28.10.

For the configuration of the Bit Timing, see Section 28.12.2.

The bits MsgVal, NewDat, IntPnd, and TxRqst of the message objects are reset to '0' by a hardware reset. The configuration of a message object is done by programming Mask, Arbitration, Control and Data bits of one of the IF1/IF2 Interface register sets to the desired values. By writing the message object number to bits [7:0] of the corresponding IF1/IF2 Command register, the IF1/IF2 Interface Register content is loaded into the addressed message object in the Message RAM.

The configuration of the bit timing requires that the CCE bit in the CAN Control register is set additionally to Init. This is not required for the configuration of the message objects.

When the Init bit in the CAN Control register is cleared, the CAN Protocol Controller state machine of the CAN Core and the message handler State Machine start to control the CAN's internal data flow. Received messages which pass the acceptance filtering are stored into the Message RAM; messages with pending transmission request are loaded into the CAN Core's Shift register and are transmitted via the CAN bus.

The CPU may enable the interrupt lines (setting IE0 and IE1 to '1') at the same time when it clears Init and CCE. The status interrupts EIE and SIE may be enabled simultaneously.

The CAN communication may be controlled interrupt-driven or in polling mode. The Interrupt Register points to those message objects with IntPnd = '1'. It is updated even if the interrupt lines to the CPU are disabled (IE0 / IE1 are zero).

The CPU may poll all MessageObject's NewDat and TxRqst bits in parallel from the NewData registers and the Transmission Request registers. Polling can be made easier if all Transmit Objects are grouped at the low numbers; all Receive Objects are grouped at the high numbers.

28.10 Configuration of Message Objects

The entire Message RAM should to be configured before the end of the initialization; however, it is also possible to change the configuration of message objects during CAN communication.

28.10.1 Configuration of a Transmit Object for Data Frames

Figure 28-9 shows how a transmit object can be initialized.

Figure 28-9. Initialization of a Transmit Object

MsgVal	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
1	appl.	appl.	appl.	1	1	0	0	0	appl.	0	appl.	0

- The arbitration bits (ID[28:0] and Xtd bit) are given by the application. They define the identifier and type of the outgoing message. If an 11-bit Identifier (standard frame) is used (Xtd = '0'), it is programmed to ID[28:18]. In this case, ID[17:0] can be ignored.
- The data registers (DLC[3:0] and Data0-7) are given by the application. TxRqst and RmtEn should not be set before the data is valid.
- If the TxIE bit is set, the IntPnd bit will be set after a successful transmission of the message object.
- If the RmtEn bit is set, a matching received remote frame will cause the TxRqst bit to be set; the remote frame will autonomously be answered by a data frame.
- The Mask bits (Msk[28:0], UMask, MXtd, and MDir bits) may be used (UMask='1') to allow groups of remote frames with similar identifiers to set the TxRqst bit. The Dir bit should not be masked. For details, see Section 28.11.8. Identifier masking must be disabled (UMask = '0') if no remote frames are allowed to set the TxRqst bit (RmtEn = '0').

28.10.2 Configuration of a Transmit Object for Remote Frames

It is not necessary to configure transmit objects for the transmission of remote frames. Setting TxRqst for a receive object will cause the transmission of a remote frame with the same identifier as the data frame for which this receive object is configured.

28.10.3 Configuration of a Single Receive Object for Data Frames

Figure 28-10 shows how a receive object for data frames can be initialized.

Figure 28-10. Initialization of a Single Receive Object for Data Frames

MsgVal	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
1	appl.	appl.	appl.	1	0	0	0	appl.	0	0	0	0

- The arbitration bits (ID[28:0] and Xtd bit) are given by the application. They define the identifier and type of accepted received messages. If an 11-bit Identifier (Standard Frame) is used (Xtd = '0'), it is programmed to ID[28:18]. In this case, ID[17:0] can be ignored. When a data frame with an 11-bit Identifier is received, ID[17:0] will be set to '0'.
- When the message handler stores a data frame in the message object, it will store the received data length code and the corresponding number of data bytes. If the data length code is less than 8, the remaining bytes of the message object may be overwritten by non specified values.
- The mask bits (Msk[28:0], UMask, MXtd, and MDir bits) may be used (UMask = '1') to allow groups of data frames with similar identifiers to be accepted. The Dir bit should not be masked in typical applications. If some bits of the Mask bits are set to "don't care", the corresponding bits of the Arbitration Register will be overwritten by the bits of the stored data frame.
- If the RxIE bit is set, the IntPnd bit will be set when a received data frame is accepted and stored in the message object.
- If the TxRqst bit is set, the transmission of a remote frame with the same identifier as actually stored in the Arbitration bits will be triggered. The content of the Arbitration bits may change if the Mask bits are used (UMask = '1') for acceptance filtering.

28.10.4 Configuration of a Single Receive Object for Remote Frames

Figure 28-11 shows how a receive object for remote frames can be initialized.

Figure 28-11. Initialization of a Single Receive Object for Remote Frames

MsgVal	Arb	Data	Mask	EoB	Dir	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
1	appl.	appl.	appl.	1	1	0	0	appl.	0	0	0	0

- Receive objects for remote frames may be used to monitor remote frames on the CAN bus. The remote frame stored in the receive object will not trigger the transmission of a data frame. Receive objects for remote frames may be expanded to a FIFO buffer, see Section 28.10.5.
- UMask must be set to '1'. The Mask bits (Msk[28:0], UMask, MXtd, and MDir bits) may be set to "must-match" or to "don't care", to allow groups of remote frames with similar identifiers to be accepted. The Dir bit should not be masked in typical applications. For details, see Section 28.11.8.
- The arbitration bits (ID[28:0] and Xtd bit) may be given by the application. They define the identifier and type of accepted received remote frames. If some bits of the Mask bits are set to "don't care", the corresponding bits of the arbitration bits will be overwritten by the bits of the stored remote frame. If an 11-bit Identifier (standard frame) is used (Xtd = '0'), it is programmed to ID[28:18]. In this case, ID[17:0] can be ignored. When a remote frame with an 11-bit Identifier is received, ID[17:0] will be set to '0'.
- The data length code (DLC[3:0]) may be given by the application. When the message handler stores a remote frame in the message object, it will store the received data length code. The data bytes of the message object will remain unchanged.
- If the RxIE bit is set, the IntPnd bit will be set when a received remote frame is accepted and stored in the message object.

28.10.5 Configuration of a FIFO Buffer

With the exception of the EoB bit, the configuration of receive objects belonging to a FIFO buffer is the same as the configuration of a single receive object.

To concatenate multiple message objects to a FIFO buffer, the identifiers and masks (if used) of these message objects have to be programmed to matching values. Due to the implicit priority of the message objects, the message object with the lowest number will be the first message object of the FIFO buffer. The EoB bit of all message objects of a FIFO buffer except the last one must be programmed to zero. The EoB bits of the last message object of a FIFO buffer is set to one, configuring it as the end of the block.

28.11 Message Handling

When initialization is finished, the CAN module synchronizes itself to the traffic on the CAN bus. It does acceptance filtering on received messages and stores those frames that are accepted into the designated message objects. The application must update the data of the messages to be transmitted and to enable and request their transmission. The transmission is requested automatically when a matching remote frame is received.

The application may read messages that are received and accepted. Messages that are not read before the next messages are accepted for the same message object will be overwritten. Messages may be read interrupt-driven or after polling of NewDat.

28.11.1 Message Handler Overview

The message handler state machine controls the data transfer between the Rx/Tx Shift Register of the CAN Core and the Message RAM. It performs the following tasks:

- Data transfer from Message RAM to CAN Core (messages to be transmitted).
- Data transfer from CAN Core to the Message RAM (received messages).
- Data transfer from CAN Core to the Acceptance Filtering unit.
- Scanning of Message RAM for a matching message object (acceptance filtering).
- Scanning the same message object after being changed by IF1/IF2 registers when priority is same or higher as message the object found by last scanning.
- Handling of TxRqst flags.
- Handling of interrupt flags.

The message handler registers contains status flags of all message objects grouped into the following topics:

- Transmission request flags
- New data flags
- Interrupt pending flags
- Message valid registers

Instead of collecting above listed status information of each message object via IFx registers separately, these message handler registers provide a fast and easy way to get an overview, for example, about all pending transmission requests.

All message handler registers are read-only.

28.11.2 Receive/Transmit Priority

The receive/transmit priority for the message objects is attached to the message number, not to the CAN identifier. Message object 1 has the highest priority, while message object 32 has the lowest priority. If more than one transmission request is pending, they are serviced due to the priority of the corresponding message object, so for example, messages with the highest priority can be placed in the message objects with the lowest numbers.

The acceptance filtering for received data frames or remote frames is also done in ascending order of message objects, so a frame that has been accepted by a message object cannot be accepted by another message object with a higher message number. The last message object may be configured to accept any data frame or remote frame that was not accepted by any other message object, for nodes that need to log the complete message traffic on the CAN bus.

28.11.3 Transmission of Messages in Event Driven CAN Communication

If the shift register of the CAN Core is ready for loading and if there is no data transfer between the IFx registers and Message RAM, the MsgVal bits in the Message Valid register and the TxRqst bits in the transmission request register are evaluated. The valid message object with the highest priority pending transmission request is loaded into the shift register by the message handler and the transmission is started. The message object's NewDat bit is reset.

After a successful transmission and if no new data was written to the message object (NewDat = '0') since the start of the transmission, the TxRqst bit will be reset. If TxIE is set, IntPnd will be set after a successful transmission. If the CAN has lost the arbitration or if an error occurred during the transmission, the message will be retransmitted as soon as the CAN bus is free again. If meanwhile the transmission of a message with higher priority has been requested, the messages will be transmitted in the order of their priority.

If automatic retransmission mode is disabled by setting the DAR bit in the CAN Control register, the behavior of bits TxRqst and NewDat in the Message Control register of the Interface register set is as follows:

- When a transmission starts, the TxRqst bit of the respective Interface register set is reset, while bit NewDat remains set.
- When the transmission has been successfully completed, the NewDat bit is reset.

When a transmission failed (lost arbitration or error) bit NewDat remains set. To restart the transmission, the application has to set TxRqst again.

Received remote frames do not require a receive object. They will automatically trigger the transmission of a data frame, if in the matching Transmit Object the RmtEn bit is set.

28.11.4 Updating a Transmit Object

The CPU may update the data bytes of a transmit object any time via the IF1/IF2 interface registers, neither MsgVal nor TxRqst have to be reset before the update.

Even if only a part of the data bytes are to be updated, all four bytes in the corresponding IF1/IF2 Data A register or IF1/IF2 Data B register have to be valid before the content of that register is transferred to the message object. Either the CPU has to write all four bytes into the IF1/IF2 Data register or the message object is transferred to the IF1/IF2 Data Register before the CPU writes the new data bytes.

When only the data bytes are updated, first 0x87 can be written to bits [23:16] of the Command register and then the number of the message object is written to bits [7:0] of the Command register, concurrently updating the data bytes and setting TxRqst with NewDat.

To prevent the reset of TxRqst at the end of a transmission that may already be in progress while the data is updated, NewDat has to be set together with TxRqst in event driven CAN communication. For details see Section 28.11.3.

When NewDat is set together with TxRqst, NewDat will be reset as soon as the new transmission has started.

28.11.5 Changing a Transmit Object

If the number of implemented message objects is not sufficient to be used as permanent message objects only, the transmit objects may be managed dynamically. The CPU can write the whole message (arbitration, control, and data) into the Interface register. The bits [23:16] of the Command register can be set to 0xB7 for the transfer of the whole message object content into the message object. Neither MsgVal nor TxRqst have to be reset before this operation.

If a previously requested transmission of this message object is not completed but already in progress, it will be continued; however it will not be repeated if it is disturbed.

To update only the data bytes of a message being transmitted, set bits [23:16] of the Command register to 0x87.

Note

After the update of the transmit object, the interface register set will contain a copy of the actual contents of the object, including the part that had not been updated.

28.11.6 Acceptance Filtering of Received Messages

When the arbitration and control bits (Identifier + IDE + RTR + DLC) of an incoming message is completely shifted into the shift register of the CAN Core, the message handler starts to scan the message RAM for a matching valid message object:

- The acceptance filtering unit is loaded with the arbitration bits from the CAN Core shift register.
- Then the arbitration and mask bits (including MsgVal, UMask, NewDat, and EoB) of Message Object 1 are loaded into the Acceptance Filtering unit and are compared with the arbitration bits from the shift register. This is repeated for all following message objects until a matching message object is found, or until the end of the Message RAM is reached.
- If a match occurs, the scanning is stopped and the message handler proceeds depending on the type of the frame (data frame or remote frame) received.

28.11.7 Reception of Data Frames

The message handler stores the message from the CAN Core shift register into the respective message object in the Message RAM. Not only the data bytes, but all arbitration bits and the data length code are stored into the corresponding message object. This ensures that the data bytes stay associated to the identifier even if arbitration mask registers are used.

The NewDat bit is set to indicate that new data (not yet seen by the CPU) has been received. The CPU should reset the NewDat bit when it reads the message object. If at the time of the reception the NewDat bit was already set, MsgLst is set to indicate that the previous data (not seen by the CPU) is lost. If the RxIE bit is set, the IntPnd bit is set, causing the Interrupt Register to point to this message object.

The TxRqst bit of this message object is reset to prevent the transmission of a remote frame, while the requested data frame has just been received.

28.11.8 Reception of Remote Frames

When a remote frame is received, three different configurations of the matching message object have to be considered:

1. Dir = '1' (direction = transmit), RmtEn = '1', UMask = '1' or '0'
   The TxRqst bit of this message object is set at the reception of a matching remote frame. The rest of the message object remains unchanged.
2. Dir = '1' (direction = transmit), RmtEn = '0', UMask = '0'
   The remote frame is ignored, this message object remains unchanged.
3. Dir = '1' (direction = transmit), RmtEn = '0', UMask = '1'
   The remote frame is treated similar to a received data frame. At the reception of a matching remote frame, the TxRqst bit of this message object is reset. The arbitration and control bits (Identifier + IDE + RTR + DLC) from the shift register are stored in the message object in the Message RAM and the NewDat bit of this message object is set. The data bytes of the message object remain unchanged

28.11.9 Reading Received Messages

The CPU may read a received message any time via the IFx interface registers, the data consistency is guaranteed by the message handler state machine.

Typically the CPU will write 0x7F to bits [23:16] and then the number of the message object to bits [7:0] of the Command Register. That combination will transfer the whole received message from the Message RAM into the Interface Register set. Additionally, the bits NewDat and IntPnd are cleared in the Message RAM (not in the Interface Register set). The values of these bits in the Message Control Register always reflect the status before resetting the bits.

If the message object uses masks for acceptance filtering, the arbitration bits show which of the different matching messages has been received.

The actual value of NewDat shows whether a new message has been received since last time when this message object was read. The actual value of MsgLst shows whether more than one message have been received since the last time when this message object was read. MsgLst will not be automatically reset.

28.11.10 Requesting New Data for a Receive Object

By means of a remote frame, the CPU may request another CAN node to provide new data for a receive object. Setting the TxRqst bit of a receive object will cause the transmission of a remote frame with the receive object's identifier. This remote frame triggers the other CAN node to start the transmission of the matching data frame. If the matching data frame is received before the remote frame could be transmitted, the TxRqst bit is automatically reset.

Setting the TxRqst bit without changing the contents of a message object requires the value 0x84 in bits [23:16] of the Command Register.

28.11.11 Storing Received Messages in FIFO Buffers

Several message objects may be grouped to form one or more FIFO Buffers. Each FIFO Buffer configured to store received messages with a particular (group of) Identifier(s). Arbitration and Mask registers of the FIFO Buffer's message objects are identical. The EoB (End of Buffer) bits of all but the last of the FIFO Buffer's message objects are '0', in the last one the EoB bit is '1'.

Received messages with identifiers matching to a FIFO Buffer are stored into a message object of this FIFO Buffer, starting with the message object with the lowest message number.

When a message is stored into a message object of a FIFO Buffer the NewDat bit of this message object is set. By setting NewDat while EoB is '0' the message object is locked for further write accesses by the message handler until the CPU has cleared the NewDat bit.

Messages are stored into a FIFO Buffer until the last message object of this FIFO Buffer is reached. If none of the preceding message objects is released by writing NewDat to '0', all further messages for this FIFO Buffer will be written into the last message object of the FIFO Buffer (EoB = '1') and therefore overwrite previous messages in this message object.

28.11.12 Reading from a FIFO Buffer

Several messages may be accumulated in a set of message objects which are concatenated to form a FIFO Buffer before the application program is required (in order to avoid the loss of data) to empty the buffer. A FIFO Buffer of length N will store N-1 plus the last received message since last time it was cleared. A FIFO Buffer is cleared by reading and resetting the NewDat bits of all its message objects, starting at the FIFO Object with the lowest message number. This should be done in a subroutine following the example shown in Figure 28-12.

Note

All message objects of a FIFO buffer needs to be read and cleared before the next batch of messages can be stored. Otherwise true FIFO functionality cannot be guaranteed, since the message objects of a partly read buffer will be re-filled according to the normal (descending) priority.

Reading from a FIFO Buffer message object and resetting its NewDat bit is handled the same way as reading from a single message object.

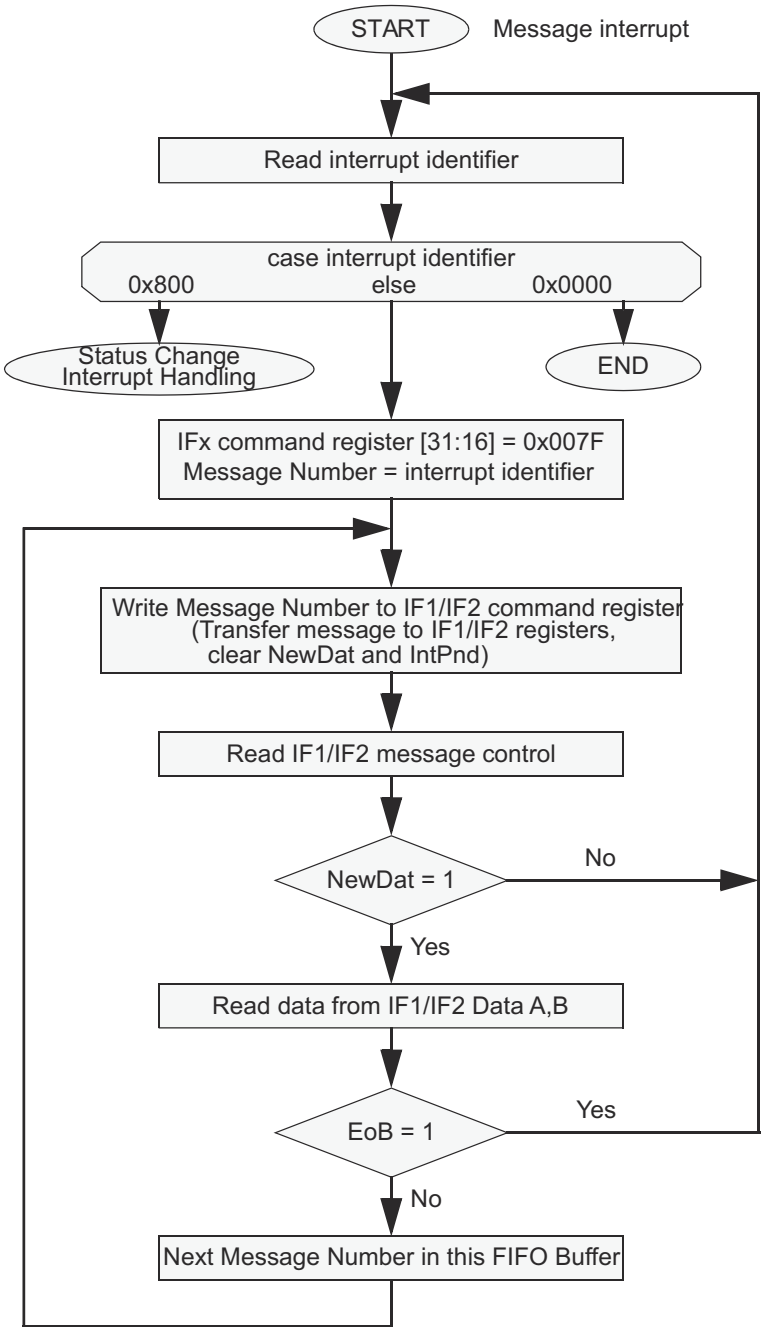


Figure 28-12. CPU Handling of a FIFO Buffer (Interrupt Driven)

28.12 CAN Bit Timing

The CAN supports bit rates between less than 1 kBit/s and 1000 kBit/s.

Each member of the CAN network has its own clock generator, typically derived from a crystal oscillator. The Bit timing parameters can be configured individually for each CAN node, creating a common Bit rate even though the CAN nodes' oscillator periods (F_{osc}) may be different.

The frequencies of these oscillators are not absolutely stable. Small variations are caused by changes in temperature or voltage and by deteriorating components. As long as the variations remain inside a specific oscillator tolerance range (df), the CAN nodes are able to compensate for the different bit rates by resynchronizing to the bit stream.

In many cases, the CAN bit synchronization will amend a faulty configuration of the CAN bit timing to such a degree that only occasionally an error frame is generated. In the case of arbitration however, when two or more CAN nodes simultaneously try to transmit a frame, a misplaced sample point may cause one of the transmitters to become error passive.

The analysis of such sporadic errors requires a detailed knowledge of the CAN bit synchronization inside a CAN node and of the CAN nodes' interaction on the CAN bus.

Even if minor errors in the configuration of the CAN bit timing do not result in immediate failure, the performance of a CAN network can be reduced significantly.

28.12.1 Bit Time and Bit Rate

According to the CAN specification, the Bit time is divided into four segments (see Figure 28-13):

- Synchronization Segment (Sync_Seg)
- Propagation Time Segment (Prop_Seg)
- Phase Buffer Segment 1 (Phase_Seg1)
- Phase Buffer Segment 2 (Phase_Seg2)

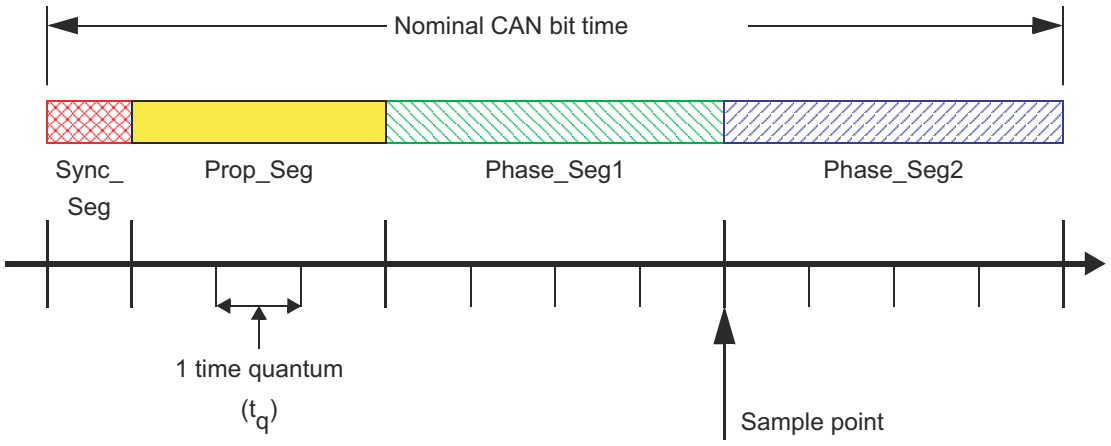


Figure 28-13. Bit Timing

Each segment consists of a specific number of time quanta. The length of one time quantum (t_q), which is the basic time unit of the bit time, is given by the CAN_CLK and the Baud Rate Prescalers (BRPE and BRP). With these two Baud Rate Prescalers combined, divider values from 1 to 1024 can be programmed:

t_q = Baud Rate Prescaler/CAN_CLK

Apart from the fixed length of the synchronization segment, these numbers are programmable. Table 28-4 describes the minimum programmable ranges required by the CAN protocol.

A given bit rate may be met by different bit time configurations.

Table 28-4. Programmable Ranges Required by CAN Protocol

Parameter	Range	Remark
Sync_Seg	1 t_q (fixed)	Synchronization of bus input to CAN_CLK
Prop_Seg	[1 … 8] t_q	Compensates for the physical delay times
Phase_Seg1	[1 … 8] t_q	May be lengthened temporarily by synchronization
Phase_Seg2	[1 … 8] t_q	May be shortened temporarily by synchronization
Synchronization Jump Width (SJW)	[1 … 4] t_q	May not be longer than either phase buffer segment

Note

For proper functionality of the CAN network, the physical delay times and the oscillator's tolerance range have to be considered.

28.12.1.1 Synchronization Segment

The Synchronization Segment (Sync_Seg) is the part of the bit time where edges of the CAN bus level are expected to occur. If an edge occurs outside of Sync_Seg, its distance to the Sync_Seg is called the phase error of this edge.

28.12.1.2 Propagation Time Segment

This part of the bit time is used to compensate physical delay times within the CAN network. These delay times consist of the signal propagation time on the bus and the internal delay time of the CAN nodes.

Any CAN node synchronized to the bit stream on the CAN bus can be out of phase with the transmitter of the bit stream, caused by the signal propagation time between the two nodes. The CAN protocol's nondestructive bitwise arbitration and the dominant acknowledge bit provided by receivers of CAN messages require that a CAN node transmitting a bit stream must also be able to receive dominant bits transmitted by other CAN nodes that are synchronized to that bit stream. The example in Figure 28-14 shows the phase shift and propagation times between two CAN nodes.

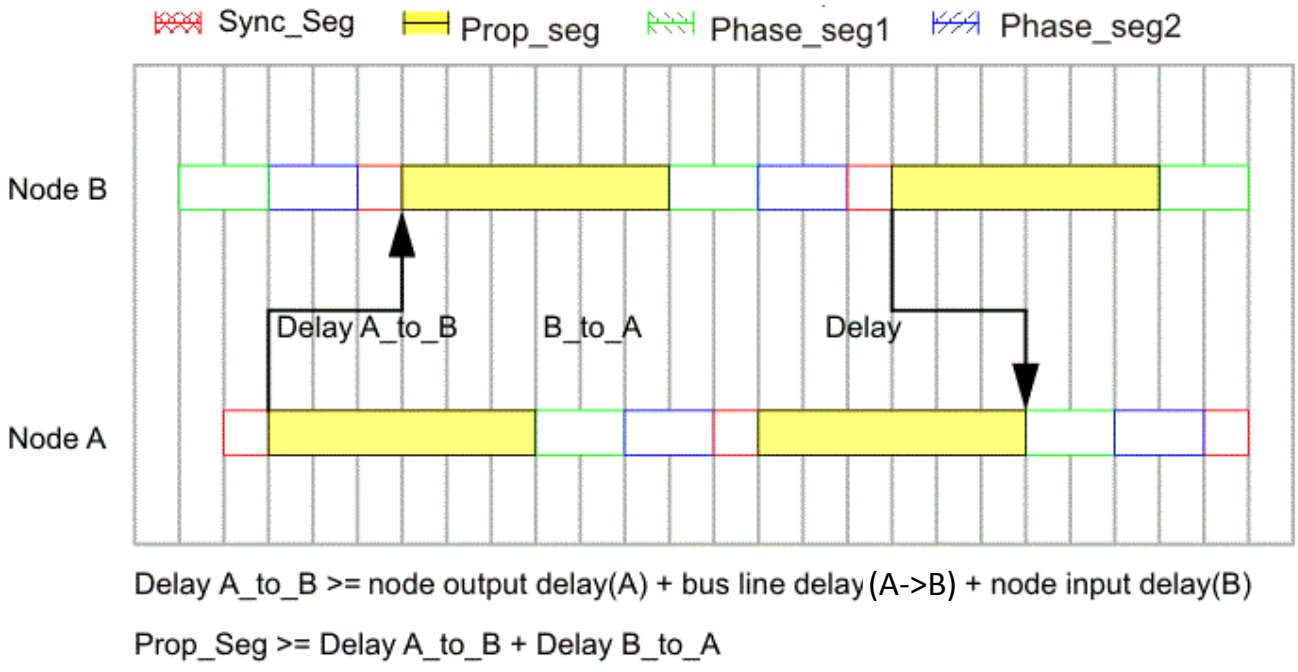


Figure 28-14. The Propagation Time Segment

In this example, both nodes A and B are transmitters performing an arbitration for the CAN bus. The node A has sent its Start of Frame bit less than one bit time earlier than node B, therefore node B has synchronized itself to the received edge from recessive to dominant. Since node B has received this edge delay(A_to_B) after it has been transmitted, node B's bit timing segments are shifted with regard to node A. Node B sends an identifier with higher priority and so it will win the arbitration at a specific identifier bit when it transmits a dominant bit while node A transmits a recessive bit. The dominant bit transmitted by node B will arrive at node A after the delay(B_to_A).

Due to oscillator tolerances, the actual position of node A's Sample Point can be anywhere inside the nominal range of node A's Phase Buffer Segments, so the bit transmitted by node B must arrive at node A before the start of Phase_Seg1. This condition defines the length of Prop_Seg.

If the edge from recessive to dominant transmitted by node B would arrive at node A after the start of Phase_Seg1, it could happen that node A samples a recessive bit instead of a dominant bit, resulting in a bit error and the destruction of the current frame by an error flag.

This error only occurs when two nodes arbitrate for the CAN bus which have oscillators of opposite ends of the tolerance range and are separated by a long bus line; this is an example of a minor error in the Bit timing configuration (Prop_Seg too short) that causes sporadic bus errors.

Some CAN implementations provide an optional 3-Sample Mode. The CAN module on this device does not. In this mode, the CAN bus input signal passes a digital low-pass filter, using three samples and a majority logic to determine the valid bit value. This results in an additional input delay of 1 t_q, requiring a longer Prop_Seg.

28.12.1.3 Phase Buffer Segments and Synchronization

The phase buffer segments (Phase_Seg1 and Phase_Seg2) and the synchronization jump width (SJW) are used to compensate for the oscillator tolerance.

The phase buffer segments surround the sample point. The phase buffer segments may be lengthened or shortened by synchronization.

The synchronization jump width (SJW) defines how far the resynchronizing mechanism may move the sample point inside the limits defined by the phase buffer segments to compensate for edge phase errors.

Synchronizations occur on edges from recessive to dominant. Their purpose is to control the distance between edges and sample points.

Edges are detected by sampling the actual bus level in each time quantum and comparing it with the bus level at the previous sample point. A synchronization may be done only if a recessive bit was sampled at the previous sample point and if the actual time quantum's bus level is dominant.

An edge is synchronous if it occurs inside of Sync_Seg, otherwise its distance to the Sync_Seg is the edge phase error, measured in time quanta. If the edge occurs before Sync_Seg, the phase error is negative, else it is positive.

Two types of synchronization exist: hard synchronization and resynchronizing. A hard synchronization is done once at the start of a frame; inside a frame only resynchronization is possible.

- Hard Synchronization

  After a hard synchronization, the bit time is restarted with the end of Sync_Seg, regardless of the edge phase error. Thus hard synchronization forces the edge which has caused the hard synchronization to lie within the synchronization segment of the restarted bit time.
- Bit Resynchronizations

  Resynchronization leads to a shortening or lengthening of the bit time such that the position of the sample point is shifted with regard to the edge.

  When the phase error of the edge which causes resynchronization is positive, Phase_Seg1 is lengthened. If the magnitude of the phase error is less than SJW, Phase_Seg1 is lengthened by the magnitude of the phase error, else it is lengthened by SJW.

When the phase error of the edge which causes resynchronization is negative, Phase_Seg2 is shortened. If the magnitude of the phase error is less than SJW, Phase_Seg2 is shortened by the magnitude of the phase error, else it is shortened by SJW.

If the magnitude of the phase error of the edge is less than or equal to the programmed value of SJW, the results of hard synchronization and resynchronization are the same. If the magnitude of the phase error is larger than SJW, the resynchronization cannot compensate the phase error completely, and an error of (phase error - SJW) remains.

Only one synchronization may be done between two sample points. The synchronizations maintain a minimum distance between edges and sample points, giving the bus level time to stabilize and filtering out spikes that are shorter than (Prop_Seg + Phase_Seg1).

Apart from noise spikes, most synchronizations are caused by arbitration. All nodes synchronize "hard" on the edge transmitted by the "leading" transceiver that started transmitting first, but due to propagation delay times, they cannot become ideally synchronized. The "leading" transmitter does not necessarily win the arbitration, therefore the receivers have to synchronize themselves to different transmitters that subsequently "take the lead" and that are differently synchronized to the previously "leading" transmitter. The same happens at the acknowledge field, where the transmitter and some of the receivers will have to synchronize to that receiver that "takes the lead" in the transmission of the dominant acknowledge bit.

Synchronizations after the end of the arbitration will be caused by oscillator tolerance, when the differences in the oscillator's clock periods of transmitter and receivers sum up during the time between synchronizations (at most ten bits). These summarized differences may not be longer than the SJW, limiting the oscillator's tolerance range.

The examples in Figure 28-15 show how the phase buffer segments are used to compensate for phase errors. There are three drawings of each two consecutive bit timings. The upper drawing shows the synchronization on a "late" edge, the lower drawing shows the synchronization on an "early" edge, and the middle drawing is the reference without synchronization.

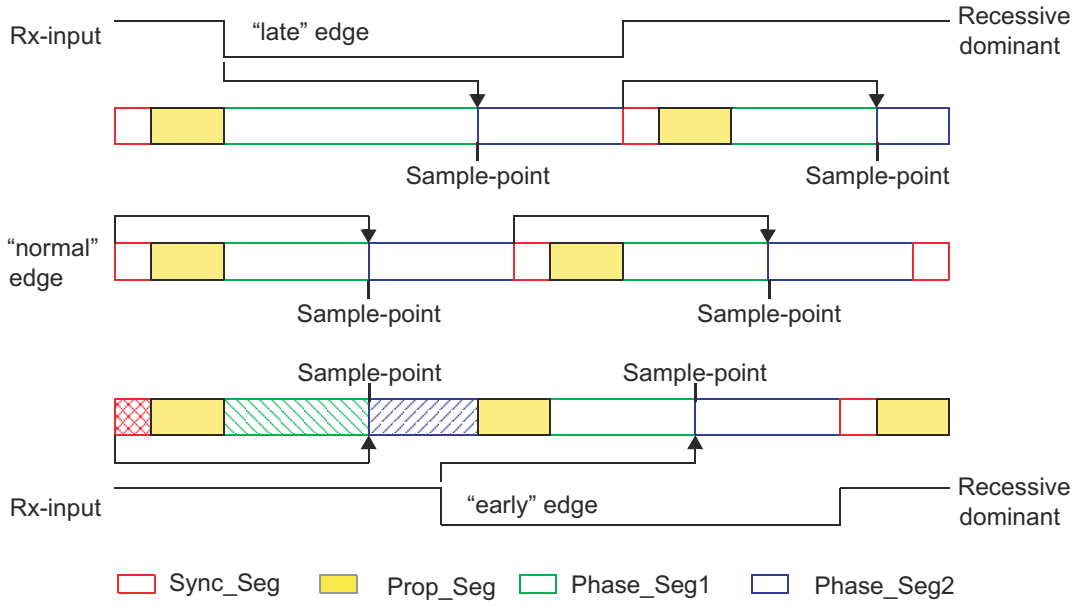


Figure 28-15. Synchronization on Late and Early Edges

In the first example, an edge from recessive to dominant occurs at the end of Prop_Seg. The edge is "late" since it occurs after the Sync_Seg. Reacting to the "late" edge, Phase_Seg1 is lengthened so that the distance from the edge to the sample point is the same as it would have been from the Sync_Seg to the sample point if no edge had occurred. The phase error of this "late" edge is less than SJW, so it is fully compensated and the edge from dominant to recessive at the end of the bit, which is one nominal bit time long, occurs in the Sync_Seg.

In the second example, an edge from recessive to dominant occurs during Phase_Seg2. The edge is "early" since it occurs before a Sync_Seg. Reacting to the "early" edge, Phase_Seg2 is shortened and Sync_Seg is omitted, so that the distance from the edge to the sample point is the same as it would have been from a Sync_Seg to the sample point if no edge had occurred. As in the previous example, the magnitude of this "early" edge's phase error is less than SJW, so it is fully compensated.

The phase buffer segments are lengthened or shortened temporarily only; at the next bit time, the segments return to their nominal programmed values.

In these examples, the bit timing is seen from the point of view of the CAN implementation's state machine, where the bit time starts and ends at the sample points. The state machine omits Sync_Seg when synchronizing on an "*early*" edge because it cannot subsequently redefine that time quantum of Phase_Seg2 where the edge occurs to be the Sync_Seg.

The examples in Figure 28-16 show how short dominant noise spikes are filtered by synchronizations. In both examples, the spike starts at the end of Prop_Seg and has the length of (Prop_Seg + Phase_Seg1).

In the first example, the synchronization jump width is greater than or equal to the phase error of the spike's edge from recessive to dominant. Therefore the sample point is shifted after the end of the spike; a recessive bus level is sampled.

In the second example, SJW is shorter than the phase error, so the sample point cannot be shifted far enough; the dominant spike is sampled as actual bus level.

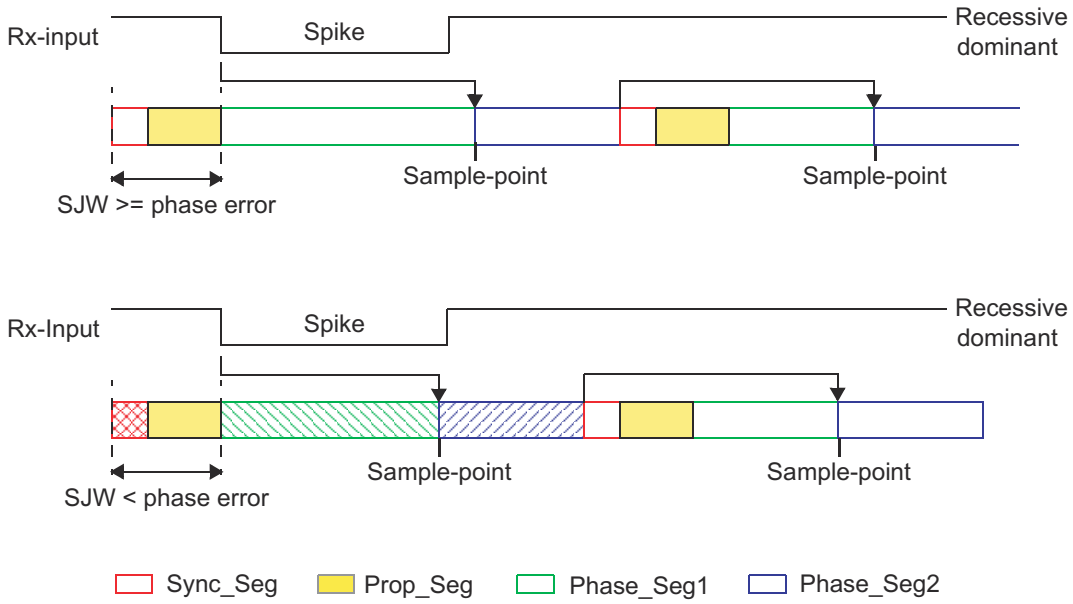


Figure 28-16. Filtering of Short Dominant Spikes

28.12.1.4 Oscillator Tolerance Range

With the introduction of CAN protocol version 1.2, the option to synchronize on edges from dominant to recessive became obsolete. Only edges from recessive to dominant are considered for synchronization. The protocol update to version 2.0 (A and B) had no influence on the oscillator tolerance.

The tolerance range df for an oscillator's frequency f_{osc} around the nominal frequency f_{nom} with:

$$(1 - df) \bullet f_{nom} \leq f_{osc} \leq (1 + df) \bullet f_{nom} \quad (28)$$

depends on the proportions of Phase_Seg1, Phase_Seg2, SJW, and the bit time. The maximum tolerance df is defined by two conditions (both shall be met):

$$df \leq \frac{\min(Tseg1, Tseg2)}{2((13 \times bit\ time) - Tseg2)}$$
$$df \leq \frac{SJW}{20 \times bit\_time} \quad (29)$$

It has to be considered that SJW may not be larger than the smaller of the phase buffer segments and that the propagation time segment limits that part of the bit time that may be used for the phase buffer segments.

The combination Prop_Seg = 1 and Phase_Seg1 = Phase_Seg2 = SJW = 4 allows the largest possible oscillator tolerance of 1.58%. This combination with a Propagation Time Segment of only 10% of the bit time is not suitable for short bit times; it can be used for bit rates of up to 125 kBit/s (bit time = 8 µs) with a bus length of 40 m.

28.12.2 Configuration of the CAN Bit Timing

In the CAN, the bit timing configuration is programmed in two register bytes, additionally a third byte for a baud rate prescaler extension of 4 bits (BRPE) is provided. The sum of Prop_Seg and Phase_Seg1 (as TSEG1) is combined with Phase_Seg2 (as TSEG2) in one byte, SJW and BRP (plus BRPE in third byte) are combined in the other byte (see Figure 28-17).

In this bit timing register, the components TSEG1, TSEG2, SJW and BRP have to be programmed to a numerical value that is one less than its functional value; so instead of values in the range of [1…n], values in the range of [0…n-1] are programmed. That way, for example, SJW (functional range of [1…4]) is represented by only two bits.

Therefore the length of the Bit time is (programmed values) [TSEG1 + TSEG2 + 3] t_q or (functional values) [Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] t_q.

The data in the Bit Timing Register is the configuration input of the CAN protocol controller. The baud rate prescaler (configured by BRPE/BRP) defines the length of the time quantum (the basic time unit of the bit time); the bit timing logic (configured by TSEG1, TSEG2, and SJW) defines the number of time quanta in the bit time.

The processing of the bit time, the calculation of the position of the Sample Point, and occasional synchronizations are controlled by the Bit timing state machine, which is evaluated once each time quantum. The rest of the CAN protocol controller, the Bit Stream Processor (BSP) state machine, is evaluated once each bit time, at the Sample Point.

The Shift register serializes the messages to be sent and parallelizes received messages. Its loading and shifting is controlled by the BSP.

The BSP translates messages into frames and vice versa. It generates and discards the enclosing fixed format bits, inserts and extracts stuff bits, calculates and checks the CRC code, performs the error management, and decides which type of synchronization is to be used. It is evaluated at the sample point and processes the sampled bus input bit. The time after the sample point that is needed to calculate the next bit to be sent (for example, data bit, CRC bit, stuff bit, error flag, or idle) is called the Information Processing Time (IPT), which is 0 t_q for the CAN.

Generally, the IPT is CAN controller specific, but may not be longer than 2 t_q. The IPT length is the lower limit of the programmed length of Phase_Seg2. In case of a synchronization, Phase_Seg2 may be shortened to a value less than IPT, which does not affect bus timing.

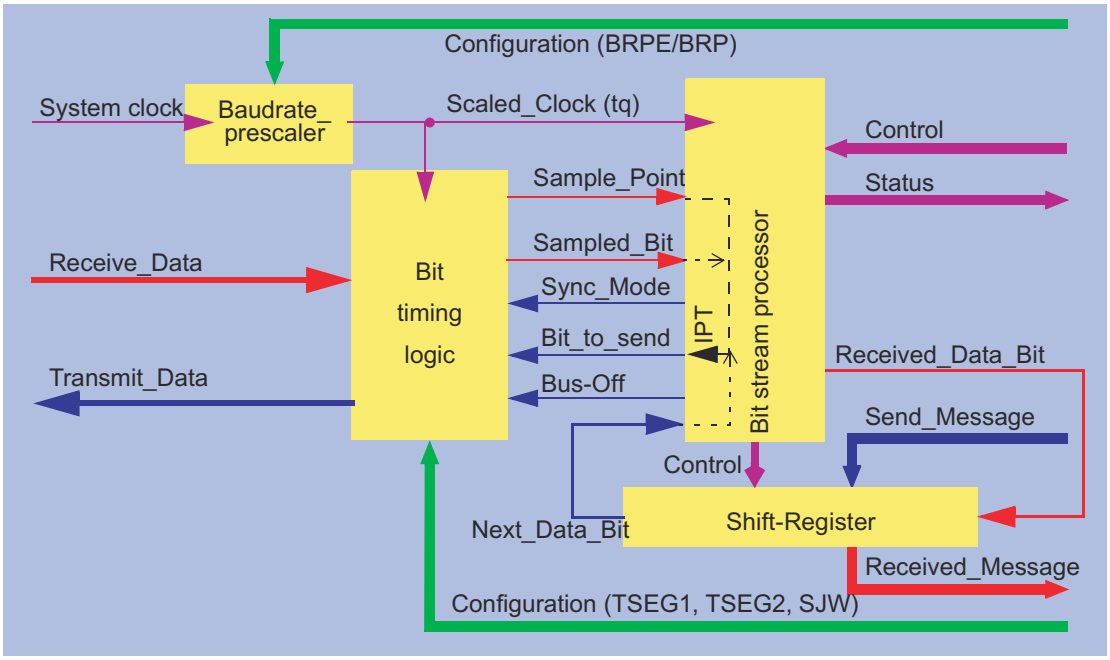


Figure 28-17. Structure of the CAN Core's CAN Protocol Controller

28.12.2.1 Calculation of the Bit Timing Parameters

Usually, the calculation of the bit timing configuration starts with a desired bit rate or bit time. The resulting Bit time (1 / Bit rate) must be an integer multiple of the CAN clock period.

Note

8 MHz is the minimum CAN clock frequency required to operate the CAN at a bit rate of 1 MBit/s.

The bit time may consist of 8 to 25 time quanta. The length of the time quantum t_q is defined by the Baud Rate Prescaler with t_q = (Baud Rate Prescaler) / CAN_CLK. Several combinations may lead to the desired bit time, allowing iterations of the following steps.

The first part of the bit time to be defined is the Prop_Seg. Its length depends on the delay times measured in the system. A maximum bus length as well as a maximum node delay has to be defined for expandible CAN bus systems. The resulting time for Prop_Seg is converted into time quanta (rounded up to the nearest integer multiple of t_q).

The Sync_Seg is 1 t_q long (fixed), leaving (bit time - Prop_Seg - 1) t_q for the two Phase Buffer Segments. If the number of remaining t_q is even, the Phase Buffer Segments have the same length, Phase_Seg2 = Phase_Seg1, else Phase_Seg2 = Phase_Seg1 + 1.

The minimum nominal length of Phase_Seg2 has to be regarded as well. Phase_Seg2 may not be shorter than any CAN controller's Information Processing Time in the network, which is device dependent and can be in the range of [0…2] t_q.

The length of the synchronization jump width is set to its maximum value, which is the minimum of 4 and Phase_Seg1.

The oscillator tolerance range necessary for the resulting configuration is calculated by the formulas given in Section 28.12.1.4.

If more than one configurations are possible to reach a certain Bit rate, it is recommended to choose the configuration which allows the highest oscillator tolerance range.

CAN nodes with different clocks require different configurations to come to the same bit rate. The calculation of the propagation time in the CAN network, based on the nodes with the longest delay times, is done once for the whole network.

The CAN system's oscillator tolerance range is limited by the node with the lowest tolerance range.

The calculation may show that bus length or bit rate have to be decreased or that the oscillator frequencies' stability has to be increased in order to find a protocol compliant configuration of the CAN bit timing.

The resulting configuration is written into the Bit Timing register:

(Phase_Seg2-1)&(Phase_Seg1+Prop_Seg-1)&

(SynchronizationJumpWidth-1)&(Prescaler-1)

28.12.2.2 Example for Bit Timing at High Baudrate

In this example, the frequency of CAN_CLK is 10 MHz, BRP is 0, the bit rate is 1 MBit/s.

t_q	100	ns	=	t_{CAN_CLK}
delay of bus driver	90	ns	=	
delay of receiver circuit	40	ns	=	
delay of bus line (40m)	220	ns	=	
t_{Prop}	700	ns	=	2*delays = 7 • t_q
t_{SJW}	100	ns	=	1 • t_q
t_{TSeg1}	800	ns	=	t_{Prop} + t_{SJW}
t_{TSeg2}	100	ns	=	Information Processing Time + 1 • t_q
$t_{Sync-Seg}$	100	ns	=	1 • t_q
bit time	1000	ns	=	$t_{Sync-Seg}$ + t_{TSeg1} + t_{TSeg2}
tolerance for CAN_CLK	0.35	%	=	$\frac{\min(Tseg1, Tseg2)}{2((13 \times bit\ time) - Tseg2)}$ $= \frac{0.1\,\mu s}{2((13 \times 1\,\mu s) - 0.1\,\mu s)}$

In this example, the concatenated bit time parameters are $(1\text{-}1)_3$&$(8\text{-}1)_4$&$(1\text{-}1)_2$&$(1\text{-}1)_6$, so the Bit Timing Register is programmed to = 0x00000700.

28.12.2.3 Example for Bit Timing at Low Baudrate

In this example, the frequency of CAN_CLK is 2 MHz, BRP is 1, the bit rate is 100 KBit/s.

t_q	1	µs	=	2 • t_{CAN_CLK}
delay of bus driver	200	ns	=	
delay of receiver circuit	80	ns	=	
delay of bus line (40m)	220	ns	=	
t_{Prop}	1	µs	=	1 • t_q
t_{SJW}	4	µs	=	4 • t_q
t_{TSeg1}	5	µs	=	t_{Prop} + t_{SJW}
t_{TSeg2}	4	µs	=	Information Processing Time + 4 • t_q
$t_{Sync-Seg}$	1	µs	=	1 • t_q
bit time	10	µs	=	$t_{Sync-Seg}$ + t_{TSeg1} + t_{TSeg2}
tolerance for CAN_CLK	1.58	%	=	$\frac{\min(Tseg1, Tseg2)}{2((13 \times bit\ time) - Tseg2)}$ $= \frac{4\,\mu s}{2((13 \times 10\,\mu s) - 4\,\mu s)}$

In this example, the concatenated bit time parameters are $(4\text{-}1)_3$&$(5\text{-}1)_4$&$(4\text{-}1)_2$&$(2\text{-}1)_6$, so the Bit Timing register is programmed to = 0x000034C1.

28.13 Message Interface Register Sets

The interface register sets control the CPU read and write accesses to the Message RAM. There are two interface register sets for read and write access (IF1 and IF2) and one Interface Register Set for read access only (IF3).

Due to the structure of the Message RAM, it is not possible to change single bits or bytes of a message object. Instead, always a complete message object in the Message RAM is accessed. Therefore the data transfer from the IF1/IF2 registers to the Message RAM requires the message handler to perform a read-modifywrite cycle. First those parts of the message object that are not to be changed are read from the Message RAM into the Interface Register set, and after the update the whole content of the Interface Register set is written into the message object.

After the partial write of a message object, those parts of the Interface Register set which are not selected in the Command Register, will be set to the actual contents of the selected message object. After the partial read of a message object, those parts of the Interface Register set which are not selected in the Command Register, will be left unchanged.

By buffering the data to be transferred, the Interface Register sets avoid conflicts between concurrent CPU accesses to the Message RAM and CAN message reception and transmission. A complete message object (see Section 28.14.1) or parts of the message object may be transferred between the Message RAM and the IF1/IF2 Register set in one single transfer. This transfer, performed in parallel on all selected parts of the message object, guarantees the data consistency of the CAN message.

That being said, there is one condition that can cause a write access to the message RAM to be lost. If MsgVal = 1 for the message object which is accessed and CAN communication is ongoing, a transfer from the IFx register to message RAM may be lost. The reason for this is that it might happen that the IFx register write to the message RAM occurs in between a read-modify-write access of the Host Message Handler when it is in the process of receiving a message for the same message object.

To avoid this issue with receive mail boxes, reset MsgVal before changing any of the following: Id28-0, Xtd, Dir, DLC3-0, RxIE, TxIE, RmtEn, EoB, Umask, Msk28-0, MXtd, and MDir.

To avoid this issue with transmit mail boxes, reset MsgVal before changing any of the following: Dir, RxIE, TxIE, RmtEn, EoB, Umask, Msk28-0, MXtd, and MDir. Other fields not listed above, like Data, may be changed without fear of losing a write to the message RAM.

28.13.1 Message Interface Register Sets 1 and 2 (IF1 and IF2)

The IF1 and IF2 register sets allow data transfers to and from the message objects. The IFxCMD register for an interface control the direction of the data transfer. If the IFxCMD register is set to write, then the message object fields selected by the IFxCMD register will be overwritten by values taken from the other IFx registers. Is the IFxCMD register is set to read, then the message object fields selected by the IFxCMD register will be copied from the message object to the other IFx registers. The interfaces allow for transfers of a complete message object as well as individual parts. The transfer begins with the desired message object number is written to bits 7:0 of the IFxCMD register.

When the CPU initiates a data transfer between the IF1/IF2 registers and Message RAM, the message handler sets the Busy bit in the respective Command Register to '1'. After the transfer has completed, the Busy bit is set back to '0' (see Figure 28-18).

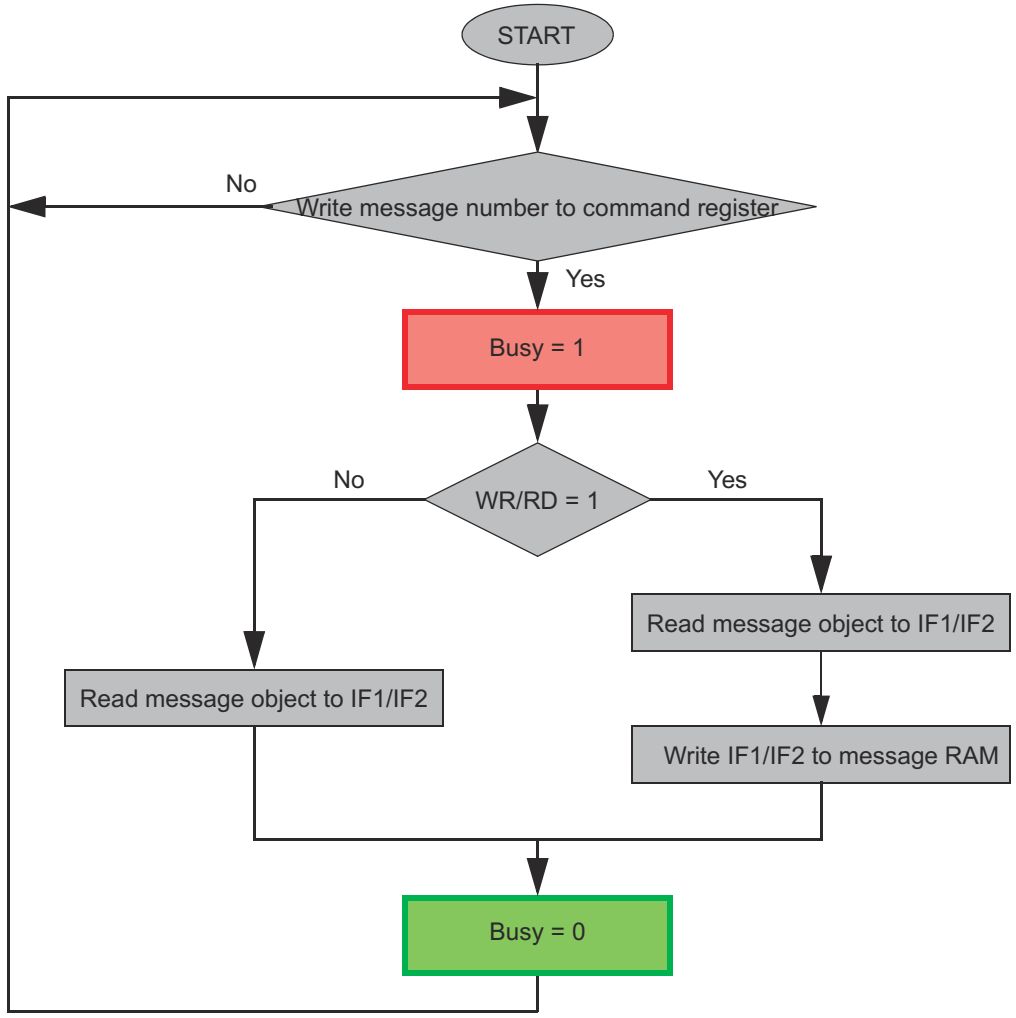


Figure 28-18. Data Transfer Between IF1 / IF2 Registers and Message RAM

28.13.2 Message Interface Register Set 3 (IF3)

The IF3 register set can automatically be updated with received message objects without the need to initiate the transfer from Message RAM by CPU. The automatic update functionality can be programmed for each message object (see the IF3 Update Enable register).

All valid message objects in Message RAM which are configured for automatic update, will be checked for active NewDat flags. If such a message object is found, it will be transferred to the IF3 register (if no previous DMA transfers are ongoing), controlled by IF3 Observation register. If more than one NewDat flag is active, the message object with the lowest number has the highest priority for automatic IF3 update.

The NewDat bit in the message object will be reset by a transfer to IF3.

If DCAN internal IF3 update is complete, a DMA will be requested. The DMA request stays active until the first read access to one of the IF3 registers. The DMA functionality has to be enabled by setting bit DE3 in the CAN Control register.

Note

The IF3 register set can not be used for transferring data into message objects.

28.14 Message RAM

The CAN Message RAM contains message objects and parity bits for the message objects. There are 32 message objects in the Message RAM.

During normal operation, accesses to the Message RAM are performed via the Interface Register sets, and the CPU cannot directly access the Message RAM.

The Interface Register sets IF1 and IF2 provide indirect read/write access from the CPU to the Message RAM. The IF1 and IF2 register sets can buffer control and user data to be transferred to and from the message objects.

The third Interface Register set IF3 can be configured to automatically receive control and user data from the Message RAM when a message object has been updated after reception of a CAN message. The CPU does not need to initiate the transfer from Message RAM to IF3 Register set.

The message handler avoids potential conflicts between concurrent accesses to Message RAM and CAN frame reception/transmission.

The message RAM can only be accessed in debug mode. The message RAM base address is 0x1000 above the base address of the CAN peripheral.

28.14.1 Structure of Message Objects

Figure 28-19 shows the structure of a message object.

The grayed fields are those parts of the message object which are represented in dedicated registers. For example, the transmit request flags of all message objects are represented in centralized transmit request registers.

Figure 28-19. Structure of a Message Object

Message Object												
UMask	Msk[28:0]	MXtd	MDir	EoB	unused	NewDat	MsgLst	RxIE	TxIE	IntPnd	RmtEn	TxRqst
MsgVal	ID[28:0]	Xtd	Dir	DLC[3:0]	Data 0	Data 1	Data 2	Data 3	Data 4	Data 5	Data 6	Data 7

Table 28-5. Message Object Field Descriptions

Name	Value	Description
MsgVal		Message valid
	0	The message object is ignored by the message handler.
	1	The message object is to be used by the message handler.
		Note: This bit may be kept at level '1' even when the identifier bits ID[28:0], the control bits Xtd, Dir, or the data length code DLC[3:0] are changed.
UMask		Use Acceptance Mask
	0	Mask bits (Msk[28:0], MXtd and MDir) are ignored and not used for acceptance filtering.
	1	Mask bits are used for acceptance filtering.
		Note: If the UMask bit is set to one, the message object's mask bits have to be programmed during initialization of the message object before MsgVal is set to one.
ID[28:0]		Message Identifier
	ID[28:0]	29-bit ("extended") identifier bits
	ID[28:18]	11-bit ("standard") identifier bits

Table 28-5. Message Object Field Descriptions (continued)

Name	Value	Description
Msk[28:0]		Identifier Mask
	0	The corresponding bit in the message identifier is not used for acceptance filtering (don't care).
	1	The corresponding bit in the message identifier is used for acceptance filtering.
		Note: The bit functionality in the DCAN module is the opposite of the Local Acceptance Mask bit functionality in the eCAN module found in older C28xx devices, where a "1" means the corresponding bit is NOT used for filtering, and "0" means it is used.
Xtd		Extended Identifier
	0	The 11-bit ("standard") identifier will be used for this message object.
	1	The 29-bit ("extended") identifier will be used for this message object.
MXtd		Mask Extended Identifier
	0	The extended identifier bit (IDE) has no effect on the acceptance filtering.
	1	The extended identifier bit (IDE) is used for acceptance filtering.
		Note: When 11-bit ("standard") Identifiers are used for a message object, the identifiers of received data frames are written into bits ID[28:18]. For acceptance filtering, only these bits together with mask bits Msk[28:18] are considered.
Dir		Message Direction
	0	Direction = receive: On TxRqst, a remote frame with the identifier of this message object is transmitted. On reception of a data frame with matching identifier, the message is stored in this message object.
	1	Direction = transmit: On TxRqst, a data frame is transmitted. On reception of a remote frame with matching identifier, the TxRqst bit of this message object is set (if RmtEn = 1).
MDir		Mask Message Direction
	0	The message direction bit (Dir) has no effect on the acceptance filtering.
	1	The message direction bit (Dir) is used for acceptance filtering.
EOB		End of Block
	0	The message object is part of a FIFO Buffer block and is not the last message object of this FIFO Buffer block.
	1	The message object is a single message object or the last message object in a FIFO Buffer Block.
		Note: This bit is used to concatenate multiple message objects to build a FIFO Buffer. For single message objects (not belonging to a FIFO Buffer), this bit must always be set to one.
NewDat		New Data
	0	No new data has been written into the data bytes of this message object by the message handler since the last time when this flag was cleared by the CPU.
	1	The message handler or the CPU has written new data into the data bytes of this message object.
MsgLst		Message Lost (only valid for Message Objects with direction = receive)
	0	No message was lost since the last time when this bit was reset by the CPU.
	1	The message handler stored a new message into this message object when NewDat was still set, so the previous message has been overwritten.
RxIE		Receive Interrupt Enable
	0	IntPnd will not be triggered after the successful reception of a frame.
	1	IntPnd will be triggered after the successful reception of a frame.
TxIE		Transmit Interrupt Enable
	0	IntPnd will not be triggered after the successful transmission of a frame.
	1	IntPnd will be triggered after the successful transmission of a frame.
IntPnd		Interrupt Pending
	0	This message object is not the source of an interrupt.
	1	This message object is the source of an interrupt. The Interrupt Identifier in the Interrupt Register will point to this message object if there is no other interrupt source with higher priority.

Table 28-5. Message Object Field Descriptions (continued)

Name	Value	Description
RmtEn		Remote Enable
	0	At the reception of a remote frame, TxRqst is not changed.
	1	At the reception of a remote frame, TxRqst is set.
		Note: See Section 28.11.8 for details on the setup of RmtEn and UMask for remote frames.
TxRqst		Transmit Request
	0	This message object is not waiting for a transmission.
	1	The transmission of this message object is requested and is not yet done.
DLC[3:0]		Data length code
	0-8	Data frame has 0-8 data bytes.
	9-15	Data frame has 8 data bytes.
		Note: The data length code of a message object must be defined to the same value as in the corresponding objects with the same identifier at other nodes. When the message handler stores a data frame, it will write the DLC to the value given by the received message.
Data 0		1st data byte of a CAN data frame
Data 1		2nd data byte of a CAN data frame
Data 2		3rd data byte of a CAN data frame
Data 3		4th data byte of a CAN data frame
Data 4		5th data byte of a CAN data frame
Data 5		6th data byte of a CAN data frame
Data 6		7th data byte of a CAN data frame
Data 7		8th data byte of a CAN data frame
		Note: Byte Data 0 is the first data byte shifted into the shift register of the CAN Core during a reception, byte Data 7 is the last. When the message handler stores a data frame, it will write all the eight data bytes into a message object. If the data length code is less than 8, the remaining bytes of the message object may be overwritten by undefined values.

28.14.2 Addressing Message Objects in RAM

The starting location of a particular message object in RAM is:

Message RAM base address + (message object number) * 0x20.

This means that Message Object 1 starts at offset 0x0020; Message Object 2 starts at offset 0x0040, and so on.

Note

'0' is not a valid message object number. At address 0x0000, the last message object (32) (with the lowest priority) is located. Writing to the address of an unimplemented message object may overwrite an implemented message object.

Message Object number 1 has the highest priority.

Table 28-6. Message RAM Addressing in Debug Mode

Message Object Number	Offset From Base Address	Word Number	Debug Mode(1)
last implemented (here:32)	0x0000	1	Parity
	0x0004	2	MXtd,MDir,Mask
	0x0008	3	Xtd,Dir,ID
	0x000C	4	Ctrl
	0x0010	5	Data Bytes 3-0
	0x0014	6	Data Bytes 7-4
1	0x0020	1	Parity
	0x0024	2	MXtd,MDir,Mask
	0x0028	3	Xtd,Dir,ID
	0x002C	4	Ctrl
	0x0030	5	Data Bytes 3-0
	0x0034	6	Data Bytes 7-4
2	0x0040	1	Parity
	0x0044	2	MXtd,MDir,Mask
	0x0048	3	Xtd,Dir,ID
	0x004C	4	Ctrl
	0x0050	5	Data Bytes 3-0
	0x0054	6	Data Bytes 7-4
…	…	…	…
31	0x03E0	1	Parity
	0x03E4	2	MXtd,MDir,Mask
	0x03E8	3	Xtd,Dir,ID
	0x03EC	4	Ctrl
	0x03F0	5	Data Bytes 3-0
	0x03F4	6	Data Bytes 7-4

(1) See Section 28.14.3.

28.14.3 Message RAM Representation in Debug Mode

In debug mode, the Message RAM will be memory mapped. This allows the external debug unit to access the Message RAM.

Note

During debug mode, the Message RAM cannot be accessed via the IFx register sets.

Figure 28-20. Message RAM Representation in Debug Mode

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0

MsgAddr + 0x00

Reserved	
Reserved	Parity[4:0]

MsgAddr + 0x04

MXtd	MDir	Rsvd	Msk[28:16]
Msk[15:0]			

MsgAddr + 0x08

Rsvd	Xtd	Dir	ID[28:16]
ID[15:0]			

MsgAddr + 0x0C

Reserved										
Rsvd	MsgLst	Rsvd	UMask	TxIE	RxIE	RmtEn	Rsvd	EOB	Reserved	DLC[3:0]

MsgAddr + 0x10

Data 3	Data 2
Data 1	Data 0

MsgAddr + 0x14

Data 7	Data 6
Data 5	Data 4

28.15 Software

28.15.1 CAN Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/can

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

28.15.1.1 CAN External Loopback

FILE: can_ex1_loopback.c

This example shows the basic setup of CAN in order to transmit and receive messages on the CAN bus. The CAN peripheral is configured to transmit messages with a specific CAN ID. A message is then transmitted once per second, using a simple delay loop for timing. The message that is sent is a 2 byte message that contains an incrementing pattern.

This example sets up the CAN controller in External Loopback test mode. Data transmitted is visible on the CANTXA pin and is received internally back to the CAN Core. Please refer to details of the External Loopback Test Mode in the CAN Chapter in the Technical Reference Manual. Please refer to the appnote Programming Examples and Debug Strategies for the DCAN Module (www.ti.com/lit/SPRACE5) for useful information about this example

External Connections

- None.

Watch Variables

- msgCount - A counter for the number of successful messages received
- txMsgData - An array with the data being sent
- rxMsgData - An array with the data that was received

28.15.1.2 CAN External Loopback with Interrupts

FILE: can_ex2_loopback_interrupts.c

This example shows the basic setup of CAN in order to transmit and receive messages on the CAN bus. The CAN peripheral is configured to transmit messages with a specific CAN ID. A message is then transmitted once per second, using a simple delay loop for timing. The message that is sent is a 4 byte message that contains an incrementing pattern. A CAN interrupt handler is used to confirm message transmission and count the number of messages that have been sent.

This example sets up the CAN controller in External Loopback test mode. Data transmitted is visible on the CANTXA pin and is received internally back to the CAN Core. Please refer to details of the External Loopback Test Mode in the CAN Chapter in the Technical Reference Manual. Please refer to the appnote Programming Examples and Debug Strategies for the DCAN Module (www.ti.com/lit/SPRACE5) for useful information about this example

External Connections

- None.

Watch Variables

- txMsgCount - A counter for the number of messages sent
- rxMsgCount - A counter for the number of messages received
- txMsgData - An array with the data being sent
- rxMsgData - An array with the data that was received
- errorFlag - A flag that indicates an error has occurred

28.15.1.3 CAN External Loopback with DMA

FILE: can_ex4_loopback_dma.c

This example sets up the CAN module to transmit and receive messages on the CAN bus. The CAN module is set to transmit a 4 byte message internally. An interrupt is used to assert the DMA request line which then triggers the DMA to transfer the received data from the CAN interface register to the receive buffer array. A data check is performed once the transfer is complete.

This example sets up the CAN controller in External Loopback test mode. Data transmitted is visible on the CANTXA pin and is received internally back to the CAN Core. Refer to details of the External Loopback Test Mode in the *CAN Chapter*. Refer to the *Programming Examples and Debug Strategies for the DCAN Module Application Report* for useful information about this example.

External Connections

- None.

Watch Variables

- txMsgCount - A counter for the number of messages sent
- rxMsgCount - A counter for the number of messages received
- txMsgData - An array with the data being sent
- rxMsgData - An array with the data that was received

28.15.1.4 CAN Transmit and Receive Configurations

FILE: can_ex5_transmit_receive.c

This example shows the basic setup of CAN in order to transmit or receive messages on the CAN bus with a specific Message ID. The CAN Controller is configured according to the selection of the define.

When the TRANSMIT define is selected, the CAN Controller acts as a Transmitter and sends data to the second CAN Controller connected externally.If TRANMSIT is not defined the CAN Controller acts as a Receiver and waits for message to be transmitted by the External CAN Controller. Refer to the *Programming Examples and Debug Strategies for the DCAN Module Application Report* for useful information about this example

CAN modules on the device need to be connected to via CAN transceivers.

Hardware Required

- A C2000 board with CAN transceiver.

External Connections

- ControlCARD CANA is on DEVICE_GPIO_PIN_CANTXA (CANTXA)
- and DEVICE_GPIO_PIN_CANRXA (CANRXA)

Watch Variables Transmit\Configuration

- MSGCOUNT - Adjust to set the number of messages
- txMsgCount - A counter for the number of messages sent
- txMsgData - An array with the data being sent
- errorFlag - A flag that indicates an error has occurred
- rxMsgCount - Has the initial value as Number of Messages to be received and decrements with each message.

28.15.1.5 CAN Error Generation Example

FILE: can_ex6_error_generation.c

This example demonstrates the ways of handling CAN Error conditions It generates the CAN Packets and sends them over GPIO It is looped back externally to be received in CAN module The CAN Interrupt service routine reads the Error status and demonstrates how different Error conditions can be detected.

Change ERR_CFG define to the different Error Scenarios and run the example. The corresponding Error Flag will be set in status variable of canaISR() routine. Uses a CPU Timer (Timer 0) for periodic timer interrupt of CANBITRATE uSec On the Timer interrupt it sends the required CAN Frame type with the specified error conditions CAN modules on the device need to be connected to via CAN transceivers. Refer to the *Configurable Error Generator for Controller Area Network Application Report* for further details on this example.

External Connections

- ControlCARD GPIOTX_PIN should be connected to
- DEVICE_GPIO_PIN_CANRXA(CANRXA)

Watch Variables Transmit\Configuration

- status - variable in canaISR for checking error Status

28.15.1.6 CAN Remote Request Loopback

FILE: can_ex7_loopback_tx_rx_remote_frame.c

This example shows the basic setup of CAN in order to transmit a remote frame and get a response for the remote frame and store it in a receive Object. The CAN peripheral is configured to transmit remote request frame and a remote answer frame messages with a specific CAN ID. Message object 3 is configured to transmit a remote request. Message object 2 is configured as a remote answer object with filter mask such that it accepts remote frame with any message ID and transmit's remote answer with message ID 7 and data length 8. Message object 1 is configured as a received object with filter message ID 7 so as to store the remote answer data transmitted by message object 2.

This example sets up the CAN controller in External Loopback test mode. Data transmitted is visible on the CANTXA pin and is received internally back to the CAN Core. Please refer to details of the External Loopback Test Mode in the CAN Chapter in the Technical Reference Manual.

External Connections

- None.

Watch Variables

- txMsgData - An array with the data being sent
- rxMsgData - An array with the data that was received

28.15.1.7 CAN Example that Illustrates the Usage of Mask Registers

FILE: can_ex8_mask.c

This example initializes CAN module A for Reception. When a frame with a matching filter criterion is received, the data will be copied in mailbox 1 and LED will be toggled a few times and the code gets ready for the next frame. If a message of any other MSGID is received, an ACK will be provided Completion of reception is determined by polling CAN_NDAT_21 register. No interrupts are used. Refer to the *Programming Examples and Debug Strategies for the DCAN Module Application Report* for useful information about this example

Hardware Required

- An external CAN node that transmits to CAN-A on the C2000 MCU

Watch Variables

- rxMsgCount - A counter for the number of messages received
- rxMsgData - An array with the data that was received

28.16 CAN Registers

This section describes the Controller Area Network registers.

28.16.1 CAN Base Address Table

Table 28-7. CAN Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
CanaRegs	CAN_REGS	CANA_BASE	0x0004_8000	YES	YES	YES	-	YES

28.16.2 CAN_REGS Registers

Table 28-8 lists the memory-mapped registers for the CAN_REGS registers. All register offset addresses not listed in Table 28-8 should be considered as reserved locations and the register contents should not be modified.

Table 28-8. CAN_REGS Registers

Offset (x8)	Offset (x16)	Acronym	Register Name	Write Protection	Section
0h	0h	CAN_CTL	CAN Control Register		Go
8h	4h	CAN_ES	Error and Status Register		Go
10h	8h	CAN_ERRC	Error Counter Register		Go
18h	Ch	CAN_BTR	Bit Timing Register		Go
20h	10h	CAN_INT	Interrupt Register		Go
28h	14h	CAN_TEST	Test Register		Go
38h	1Ch	CAN_PERR	CAN Parity Error Code Register		Go
80h	40h	CAN_RAM_INIT	CAN RAM Initialization Register		Go
A0h	50h	CAN_GLB_INT_EN	CAN Global Interrupt Enable Register		Go
A8h	54h	CAN_GLB_INT_FLG	CAN Global Interrupt Flag Register		Go
B0h	58h	CAN_GLB_INT_CLR	CAN Global Interrupt Clear Register		Go
100h	80h	CAN_ABOTR	Auto-Bus-On Time Register		Go
108h	84h	CAN_TXRQ_X	CAN Transmission Request Register		Go
110h	88h	CAN_TXRQ_21	CAN Transmission Request 2_1 Register		Go
130h	98h	CAN_NDAT_X	CAN New Data Register		Go
138h	9Ch	CAN_NDAT_21	CAN New Data 2_1 Register		Go
158h	ACh	CAN_IPEN_X	CAN Interrupt Pending Register		Go
160h	B0h	CAN_IPEN_21	CAN Interrupt Pending 2_1 Register		Go
180h	C0h	CAN_MVAL_X	CAN Message Valid Register		Go
188h	C4h	CAN_MVAL_21	CAN Message Valid 2_1 Register		Go
1B0h	D8h	CAN_IP_MUX21	CAN Interrupt Multiplexer 2_1 Register		Go
200h	100h	CAN_IF1CMD	IF1 Command Register		Go
208h	104h	CAN_IF1MSK	IF1 Mask Register		Go
210h	108h	CAN_IF1ARB	IF1 Arbitration Register		Go
218h	10Ch	CAN_IF1MCTL	IF1 Message Control Register		Go
220h	110h	CAN_IF1DATA	IF1 Data A Register		Go
228h	114h	CAN_IF1DATB	IF1 Data B Register		Go
240h	120h	CAN_IF2CMD	IF2 Command Register		Go
248h	124h	CAN_IF2MSK	IF2 Mask Register		Go
250h	128h	CAN_IF2ARB	IF2 Arbitration Register		Go
258h	12Ch	CAN_IF2MCTL	IF2 Message Control Register		Go
260h	130h	CAN_IF2DATA	IF2 Data A Register		Go
268h	134h	CAN_IF2DATB	IF2 Data B Register		Go
280h	140h	CAN_IF3OBS	IF3 Observation Register		Go
288h	144h	CAN_IF3MSK	IF3 Mask Register		Go
290h	148h	CAN_IF3ARB	IF3 Arbitration Register		Go
298h	14Ch	CAN_IF3MCTL	IF3 Message Control Register		Go
2A0h	150h	CAN_IF3DATA	IF3 Data A Register		Go
2A8h	154h	CAN_IF3DATB	IF3 Data B Register		Go
2C0h	160h	CAN_IF3UPD	IF3 Update Enable Register		Go

Complex bit access types are encoded to fit into small table cells. Table 28-9 shows the codes that are used for access types in this section.

Table 28-9. CAN_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

28.16.2.1 CAN_CTL Register (Offset (x8) = 0h, Offset (x16) = 0h) [Reset = 1401h]

CAN_CTL is shown in Figure 28-21 and described in Table 28-10.

Return to the Summary Table.

This register is used for configuring the CAN module in terms of interrupts, parity, debug-mode behavior etc.

Figure 28-21. CAN_CTL Register

31	30	29	28	27	26	25	24
RESERVED						RESERVED	RESERVED
R-0h						R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED			DE3	DE2	DE1	IE1	INITDBG
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h
15	14	13	12	11	10	9	8
SWR	RESERVED	PMD				ABO	IDS
R-0/W1C-0h	R-0h	R/W-5h				R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
Test	CCE	DAR	RESERVED	EIE	SIE	IE0	Init
R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-1h

Table 28-10. CAN_CTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved
25	RESERVED	R/W	0h	Reserved
24	RESERVED	R/W	0h	Reserved
23-21	RESERVED	R	0h	Reserved
20	DE3	R/W	0h	Enable DMA request line for IF3 0 Disabled 1 Enabled Note: A pending DMA request for IF3 remains active until first access to one of the IF3 registers. Reset type: SYSRSn
19	DE2	R/W	0h	Enable DMA request line for IF2 0 Disabled 1 Enabled Note: A pending DMA request for IF1 remains active until first access to one of the IF2 registers. Reset type: SYSRSn
18	DE1	R/W	0h	Enable DMA request line for IF1 0 Disabled 1 Enabled Note: A pending DMA request for IF1 remains active until first access to one of the IF1 registers. Reset type: SYSRSn
17	IE1	R/W	0h	Interrupt line 1 Enable 0 CANINT1 is disabled. 1 CANINT1 is enabled. Interrupts will assert CANINT1 line to 1 line remains active until pending interrupts are processed. Reset type: SYSRSn
16	INITDBG	R	0h	Debug Mode Status Bit: This bit indicates the internal init state for a debug access 0 Not in debug mode, or debug mode requested but not entered. 1 Debug mode requested and internally entered the CAN module is ready for debug accesses. Reset type: SYSRSn

Table 28-10. CAN_CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15	SWR	R-0/W1C	0h	Software Reset Enable Bit: This bit activates the software reset. 0 Normal Operation. 1 Module is forced to reset state. This bit will get cleared automatically one clock cycle after execution of software reset. Note: To execute software reset, the following procedure is necessary: 1. Set INIT bit to shut down CAN communication. 2. Set SWR bit. Note: This bit is write-protected by Init bit. If module is reset using the SWR bit, no user configuration is lost. Only status bits get reset along with logic which needs to be reset for the next CAN transaction. If module is reset using SOFTPRES register, entire module will get reset, including configuration registers. Reset type: SYSRSn
14	RESERVED	R	0h	Reserved
13-10	PMD	R/W	5h	Parity on/off 0101 Parity function disabled Any other value - Parity function enabled Reset type: SYSRSn
9	ABO	R/W	0h	Auto-Bus-On Enable 0 The Auto-Bus-On feature is disabled 1 The Auto-Bus-On feature is enabled Reset type: SYSRSn
8	IDS	R/W	0h	Interruption Debug Support Enable 0 When Debug mode is requested, the CAN module will wait for a started transmission or reception to be completed before entering Debug mode 1 When Debug mode is requested, the CAN module will interrupt any transmission or reception, and enter Debug mode immediately. Reset type: SYSRSn
7	Test	R/W	0h	Test Mode Enable 0 Disable Test Mode (Normal operation) 1 Enable Test Mode Reset type: SYSRSn
6	CCE	R/W	0h	Configuration Change Enable 0 The CPU has no write access to the configuration registers. 1 The CPU has write access to the configuration registers (when Init bit is set). Reset type: SYSRSn
5	DAR	R/W	0h	Disable Automatic Retransmission 0 Automatic Retransmission of "not successful" messages enabled. 1 Automatic Retransmission disabled. Reset type: SYSRSn
4	RESERVED	R	0h	Reserved
3	EIE	R/W	0h	Error Interrupt Enable 0 Disabled - PER, BOff and EWarn bits cannot generate an interrupt. 1 Enabled - PER, BOff and EWarn bits can generate an interrupt at CANINT0 line and affect the Interrupt Register. Reset type: SYSRSn
2	SIE	R/W	0h	Status Change Interrupt Enable 0 Disabled - RxOk, TxOk and LEC bits cannot generate an interrupt. 1 Enabled - RxOk, TxOk and LEC can generate an interrupt on the CANINT0 line Reset type: SYSRSn
1	IE0	R/W	0h	Interrupt line 0 Enable 0 CANINT0 is disabled. 1 CANINT0 is enabled. Interrupts will assert CANINT0 line to 1 line remains active until pending interrupts are processed. Reset type: SYSRSn

Table 28-10. CAN_CTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	Init	R/W	1h	Initialization Mode This bit is used to keep the CAN module inactive during bit timing configuration and message RAM initialization. It is set automatically during a bus off event. Clearing this bit will not shorten the bus recovery time. 0 CAN module processes messages normally 1 CAN module ignores bus activity Reset type: SYSRSn

28.16.2.2 CAN_ES Register (Offset (x8) = 8h, Offset (x16) = 4h) [Reset = 7h]

CAN_ES is shown in Figure 28-22 and described in Table 28-11.

Return to the Summary Table.

This register indicates error conditions, if any, of the CAN module. Interrupts are generated by PER, BOff and EWarn bits (if EIE bit in CAN Control Register is set) and by RxOk, TxOk, and LEC bits (if SIE bit in CAN Control Register is set). A change of bit EPass will not generate an Interrupt.
Reading the Error and Status Register clears the PER, RxOk and TxOk bits and sets the LEC to value '7'. Additionally, the Status Interrupt value (0x8000) in the Interrupt Register will be replaced by the next lower priority interrupt value.
For debug support, the auto clear functionality of Error and Status Register (clear of status flags by read) is disabled when in Debug/Suspend mode.

Figure 28-22. CAN_ES Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					RESERVED	RESERVED	PER
R-0h					R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
BOff	EWarn	EPass	RxOk	TxOk	LEC		
R-0h	R-0h	R-0h	R-0h	R-0h	R-7h		

Table 28-11. CAN_ES Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	Reserved
10	RESERVED	R	0h	Reserved
9	RESERVED	R	0h	Reserved
8	PER	R	0h	Parity Error Detected: This bit will be reset after the CPU reads the register. 0 No parity error has been detected since last read access. 1 The parity check mechanism has detected a parity error in the Message RAM. Reset type: SYSRSn
7	BOff	R	0h	Bus-off Status Bit: 0 The CAN module is not in Bus-Off state. 1 The CAN module is in Bus-Off state. Reset type: SYSRSn
6	EWarn	R	0h	Warning State Bit: 0 Both error counters are below the error warning limit of 96. 1 At least one of the error counters has reached the error warning limit of 96. Reset type: SYSRSn
5	EPass	R	0h	Error Passive State 0 On CAN Bus error, the CAN could send active error frames. 1 The CAN Core is in the error passive state as defined in the CAN Specification. Reset type: SYSRSn

Table 28-11. CAN_ES Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	RxOk	R	0h	Reception status Bit: This bit indicates the status of reception. The bit will be reset after the CPU reads the register. 0 No message has been successfully received since the last time when this bit was read by the CPU. This bit is never reset by CAN internal events. 1 A message has been successfully received since the last time when this bit was reset by a read access of the CPU. This bit will be set independent of the result of acceptance filtering. Reset type: SYSRSn
3	TxOk	R	0h	Transmission status Bit: This bit indicates the status of transmission. The bit will be reset after the CPU reads the register. 0 No message has been successfully transmitted since the last time when this bit was read by the CPU. This bit is never reset by CAN internal events. 1 A message has been successfully transmitted (error free and acknowledged by at least one other node) since the last time when this bit was cleared by a read access of the CPU. Reset type: SYSRSn
2-0	LEC	R	7h	Last Error Code The LEC field indicates the type of the last error on the CAN bus. This field will be cleared to '0' when a message has been transferred (reception or transmission) without error. This field will be reset to '7' whenever the CPU reads the register. 0 No Error 1 Stuff Error: More than five equal bits in a row have been detected in a part of a received message where this is not allowed. 2 Form Error: A fixed format part of a received frame has the wrong format. 3 Ack Error: The message this CAN Core transmitted was not acknowledged by another node. 4 Bit1 Error: During the transmission of a message (with the exception of the arbitration field), the device wanted to send a recessive level (bit of logical value '1'), but the monitored bus value was dominant. 5 Bit0 Error: During the transmission of a message (or acknowledge bit, or active error flag, or overload flag), the device wanted to send a dominant level (logical value '0'), but the monitored bus level was recessive. During Bus-Off recovery, this status is set each time a sequence of 11 recessive bits has been monitored. This enables the CPU to monitor the proceeding of the Bus-Off recovery sequence (indicating the bus is not stuck at dominant or continuously disturbed). 6 CRC Error: In a received message, the CRC check sum was incorrect. (CRC received for an incoming message does not match the calculated CRC for the received data). 7 No CAN bus event was detected since the last time when CPU has read the Error and Status Register. Any read access to the Error and Status Register re-initializes the LEC to value '7'. Reset type: SYSRSn

28.16.2.3 CAN_ERRC Register (Offset (x8) = 10h, Offset (x16) = 8h) [Reset = 0h]

CAN_ERRC is shown in Figure 28-23 and described in Table 28-12.

Return to the Summary Table.

This register reflects the value of the Transmit and Receive error counters

Figure 28-23. CAN_ERRC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RP	REC							TEC							
R-0h	R-0h							R-0h							

Table 28-12. CAN_ERRC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	RP	R	0h	Receive Error Passive 0 The Receive Error Counter is below the error passive level. 1 The Receive Error Counter has reached the error passive level as defined in the CAN Specification. Reset type: SYSRSn
14-8	REC	R	0h	Receive Error Counter Actual state of the Receive Error Counter (values from 0 to 127). Reset type: SYSRSn
7-0	TEC	R	0h	Transmit Error Counter Actual state of the Transmit Error Counter. (values from 0 to 255). Reset type: SYSRSn

28.16.2.4 CAN_BTR Register (Offset (x8) = 18h, Offset (x16) = Ch) [Reset = 2301h]

CAN_BTR is shown in Figure 28-24 and described in Table 28-13.

Return to the Summary Table.

This register is used to configure the bit-timing parameters for the CAN module. This register is only writable if CCE and Init bits in the CAN Control Register are set.
The CAN bit time may be programmed in the range of 8 to 25 time quanta.
The CAN time quantum may be programmed in the range of 1 to1024 CAN_CLK periods.

Figure 28-24. CAN_BTR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED				BRPE			
R-0h				R/W-0h			
15	14	13	12	11	10	9	8
RESERVED	TSEG2			TSEG1			
R-0h	R/W-2h			R/W-3h			
7	6	5	4	3	2	1	0
SJW		BRP					
R/W-0h		R/W-1h					

Table 28-13. CAN_BTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19-16	BRPE	R/W	0h	Baud Rate Prescaler Extension Valid programmed values are 0 to 15. By programming BRPE the Baud Rate Prescaler can be extended to values up to 1024. Note: This bit is Write Protected by CCE bit. Reset type: SYSRSn
15	RESERVED	R	0h	Reserved
14-12	TSEG2	R/W	2h	Time segment after the sample point Valid programmed values are 0 to 7. The actual TSeg2 value which is interpreted for the Bit Timing will be the programmed TSeg2 value + 1. Note: This bit is Write Protected by CCE bit. Reset type: SYSRSn
11-8	TSEG1	R/W	3h	Time segment before the sample point Valid programmed values are 1 to 15. The actual TSeg1 value interpreted for the Bit Timing will be the programmed TSeg1 value + 1. Note: This bit is Write Protected by CCE bit. Reset type: SYSRSn
7-6	SJW	R/W	0h	Synchronization Jump Width Valid programmed values are 0 to 3. The actual SJW value interpreted for the Synchronization will be the programmed SJW value + 1. Note: This bit is Write Protected by CCE bit. Reset type: SYSRSn

Table 28-13. CAN_BTR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-0	BRP	R/W	1h	Baud Rate Prescaler- Value by which the CAN_CLK frequency is divided for generating the bit time quanta. The bit time is built up from a multiple of this quanta. Valid programmed values are 0 to 63. The actual BRP value interpreted for the Bit Timing will be the programmed BRP value + 1. Note: This bit is Write Protected by CCE bit. Reset type: SYSRSn

28.16.2.5 CAN_INT Register (Offset (x8) = 20h, Offset (x16) = 10h) [Reset = 0h]

CAN_INT is shown in Figure 28-25 and described in Table 28-14.

Return to the Summary Table.

This register is used to identify the source of the interrupt(s).

Figure 28-25. CAN_INT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								INT1ID								INT0ID															
R-0h								R-0h								R-0h															

Table 28-14. CAN_INT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	INT1ID	R	0h	Interrupt 1 Cause 0x00 No interrupt is pending. 0x01-0x20 Number of message object (mailbox) which caused the interrupt. 0x21-0xFF Unused. If several interrupts are pending, the CAN Interrupt Register will point to the pending interrupt with the highest priority. Note: The CANINT1 interrupt line remains active until INT1ID reaches value 0 (the cause of the interrupt is reset) or until IE0 is cleared. A message interrupt is cleared by clearing the mailbox's IntPnd bit. Among the message interrupts, the mailbox's interrupt priority decreases with increasing message number. Reset type: SYSRSn
15-0	INT0ID	R	0h	Interrupt 0 Cause 0x0000 - No interrupt is pending. 0x0001 - 0x0020 - Number of message object which caused the interrupt. 0x0021 - 0x7FFF - Unused. 0x8000 - Error and Status Register value is not 0x07. 0x8001 - 0xFFFF - Unused. If several interrupts are pending, the CAN Interrupt Register will point to the pending interrupt with the highest priority. Note: The CANINT0 interrupt line remains active until INT0ID reaches value 0 (the cause of the interrupt is reset) or until IE0 is cleared. The Status Interrupt has the highest priority. Among the message interrupts, the message object's interrupt priority decreases with increasing message number. Reset type: SYSRSn

28.16.2.6 CAN_TEST Register (Offset (x8) = 28h, Offset (x16) = 14h) [Reset = 0h]

CAN_TEST is shown in Figure 28-26 and described in Table 28-15.

Return to the Summary Table.

This register is used to configure the various test options supported. For all test modes, the Test bit in CAN Control Register needs to be set to one. If Test bit is set, the RDA, EXL, Tx1, Tx0, LBack and Silent bits are writable. Bit Rx monitors the state of CANRX pin and therefore is only readable. All Test Register functions are disabled when Test bit is cleared.
Note: Setting Tx[1:0] other than '00' will disturb message transfer.
Note: When the internal loop back mode is active (bit LBack is set), bit EXL will be ignored.

Figure 28-26. CAN_TEST Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						RDA	EXL
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RX	TX		LBACK	SILENT	RESERVED		
R-0h	R/W-0h		R/W-0h	R/W-0h	R-0h		

Table 28-15. CAN_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9	RDA	R/W	0h	RAM Direct Access Enable: 0 Normal Operation. 1 Direct access to the RAM is enabled while in Test Mode. Reset type: SYSRSn
8	EXL	R/W	0h	External Loop Back Mode: 0 Disabled. 1 Enabled. Reset type: SYSRSn
7	RX	R	0h	Monitors the actual value of the CANRX pin: 0 The CAN bus is dominant. 1 The CAN bus is recessive. Reset type: SYSRSn
6-5	TX	R/W	0h	Control of CANTX pin: 00 Normal operation, CANTX is controlled by the CAN Core. 01 Sample Point can be monitored at CANTX pin. 10 CANTX pin drives a dominant value. 11 CANTX pin drives a recessive value. Reset type: SYSRSn
4	LBACK	R/W	0h	Loop Back Mode: 0 Disabled. 1 Enabled. Reset type: SYSRSn
3	SILENT	R/W	0h	Silent Mode: 0 Disabled. 1 Enabled. Reset type: SYSRSn

Table 28-15. CAN_TEST Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-0	RESERVED	R	0h	Reserved

28.16.2.7 CAN_PERR Register (Offset (x8) = 38h, Offset (x16) = 1Ch) [Reset = 100h]

CAN_PERR is shown in Figure 28-27 and described in Table 28-16.

Return to the Summary Table.

This register indicates the Word/Mailbox number where a parity error has been detected. If a parity error is detected, the PER flag will be set in the Error and Status Register. This bit is not reset by the parity check mechanism
it must be reset by reading the Error and Status Register. In addition to the PER flag, the Parity Error Code Register will indicate the memory area where the parity error has been detected. If more than one word with a parity error was detected, the highest word number with a parity error will be displayed. After a parity error has been detected, the register will hold the last error code until power is removed.

Figure 28-27. CAN_PERR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED					WORD_NUM			MSG_NUM							
R-0h					R-1h			R-0h							

Table 28-16. CAN_PERR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	Reserved
10-8	WORD_NUM	R	1h	0x01-0x05 Word number where parity error has been detected. RDA word number (1 to 5) of the mailbox (according to the Message RAM representation in RDA mode). Reset type: SYSRSn
7-0	MSG_NUM	R	0h	0x01-0x21 Mailbox number where parity error has been detected Reset type: SYSRSn

28.16.2.8 CAN_RAM_INIT Register (Offset (x8) = 80h, Offset (x16) = 40h) [Reset = 5h]

CAN_RAM_INIT is shown in Figure 28-28 and described in Table 28-17.

Return to the Summary Table.

This register is used to initialize the Mailbox RAM. It clears the entire mailbox RAM, including the MsgVal bits.

Figure 28-28. CAN_RAM_INIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		RAM_INIT_DONE	CAN_RAM_INIT	KEY3	KEY2	KEY1	KEY0
R-0h		R-0h	R/W-0h	R/W-0h	R/W-1h	R/W-0h	R/W-1h

Table 28-17. CAN_RAM_INIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5	RAM_INIT_DONE	R	0h	CAN Mailbox RAM initialization status: 0 Read: Initialization is on-going or initialization not initiated. 1 Read: Initialization complete Reset type: SYSRSn
4	CAN_RAM_INIT	R/W	0h	Initiate CAN Mailbox RAM initialization: 0 Read: Initialization complete or initialization not initiated. Write: No action 1 Read: Initialization is on-going Write: Initiate CAN Mailbox RAM initialization. After initialization, this bit will be automatically cleared to 0. Reset type: SYSRSn
3	KEY3	R/W	0h	See Key 0 Reset type: SYSRSn
2	KEY2	R/W	1h	See Key 0 Reset type: SYSRSn
1	KEY1	R/W	0h	See Key 0 Reset type: SYSRSn
0	KEY0	R/W	1h	KEY3-KEY0 should be 1010 for any write to this register to be valid. These bits will be restored to their reset state after the CAN RAM initialization is complete. Reset type: SYSRSn

28.16.2.9 CAN_GLB_INT_EN Register (Offset (x8) = A0h, Offset (x16) = 50h) [Reset = 0h]

CAN_GLB_INT_EN is shown in Figure 28-29 and described in Table 28-18.

Return to the Summary Table.

This register is used to enable the interrupt lines to the PIE.

Figure 28-29. CAN_GLB_INT_EN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						GLBINT1_EN	GLBINT0_EN
R-0h						R/W-0h	R/W-0h

Table 28-18. CAN_GLB_INT_EN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	GLBINT1_EN	R/W	0h	Global Interrupt Enable for CANINT1 0 CANINT1 does not generate interrupt to PIE 1 CANINT1 generates interrupt to PIE if interrupt condition occurs Reset type: SYSRSn
0	GLBINT0_EN	R/W	0h	Global Interrupt Enable for CANINT0 0 CANINT0 does not generate interrupt to PIE 1 CANINT0 generates interrupt to PIE if interrupt condition occurs Reset type: SYSRSn

28.16.2.10 CAN_GLB_INT_FLG Register (Offset (x8) = A8h, Offset (x16) = 54h) [Reset = 0h]

CAN_GLB_INT_FLG is shown in Figure 28-30 and described in Table 28-19.

Return to the Summary Table.

This register indicates if and when the interrupt line to the PIE is active.

Figure 28-30. CAN_GLB_INT_FLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						INT1_FLG	INT0_FLG
R-0h						R-0h	R-0h

Table 28-19. CAN_GLB_INT_FLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	INT1_FLG	R	0h	CANINT1 Flag 0 No interrupt generated 1 Interrupt is generated due to CANINT1 (refer to CAN Interrupt Status Register for the condition) Reset type: SYSRSn
0	INT0_FLG	R	0h	CANINT0 Flag 0 No interrupt generated 1 Interrupt is generated due to CANINT0 (refer to CAN Interrupt Status Register for the condition) Reset type: SYSRSn

28.16.2.11 CAN_GLB_INT_CLR Register (Offset (x8) = B0h, Offset (x16) = 58h) [Reset = 0h]

CAN_GLB_INT_CLR is shown in Figure 28-31 and described in Table 28-20.

Return to the Summary Table.

This register is used to clear the interrupt to the PIE.

Figure 28-31. CAN_GLB_INT_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						INT1_FLG_CLR	INT0_FLG_CLR
R-0h						W-0h	W-0h

Table 28-20. CAN_GLB_INT_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	INT1_FLG_CLR	W	0h	Global Interrupt flag clear for CANINT1 0 No effect 1 Write 1 to clear the corresponding bit of the Global Interrupt Flag Register and allow the PIE to receive another interrupt from CANINT1. Reset type: SYSRSn
0	INT0_FLG_CLR	W	0h	Global Interrupt flag clear for CANINT0 0 No effect 1 Write 1 to clear the corresponding bit of the Global Interrupt Flag Register and allow the PIE to receive another interrupt from CANINT0. Reset type: SYSRSn

28.16.2.12 CAN_ABOTR Register (Offset (x8) = 100h, Offset (x16) = 80h) [Reset = 0h]

CAN_ABOTR is shown in Figure 28-32 and described in Table 28-21.

Return to the Summary Table.

This register is used to introduce a variable delay before the Bus-off recovery sequence is started.

Figure 28-32. CAN_ABOTR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ABO_Time																															
R/W-0h																															

Table 28-21. CAN_ABOTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ABO_Time	R/W	0h	Auto-Bus-On Timer Number of clock cycles before a Bus-Off recovery sequence is started by clearing the Init bit. "Clock" refers to the input clock to the CAN module. This function has to be enabled by setting bit ABO in CAN Control Register. The Auto-Bus-On timer is realized by a 32-bit counter which starts to count down to zero when the module goes Bus-Off. The counter will be reloaded with the preload value of the ABO Time register after this phase. NOTE: On write access to the CAN Control register while Auto-Bus-On timer is running, the Auto-Bus-On procedure will be aborted. NOTE: During Debug mode, running Auto-Bus-On timer will be paused. Reset type: SYSRSn

28.16.2.13 CAN_TXRQ_X Register (Offset (x8) = 108h, Offset (x16) = 84h) [Reset = 0h]

CAN_TXRQ_X is shown in Figure 28-33 and described in Table 28-22.

Return to the Summary Table.

With these bits, the CPU can detect if one or more bits in the CAN Transmission Request 21 Register (CAN_TXRQ_21) is set. Each bit in this register represents a group of eight mailboxes. If at least one of the TxRqst bits of these message objects is set, the corresponding bit in this register will be set.

Figure 28-33. CAN_TXRQ_X Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				TxRqstReg2		TxRqstReg1	
R-0h				R-0h		R-0h	

Table 28-22. CAN_TXRQ_X Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3-2	TxRqstReg2	R	0h	Transmit Request Register 2 flag: Bit 2 represents byte 2 of CAN_TXRQ_21. If one or more bits in that byte are set, then bit 2 will be set. Bit 3 represents byte 3 of CAN_TXRQ_21 Register. If one or more bits in that byte are set, then bit 3 will be set. Reset type: SYSRSn
1-0	TxRqstReg1	R	0h	Transmit Request Register 1 flag: Bit 0 represents byte 0 of CAN_TXRQ_21 Register. If one or more bits in that byte are set, then bit 0 will be set. Bit 1 represents byte 1 of CAN_TXRQ_21 Register. If one or more bits in that byte are set, then bit 1 will be set. Reset type: SYSRSn

28.16.2.14 CAN_TXRQ_21 Register (Offset (x8) = 110h, Offset (x16) = 88h) [Reset = 0h]

CAN_TXRQ_21 is shown in Figure 28-34 and described in Table 28-23.

Return to the Summary Table.

This register holds the TxRqst bits of the mailboxes. By reading out these bits, the CPU can check for pending transmission requests. The TxRqst bit in a specific mailbox can be set/reset by the CPU via the IF1/IF2 message interface registers, or by the message handler after reception of a remote frame or after a successful transmission.

Figure 28-34. CAN_TXRQ_21 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TxRqst																															
R-0h																															

Table 28-23. CAN_TXRQ_21 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TxRqst	R	0h	Transmission Request Bits (for all message objects) 0 No transmission has been requested for this message object. 1 The transmission of this message object is requested and is not yet done. Note: Bit 0 is for mailbox 1, Bit 1 is for mailbox 2, Bit 2 is for mailbox 3,..., Bit 31 is for mailbox 32 Reset type: SYSRSn

28.16.2.15 CAN_NDAT_X Register (Offset (x8) = 130h, Offset (x16) = 98h) [Reset = 0h]

CAN_NDAT_X is shown in Figure 28-35 and described in Table 28-24.

Return to the Summary Table.

With these bits, the CPU can detect if one or more bits in the CAN New Data 21 Register (CAN_NDAT _21) is set. Each bit in this register represents a group of eight mailboxes. If at least one of the NewDat bits of these mailboxes are set, the corresponding bit in this register will be set.

Figure 28-35. CAN_NDAT_X Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				NewDatReg2		NewDatReg1	
R-0h				R-0h		R-0h	

Table 28-24. CAN_NDAT_X Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3-2	NewDatReg2	R	0h	New Data Register 2 flag: Bit 2 represents byte 2 of CAN_NDAT _21 Register. If one or more bits in that byte are set, then bit 2 will be set. Bit 3 represents byte 3 of CAN_NDAT _21 Register. If one or more bits in that byte are set, then bit 3 will be set. Reset type: SYSRSn
1-0	NewDatReg1	R	0h	New Data Register 1 flag: Bit 0 represents byte 0 of CAN_NDAT _21 Register. If one or more bits in that byte are set, then bit 0 will be set. Bit 1 represents byte 1 of CAN_NDAT _21 Register. If one or more bits in that byte are set, then bit 1 will be set. Reset type: SYSRSn

28.16.2.16 CAN_NDAT_21 Register (Offset (x8) = 138h, Offset (x16) = 9Ch) [Reset = 0h]

CAN_NDAT_21 is shown in Figure 28-36 and described in Table 28-25.

Return to the Summary Table.

This register holds the NewDat bits of all mailboxes. By reading out the NewDat bits, the CPU can check for which mailboxes the data portion was updated. The NewDat bit of a specific mailbox can be set/reset by the CPU via the IFx "Message Interface" Registers or by the Message Handler after reception of a Data Frame or after a successful transmission.

Figure 28-36. CAN_NDAT_21 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NewDat																															
R-0h																															

Table 28-25. CAN_NDAT_21 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	NewDat	R	0h	New Data Bits (for all message objects) 0 No new data has been written into the data portion of this message object by the message handler since the last time when this flag was cleared by the CPU. 1 The message handler or the CPU has written new data into the data portion of this message object. Note: Bit 0 is for mailbox 1, Bit 1 is for mailbox 2, Bit 2 is for mailbox 3,..., Bit 31 is for mailbox 32 Reset type: SYSRSn

28.16.2.17 CAN_IPEN_X Register (Offset (x8) = 158h, Offset (x16) = ACh) [Reset = 0h]

CAN_IPEN_X is shown in Figure 28-37 and described in Table 28-26.

Return to the Summary Table.

With these bits, the CPU can detect if one or more bits in the CAN Interrupt Pending 21 Register (CAN_IPEN_21) is set. Each bit in this register represents a group of eight mailboxes. If at least one of the IntPnd bits of these mailboxes are set, the corresponding bit in this register will be set.

Figure 28-37. CAN_IPEN_X Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				IntPndReg2		IntPndReg1	
R-0h				R-0h		R-0h	

Table 28-26. CAN_IPEN_X Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3-2	IntPndReg2	R	0h	Interrupt Pending Register 2 flag: Bit 2 represents byte 2 of CAN_IPEN_21 Register. If one or more bits in that byte are set, then bit 2 will be set. Bit 3 represents byte 3 of CAN_IPEN_21 Register. If one or more bits in that byte are set, then bit 3 will be set. Reset type: SYSRSn
1-0	IntPndReg1	R	0h	Interrupt Pending Register 1 flag: Bit 0 represents byte 0 of CAN_IPEN_21 Register. If one or more bits in that byte are set, then bit 0 will be set. Bit 1 represents byte 1 of CAN_IPEN_21 Register. If one or more bits in that byte are set, then bit 1 will be set. Reset type: SYSRSn

28.16.2.18 CAN_IPEN_21 Register (Offset (x8) = 160h, Offset (x16) = B0h) [Reset = 0h]

CAN_IPEN_21 is shown in Figure 28-38 and described in Table 28-27.

Return to the Summary Table.

This register holds the IntPnd bits of the mailboxes. By reading out these bits, the CPU can check for pending interrupts in the mailboxes. The IntPnd bit of a specific mailbox can be set/reset by the CPU via the IF1/IF2 interface register sets, or by the message handler after a reception or a successful transmission.

Figure 28-38. CAN_IPEN_21 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IntPnd																															
R-0h																															

Table 28-27. CAN_IPEN_21 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IntPnd	R	0h	Interrupt Pending bits: This register contains the bits that indicate the pending interrupts in each one of the 32 mailboxes. 0 This mailbox is not the source of an interrupt. 1 This mailbox is the source of an interrupt. Note: Bit 0 is for mailbox 1, Bit 1 is for mailbox 2, Bit 2 is for mailbox 3,..., Bit 31 is for mailbox 32 Reset type: SYSRSn

28.16.2.19 CAN_MVAL_X Register (Offset (x8) = 180h, Offset (x16) = C0h) [Reset = 0h]

CAN_MVAL_X is shown in Figure 28-39 and described in Table 28-28.

Return to the Summary Table.

With these bits, the CPU can detect if one or more bits in the CAN Message Valid 2_1 Register (CAN_MVAL_21) is set.Each bit in this register represents a group of eight mailboxes. If at least one of the MsgVal bits of these mailboxes are set, the corresponding bit in this register will be set.

Figure 28-39. CAN_MVAL_X Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				MsgValReg2		MsgValReg1	
R-0h				R-0h		R-0h	

Table 28-28. CAN_MVAL_X Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3-2	MsgValReg2	R	0h	Message Valid Register 2 flag: Bit 2 represents byte 2 of CAN_ MVAL _21 Register. If one or more bits in that byte are set, then bit 2 will be set. Bit 3 represents byte 3 of CAN_ MVAL _21 Register. If one or more bits in that byte are set, then bit 3 will be set. Reset type: SYSRSn
1-0	MsgValReg1	R	0h	Message Valid Register 1 flag: Bit 0 represents byte 0 of CAN_ MVAL _21 Register. If one or more bits in that byte are set, then bit 0 will be set. Bit 1 represents byte 1 of CAN_ MVAL _21 Register. If one or more bits in that byte are set, then bit 1 will be set. Reset type: SYSRSn

28.16.2.20 CAN_MVAL_21 Register (Offset (x8) = 188h, Offset (x16) = C4h) [Reset = 0h]

CAN_MVAL_21 is shown in Figure 28-40 and described in Table 28-29.

Return to the Summary Table.

This registers hold the MsgVal bits of all mailboxes. By reading out the MsgVal bits, the CPU can check which mailbox is valid. The MsgVal bit of a specific mailbox can be set/reset by the CPU via the IF1/2 "Message Interface" Registers.

Figure 28-40. CAN_MVAL_21 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MsgValReg																															
R-0h																															

Table 28-29. CAN_MVAL_21 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	MsgValReg	R	0h	Message Valid Bits (for all message objects) 0 This message object is ignored by the message handler. 1 This message object is configured and will be considered by the message handler. Note: Bit 0 is for mailbox 1, Bit 1 is for mailbox 2, Bit 2 is for mailbox 3,..., Bit 31 is for mailbox 32 Reset type: SYSRSn

28.16.2.21 CAN_IP_MUX21 Register (Offset (x8) = 1B0h, Offset (x16) = D8h) [Reset = 0h]

CAN_IP_MUX21 is shown in Figure 28-41 and described in Table 28-30.

Return to the Summary Table.

The IntMux bit determines for each mailbox, which of the two interrupt lines (CANINT0 or CANINT1) will be asserted when the IntPnd bit of that mailbox is set. Both interrupt lines can be globally enabled or disabled by setting or clearing IE0 and IE1 bits in CAN Control Register. This will also affect the INT0ID or INT1ID flags in the Interrupt Register.

Figure 28-41. CAN_IP_MUX21 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IntMux																															
R/W-0h																															

Table 28-30. CAN_IP_MUX21 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IntMux	R/W	0h	Interrupt Mux bits: 0 CANINT0 line is active if corresponding IntPnd flag is one. 1 CANINT1 line is active if corresponding IntPnd flag is one. Note: Bit 0 is for mailbox 32, Bit 1 is for mailbox 1, Bit 2 is for mailbox 2,..., Bit 31 is for mailbox 31 Reset type: SYSRSn

28.16.2.22 CAN_IF1CMD Register (Offset (x8) = 200h, Offset (x16) = 100h) [Reset = 1h]

CAN_IF1CMD is shown in Figure 28-42 and described in Table 28-31.

Return to the Summary Table.

The IF1/IF2 Command Registers configure and initiate the transfer between the IF1/IF2 Register sets and the Message RAM. It is configurable which portions of the message object should be transferred. A transfer is started when the CPU writes the message number to bits [7:0] of the IF1/IF2 Command Register. With this write operation, the Busy bit is automatically set to '1' to indicate that a transfer is in progress. After 4 to 14 clock cycles, the transfer between the Interface Register and the Message RAM will be completed and the Busy bit is cleared. The maximum number of cycles is needed when the message transfer coincides with a CAN message transmission, acceptance filtering, or message storage.
If the CPU writes to both IF1/IF2 Command Registers consecutively (request of a second transfer while first transfer is still in progress), the second transfer will start after the first one has been completed. The following points must be borne in mind while writing to this register: (1) Do not write zeros to the whole register. (2) Write to the register in a single 32-bit write or write the upper 16-bits before writing to the lower 16- bits.
Note: While Busy bit is one, IF1/IF2 Register sets are write protected.
Note: For debug support, the auto clear functionality of the IF1/IF2 Command Registers (clear of DMAactive flag by R/W, for devices with DMA support) is disabled during Debug/Suspend mode.
Note: If an invalid Message Number is written to bits [7:0] of the IF1/IF2 Command Register, the Message Handler may access an implemented (valid) message object instead.

Figure 28-42. CAN_IF1CMD Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
DIR	Mask	Arb	Control	ClrIntPnd	TXRQST	DATA_A	DATA_B
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
Busy	DMAactive	RESERVED					
R-0h	R/W-0h	R-0h					
7	6	5	4	3	2	1	0
MSG_NUM							
R/W-1h							

Table 28-31. CAN_IF1CMD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23	DIR	R/W	0h	Write/Read 0 Direction = Read: Transfer direction is from the message object addressed by Message Number (Bits [7:0]) to the IF1/IF2 Register set. That is, transfer data from the mailbox into the selected IF1/IF2 Message Buffer Registers. 1 Direction = Write: Transfer direction is from the IF1/IF2 Register set to the message object addressed by Message Number (Bits [7:0]) . That is, transfer data from the selected IF1/IF2 Message Buffer Registers to the mailbox. The other bits of IF1/IF2 Command Mask Register have different functions depending on the transfer direction. Note: This bit is write protected by Busy bit. Reset type: SYSRSn

Table 28-31. CAN_IF1CMD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
22	Mask	R/W	0h	Access Mask Bits 0 Mask bits will not be changed 1 (Direction = Read): The Mask bits (Identifier Mask + MDir + MXtd) will be transferred from the message object addressed by Message Number (Bits [7:0]) to the IF1/IF2 Register set. 1 (Direction = Write): The Mask bits (Identifier Mask + MDir + MXtd) will be transferred from the IF1/IF2 Register set to the message object addressed by Message Number (Bits [7:0]). Note: This bit is write protected by Busy bit. Reset type: SYSRSn
21	Arb	R/W	0h	Access Arbitration Bits 0 Arbitration bits will not be changed 1 (Direction = Read): The Arbitration bits (Identifier + Dir + Xtd + MsgVal) will be transferred from the message object addressed by Message Number (Bits [7:0]) to the corresponding IF1/IF2 Register set. 1 (Direction = Write): The Arbitration bits (Identifier + Dir + Xtd + MsgVal) will be transferred from the IF1/IF2 Register set to the message object addressed by Message Number (Bits [7:0]). Note: This bit is write protected by Busy bit. Reset type: SYSRSn
20	Control	R/W	0h	Access control bits. If the TxRqst/NewDat bit in this register(Bit [18]) is set, the TxRqst/NewDat bit in the IF1 message control register will be ignored. 0 Control bits will not be changed. 1 (Direction = Read): The message control bits will be transferred from the message object addressed by message number (Bits [7:0]) to the IF1 register set. 1 (Direction = Write): The message control bits will be transferred from the IF1 register set to the message object addressed by message number (Bits [7:0]). Note: This bit is write protected by the Busy bit. Reset type: SYSRSn
19	ClrIntPnd	R/W	0h	Clear Interrupt Pending Bit 0 IntPnd bit will not be changed 1 (Direction = Read): Clears IntPnd bit in the message object. 1 (Direction = Write): This bit is ignored. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
18	TXRQST	R/W	0h	Access Transmission Request (TxRqst) / New Data (NewDat) Bit 0 (Direction = Read): NewDat bit will not be changed. 0 (Direction = Write): TxRqst/NewDat bit will be handled according to the Control bit. 1 (Direction = Read): Clears NewDat bit in the message object. 1 (Direction = Write): Sets TxRqst/NewDat in message object. Note: If a CAN transmission is requested by setting TxRqst/NewDat in this register, the TxRqst/NewDat bits in the message object will be set to one independent of the values in IF1/IF2 Message Control Register. Note: A read access to a message object can be combined with the reset of the control bits IntPnd and NewDat. The values of these bits transferred to the IF1/IF2 Message Control Register always reflect the status before resetting them. Note: This bit is write protected by Busy bit. Reset type: SYSRSn

Table 28-31. CAN_IF1CMD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	DATA_A	R/W	0h	Access Data Bytes 0-3 0 Data Bytes 0-3 will not be changed. 1 (Direction = Read): The Data Bytes 0-3 will be transferred from the message object addressed by the Message Number (Bits [7:0]) to the corresponding IF1/IF2 Register set. 1 (Direction = Write): The Data Bytes 0-3 will be transferred from the IF1/IF2 Register set to the message object addressed by the Message Number (Bits [7:0]). Note: The duration of the message transfer is independent of the number of bytes to be transferred. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
16	DATA_B	R/W	0h	Access Data Bytes 4-7 0 Data Bytes 4-7 will not be changed. 1 (Direction = Read): The Data Bytes 4-7 will be transferred from the message object addressed by Message Number (Bits [7:0]) to the corresponding IF1/IF2 Register set. 1 (Direction = Write): The Data Bytes 4-7 will be transferred from the IF1/IF2 Register set to the message object addressed by Message Number (Bits [7:0]). Note: The duration of the message transfer is independent of the number of bytes to be transferred. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
15	Busy	R	0h	Busy Flag 0 No transfer between IF1/IF2 Register Set and Message RAM is in progress. 1 Transfer between IF1/IF2 Register Set and Message RAM is in progress. This bit is set to one after the message number has been written to bits [7:0]. IF1/IF2 Register Set will be write protected. The bit is cleared after read/write action has been finished. Reset type: SYSRSn
14	DMAactive	R/W	0h	DMA trigger status due to IF1 update. 0 No IF1 DMA request is active. 1 DMA is requested after a completed transfer between IF1 and the message RAM. The DMA request remains active until the first read or write to one of the IF1 registers an exception is a write to Message Number (Bits [7:0]) when DMAactive is one. Note: Due to the auto reset feature of the DMAactive bit, this bit has to be set for each subsequent DMA cycle separately. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
13-8	RESERVED	R	0h	Reserved
7-0	MSG_NUM	R/W	1h	Number of message object in Message RAM which is used for data transfer 0x00 Invalid message number 0x01-0x20 Valid message numbers 0x21-0xFF Invalid message numbers Note: This bit is write protected by Busy bit. Reset type: SYSRSn

28.16.2.23 CAN_IF1MSK Register (Offset (x8) = 208h, Offset (x16) = 104h) [Reset = FFFFFFFFh]

CAN_IF1MSK is shown in Figure 28-43 and described in Table 28-32.

Return to the Summary Table.

The bits of the IF1/IF2 Mask Registers mirror the mask bits of a message object.
Note: While Busy bit of IF1/IF2 Command Register is one, IF1/IF2 Register Set is write-protected.

Figure 28-43. CAN_IF1MSK Register

31	30	29	28	27	26	25	24
MXtd	MDir	RESERVED	Msk				
R/W-1h	R/W-1h	R-0h	R/W-1FFFFFFFh				
23	22	21	20	19	18	17	16
Msk							
R/W-1FFFFFFFh							
15	14	13	12	11	10	9	8
Msk							
R/W-1FFFFFFFh							
7	6	5	4	3	2	1	0
Msk							
R/W-1FFFFFFFh							

Table 28-32. CAN_IF1MSK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MXtd	R/W	1h	Mask Extended Identifier 0 The extended identifier bit (Xtd) has no effect on the acceptance filtering. 1 The extended identifier bit (Xtd) is used for acceptance filtering. When 11-bit ("standard") identifiers are used for a message object, the identifiers of received data frames are written into bits ID[28:18]. For acceptance filtering, only these bits together with mask bits Msk[28:18] are considered. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
30	MDir	R/W	1h	Mask Message Direction 0 The message direction bit (Dir) has no effect on the acceptance filtering. 1 The message direction bit (Dir) is used for acceptance filtering. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
29	RESERVED	R	0h	Reserved
28-0	Msk	R/W	1FFFFFFFh	Identifier Mask- 0 The corresponding bit in the identifier of the message object is not used for acceptance filtering (don't care). 1 The corresponding bit in the identifier of the message object is used for acceptance filtering. Note: This bit is write protected by Busy bit. Reset type: SYSRSn

28.16.2.24 CAN_IF1ARB Register (Offset (x8) = 210h, Offset (x16) = 108h) [Reset = 0h]

CAN_IF1ARB is shown in Figure 28-44 and described in Table 28-33.

Return to the Summary Table.

The bits of the IF1/IF2 Arbitration Registers mirror the arbitration bits of a message object. The Arbitration bits ID[28:0], Xtd, and Dir are used to define the identifier and type of outgoing messages and (together with the Mask bits Msk[28:0], MXtd, and MDir) for acceptance filtering of incoming messages.
A received message is stored into the valid message object with matching identifier and Direction = receive (Data Frame) or Direction = transmit (Remote Frame).
Extended frames can be stored only in message objects with Xtd = one, standard frames in message objects with Xtd = zero.
If a received message (Data Frame or Remote Frame) matches more than one valid message objects, it is stored into the one with the lowest message number.
Note: While Busy bit of IF1/IF2 Command Register is one, IF1/IF2 Register Set is write-protected.

Figure 28-44. CAN_IF1ARB Register

31	30	29	28	27	26	25	24
MsgVal	Xtd	Dir	ID				
R/W-0h	R/W-0h	R/W-0h	R/W-0h				
23	22	21	20	19	18	17	16
ID							
R/W-0h							
15	14	13	12	11	10	9	8
ID							
R/W-0h							
7	6	5	4	3	2	1	0
ID							
R/W-0h							

Table 28-33. CAN_IF1ARB Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MsgVal	R/W	0h	Message Valid 0 The mailbox is disabled. (The message object is ignored by the message handler). 1 The mailbox is enabled. (The message object is to be used by the message handler). The CPU should reset the MsgVal bit of all unused Messages Objects during the initialization before it resets the Init bit in the CAN Control Register. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
30	Xtd	R/W	0h	Extended Identifier 0 The 11-bit ("standard") Identifier is used for this message object. 1 The 29-bit ("extended") Identifier is used for this message object. Note: This bit is write protected by Busy bit. Reset type: SYSRSn

Table 28-33. CAN_IF1ARB Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
29	Dir	R/W	0h	Message Direction 0 Direction = receive: On TxRqst, a remote frame with the identifier of this message object is transmitted. On reception of a data frame with matching identifier, that frame is stored in this message object. 1 Direction = transmit: On TxRqst, the respective message object is transmitted as a data frame. On reception of a remote frame with matching identifier, the TxRqst bit of this message object is set (if RmtEn = one). Note: This bit is write protected by Busy bit. Reset type: SYSRSn
28-0	ID	R/W	0h	Message Identifier ID[28:0] 29-bit Identifier ("Extended Frame") ID[28:18] 11-bit Identifier ("Standard Frame") Note: This bit is write protected by Busy bit. Reset type: SYSRSn

28.16.2.25 CAN_IF1MCTL Register (Offset (x8) = 218h, Offset (x16) = 10Ch) [Reset = 0h]

CAN_IF1MCTL is shown in Figure 28-45 and described in Table 28-34.

Return to the Summary Table.

The bits of the IF1/IF2 Message Control Registers mirror the message control bits of a message object. This register has control/status bits pertaining to interrupts, acceptance mask, remote frames and FIFO option.

Figure 28-45. CAN_IF1MCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
NewDat	MsgLst	IntPnd	UMask	TxIE	RxIE	RmtEn	TxRqst
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
EoB	RESERVED			DLC			
R/W-0h	R-0h			R/W-0h			

Table 28-34. CAN_IF1MCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	NewDat	R/W	0h	New Data 0 No new data has been written into the data portion of this message object by the message handler since the last time when this flag was cleared by the CPU. 1 The message handler or the CPU has written new data into the data portion of this message object. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
14	MsgLst	R/W	0h	Message Lost (only valid for message objects with direction = receive) 0 No message lost since the last time when this bit was reset by the CPU. 1 The message handler stored a new message into this object when NewDat was still set, so the previous message has been overwritten. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
13	IntPnd	R/W	0h	Interrupt Pending 0 This message object is not the source of an interrupt. 1 This message object is the source of an interrupt. The Interrupt Identifier in the Interrupt Register will point to this message object if there is no other interrupt source with higher priority. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
12	UMask	R/W	0h	Use Acceptance Mask 0 Mask ignored 1 Use Mask (Msk[28:0], MXtd, and MDir) for acceptance filtering If the UMask bit is set to one, the message object's mask bits have to be programmed during initialization of the message object before MsgVal is set to one. Note: This bit is write protected by Busy bit. Reset type: SYSRSn

Table 28-34. CAN_IF1MCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	TxIE	R/W	0h	Transmit Interrupt Enable 0 IntPnd will not be triggered after the successful transmission of a frame. 1 IntPnd will be triggered after the successful transmission of a frame. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
10	RxIE	R/W	0h	Receive Interrupt Enable 0 IntPnd will not be triggered after the successful reception of a frame. 1 IntPnd will be triggered after the successful reception of a frame. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
9	RmtEn	R/W	0h	Remote Enable 0 At the reception of a remote frame, TxRqst is not changed. 1 At the reception of a remote frame, TxRqst is set. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
8	TxRqst	R/W	0h	Transmit Request 0 This message object is not waiting for a transmission. 1 The transmission of this message object is requested and is not yet done. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
7	EoB	R/W	0h	End of Block 0 The message object is part of a FIFO Buffer block and is not the last message object of the FIFO Buffer block. 1 The message object is a single message object or the last message object in a FIFO Buffer Block. Note: This bit is used to concatenate multiple message objects to build a FIFO Buffer. For single message objects (not belonging to a FIFO Buffer), this bit must always be set to one. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
6-4	RESERVED	R	0h	Reserved
3-0	DLC	R/W	0h	Data length code 0-8 Data frame has 0-8 data bytes. 9-15 Data frame has 8 data bytes. Note: The data length code of a message object must be defined the same as in all the corresponding objects with the same identifier at other nodes. When the message handler stores a data frame, it will write the DLC to the value given by the received message. Note: This bit is write protected by Busy bit. Reset type: SYSRSn

28.16.2.26 CAN_IF1DATA Register (Offset (x8) = 220h, Offset (x16) = 110h) [Reset = 0h]

CAN_IF1DATA is shown in Figure 28-46 and described in Table 28-35.

Return to the Summary Table.

This register provides a window to the data bytes of the CAN message. The data bytes of CAN messages are stored in the IF1/IF2 registers in the following order. In a CAN Data Frame, Data 0 is the first, and Data 7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first. All bits in this register are write-protected by the Busy bit.

Figure 28-46. CAN_IF1DATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data_3								Data_2								Data_1								Data_0							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 28-35. CAN_IF1DATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	Data_3	R/W	0h	Data Byte 3 Reset type: SYSRSn
23-16	Data_2	R/W	0h	Data Byte 2 Reset type: SYSRSn
15-8	Data_1	R/W	0h	Data Byte 1 Reset type: SYSRSn
7-0	Data_0	R/W	0h	Data Byte 0 Reset type: SYSRSn

28.16.2.27 CAN_IF1DATB Register (Offset (x8) = 228h, Offset (x16) = 114h) [Reset = 0h]

CAN_IF1DATB is shown in Figure 28-47 and described in Table 28-36.

Return to the Summary Table.

This register provides a window to the data bytes of the CAN message. The data bytes of CAN messages are stored in the IF1/IF2 registers in the following order. In a CAN Data Frame, Data 0 is the first, and Data 7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first. All bits in this register are write-protected by the Busy bit.

Figure 28-47. CAN_IF1DATB Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data_7								Data_6								Data_5								Data_4							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 28-36. CAN_IF1DATB Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	Data_7	R/W	0h	Data Byte 7 Reset type: SYSRSn
23-16	Data_6	R/W	0h	Data Byte 6 Reset type: SYSRSn
15-8	Data_5	R/W	0h	Data Byte 5 Reset type: SYSRSn
7-0	Data_4	R/W	0h	Data Byte 4 Reset type: SYSRSn

28.16.2.28 CAN_IF2CMD Register (Offset (x8) = 240h, Offset (x16) = 120h) [Reset = 1h]

CAN_IF2CMD is shown in Figure 28-48 and described in Table 28-37.

Return to the Summary Table.

The IF1/IF2 Command Registers configure and initiate the transfer between the IF1/IF2 Register sets and the Message RAM. It is configurable which portions of the message object should be transferred. A transfer is started when the CPU writes the message number to bits [7:0] of the IF1/IF2 Command Register. With this write operation, the Busy bit is automatically set to '1' to indicate that a transfer is in progress. After 4 to 14 clock cycles, the transfer between the Interface Register and the Message RAM will be completed and the Busy bit is cleared. The maximum number of cycles is needed when the message transfer coincides with a CAN message transmission, acceptance filtering, or message storage.
If the CPU writes to both IF1/IF2 Command Registers consecutively (request of a second transfer while first transfer is still in progress), the second transfer will start after the first one has been completed. The following points must be borne in mind while writing to this register: (1) Do not write zeros to the whole register. (2) Write to the register in a single 32-bit write or write the upper 16-bits before writing to the lower 16- bits.
Note: While Busy bit is one, IF1/IF2 Register sets are write protected.
Note: For debug support, the auto clear functionality of the IF1/IF2 Command Registers (clear of DMAactive flag by R/W, for devices with DMA support) is disabled during Debug/Suspend mode.
Note: If an invalid Message Number is written to bits [7:0] of the IF1/IF2 Command Register, the Message Handler may access an implemented (valid) message object instead.

Figure 28-48. CAN_IF2CMD Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
DIR	Mask	Arb	Control	ClrIntPnd	TxRqst	DATA_A	DATA_B
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
Busy	DMAactive	RESERVED					
R-0h	R/W-0h	R-0h					
7	6	5	4	3	2	1	0
MSG_NUM							
R/W-1h							

Table 28-37. CAN_IF2CMD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23	DIR	R/W	0h	Write/Read 0 Direction = Read: Transfer direction is from the message object addressed by Message Number (Bits [7:0]) to the IF1/IF2 Register set. That is, transfer data from the mailbox into the selected IF1/IF2 Message Buffer Registers. 1 Direction = Write: Transfer direction is from the IF1/IF2 Register set to the message object addressed by Message Number (Bits [7:0]) . That is, transfer data from the selected IF1/IF2 Message Buffer Registers to the mailbox. The other bits of IF1/IF2 Command Mask Register have different functions depending on the transfer direction. Note: This bit is write protected by Busy bit. Reset type: SYSRSn

Table 28-37. CAN_IF2CMD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
22	Mask	R/W	0h	Access Mask Bits 0 Mask bits will not be changed 1 (Direction = Read): The Mask bits (Identifier Mask + MDir + MXtd) will be transferred from the message object addressed by Message Number (Bits [7:0]) to the IF1/IF2 Register set. 1 (Direction = Write): The Mask bits (Identifier Mask + MDir + MXtd) will be transferred from the IF1/IF2 Register set to the message object addressed by Message Number (Bits [7:0]). Note: This bit is write protected by Busy bit. Reset type: SYSRSn
21	Arb	R/W	0h	Access Arbitration Bits 0 Arbitration bits will not be changed 1 (Direction = Read): The Arbitration bits (Identifier + Dir + Xtd + MsgVal) will be transferred from the message object addressed by Message Number (Bits [7:0]) to the corresponding IF1/IF2 Register set. 1 (Direction = Write): The Arbitration bits (Identifier + Dir + Xtd + MsgVal) will be transferred from the IF1/IF2 Register set to the message object addressed by Message Number (Bits [7:0]). Note: This bit is write protected by Busy bit. Reset type: SYSRSn
20	Control	R/W	0h	Access control bits. If the TxRqst/NewDat bit in this register(Bit [18]) is set, the TxRqst/NewDat bit in the IF1 message control register will be ignored. 0 Control bits will not be changed. 1 (Direction = Read): The message control bits will be transferred from the message object addressed by message number (Bits [7:0]) to the IF1 register set. 1 (Direction = Write): The message control bits will be transferred from the IF1 register set to the message object addressed by message number (Bits [7:0]). Note: This bit is write protected by the Busy bit. Reset type: SYSRSn
19	ClrIntPnd	R/W	0h	Clear Interrupt Pending Bit 0 IntPnd bit will not be changed 1 (Direction = Read): Clears IntPnd bit in the message object. 1 (Direction = Write): This bit is ignored. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
18	TxRqst	R/W	0h	Access Transmission Request (TxRqst) / New Data (NewDat) Bit 0 (Direction = Read): NewDat bit will not be changed. 0 (Direction = Write): TxRqst/NewDat bit will be handled according to the Control bit. 1 (Direction = Read): Clears NewDat bit in the message object. 1 (Direction = Write): Sets TxRqst/NewDat in message object. Note: If a CAN transmission is requested by setting TxRqst/NewDat in this register, the TxRqst/NewDat bits in the message object will be set to one independent of the values in IF1/IF2 Message Control Register. Note: A read access to a message object can be combined with the reset of the control bits IntPnd and NewDat. The values of these bits transferred to the IF1/IF2 Message Control Register always reflect the status before resetting them. Note: This bit is write protected by Busy bit. Reset type: SYSRSn

Table 28-37. CAN_IF2CMD Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	DATA_A	R/W	0h	Access Data Bytes 0-3 0 Data Bytes 0-3 will not be changed. 1 (Direction = Read): The Data Bytes 0-3 will be transferred from the message object addressed by the Message Number (Bits [7:0]) to the corresponding IF1/IF2 Register set. 1 (Direction = Write): The Data Bytes 0-3 will be transferred from the IF1/IF2 Register set to the message object addressed by the Message Number (Bits [7:0]). Note: The duration of the message transfer is independent of the number of bytes to be transferred. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
16	DATA_B	R/W	0h	Access Data Bytes 4-7 0 Data Bytes 4-7 will not be changed. 1 (Direction = Read): The Data Bytes 4-7 will be transferred from the message object addressed by Message Number (Bits [7:0]) to the corresponding IF1/IF2 Register set. 1 (Direction = Write): The Data Bytes 4-7 will be transferred from the IF1/IF2 Register set to the message object addressed by Message Number (Bits [7:0]). Note: The duration of the message transfer is independent of the number of bytes to be transferred. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
15	Busy	R	0h	Busy Flag 0 No transfer between IF1/IF2 Register Set and Message RAM is in progress. 1 Transfer between IF1/IF2 Register Set and Message RAM is in progress. This bit is set to one after the message number has been written to bits [7:0]. IF1/IF2 Register Set will be write protected. The bit is cleared after read/write action has been finished. Reset type: SYSRSn
14	DMAactive	R/W	0h	DMA trigger status due to IF1 update. 0 No IF1 DMA request is active. 1 DMA is requested after a completed transfer between IF1 and the message RAM. The DMA request remains active until the first read or write to one of the IF1 registers an exception is a write to Message Number (Bits [7:0]) when DMAactive is one. Note: Due to the auto reset feature of the DMAactive bit, this bit has to be set for each subsequent DMA cycle separately. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
13-8	RESERVED	R	0h	Reserved
7-0	MSG_NUM	R/W	1h	Number of message object in Message RAM which is used for data transfer 0x00 Invalid message number 0x01-0x20 Valid message numbers 0x21-0xFF Invalid message numbers Note: This bit is write protected by Busy bit. Reset type: SYSRSn

28.16.2.29 CAN_IF2MSK Register (Offset (x8) = 248h, Offset (x16) = 124h) [Reset = FFFFFFFFh]

CAN_IF2MSK is shown in Figure 28-49 and described in Table 28-38.

Return to the Summary Table.

The bits of the IF1/IF2 Mask Registers mirror the mask bits of a message object.
Note: While Busy bit of IF1/IF2 Command Register is one, IF1/IF2 Register Set is write-protected.

Figure 28-49. CAN_IF2MSK Register

31	30	29	28	27	26	25	24
MXtd	MDir	RESERVED	Msk				
R/W-1h	R/W-1h	R-0h	R/W-1FFFFFFFh				
23	22	21	20	19	18	17	16
Msk							
R/W-1FFFFFFFh							
15	14	13	12	11	10	9	8
Msk							
R/W-1FFFFFFFh							
7	6	5	4	3	2	1	0
Msk							
R/W-1FFFFFFFh							

Table 28-38. CAN_IF2MSK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MXtd	R/W	1h	Mask Extended Identifier 0 The extended identifier bit (Xtd) has no effect on the acceptance filtering. 1 The extended identifier bit (Xtd) is used for acceptance filtering. When 11-bit ("standard") identifiers are used for a message object, the identifiers of received data frames are written into bits ID[28:18]. For acceptance filtering, only these bits together with mask bits Msk[28:18] are considered. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
30	MDir	R/W	1h	Mask Message Direction 0 The message direction bit (Dir) has no effect on the acceptance filtering. 1 The message direction bit (Dir) is used for acceptance filtering. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
29	RESERVED	R	0h	Reserved
28-0	Msk	R/W	1FFFFFFFh	Identifier Mask 0 The corresponding bit in the identifier of the message object is not used for acceptance filtering (don't care). 1 The corresponding bit in the identifier of the message object is used for acceptance filtering. Note: This bit is write protected by Busy bit. Reset type: SYSRSn

28.16.2.30 CAN_IF2ARB Register (Offset (x8) = 250h, Offset (x16) = 128h) [Reset = 0h]

CAN_IF2ARB is shown in Figure 28-50 and described in Table 28-39.

Return to the Summary Table.

The bits of the IF1/IF2 Arbitration Registers mirror the arbitration bits of a message object. The Arbitration bits ID[28:0], Xtd, and Dir are used to define the identifier and type of outgoing messages and (together with the Mask bits Msk[28:0], MXtd, and MDir) for acceptance filtering of incoming messages.
A received message is stored into the valid message object with matching identifier and Direction = receive (Data Frame) or Direction = transmit (Remote Frame).
Extended frames can be stored only in message objects with Xtd = one, standard frames in message objects with Xtd = zero.
If a received message (Data Frame or Remote Frame) matches more than one valid message objects, it is stored into the one with the lowest message number.
Note: While Busy bit of IF1/IF2 Command Register is one, IF1/IF2 Register Set is write-protected.

Figure 28-50. CAN_IF2ARB Register

31	30	29	28	27	26	25	24
MsgVal	Xtd	Dir	ID				
R/W-0h	R/W-0h	R/W-0h	R/W-0h				
23	22	21	20	19	18	17	16
ID							
R/W-0h							
15	14	13	12	11	10	9	8
ID							
R/W-0h							
7	6	5	4	3	2	1	0
ID							
R/W-0h							

Table 28-39. CAN_IF2ARB Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MsgVal	R/W	0h	Message Valid 0 The mailbox is disabled. (The message object is ignored by the message handler). 1 The mailbox is enabled. (The message object is to be used by the message handler). The CPU should reset the MsgVal bit of all unused Messages Objects during the initialization before it resets the Init bit in the CAN Control Register. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
30	Xtd	R/W	0h	Extended Identifier 0 The 11-bit ("standard") Identifier is used for this message object. 1 The 29-bit ("extended") Identifier is used for this message object. Note: This bit is write protected by Busy bit. Reset type: SYSRSn

Table 28-39. CAN_IF2ARB Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
29	Dir	R/W	0h	Message Direction 0 Direction = receive: On TxRqst, a remote frame with the identifier of this message object is transmitted. On reception of a data frame with matching identifier, that frame is stored in this message object. 1 Direction = transmit: On TxRqst, the respective message object is transmitted as a data frame. On reception of a remote frame with matching identifier, the TxRqst bit of this message object is set (if RmtEn = one). Note: This bit is write protected by Busy bit. Reset type: SYSRSn
28-0	ID	R/W	0h	Message Identifier ID[28:0] 29-bit Identifier ("Extended Frame") ID[28:18] 11-bit Identifier ("Standard Frame") Note: This bit is write protected by Busy bit. Reset type: SYSRSn

28.16.2.31 CAN_IF2MCTL Register (Offset (x8) = 258h, Offset (x16) = 12Ch) [Reset = 0h]

CAN_IF2MCTL is shown in Figure 28-51 and described in Table 28-40.

Return to the Summary Table.

The bits of the IF1/IF2 Message Control Registers mirror the message control bits of a message object. This register has control/status bits pertaining to interrupts, acceptance mask, remote frames and FIFO option.

Figure 28-51. CAN_IF2MCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
NewDat	MsgLst	IntPnd	UMask	TxIE	RxIE	RmtEn	TxRqst
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
EoB	RESERVED			DLC			
R/W-0h	R-0h			R/W-0h			

Table 28-40. CAN_IF2MCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	NewDat	R/W	0h	New Data 0 No new data has been written into the data portion of this message object by the message handler since the last time when this flag was cleared by the CPU. 1 The message handler or the CPU has written new data into the data portion of this message object. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
14	MsgLst	R/W	0h	Message Lost (only valid for message objects with direction = receive) 0 No message lost since the last time when this bit was reset by the CPU. 1 The message handler stored a new message into this object when NewDat was still set, so the previous message has been overwritten. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
13	IntPnd	R/W	0h	Interrupt Pending 0 This message object is not the source of an interrupt. 1 This message object is the source of an interrupt. The Interrupt Identifier in the Interrupt Register will point to this message object if there is no other interrupt source with higher priority. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
12	UMask	R/W	0h	Use Acceptance Mask 0 Mask ignored 1 Use Mask (Msk[28:0], MXtd, and MDir) for acceptance filtering If the UMask bit is set to one, the message object's mask bits have to be programmed during initialization of the message object before MsgVal is set to one. Note: This bit is write protected by Busy bit. Reset type: SYSRSn

Table 28-40. CAN_IF2MCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	TxIE	R/W	0h	Transmit Interrupt Enable 0 IntPnd will not be triggered after the successful transmission of a frame. 1 IntPnd will be triggered after the successful transmission of a frame. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
10	RxIE	R/W	0h	Receive Interrupt Enable 0 IntPnd will not be triggered after the successful reception of a frame. 1 IntPnd will be triggered after the successful reception of a frame. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
9	RmtEn	R/W	0h	Remote Enable 0 At the reception of a remote frame, TxRqst is not changed. 1 At the reception of a remote frame, TxRqst is set. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
8	TxRqst	R/W	0h	Transmit Request 0 This message object is not waiting for a transmission. 1 The transmission of this message object is requested and is not yet done. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
7	EoB	R/W	0h	End of Block 0 The message object is part of a FIFO Buffer block and is not the last message object of the FIFO Buffer block. 1 The message object is a single message object or the last message object in a FIFO Buffer Block. Note: This bit is used to concatenate multiple message objects to build a FIFO Buffer. For single message objects (not belonging to a FIFO Buffer), this bit must always be set to one. Note: This bit is write protected by Busy bit. Reset type: SYSRSn
6-4	RESERVED	R	0h	Reserved
3-0	DLC	R/W	0h	Data length code 0-8 Data frame has 0-8 data bytes. 9-15 Data frame has 8 data bytes. Note: The data length code of a message object must be defined the same as in all the corresponding objects with the same identifier at other nodes. When the message handler stores a data frame, it will write the DLC to the value given by the received message. Note: This bit is write protected by Busy bit. Reset type: SYSRSn

28.16.2.32 CAN_IF2DATA Register (Offset (x8) = 260h, Offset (x16) = 130h) [Reset = 0h]

CAN_IF2DATA is shown in Figure 28-52 and described in Table 28-41.

Return to the Summary Table.

This register provides a window to the data bytes of the CAN message. The data bytes of CAN messages are stored in the IF1/IF2 registers in the following order. In a CAN Data Frame, Data 0 is the first, and Data 7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first. All bits in this register are write-protected by the Busy bit.

Figure 28-52. CAN_IF2DATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data_3								Data_2								Data_1								Data_0							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 28-41. CAN_IF2DATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	Data_3	R/W	0h	Data Byte 3 Reset type: SYSRSn
23-16	Data_2	R/W	0h	Data Byte 2 Reset type: SYSRSn
15-8	Data_1	R/W	0h	Data Byte 1 Reset type: SYSRSn
7-0	Data_0	R/W	0h	Data Byte 0 Reset type: SYSRSn

28.16.2.33 CAN_IF2DATB Register (Offset (x8) = 268h, Offset (x16) = 134h) [Reset = 0h]

CAN_IF2DATB is shown in Figure 28-53 and described in Table 28-42.

Return to the Summary Table.

This register provides a window to the data bytes of the CAN message. The data bytes of CAN messages are stored in the IF1/IF2 registers in the following order. In a CAN Data Frame, Data 0 is the first, and Data 7 is the last byte to be transmitted or received. In CAN's serial bit stream, the MSB of each byte will be transmitted first. All bits in this register are write-protected by the Busy bit.

Figure 28-53. CAN_IF2DATB Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data_7								Data_6								Data_5								Data_4							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 28-42. CAN_IF2DATB Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	Data_7	R/W	0h	Data Byte 7 Reset type: SYSRSn
23-16	Data_6	R/W	0h	Data Byte 6 Reset type: SYSRSn
15-8	Data_5	R/W	0h	Data Byte 5 Reset type: SYSRSn
7-0	Data_4	R/W	0h	Data Byte 4 Reset type: SYSRSn

28.16.2.34 CAN_IF3OBS Register (Offset (x8) = 280h, Offset (x16) = 140h) [Reset = 0h]

CAN_IF3OBS is shown in Figure 28-54 and described in Table 28-43.

Return to the Summary Table.

The IF3 register set can automatically be updated with received message objects without the need to initiate the transfer from Message RAM by CPU.
The observation flags (Bits [4:0]) in the IF3 Observation register are used to determine, which data sections of the IF3 Interface Register set have to be read in order to complete a DMA read cycle. After all marked data sections are read, the DCAN is enabled to update the IF3 Interface Register set with new data.
Any access order of single bytes or half-words is supported. When using byte or half-word accesses, a data section is marked as completed, if all bytes are read.
Note: If IF3 Update Enable is used and no Observation flag is set, the corresponding message objects will be copied to IF3 without activating the DMA request line and without waiting for DMA read accesses.
A write access to this register aborts a pending DMA cycle by resetting the DMA line and enables updating of IF3 Interface Register set with new data. To avoid data inconsistency, the DMA controller should be disabled before reconfiguring IF3 observation register. The status of the current read-cycle can be observed via status flags (Bits [12:8]).
With this, the observation status bits and the IF3Upd bit could be used by the application to realize the notification about new IF3 content in polling or interrupt mode

Figure 28-54. CAN_IF3OBS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
IF3Upd	RESERVED		IF3SDB	IF3SDA	IF3SC	IF3SA	IF3SM
R-0h	R-0h		R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED			Data_B	Data_A	Ctrl	Arb	Mask
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 28-43. CAN_IF3OBS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	IF3Upd	R	0h	IF3 Update Data 0 No new data has been loaded since last IF3 read. 1 New data has been loaded since last IF3 read. Reset type: SYSRSn
14-13	RESERVED	R	0h	Reserved
12	IF3SDB	R	0h	IF3 Status of Data B read access 0 All Data B bytes are already read out, or are not marked to be read. 1 Data B section has still data to be read out. Reset type: SYSRSn
11	IF3SDA	R	0h	IF3 Status of Data A read access 0 All Data A bytes are already read out, or are not marked to be read. 1 Data A section has still data to be read out. Reset type: SYSRSn

Table 28-43. CAN_IF3OBS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	IF3SC	R	0h	IF3 Status of Control bits read access 0 All Control section bytes are already read out, or are not marked to be read. 1 Control section has still data to be read out. Reset type: SYSRSn
9	IF3SA	R	0h	IF3 Status of Arbitration data read access 0 All Arbitration data bytes are already read out, or are not marked to be read. 1 Arbitration section has still data to be read out. Reset type: SYSRSn
8	IF3SM	R	0h	IF3 Status of Mask data read access 0 All Mask data bytes are already read out, or are not marked to be read. 1 Mask section has still data to be read out. Reset type: SYSRSn
7-5	RESERVED	R	0h	Reserved
4	Data_B	R/W	0h	Data B read observation 0 Data B section not to be read. 1 Data B section has to be read to enable next IF3 update. Reset type: SYSRSn
3	Data_A	R/W	0h	Data A read observation 0 Data A section not to be read. 1 Data A section has to be read to enable next IF3 update. Reset type: SYSRSn
2	Ctrl	R/W	0h	Ctrl read observation 0 Ctrl section not to be read. 1 Ctrl section has to be read to enable next IF3 update. Reset type: SYSRSn
1	Arb	R/W	0h	Arbitration data read observation 0 Arbitration data not to be read. 1 Arbitration data has to be read to enable next IF3 update. Reset type: SYSRSn
0	Mask	R/W	0h	Mask data read observation 0 Mask data not to be read. 1 Mask data has to be read to enable next IF3 update. Reset type: SYSRSn

28.16.2.35 CAN_IF3MSK Register (Offset (x8) = 288h, Offset (x16) = 144h) [Reset = FFFFFFFFh]

CAN_IF3MSK is shown in Figure 28-55 and described in Table 28-44.

Return to the Summary Table.

This register provides a window to the acceptance mask for the chosen mailbox.

Figure 28-55. CAN_IF3MSK Register

31	30	29	28	27	26	25	24
MXtd	MDir	RESERVED	Msk				
R-1h	R-1h	R-0h	R-1FFFFFFFh				
23	22	21	20	19	18	17	16
Msk							
R-1FFFFFFFh							
15	14	13	12	11	10	9	8
Msk							
R-1FFFFFFFh							
7	6	5	4	3	2	1	0
Msk							
R-1FFFFFFFh							

Table 28-44. CAN_IF3MSK Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MXtd	R	1h	Mask Extended Identifier 0 The extended identifier bit (Xtd) has no effect on the acceptance filtering. 1 The extended identifier bit (Xtd) is used for acceptance filtering. Note: When 11-bit ("standard") identifiers are used for a message object, the identifiers of received data frames are written into bits ID[28:18]. For acceptance filtering, only these bits together with mask bits Msk[28:18] are considered. Reset type: SYSRSn
30	MDir	R	1h	Mask Message Direction 0 The message direction bit (Dir) has no effect on the acceptance filtering. 1 The message direction bit (Dir) is used for acceptance filtering. Reset type: SYSRSn
29	RESERVED	R	0h	Reserved
28-0	Msk	R	1FFFFFFFh	Identifier Mask Identifier Mask 0 The corresponding bit in the identifier of the message object is not used for acceptance filtering (don't care). 1 The corresponding bit in the identifier of the message object is used for acceptance filtering. Identifier Mask Reset type: SYSRSn

28.16.2.36 CAN_IF3ARB Register (Offset (x8) = 290h, Offset (x16) = 148h) [Reset = 0h]

CAN_IF3ARB is shown in Figure 28-56 and described in Table 28-45.

Return to the Summary Table.

The bits of the IF3 Arbitration Register mirrors the arbitration bits of a message object.

Figure 28-56. CAN_IF3ARB Register

31	30	29	28	27	26	25	24
MsgVal	Xtd	Dir	ID				
R-0h	R-0h	R-0h	R-0h				
23	22	21	20	19	18	17	16
ID							
R-0h							
15	14	13	12	11	10	9	8
ID							
R-0h							
7	6	5	4	3	2	1	0
ID							
R-0h							

Table 28-45. CAN_IF3ARB Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MsgVal	R	0h	Message Valid 0 The message object is ignored by the message handler. 1 The message object is to be used by the message handler. The CPU should reset the MsgVal bit of all unused Messages Objects during the initialization before it resets bit Init in the CAN Control Register. Reset type: SYSRSn
30	Xtd	R	0h	Extended Identifier 0 The 11-bit ("standard") Identifier is used for this message object. 1 The 29-bit ("extended") Identifier is used for this message object. Reset type: SYSRSn
29	Dir	R	0h	Message Direction 0 Direction = receive: On TxRqst, a remote frame with the identifier of this message object is transmitted. On reception of a data frame with matching identifier, that message is stored in this message object. 1 Direction = transmit: On TxRqst, the respective message object is transmitted as a data frame. On reception of a remote frame with matching identifier, the TxRqst bit of this message object is set (if RmtEn = one). Reset type: SYSRSn
28-0	ID	R	0h	Message Identifier ID[28:0] 29-bit Identifier ("Extended Frame") ID[28:18] 11-bit Identifier ("Standard Frame") Reset type: SYSRSn

28.16.2.37 CAN_IF3MCTL Register (Offset (x8) = 298h, Offset (x16) = 14Ch) [Reset = 0h]

CAN_IF3MCTL is shown in Figure 28-57 and described in Table 28-46.

Return to the Summary Table.

The bits of the IF3 Message Control Register mirrors the message control bits of a message object.

Figure 28-57. CAN_IF3MCTL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
NewDat	MsgLst	IntPnd	UMask	TxIE	RxIE	RmtEn	TxRqst
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
EoB	RESERVED			DLC			
R-0h	R-0h			R-0h			

Table 28-46. CAN_IF3MCTL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	NewDat	R	0h	New Data 0 No new data has been written into the data portion of this message object by the message handler since the last time when this flag was cleared by the CPU. 1 The message handler or the CPU has written new data into the data portion of this message object. Reset type: SYSRSn
14	MsgLst	R	0h	Message Lost (only valid for message objects with direction = receive) 0 No message lost since the last time when this bit was reset by the CPU. 1 The message handler stored a new message into this object when NewDat was still set, so the previous message has been overwritten. Reset type: SYSRSn
13	IntPnd	R	0h	Interrupt Pending 0 This message object is not the source of an interrupt. 1 This message object is the source of an interrupt. The Interrupt Identifier in the Interrupt Register will point to this message object if there is no other interrupt source with higher priority. Reset type: SYSRSn
12	UMask	R	0h	Use Acceptance Mask 0 Mask ignored 1 Use Mask (Msk[28:0], MXtd, and MDir) for acceptance filtering If the UMask bit is set to one, the message object's mask bits have to be programmed during initialization of the message object before MsgVal is set to one. Reset type: SYSRSn
11	TxIE	R	0h	Transmit Interrupt Enable 0 IntPnd will not be triggered after the successful transmission of a frame. 1 IntPnd will be triggered after the successful transmission of a frame. Reset type: SYSRSn

Table 28-46. CAN_IF3MCTL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	RxIE	R	0h	Receive Interrupt Enable 0 IntPnd will not be triggered after the successful reception of a frame. 1 IntPnd will be triggered after the successful reception of a frame. Reset type: SYSRSn
9	RmtEn	R	0h	Remote Enable 0 At the reception of a remote frame, TxRqst is not changed. 1 At the reception of a remote frame, TxRqst is set. Reset type: SYSRSn
8	TxRqst	R	0h	Transmit Request 0 This message object is not waiting for a transmission. 1 The transmission of this message object is requested and is not yet done. Reset type: SYSRSn
7	EoB	R	0h	End of Block 0 The message object is part of a FIFO Buffer block and is not the last message object of the FIFO Buffer block. 1 The message object is a single message object or the last message object in a FIFO Buffer Block. Note: This bit is used to concatenate multiple message objects to build a FIFO Buffer. For single message objects (not belonging to a FIFO Buffer), this bit must always be set to one. Reset type: SYSRSn
6-4	RESERVED	R	0h	Reserved
3-0	DLC	R	0h	Data length code 0-8 Data frame has 0-8 data bytes. 9-15 Data frame has 8 data bytes. Note: The data length code of a message object must be defined the same as in all the corresponding objects with the same identifier at other nodes. When the message handler stores a data frame, it will write the DLC to the value given by the received message. Reset type: SYSRSn

28.16.2.38 CAN_IF3DATA Register (Offset (x8) = 2A0h, Offset (x16) = 150h) [Reset = 0h]

CAN_IF3DATA is shown in Figure 28-58 and described in Table 28-47.

Return to the Summary Table.

This register provides a window to the data bytes of the CAN message.

Figure 28-58. CAN_IF3DATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data_3								Data_2								Data_1								Data_0							
R-0h								R-0h								R-0h								R-0h							

Table 28-47. CAN_IF3DATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	Data_3	R	0h	Data Byte 3 Reset type: SYSRSn
23-16	Data_2	R	0h	Data Byte 2 Reset type: SYSRSn
15-8	Data_1	R	0h	Data Byte 1 Reset type: SYSRSn
7-0	Data_0	R	0h	Data Byte 0 Reset type: SYSRSn

28.16.2.39 CAN_IF3DATB Register (Offset (x8) = 2A8h, Offset (x16) = 154h) [Reset = 0h]

CAN_IF3DATB is shown in Figure 28-59 and described in Table 28-48.

Return to the Summary Table.

This register provides a window to the data bytes of the CAN message.

Figure 28-59. CAN_IF3DATB Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Data_7								Data_6								Data_5								Data_4							
R-0h								R-0h								R-0h								R-0h							

Table 28-48. CAN_IF3DATB Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	Data_7	R	0h	Data Byte 7 Reset type: SYSRSn
23-16	Data_6	R	0h	Data Byte 6 Reset type: SYSRSn
15-8	Data_5	R	0h	Data Byte 5 Reset type: SYSRSn
7-0	Data_4	R	0h	Data Byte 4 Reset type: SYSRSn

28.16.2.40 CAN_IF3UPD Register (Offset (x8) = 2C0h, Offset (x16) = 160h) [Reset = 0h]

CAN_IF3UPD is shown in Figure 28-60 and described in Table 28-49.

Return to the Summary Table.

The automatic update functionality of the IF3 register set can be configured for each message object. A message object is enabled for automatic IF3 update, if the dedicated IF3UpdEn flag is set. This means that an active NewDat flag of this message object (e.g due to reception of a CAN frame) will trigger an automatic copy of the whole message object to IF3 register set. Note: IF3 Update enable should not be set for transmit objects.

Figure 28-60. CAN_IF3UPD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IF3UpdEn																															
R/W-0h																															

Table 28-49. CAN_IF3UPD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	IF3UpdEn	R/W	0h	IF3 Update Enabled (for all message objects) 0 Automatic IF3 update is disabled for this message object. 1 Automatic IF3 update is enabled for this message object. A message object is scheduled to be copied to IF3 register set, if NewDat flag of the message object is active. Reset type: SYSRSn

28.16.3 CAN Registers to Driverlib Functions

Table 28-50. CAN Registers to Driverlib Functions

File	Driverlib Function
CAN_CTL	
can.c	CAN_initModule
can.c	CAN_setBitTiming
can.h	CAN_startModule
can.h	CAN_enableController
can.h	CAN_disableController
can.h	CAN_enableTestMode
can.h	CAN_disableTestMode
can.h	CAN_setInterruptionDebugMode
can.h	CAN_enableDMARequests
can.h	CAN_disableDMARequests
can.h	CAN_disableAutoBusOn
can.h	CAN_enableAutoBusOn
can.h	CAN_enableInterrupt
can.h	CAN_disableInterrupt
can.h	CAN_enableRetry
can.h	CAN_disableRetry
can.h	CAN_isRetryEnabled
CAN_ES	
can.c	CAN_clearInterruptStatus
can.h	CAN_getStatus
CAN_ERRC	
can.h	CAN_getErrorCount

Table 28-50. CAN Registers to Driverlib Functions (continued)

File	Driverlib Function
CAN_BTR	
can.c	CAN_setBitTiming
can.h	CAN_getBitTiming
CAN_INT	
can.h	CAN_getInterruptCause
CAN_TEST	
can.h	CAN_enableTestMode
can.h	CAN_disableTestMode
can.h	CAN_enableMemoryAccessMode
can.h	CAN_disableMemoryAccessMode
CAN_PERR	
-	
CAN_RAM_INIT	
can.h	CAN_initRAM
CAN_GLB_INT_EN	
can.h	CAN_enableGlobalInterrupt
can.h	CAN_disableGlobalInterrupt
CAN_GLB_INT_FLG	
can.h	CAN_getGlobalInterruptStatus
CAN_GLB_INT_CLR	
can.h	CAN_clearGlobalInterruptStatus
CAN_ABOTR	
can.h	CAN_setAutoBusOnTime
CAN_TXRQ_X	
-	
CAN_TXRQ_21	
can.h	CAN_getTxRequests
CAN_NDAT_X	
-	
CAN_NDAT_21	
can.h	CAN_getNewDataFlags
CAN_IPEN_X	
-	
CAN_IPEN_21	
can.h	CAN_getInterruptMessageSource
CAN_MVAL_X	
-	
CAN_MVAL_21	
can.h	CAN_getValidMessageObjects
CAN_IP_MUX21	
can.h	CAN_getInterruptMux
can.h	CAN_setInterruptMux
CAN_IF1CMD	
can.c	CAN_clearInterruptStatus
can.c	CAN_setupMessageObject

Table 28-50. CAN Registers to Driverlib Functions (continued)

File	Driverlib Function
can.c	CAN_sendMessage
can.c	CAN_sendRemoteRequestMessage
can.c	CAN_transferMessage
can.c	CAN_clearMessage
can.c	CAN_disableMessageObject
can.c	CAN_disableAllMessageObjects
CAN_IF1MSK	
can.c	CAN_setupMessageObject
CAN_IF1ARB	
can.c	CAN_setupMessageObject
can.c	CAN_clearMessage
can.c	CAN_disableMessageObject
can.c	CAN_disableAllMessageObjects
CAN_IF1MCTL	
can.c	CAN_setupMessageObject
can.c	CAN_sendMessage
can.c	CAN_sendRemoteRequestMessage
CAN_IF1DATA	
can.c	CAN_sendMessage
CAN_IF1DATB	
-	See IF1DATA
CAN_IF2CMD	
can.c	CAN_readMessage
can.c	CAN_transferMessage
CAN_IF2MSK	
-	
CAN_IF2ARB	
can.c	CAN_readMessageWithID
CAN_IF2MCTL	
can.c	CAN_readMessage
CAN_IF2DATA	
can.c	CAN_readMessage
CAN_IF2DATB	
-	See IF2DATA
CAN_IF3OBS	
-	
CAN_IF3MSK	
-	
CAN_IF3ARB	
-	
CAN_IF3MCTL	
-	
CAN_IF3DATA	
-	
CAN_IF3DATB	

Table 28-50. CAN Registers to Driverlib Functions (continued)

File	Driverlib Function
-	See IF3DATA
CAN_IF3UPD	
-	

This page intentionally left blank.

Chapter 29

Modular Controller Area Network (MCAN)

This chapter describes the Modular Controller Area Network (MCAN). MCAN supports both classic CAN and CAN FD protocols.

29.1 MCAN Overview

The Controller Area Network (CAN) is a serial communications protocol that efficiently supports distributed real-time control with a high level of reliability. CAN has high immunity to electrical interference and the ability to detect various type of errors. In CAN, many short messages are broadcast to the entire network, which provides data consistency in every node of the system.

The MCAN module supports both classic CAN and CAN FD (CAN with flexible data-rate) protocols. The CAN FD feature allows higher throughput and increased payload per data frame. Classic CAN and CAN FD devices may coexist on the same network without any conflict provided that partial network transceivers, which can detect and ignore CAN FD without generating bus errors, are used by the classic CAN devices. The MCAN module is compliant to ISO 11898-1:2015.

Note

The availability of the CAN FD feature is dependent on the device's part number. Refer to the device data manual for more information.

Figure 29-1 shows an overview of the MCAN module.

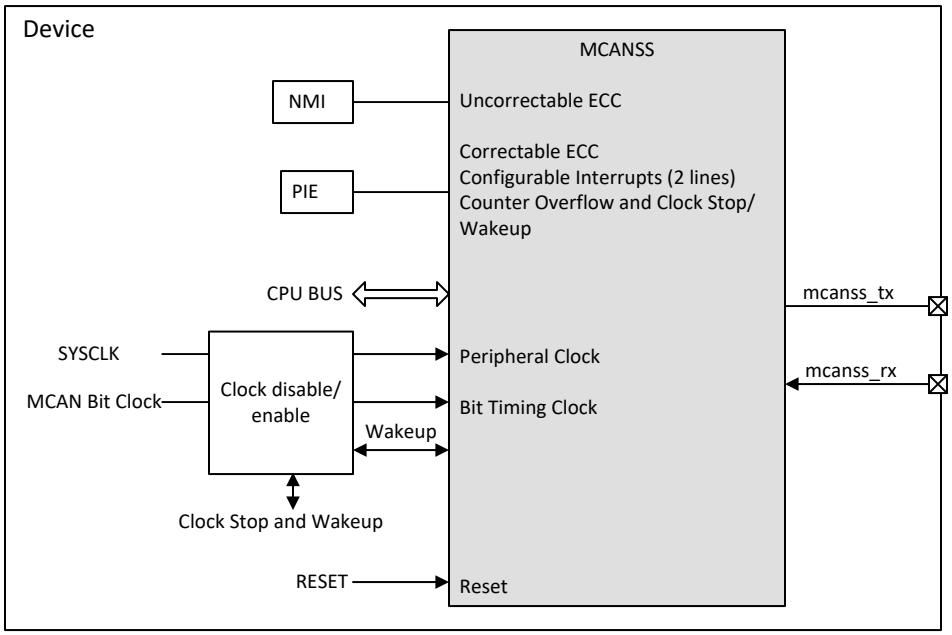


Figure 29-1. MCAN Module Overview

29.1.1 MCAN Related Collateral

Foundational Materials

- C2000 Academy - Communications
- CAN and CAN FD Overview (Video)
- CAN and CAN FD Protocol (Video)

29.1.2 MCAN Features

The MCAN module implements the following features:

- Conforms with CAN Protocol 2.0 A, B and ISO 11898-1:2015
- Full CAN FD support (up to 64 data bytes)
- AUTOSAR and SAE J1939 support
- Up to 32 dedicated transmit buffers
- Configurable transmit FIFO, up to 32 elements
- Configurable transmit queue, up to 32 elements
- Configurable transmit Event FIFO, up to 32 elements
- Up to 64 dedicated receive buffers
- Two configurable receive FIFOs, up to 64 elements each
- Up to 128 filter elements
- Loop-back mode for self-test
- Maskable interrupt (two configurable interrupt lines, correctable ECC, counter overflow and clock stop/ wakeup)
- Non-maskable interrupt (uncorrectable ECC)
- Two clock domains (CAN clock/host clock)
- ECC check for Message RAM
- Clock stop and wakeup support
- Timestamp counter

Non-supported features:

- Host bus firewall
- Clock calibration
- Debug over CAN

29.2 MCAN Environment

The CAN network physical layer consists of a two-wire differential bus, usually twisted pair, and provides a high level of interference immunity. An external CAN transceiver IC is needed to access the bus.

Figure 29-2 shows typical MCAN wiring.

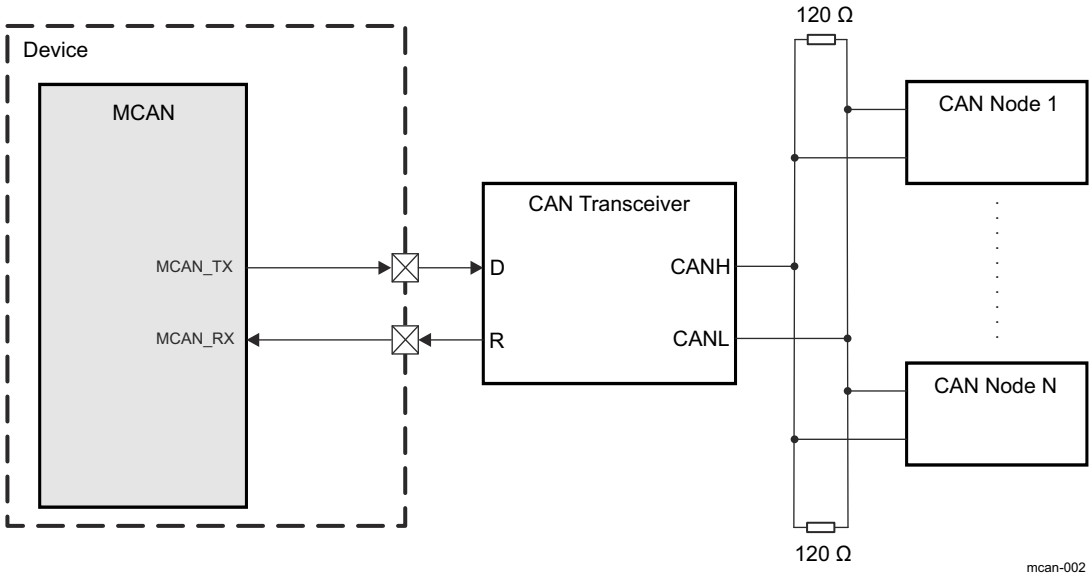


Figure 29-2. MCAN Typical Bus Wiring

Table 29-1 describes the external signals of the MCAN module.

Table 29-1. MCAN I/O Description

Module Signal	I/O	Description	Value at Reset
MCAN_RX	Input	Serial data input from external CAN transceiver.	HiZ
MCAN_TX	Output	Serial data output to external CAN transceiver.	1

Note

See the *Terminal Configurations and Functions* section in the device data manual and the *General-Purpose Input/Output (GPIO)* chapter in this TRM to configure this peripheral to be connected to the device pins.

29.3 CAN Network Basics

The network basics are:

- The CAN bus is a 2-wire differential bus using non-return-to-zero (NRZ) encoding and has two states:
  - Recessive state (logical 1)
  - Dominant state (logical 0)
- When the bus is idle, any node can initiate a transmission to any other node(s). When two or more nodes (ECUs) attempt to transmit at the same time, a non-destructive arbitration technique guarantees messages are sent in order of priority and no messages are lost.
- The message transmission is multicast. Data messages transmitted are identifier-based, not address-based.
- The content of the message is labeled by the identifier that is unique throughout the network (for example: RPM, temperature, position, pressure, and so forth).
- All nodes on the network receive the message and each performs an acceptance test on the identifier. If the message is relevant, it is processed; otherwise, it is ignored.
- The unique identifier also determines the priority of the message (the lower the numerical value of the identifier, the higher the priority).
- Data is transmitted and received using message frames, consisting of the following basic fields:
  - Arbitration field
  - Control field
  - Data field (up to 8 bytes for classical CAN and up to 64 bytes for CAN FD)
  - CRC field
  - ACK field

For more information, see *ISO 11898-1:2015: CAN data link layer and physical signaling*.

29.4 MCAN Integration

Figure 29-3 shows the integration of the MCAN module in the device.

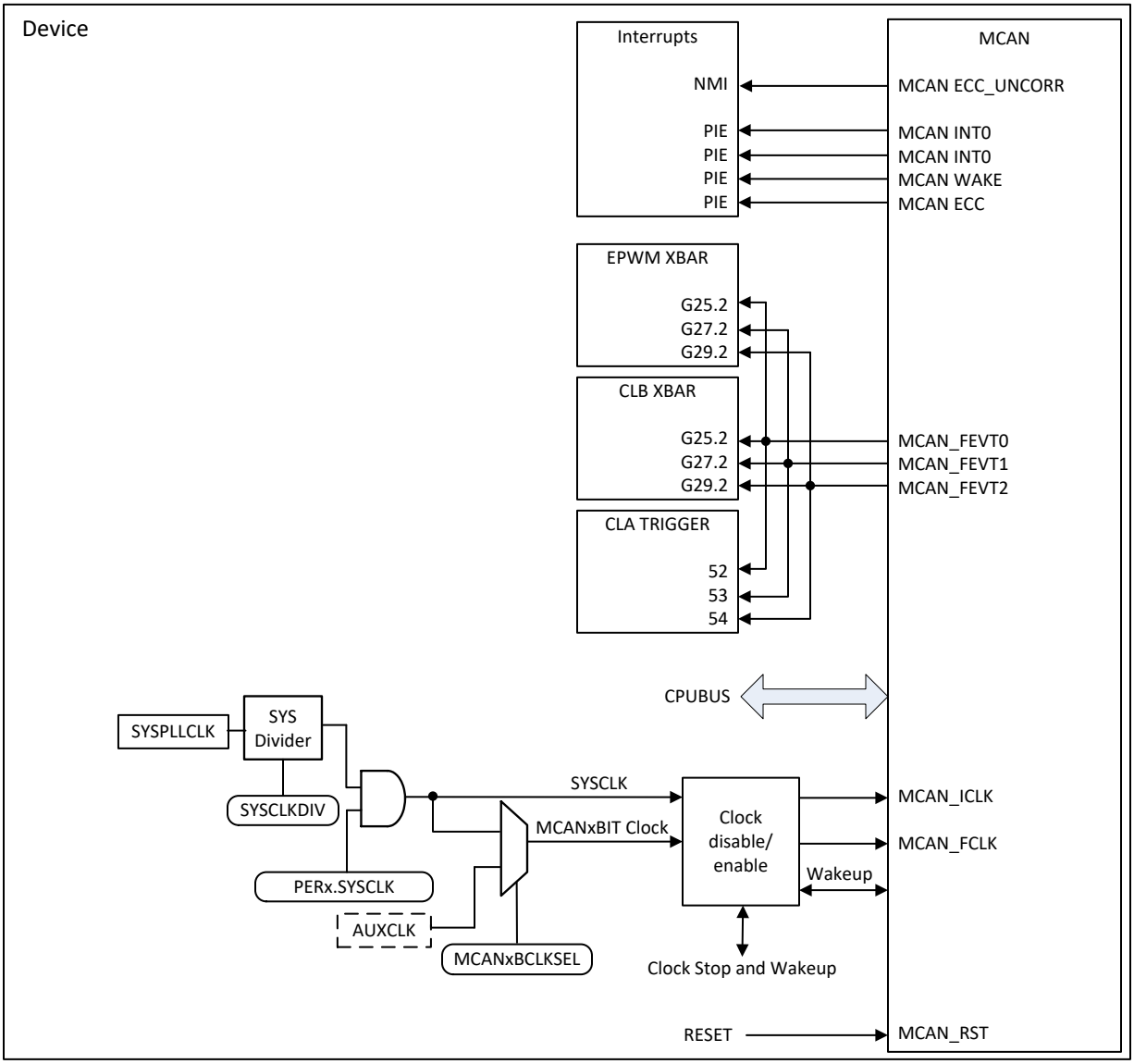


Figure 29-3. MCAN Integration

Table 29-2 and Table 29-3 summarize the integration of the MCAN module in the device.

Table 29-2. MCAN Clocks and Resets

Destination Signal Name	Source Signal Name	Description
Clocks		
MCAN_ICLK	SYSCLK	Interface clock for the MCAN module
MCAN_FCLK	MCANxBIT Clock	Bit timing clock for MCAN
Resets		
MCAN_RST	RESET	Asynchronous reset signal to the MCAN module

Table 29-3. MCAN Hardware Requests

Interrupt Requests				
Source Signal Name	**Interrupt Vector**	**28x PIE Input**	**Default Mapping**	**Description**
MCAN_IRQ_INT0	INT9.9	INT9.9	Configurable	MCAN interrupt 0
MCAN_IRQ_INT1	INT9.10	INT9.10	Configurable	MCAN interrupt 1
MCAN_IRQ_TS_WAKE	INT9.12	INT9.12	Configurable	MCAN timestamp and wakeup interrupt
MCAN_IRQ_ECC	INT9.11	INT9.11	Configurable	MCAN ECC interrupt
MCAN_IRQ_ECC_UNCORR	NMI	NMI	-2 Priority	MCAN ECC uncorrectable interrupt
Filter Event Connections				
Source Signal Name	**Trigger Input**		**Default Mapping**	**Description**
MCAN_FEVT0	EPWM XBAR CLB XBAR CLA Trigger		G25.2 G25.2 52	MCAN RX Filter Event 1
MCAN_FEVT1	EPWM XBAR CLB XBAR CLA Trigger		G27.2 G27.2 53	MCAN RX Filter Event 2
MCAN_FEVT2	EPWM XBAR CLB XBAR CLA Trigger		G29.2 G29.2 54	MCAN RX Filter Event 3

Note

For more information about the CLA_triggers, see the *Control Law Accelerator (CLA)* chapter.

For more information about the CLB XBAR, see the *Configurable Logic Block (CLB)* chapter.

For more information about the ePWM XBAR module, see the *Enhanced Pulse Width Modulator (ePWM)* chapter.

29.5 MCAN Functional Description

The MCAN module performs CAN protocol communication according to ISO 11898-1:2015. The bit rate can be programmed to values up to 5 Mbps. Additional transceiver hardware is required for the connection to the physical layer (CAN bus).

For communication on a CAN network, individual message frames can be configured. The message frames and identifier masks are stored in the Message RAM.

All functions concerning the handling of messages are implemented in the Message Handler.

The register set of the MCAN module can be accessed directly via the module interface. These registers are used to control and configure the CAN core and the Message Handler, and to access the Message RAM.

Figure 29-4 shows the MCAN module block diagram, followed by the description of the MCAN module blocks.

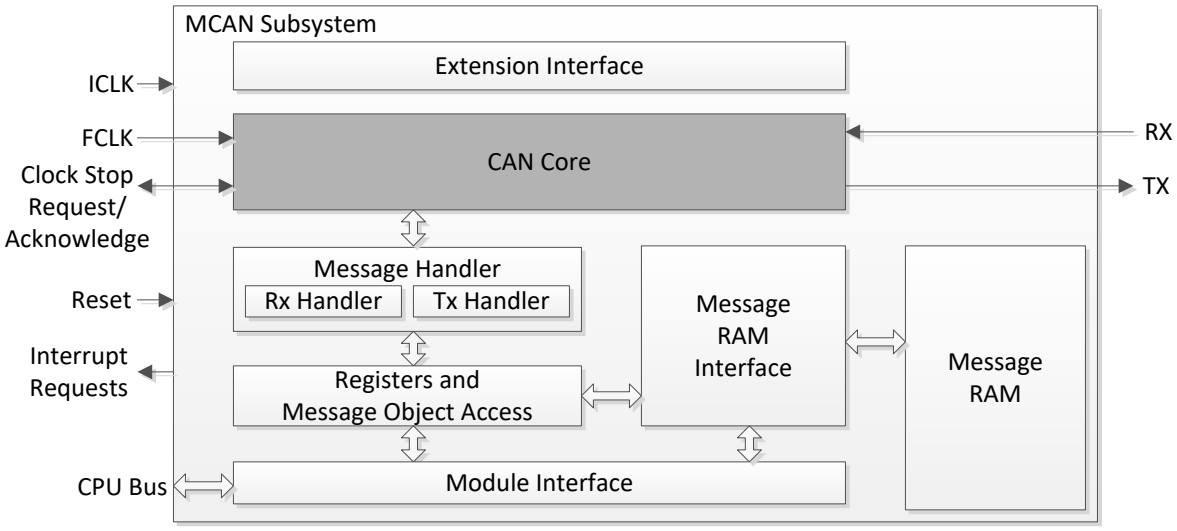


Figure 29-4. MCAN Block Diagram

- **CAN Core**: The CAN core consists of the CAN protocol controller and the Rx/Tx shift register. It handles all ISO 11898-1:2015 protocol functions and supports 11-bit and 29-bit identifiers.
- **Message Handler**: the Message Handler (Rx Handler and Tx Handler) is a state machine that controls the data transfer between the single-ported Message RAM and the CAN core's Rx/Tx shift register. It also handles the acceptance filtering and Interrupt generation as programmed in the control registers.
- **Message RAM**: the main purpose of the Message RAM is to store Rx/Tx messages, Tx Event elements, and Message ID Filter elements (for more information, see Section 29.5.16).
- **Message RAM Interface**: enables a connection between the Message RAM and the other blocks in the MCAN module.
- **Registers and Message Object Access**: Data consistency is ensured by indirect accesses to the message objects. During normal operation, all software and DMA accesses to the Message RAM are done through interface registers. The interface registers have the same word-length as the Message RAM.
- **Module Interface**: The MCAN module registers are accessed by the user's software through a 32-bit peripheral bus interface.
- **Clocking**: Two clocks are provided to the MCAN module: the peripheral synchronous clock (interface clock - MCAN_ICLK) and the peripheral asynchronous clock (functional clock - MCAN_FCLK).
- **Extension Interface**: All selected internal status and control signals are routed to this interface (except for the indication signals of configuration change enable bit (MCAN_CCCR.CCE) and Interrupt Register bits (MCAN_IR).

29.5.1 Module Clocking Requirements

Two clocks are provided to the MCAN module:

- the peripheral synchronous clock (MCAN_ICLK) as the general module clock source, and
- the peripheral asynchronous clock (MCAN_FCLK) provided to the CAN core for generating the CAN bit timing.

Within the MCAN module, there is a synchronization mechanism implemented to ensure safe data transfer between the two clock domains. There is synchronization between the signals from the Host clock domain to the CAN clock domain and conversely, and between the reset signal (MCAN_RST) to the Host clock domain and to the CAN clock domain.

Note

MCAN_ICLK must always be higher or equal to MCAN_FCLK, in order to achieve a stable functionality of the MCAN module: $f_{ICLK} \geq f_{FCLK}$

The CAN-FD supports higher speeds of operation and as such has more stringent timing requirements than the Classic CAN. For optimal performance, TI recommends using the lowest N-divider value that maintains a working PLL REF_CLK for the system. Lower N-divider values increase the loop bandwidth of the PLL, which in turn improves timing margins for CAN-FD.

29.5.2 Interrupt Requests

The MCAN module generates interrupt requests. It is configured via the Host CPU. The Suspend mode prevents the interrupt requests from propagating to the Host CPU. The MCAN core has two interrupt lines and 30 internal interrupt sources. Each source can be configured to drive one of the two interrupt lines. The interrupts are 'level high' interrupts. The MCAN core provides two interrupt requests (MCANSS_INT0 and MCANSS_INT1).

For more information, see the following registers:

- Interrupt Register (MCAN_IR)
- Interrupt Enable (MCAN_IE)
- Interrupt Line Select (MCAN_ILS)
- Interrupt Line Enable (MCAN_ILE)

The MCAN module supports External Timestamp Counter. The External Timestamp Counter produces an interrupt when it rolls over (see Section 29.5.10.1).

For more information, see the following registers:

- Interrupt Clear Shadow Register (MCANSS_ICS)
- Interrupt Raw Status Register (MCANSS_IRS)
- Interrupt Enable Clear Shadow Register (MCANSS_IECS)
- Interrupt Enable Register (MCANSS_IE)
- Interrupt Enable Status Register (MCANSS_IES)
- End Of Interrupt Register (MCANSS_EOI)
- External Timestamp Prescaler Register (MCANSS_EXT_TS_PRESCALER)
- External Timestamp Unserviced Interrupts Counter Register (MCANSS_EXT_TS_UNSERVICED_INTR_CNTR)

To clear IRQ_INT0, IRQ_INT1 and TS_WAKE interrupts, write to the EOI bit field for the corresponding interrupt number that is described in the MCANSS_EOI register. When MCAN is used by the C28x CPU, in addition to clearing the interrupt sources, it is necessary to clear the interrupt by writing '1' to PIEACK register in the bit position for group 9 (refer to PIE channel mapping) in order for successive MCAN interrupts to be recognized.

The MCAN module is capable of issuing an ECC interrupt. After clearing the ECC interrupt source, the application software must also write a 1 to the EOI registers (MCANERR_SEC_EOI.EOI_WR/ MCANERR_DED_EOI.EOI_WR). For more information, see Section 29.5.12.2.

29.5.3 Operating Modes

The operating modes are discussed in the following sections.

29.5.3.1 Software Initialization

A software initialization begins when the MCAN_CCCR.INIT bit is set to 1. This is done either by software or by a hardware reset, when an uncorrected bit error is detected in the Message RAM, or by going to a Bus_Off state. While the MCAN_CCCR.INIT bit is set, the message transfer is stopped and the status of the output TX pin is recessive (high). The counters of the Error Management Logic (EML) are unchanged. Setting the MCAN_CCCR.INIT bit does not change any configuration register. Resetting the MCAN_CCCR.INIT bit finishes the software initialization. After waiting for the occurrence of a sequence of 11 consecutive recessive bits (indication for Bus_Idle state) the message transfer starts.

Access to the MCAN configuration registers is only enabled when both MCAN_CCCR.INIT and MCAN_CCCR.CCE bits are set (write protection).

The MCAN_CCCR.CCE bit can only be set/reset while the MCAN_CCCR.INIT = 1. The MCAN_CCCR.CCE bit is automatically reset when the MCAN_CCCR.INIT bit is reset.

The following registers are reset when the MCAN_CCCR.CCE bit is set:

- MCAN_HPMS - High Priority Message Status
- MCAN_RXF0S - Rx FIFO 0 Status
- MCAN_RFX1S - Rx FIFO 1 Status
- MCAN_TXFQS - Tx FIFO/Queue Status
- MCAN_TXBRP - Tx Buffer Request Pending
- MCAN_TXBTO - Tx Buffer Transmission Occurred
- MCAN_TXBCF - Tx Buffer Cancellation Finished
- MCAN_TXEFS - Tx Event FIFO Status

The Timeout Counter value MCAN_TOCV.TOC field is preset to the value configured by the MCAN_TOCC.TOP field when the MCAN_CCCR.CCE bit is set.

In addition, the Tx Handler and Rx Handler are held in idle state while MCAN_CCCR.CCE = 1.

The following registers are only writeable while MCAN_CCCR.CCE = 0

- MCAN_TXBAR - Tx Buffer Add Request
- MCAN_TXBCR - Tx Buffer Cancellation Request

MCAN_CCCR.TEST and MCAN_CCCR.MON bits can only be set by the Host CPU while MCAN_CCCR.INIT = 1 and MCAN_CCCR.CCE = 1. Both bits may be reset at any time. The MCAN_CCCR.DAR bit can only be set/reset while MCAN_CCCR.INIT = 1 and MCAN_CCCR.CCE = 1.

Table 29-4 shows the steps to configure the MCAN module.

Table 29-4. Steps to Configure MCAN Module

Step	Operation	Description	Pseudo Code
1	Initialize MCAN_CCCR	Set MCAN_CCCR.INIT bit and check that it has been set	INIT = 1; If INIT ≠ 1, wait until set
2	Unlock protected registers	Set MCAN_CCCR.CCE bit	CCE = 1;
3	Configure CAN mode	Set MCAN_CCCR.FDOE bit to CAN FD	FDOE = 1 for CAN FD FDOE = 0 for Classic CAN
4	Configure Bit Rate Switching	Set MCAN_CCCR.BRSE bit	BRSE = 1 for bit rate switching BRSE = 0 for no bit rate switching
5	Set nominal bit timing	Set MCAN_NBTP register	
6	Lock protected registers	Clear MCAN_CCCR.CCE bit	CCE = 0;
7	Return MCAN module to normal operation	Clear MCAN_CCCR.INIT bit and check it has been cleared	INIT = 0; If INIT ≠ 0, wait until set

29.5.3.2 Normal Operation

Once the MCAN module is initialized and the MCAN_CCCR.INIT bit is reset to zero, the MCAN module synchronizes itself to the CAN bus and is ready for communication. After passing the acceptance filtering, received messages including Message Identifier (ID) and Data Length Code (DLC) are stored into a dedicated Rx Buffer or into Rx FIFO 0/Rx FIFO 1.

For messages to be transmitted,dedicated Tx buffers and/or a Tx FIFO or a Tx queue can be initialized or updated.

Note

Automated transmission upon reception of remote frames is not supported.

29.5.3.3 CAN FD Operation

The CAN FD standard allows extended frames to be sent, up to 64 data bytes in a single frame at a higher bit rate for the data phase of a frame, up to 5 Mbps. The CAN FD standard introduces the ability to switch from one bit rate to another. Extended Data Length (EDL), as shown in Figure 29-5, sets a data length of up to 8 or up to 64 data bytes. Bit Rate Switching (BRS) indicates whether two bit rates (the data phase is transmitted at a different bit rate compared to the arbitration phase) are enabled.

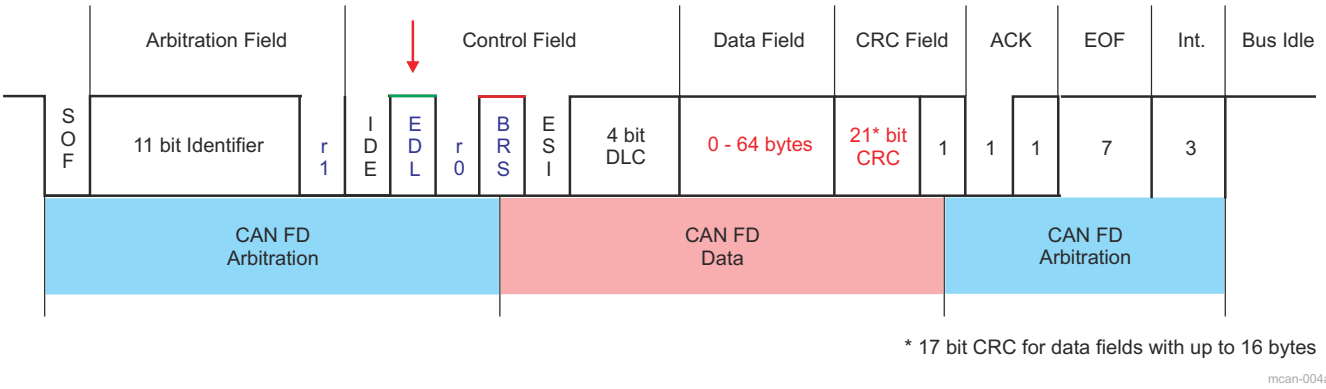


Figure 29-5. CAN FD Frame

There are two variants of CAN FD frame transmission:

- CAN FD frame transmission without bit rate switching
- CAN FD frame transmission where control field, data field, and CRC field are transmitted with a higher bit rate than the beginning and the end of the frame

In the CAN frames, FDF = recessive (logical 1) signifies a CAN FD frame, FDF = dominant (logical 0) signifies a Classic CAN frame. In a CAN FD frame, the two bits following FDF - res and BRS, decide whether the bit rate inside of this CAN FD frame is switched. A CAN FD bit rate switch is signified by res = dominant and BRS = recessive. Note that the coding of res = recessive is reserved for future expansion of the protocol. If the MCAN module receives a frame with FDF = recessive and res = recessive, it will signal a Protocol Exception Event by setting the MCAN_PSR.PXE bit. When Protocol Exception Handling is enabled (MCAN_CCCR.PXHD = 0), this causes the operation state to change from Receiver (MCAN_PSR.ACT = 10) to Integrating (MCAN_PSR.ACT = 00) at the next sample point. In case Protocol Exception Handling is disabled (MCAN_CCCR.PXHD = 1), the MCAN will treat a recessive bit as an error and will respond with an error frame.

CAN FD operation is enabled by programming the MCAN_CCCR.FDOE bit. If MCAN_CCCR.FDOE = 1, transmission and reception of CAN FD frames is enabled. Transmission and reception of Classic CAN frames is always possible. Whether a CAN FD frame or a Classic CAN frame is transmitted can be configured via the FDF bit in the respective Tx Buffer element.

With MCAN_CCCR.FDOE = 0, received frames are interpreted as Classic CAN frames, which leads to the transmission of an error frame when receiving a CAN FD frame. When CAN FD operation is disabled, no CAN FD frames are transmitted even if the FDF bit of a Tx Buffer element is set. The MCAN_CCCR.FDOE and MCAN_CCCR.BRSE bits can only be changed while the MCAN_CCCR.INIT and MCAN_CCCR.CCE bits are both set. With MCAN_CCCR.FDOE = 0, the setting of bits FDF and BRS is ignored and frames are transmitted in Classic CAN format.

With MCAN_CCCR.FDOE = 1 and MCAN_CCCR.BRSE = 0, only FDF bit of a Tx Buffer element is evaluated. With MCAN_CCCR.FDOE = 1 and MCAN_CCCR.BRSE = 1, transmission of CAN FD frames with bit rate switching is enabled. All Tx Buffer elements with bits FDF and BRS set are transmitted in CAN FD format with bit rate switching.

A mode change during CAN operation is only recommended under the following conditions:

- The failure rate in the CAN FD data phase is significantly higher than in the CAN FD arbitration phase. In this case, disable the CAN FD bit rate switching option for transmissions.
- During system startup all nodes are transmitting Classic CAN messages until it is verified that they are able to communicate in CAN FD format. If this is true, all nodes switch to CAN FD operation.
- Wakeup messages in CAN Partial Networking have to be transmitted in Classic CAN format.
- End-of-line programming in case not all nodes are CAN FD capable. Non CAN FD nodes are held in Silent mode until programming has completed. Then all nodes switch back to Classic CAN communication.

The coding of the DLC in the CAN FD format differs from the standard CAN format. The DLC codes 0 to 8 have the same coding as in standard CAN (0 to 8 data bytes), the codes 9 to 15, which in standard CAN all code a data field of 8 bytes, are coded according to Table 29-5.

Table 29-5. DLC Coding in CAN FD

DLC	9	10	11	12	13	14	15
Number of Data Bytes	12	16	20	24	32	48	64

For CAN FD frames, the bit timing will be switched inside the frame after the BRS (Bit Rate Switch) bit in case this bit is recessive. In the CAN FD arbitration phase, before the BRS bit, the nominal CAN bit timing (see Figure 29-6) is used as configured by the Nominal Bit Timing and Prescaler Register (MCAN_NBTP). In the following CAN FD data phase, the data phase bit timing is used as configured by the Data Bit Timing and Prescaler Register (MCAN_DBTP). The bit timing is switched back from the data phase timing at the CRC delimiter or when an error is detected, whichever occurs first.

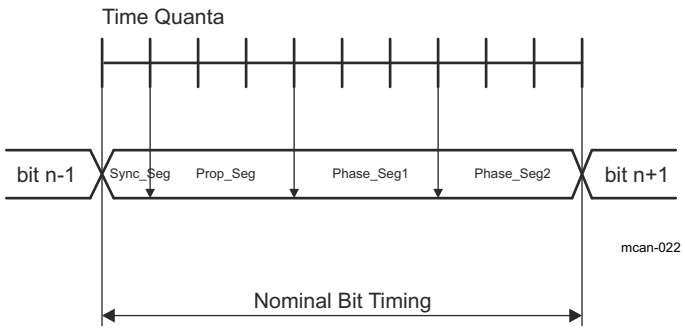


Figure 29-6. CAN Bit Timing

The maximum configurable data phase bit timing depends on the CAN clock frequency (MCAN_FCLK). Example: with MCAN_FCLK = 20 MHz and the shortest configurable bit time of 4 t_q (time quanta), the bit rate in the data phase is 5 Mbit/s.

In both data frame formats, CAN FD and CAN FD with bit rate switching, the value of the Error Status Indicator (ESI) bit depends on transmitter's error state (see MCAN_PSR.RESI bit) monitored at the start of the transmission. If the transmitter has error passive flag, the ESI bit is transmitted recessive, else it is transmitted dominant.

29.5.4 Transmitter Delay Compensation

29.5.4.1 Description

When only one CAN FD node is transmitting and all other nodes are receivers, the length of the bus line has no impact. When transmitting via the TX pin the MCAN module receives the transmitted data from its local CAN transceiver via the RX pin. The received data is delayed. If the transmitter delay is greater than TSEG1 (time segment before sample point), a bit error is detected.

The MCAN module provides a delay compensation mechanism to compensate for the transmitter delay. The compensation mechanism enables transmission with higher bit rates during the CAN FD data phase independent of the delay of a specific CAN transceiver. Without transmitter delay compensation the bit rate in the data phase is limited by the transmitter delay.

The mechanism enables configurations where the data bit time is shorter than the transmitter delay (it is described in detail in ISO 11898-1:2015). The transmitter delay compensation is enabled by setting the MCAN_DBTP.TDC bit to 1.

The delayed transmit data is compared against the received data at the Secondary Sample Point (SSP) in order to check for bit errors during the data phase of transmitting nodes. If a bit error is detected, the transmitter will react on this bit error at the next following regular sample point. During arbitration phase the delay compensation is always disabled.

The received bit is compared against the transmitted bit at the SSP. The SSP position is defined as the sum of the measured delay from the MCAN's transmit output TX pin through the transceiver to the receive input RX pin plus the transmitter delay compensation offset configured by the MCAN_TDCR.TDCO field (see Figure 29-7). The transmitter delay compensation offset is used to adjust the position of the SSP inside the received bit (example: half of the bit time in the data phase). The position of the SSP is rounded down to the next integer number of mtq.

The actual transmitter delay compensation value can be checked by reading the MCAN_PSR.TDCV field. This field is cleared when the MCAN_CCCR_INIT bit is set and is updated at each transmission of CAN FD frame while the MCAN_DBTP.TDC bit is set.

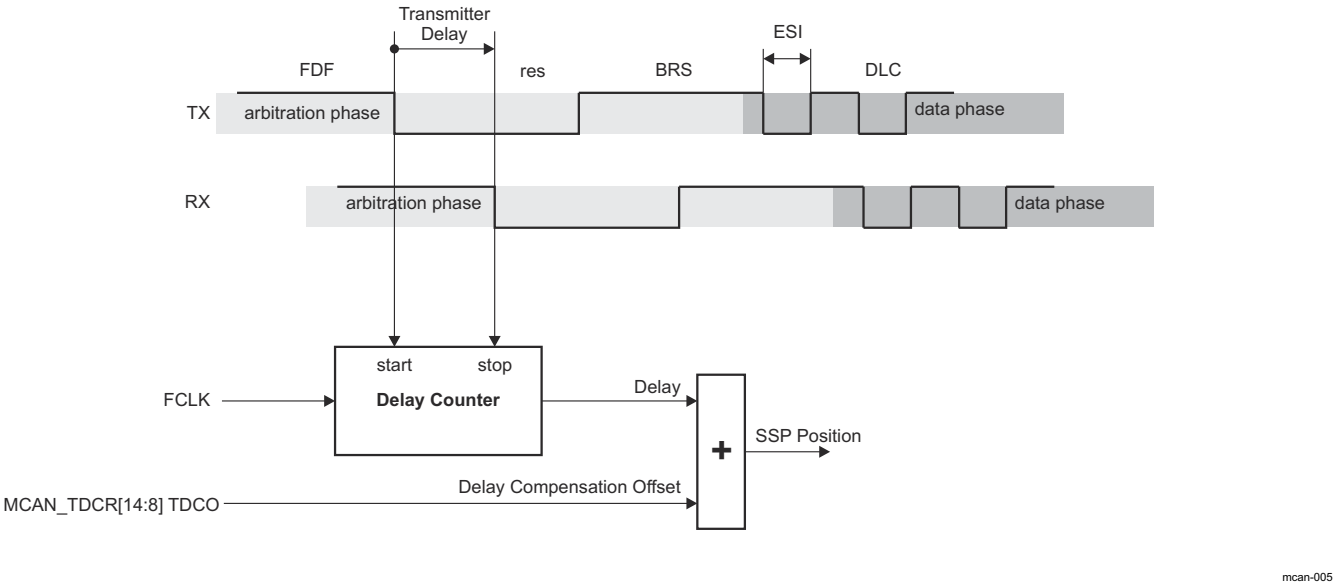


Figure 29-7. Transmitter Delay Measurement

29.5.4.2 Transmitter Delay Compensation Measurement

When transmitter delay compensation is enabled (by programming MCAN_DBTP.TDC = 1), the measurement is started within each transmitted CAN FD frame at the falling edge of FDF bit to bit r0. The measurement is stopped when this edge is seen at the receive input RX pin of the transmitter. The resolution of this measurement is one mtq (see Figure 29-7). The mtq (minimum time quantum) dimension is equal to the CAN clock period (MCAN_FCLK).

The use of a transmitter delay compensation filter window can be enabled by programming the MCAN_TDCR.TDCF field. This filter feature defines a minimum value for the SSP position to avoid the case in which a dominant glitch inside the received FDF bit ends the delay compensation measurement before the falling edge of the received res bit, resulting in an early taken SSP position. Dominant edges on the RX pin, that would result in an earlier SSP position are ignored for transmitter delay measurement. The measurement is stopped when the SSP position is at least MCAN_TDCR.TDCF field and the RX pin is low.

The following boundary conditions have to be considered:

- The sum of the measured delay from the TX pin to the RX pin and the configured transmitter delay compensation offset (MCAN_TDCR.TDCO field) has to be less than six bit times in the data phase.
- The sum of the measured delay from the TX pin to the RX pin and the configured transmitter delay compensation offset (MCAN_TDCR.TDCO) field has to be less or equal 127 mtq. In case this sum exceeds 127 mtq, the maximum value of 127 mtq is used for transmitter delay compensation.
- The data phase ends at the sample point of the CRC delimiter, that stops checking of receive bits at the SSPs.

29.5.5 Restricted Operation Mode

In restricted operation mode, the CAN node is able to receive data and remote frames and to give acknowledgment to valid frames, but it does not send data frames, remote frames, active error frames, or overload frames. In case of an error condition or overload condition, it does not send dominant bits; instead it waits for the occurrence of bus idle condition to resynchronize itself to the CAN communication. The receive and transmit error counters (MCAN_ECR.REC and MCAN_ECR.TEC) are frozen while CAN error logging (MCAN_ECR.CEL) is active. The Host CPU can set the MCAN module into Restricted Operation Mode by setting the MCAN_CCCR.ASM bit. The bit can only be set by the Host CPU at any time when both MCAN_CCCR.CCE and MCAN_CCCR.INIT bits are set to 1.

The restricted operation mode is automatically entered when the Tx Handler was not able to read data from the Message RAM in time. To leave restricted operation mode, the Host CPU has to reset the MCAN_CCCR.ASM bit. This mode can be used in applications that adapt themselves to different CAN bit rates. In this case, the application tests different bit rates and leaves the restricted operation mode after it has received a valid frame.

Note

The Restricted Operation Mode must not be combined with the Loop Back Mode.

29.5.6 Bus Monitoring Mode

Entering bus monitoring mode is done by setting the MCAN_CCCR.MON bit to 1. In this mode (see ISO 11898-1:2015, *Bus Monitoring* section), the MCAN module is able to receive valid data and remote frames, but cannot start a transmission. The MCAN module sends only recessive bits on the CAN bus. If the MCAN module is required to send a dominant bit (ACK bit, overload flag, active error flag), the bit is rerouted internally so that the MCAN module monitors this dominant bit, although the CAN bus may remain in recessive state. In bus monitoring mode the MCAN_TXBRP register is held in reset state. The bus monitoring mode can be used to analyze the traffic on a CAN bus without affecting it by the transmission of dominant bits. Figure 29-8 shows the connection of the TX and RX signals to the MCAN module in bus monitoring mode.

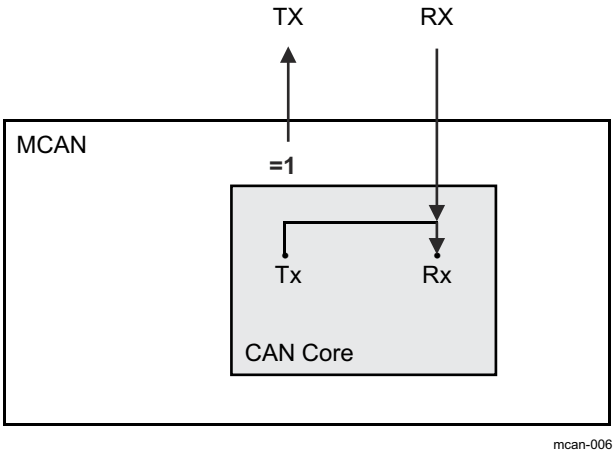


Figure 29-8. Connection of Signals in Bus Monitoring Mode

29.5.7 Disabled Automatic Retransmission (DAR) Mode

According to the CAN Specification (see ISO11898-1:2015, *Recovery Management* section), the MCAN module provides means for automatic retransmission of frames that have lost arbitration or that have been disturbed by errors during transmission. By default automatic retransmission is enabled (see the MCAN_CCCR.DAR bit).

29.5.7.1 Frame Transmission in DAR Mode

In DAR mode all transmissions are automatically cancelled after they started on the CAN bus. A Tx buffer's Tx Request Pending (MCAN_TXBRP[xx]) TRPx bit is reset after successful transmission, when a transmission has not yet been started at the point of cancellation, has been aborted due to lost arbitration, or when an error occurred during frame transmission.

Successful transmission:

- Corresponding Tx Buffer Transmission Occurred (MCAN_TXBTO[xx]) TOx bit is set
- Corresponding Tx Buffer Cancellation Finished (MCAN_TXBCF[xx]) CFx bit is not set

Successful transmission in spite of cancellation:

- Corresponding Tx Buffer Transmission Occurred (MCAN_TXBTO[xx]) TOx bit is set
- Corresponding Tx Buffer Cancellation Finished (MCAN_TXBCF[xx]) CFx bit is set

Arbitration lost or frame transmission disturbed:

- Corresponding Tx Buffer Transmission Occurred (MCAN_TXBTO[xx]) TOx bit is not set
- Corresponding Tx Buffer Cancellation Finished (MCAN_TXBCF[xx]) CFx bit is set

In the case of a successful frame transmission, and if storage of Tx events is enabled, a Tx Event FIFO element is written with Event Type ET = 10 (transmission in spite of cancellation).

29.5.8 Clock Stop Mode

Entering clock stop mode is controlled by the input clock stop request signal or MCAN_CCCR.CSR bit. As long as the clock stop request signal is active, the MCAN_CCCR.CSR bit is read as 1. When all pending transmission requests have completed, the MCAN module waits until bus idle state is detected. Then the MCAN module sets the MCAN_CCCR.INIT to 1 to prevent any further CAN transfers. The MCAN module acknowledges that it is ready for power down by setting the output clock stop acknowledge signal to 1 and the MCAN_CCCR.CSA bit to 1. In this state, before the clocks are switched off, further register accesses can be made. A write access to the MCAN_CCCR.INIT bit will have no effect. Now the module clock inputs MCAN_ICLK and MCAN_FCLK may be switched off.

To leave power down mode, the application has to turn on the module clocks before resetting the input clock stop request signal respectively the MCAN_CCCR.CSR flag bit. The MCAN will acknowledge this by resetting the output clock stop acknowledge signal respectively the MCAN_CCCR.CSA flag bit. Afterwards, the application can restart CAN communication by resetting MCAN_CCCR.INIT bit.

Restoring the clocks from clock stop mode, needs to be done according to how the clock stop was initiated.

The MCAN module supports two external clock stop modes:

- Immediate
- Graceful

In a graceful clock stop mode, when the clock stop request is asserted, the MCAN core will respond with clock stop acknowledge when all pending Tx messages have been processed and an Idle line had been detected. The MCAN_CCCR.INIT bit will be set, the MCAN core will go and stay Idle.

The automatic wakeup feature is enabled by setting the MCANSS_CTRL.AUTOWAKEUP and MCANSS_CTRL.WAKEUPREQEN bits to 1 (for more information, see Section 29.5.8.2). When external clock stop request is removed and no suspend request is active, a read-modify-write to the MCAN_CCCR.INIT bit is performed to clear it.

29.5.8.1 Suspend Mode

The MCAN module supports two suspend modes:

- Immediate
- Graceful

In a graceful suspend mode (see the MCANSS_CTRL.DBGSUSP_FREE bit), when the suspend request is asserted, a clock stop request to the MCAN core is performed. The MCAN core will respond with clock stop acknowledge when all pending Tx messages have been processed and an Idle line had been detected. At that point the MCAN_CCCR.INIT bit will be set and the MCAN core will stay Idle. The suspend state can be verified by reading MCAN_CCCR.INIT bit.

The automatic wakeup feature is enabled by setting the MCANSS_CTRL.AUTOWAKEUP and MCANSS_CTRL.WAKEUPREQEN bits to 1 (for more information, see Section 29.5.8.2). When suspend request is removed, if no external clock stop request is active, a read-modify-write to the MCAN_CCCR.INIT bit is performed to clear it.

During suspend mode the auto-clear feature is disabled. The following register fields have an auto-clear feature:

- MCAN_ECR.CEL
- MCAN_PSR.LEC
- MCAN_PSR.DLEC
- MCAN_PSR.RESI
- MCAN_PSR.RBRS
- MCAN_PSR.RFDF
- MCAN_PSR.PXE

29.5.8.2 Wakeup Request

Issuing a clock stop request puts the MCAN module into Power Down mode (Sleep Mode). During transition from IDLE to ACTIVE, if the MCANSS_CTRL.AUTOWAKEUP and MCANSS_CTRL.WAKEUPREQEN bits are enabled, after the MCAN Core respond to the removal of the clock stop request with removing the clock stop acknowledge, a read-modify-write will be issued to clear the MCAN_CCCR.INIT bit and the MCAN core will resume operation.

If the MCANSS_CTRL.WAKEUPREQEN bit is set, the MCAN module provides a wakeup request on the following wakeup event:

- The receive RX pin is dominant (logical 0)

The wakeup request is de-asserted when any of the following conditions occur:

- Clock stop request is removed and clock stop acknowledge is de-asserted
- A reset is applied to the MCAN module

29.5.9 Test Modes

The MCAN_TEST register write access is enabled by setting the test mode enable MCAN_CCCR.TEST bit to 1. The MCAN_TEST register allows the configuration of the test modes and test functions.

The CAN transmit TX pin has four output functions. One of those functions can be selected by programming the MCAN_TEST.TX field. Additionally to its default function (the serial data output) it can drive the CAN sample point signal to monitor the MCAN's bit timing and it can drive constant dominant or recessive values.

The actual value of the CAN receive RX pin can be monitored from MCAN_TEST.RX bit. Both functions can be used to check the CAN bus physical layer. Due to the synchronization mechanism between the CAN clock (MCANx_FCLK) and Host clock (MCANx_ICLK) domain, there may be a delay of several Host clock periods between writing to the MCAN_TEST.TX field until the new configuration is visible at the output TX pin. This applies also when reading input RX pin by way of the MCAN_TEST.RX bit.

Note

Test modes should be used for self-test only. The software control for TX pin interferes with all CAN protocol functions. It is not recommended to use test modes for application.

29.5.9.1 Internal Loop Back Mode

The MCAN module can be set into internal loop back mode by programming MCAN_TEST.LBCK and MCAN_CCCR.MON bits to 1. The internal loop back mode is used for a Hot Selftest. The Hot Selftest allows the MCAN module to be tested without affecting a running CAN system connected to the TX and RX pins. In this mode, the RX pin is disconnected from the MCAN module and the TX pin is held recessive. Figure 29-9 shows the connection of the TX and RX pins to the MCAN module in internal loop back mode.

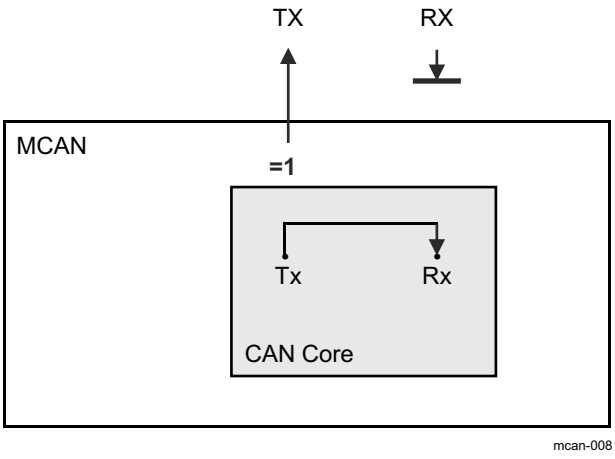


Figure 29-9. Internal Loop Back Mode

29.5.10 Timestamp Generation

The MCAN module has integrated a 16-bit wrap-around counter for timestamp generation. The timestamp counter prescaler MCAN_TSCC.TCP field can be configured to clock the counter in multiples of CAN bit times (1-16). The counter is readable by way of the MCAN_TSCV.TSC field. A write access to the MCAN_TSCV register resets the counter to zero. When the timestamp counter wraps around the interrupt MCAN_IR.TSW flag

is set. On start of a frame reception/transmission the counter value is captured and stored into the timestamp section of an Rx Buffer/Rx FIFO (RXTS[15:0]) or Tx Event FIFO (TXTS[15:0]) element. For more information, see Section 29.5.16.

29.5.10.1 External Timestamp Counter

For CAN FD operation mode the MCAN core requires an external timestamp counter. An externally generated 16-bit vector may substitute the integrated 16-bit CAN bit time counter (internal timestamp counter) for receive and transmit timestamp generation. An external 16-bit timestamp counter can be used by programming the MCAN_TSCC.TSS field.

The external timestamp counter uses the interface clock (MCANx_ICLK) as a reference clock. The MCAN core accepts a 16-bit timestamp. A 24-bit prescaler provides a programmable resolution for the timestamp (see MCANSS_EXT_TS_PRESCALER.PRESCALER bit field). The external timestamp counter can be enabled or disabled through the MCANSS_CTRL.EXT_TS_CNTR_EN bit. When disabled, the counter is reset back to zero. While enabled, the counter keeps incrementing. When the timestamp rolls over, the MCAN_IRQ_TS interrupt is generated.

When the timestamp rolls over, the MCANSS_IRS register is set. The MCANSS_IE register can be affected by writing to the MCANSS_IESS register to set or to the MCANSS_IECS register to clear. The MCANSS_IESS register is a shadow register mapped to the same address as the MCANSS_IE register. The level interrupt is a reflection of both MCANSS_IRS and MCANSS_IE being set. The MCANSS_IES register reflects the level interrupt. When an rollover event occurs, the interrupt counter is incremented. Writing to the MCANSS_ICS register to clear the MCANSS_IRS register will also decrement the interrupt counter. Writing to the MCANSS_EOI register will issue another pulse if the interrupt counter is not zero.

The rollover event can be artificially simulated by software through writing to the Interrupt Set Shadow register (MCANSS_ISS). The MCANSS_ISS register is a shadow register mapped to the same address as the MCANSS_IRS register.

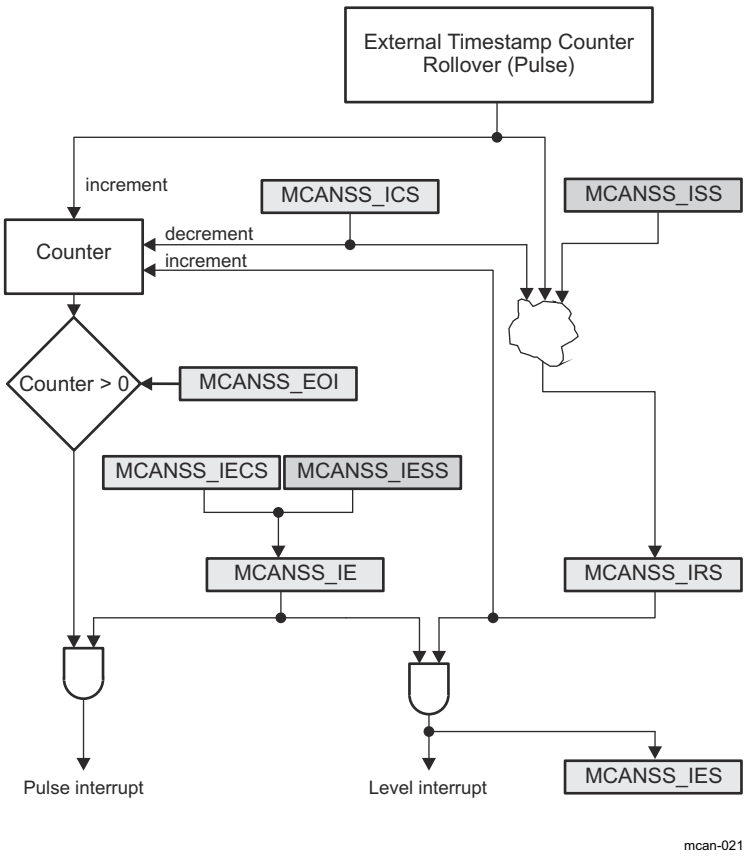


Figure 29-10. External Timestamp Counter Interrupt

29.5.11 Timeout Counter

The MCAN module has integrated a 16-bit timeout counter. It is used to signal timeout conditions for the Rx FIFO 0, Rx FIFO 1, and Tx Event FIFO Message RAM elements. The timeout counter is configured by way of the MCAN_TOCC register. It is enabled by way of the MCAN_TOCC.ETOC bit. The timeout counter operates as down-counter and uses the same prescaler programmed by the MCAN_TSCC.TCP field as the timestamp counter. The actual counter value can be monitored from the MCAN_TOCV.TOC field. The timeout counter can be started only when MCAN_CCCR.INIT = 0 and stopped when MCAN_CCCR.INIT = 1 (example: when the MCAN enters Bus_Off state). The operation mode is selected by the MCAN_TOCC.TOS field. When continuous mode is selected, the counter starts when MCAN_CCCR.INIT = 0, a write to the MCAN_TOCV register presets the counter to the value configured by the MCAN_TOCC.TOP field and continues down-counting.

In case the timeout counter is controlled by one of the FIFOs, an empty FIFO presets the counter to the value configured by the MCAN_TOCC.TOP field. Down-counting is started when the first FIFO element is stored. Writing to the MCAN_TOCV register has no effect. When the counter reaches zero, the interrupt MCAN_IR.TOO flag is set.

In continuous mode, the counter is immediately restarted at the value configured by the MCAN_TOCC.TOP field.

Note

The clock signal for the timeout counter is derived from the CAN core's sample point signal. Therefore the point in time where the timeout counter is decremented may vary due to the synchronization/re-synchronization mechanism of the CAN core. If the baud rate switch feature in CAN FD is used, the timeout counter is clocked differently in arbitration and data field.

29.5.12 Safety

The Message Memory is wrapped in an ECC wrapper providing SECDED parity functionality. The ECC wrapper is controlled by an ECC aggregator.

29.5.12.1 ECC Wrapper

The ECC wrapper provides single error correction (SEC) and double error detection (DED) parity to the message memory content. It has side band signals for error notification. The ECC Wrapper implements an error injection test mode.

The error correction is done using a lazy write back. When an error is detected, it is noted in a FIFO queue which waits for an access gap to write the data back and refresh the memory. If a transaction writes new data to the compromised entry before the lazy write back completes, the write back is discarded.

29.5.12.2 ECC Aggregator

This section describes the functional details of the ECC aggregator module.

29.5.12.2.1 ECC Aggregator Overview

The ECC aggregator module supports the following general features:

- Provides a mechanism to control and monitor the ECC RAM in the MCAN module.
- Provides software access to all the ECC related registers.
- Supports software readable status of ECC single/double-bit errors and associated info such as RAM address and data bit(s) that are in error.
- Aggregates level pending status from the ECC RAM into a single interrupt to the Host CPU.

The following feature is not supported:

- Statistics such as tracking the number of single and double-bit errors. If needed, these operations can be handled by software.

29.5.12.2.2 ECC Aggregator Registers

There are three groups of registers in the ECC aggregator module:

- **Global registers:** Aggregator Revision Register (MCANERR_REV), ECC Vector Register (MCANERR_VECTOR), Misc Status Register (MCANERR_STAT), ECC Control Register (MCANERR_CTRL), and ECC Wrapper Revision Register (MCANERR_WRAP_REV).
- **Control and status registers:** ECC Error Control Registers (MCANERR_ERR_CTRL1 and MCANERR_ERR_CTRL2) and ECC Error Status Registers (MCANERR_ERR_STAT1, MCANERR_ERR_STAT2, and MCANERR_ERR_STAT3).
- **Interrupt registers:** interrupt status, interrupt enable set, interrupt enable clear, and EOI (End Of Interrupt) registers that are part of a standard interrupt module. For more information, see the following registers:
  - MCANERR_SEC_EOI
  - MCANERR_SEC_STATUS
  - MCANERR_SEC_ENABLE_SET
  - MCANERR_SEC_ENABLE_CLR
  - MCANERR_DED_EOI
  - MCANERR_DED_STATUS
  - MCANERR_DED_ENABLE_SET
  - MCANERR_DED_ENABLE_CLR

29.5.12.3 Reads to ECC Control and Status Registers

The reads to the ECC control and status registers are triggered by writing a 'read message' to the ECC Vector Register as:

- Software writes value (the ECC RAM ID) to the MCANERR_VECTOR.ECC_VECTOR field to select the ECC RAM for control or status.
- Software writes 1 to the MCANERR_VECTOR.RD_SVBUS bit to trigger a read.
- Software writes read address to the MCANERR_VECTOR.RD_SVBUS_ADDRESS field.
- Software then polls the MCANERR_VECTOR.RD_SVBUS_DONE bit to check if it is 1. This bit indicates that the read operation has completed.
- Software reads the data from the ECC control or status register. The following clock cycle (MCAN_ICLK) returns the read data.

29.5.12.4 ECC Interrupts

The ECC aggregator module aggregates the level pending status from the ECC RAM into a single EOI-handshake based interrupt to the Host CPU. Software is expected to follow the sequence described:

- Software enables the interrupts for the ECC RAM by writing to the MCANERR_SEC_ENABLE_SET/ MCANERR_DED_ENABLE_SET register.
- Software writes the ECC RAM ID in the MCANERR_VECTOR.ECC_VECTOR.
- Software writes the MCANERR_VECTOR.RD_SVBUS bit to trigger the read.
- Software writes the MCANERR_ERR_STAT1 register address to the MCANERR_VECTOR.RD_SVBUS_ADDRESS field. Software will need to load the 'read message' in the rMCANERR_VECTOR register again if it needs to read the MCANERR_ERR_STAT2 register.
- Software polls the MCANERR_VECTOR.RD_SVBUS_DONE bit. When this bit is set, a read of the MCANERR_ERR_STAT1/MCANERR_ERR_STAT2 register is performed.
- After the interrupt has been serviced, software will clear the interrupt status by writing to the MCANERR_ERR_STAT1.CLR_ECC_SEC or MCANERR_ERR_STAT1.CLR_ECC_DED bit depending on the type of the ECC error.
- Software has to poll the MCANERR_ERR_STAT1 register to guarantee that the status bit has been cleared.
- Software will write to the MCANERR_SEC_EOI/MCANERR_DED_EOI register to clear the interrupt.
- After clearing the ECC interrupt source, the application software must also write 1 to the MCANERR_SEC_EOI.EOI_WR /MCANERR_DED_EOI.EOI_WR bits.

29.5.13 Rx Handling

The Rx Handler controls the following operations:

- Acceptance filtering
- The transfer of received messages to the Rx buffers or to one of the two Rx FIFOs (Rx FIFO 0 or Rx FIFO 1)
- Rx FIFO Put and Get Index operations

29.5.13.1 Acceptance Filtering

The MCAN module employs two sets of acceptance filters - one set for standard and one set for extended identifiers. These filters can be assigned to an Rx Buffer or to one of the two Rx FIFOs.

The main features of the filter elements are:

- Each filter element can be configured as:
  - Range filter (from - to)
  - Filter for specific IDs (for one or two dedicated IDs)
  - Classic bit mask filter
- Each filter element can be enabled/disabled individually
- Each filter element can be configured for acceptance or rejection filtering
- Filters are checked sequentially and execution (acceptance filtering procedure) stops at the first matching filter element or when the end of the filter list is reached

Related configuration registers are:

- Global Filter Configuration (MCAN_GFC) register
- Standard ID Filter Configuration (MCAN_SIDFC) register
- Extended ID Filter Configuration (MCAN_XIDFC) register
- Extended ID AND Mask (MCAN_XIDAM) register

Depending on the configuration of the filter element (see SFEC/EFEC in Section 29.5.16) if filter matches, one of the following actions is performed:

- Received frame is stored in FIFO 0 or FIFO 1
- Received frame is stored in Rx Buffer
- Received frame is stored in Rx Buffer and generation of pulse at filter event pin is performed. This is high level single MCAN_ICLK pulse.
- Received frame is rejected
- Set High Priority Message interrupt flag MCAN_IR.HPM
- Set High Priority Message interrupt flag MCAN_IR.HPM and store received frame in FIFO 0 or FIFO 1

Acceptance filtering starts when complete Message ID is received. Acceptance filtering stops at the first matching enabled filter element or when the end of the filter list is reached. If a filter element matches - the Rx Handler starts writing the received message data in portions of 32-bit to the matching Rx Buffer or Rx FIFO. If an error condition occurs (for example: CRC error), this message is rejected with the following impact on the affected Rx Buffer or Rx FIFO:

- Rx Buffer: New Data flag (MCAN_NDAT1/MCAN_NDAT2) of matching Rx Buffer is not set, but Rx Buffer (partly) overwritten with received data (for error type, see MCAN_PSR.LEC and MCAN_PSR.DLEC fields,respectively).
- Rx FIFO: Put index of matching Rx FIFO is not updated, but related Rx FIFO element (partly) overwritten with received data (for error type, see MCAN_PSR.LEC and MCAN_PSR.DLEC fields, respectively). If matching Rx FIFO is configured to operate in overwrite mode, the boundary conditions described in Section 29.5.13.2.2 have to be considered.

Note

When an accepted message is written to one of the two Rx FIFOs, or into an Rx Buffer, the unmodified received identifier is stored independent of the filter(s) used. The result of the acceptance filter process is strongly depending on the sequence of configured filter elements.

29.5.13.1.1 Range Filter

Each filter element can be configured to operate as Range Filter (Standard Filter Type SFT = 00/Extended Filter Type EFT = 00). The filter matches for all received message frames with IDs in the range from SFID1 to SFID2 (SFID2 ≥ SFID1) respectively in the range from EFID1 to EFID2 (EFID2 ≥ EFID1). For more information see Section 29.5.16.5 and Section 29.5.16.6.

There are two options for range filtering of extended frames:

- Extended Filter Type EFT = 00: The Extended ID AND Mask (MCAN_XIDAM) is used for Range Filtering. The Message ID of received frames is ANDed with the Extended ID AND Mask (MCAN_XIDAM) before the range filter is applied.
- Extended Filter Type EFT = 11: The Extended ID AND Mask (MCAN_XIDAM) is not used for Range Filtering.

29.5.13.1.2 Filter for Specific IDs

Each filter element can be configured to filter one or two dedicated Message IDs (Standard Filter Type SFT =01/ Extended Filter Type EFT =01). To filter only one specific Message ID, the filter element has to be configured with SFID1 = SFID2 respectively EFID1 = EFID2. For more information, see Section 29.5.16.5 and Section 29.5.16.6.

29.5.13.1.3 Classic Bit Mask Filter

Classic bit mask filtering can filter groups of Message IDs (Standard Filter Type SFT =10/Extended Filter Type EFT =10). This is done by masking single bits of a received Message ID. In this case SFID1/EFID1 element is used as Message ID filter, while the SFID2/EFID2 element is used as filter mask.

A 0 bit at the filter mask (SFID2/EFID2) will mask out the corresponding bit position of the configured Message ID filter (SFID1/EFID1) and the value of the received Message ID at that bit position is not relevant for acceptance filtering. Only those bits of the received Message ID where the corresponding mask bits are 1 are relevant for acceptance filtering.

There are two interesting cases:

- All mask bits are 1: a match occurs only when the received Message ID and the configured Message ID filter are identical.
- All mask bits are 0: all Message IDs match.

29.5.13.1.4 Standard Message ID Filtering

Figure 29-11 shows the standard Message ID (11-bit ID) filtering flow. Section 29.5.16.5 describes the standard Message ID filter element.

The Remote Transmission Request (RTR) and Extended Identifier (XTD) bits of the received frames are compared against the list of configured filter elements. This is controlled by the following registers:

- Global Filter Configuration (MCAN_GFC) register
- Standard ID Filter Configuration (MCAN_SIDFC) register

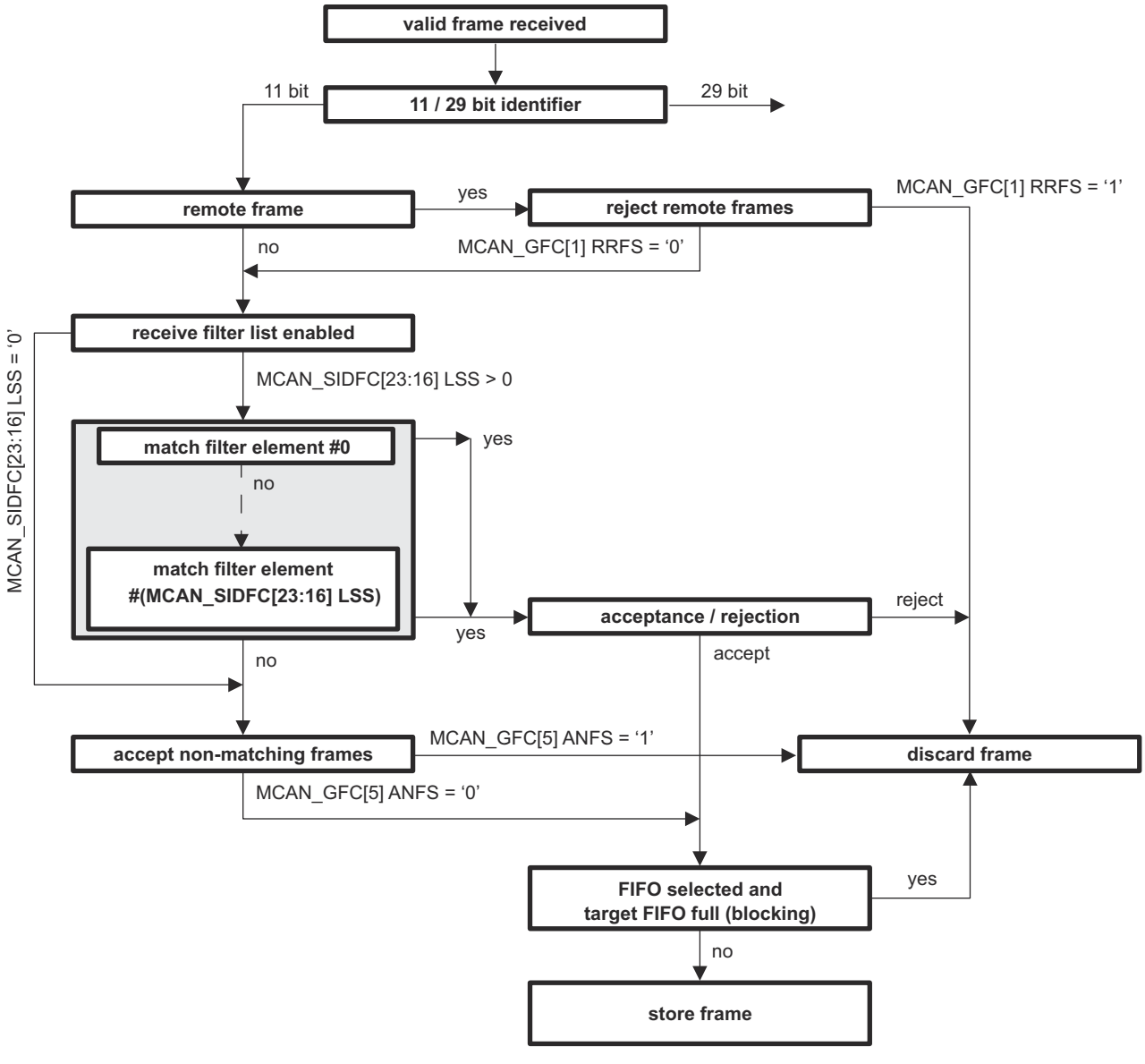


Figure 29-11. Standard Message ID Filter Path

29.5.13.1.5 Extended Message ID Filtering

Figure 29-12 shows the extended Message ID (29-bit ID) filtering flow. Section 29.5.16.6 describes the extended Message ID filter element.

The Remote Transmission Request (RTR) and Extended Identifier (XTD) bits of the received frames are compared against the list of configured filter elements. This is controlled by the following registers:

- Global Filter Configuration (MCAN_GFC) register
- Extended ID Filter Configuration (MCAN_XIDFC) register

Note that before the filter list is executed, the received identifier is ANDed with the Extended ID AND Mask (MCAN_XIDAM).

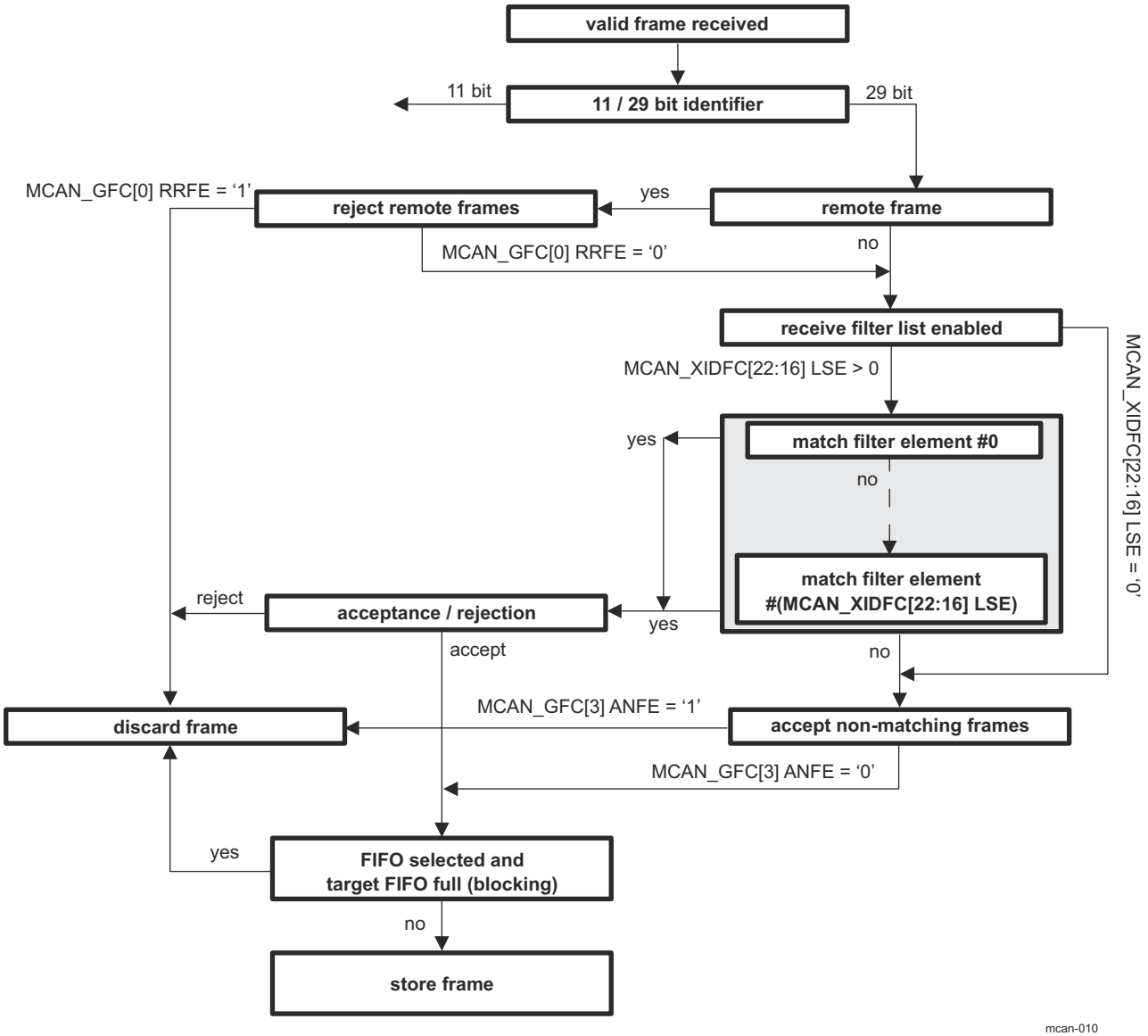


Figure 29-12. Extended Message ID Filter Path

29.5.13.2 Rx FIFOs

The configuration of the Rx FIFOs (Rx FIFO 0 and Rx FIFO 1) can be done by way of the MCAN_RXF0C and MCAN_RXF1C registers. Each Rx FIFO can be configured to store up to 64 received messages.

After acceptance filtering the received messages that passed are transferred to the Rx FIFO. The filter mechanisms available for the Rx FIFO 0 and Rx FIFO 1 are described in Section 29.5.13.1. The Rx FIFO element is described in Section 29.5.16.2.

The Rx FIFO watermark can be used to prevent an Rx FIFO overflow. If the Rx FIFO fill level reaches the Rx FIFO watermark configured by the MCAN_RXFnC[30:24] FnWM filed (where: n = 0 or 1), an interrupt flag MCAN_IR.RF0W/MCAN_IR.RF1W is set.

When the Rx FIFO Put Index reaches the Rx FIFO Get Index (MCAN_RXFnS[21:16] FnPI = MCAN_RXFnS[13:8] FnGI), an Rx FIFO Full condition is signalled by the MCAN_RXFnS[24] FnF status bit and interrupt flag MCAN_IR.RF0F/MCAN_IR.RF1F is set. Figure 29-13 shows Rx FIFO Status. The FIFOs fill level is presented in the MCAN_RXFnS[6:0] FnFL field (the number of elements stored in Rx FIFO).

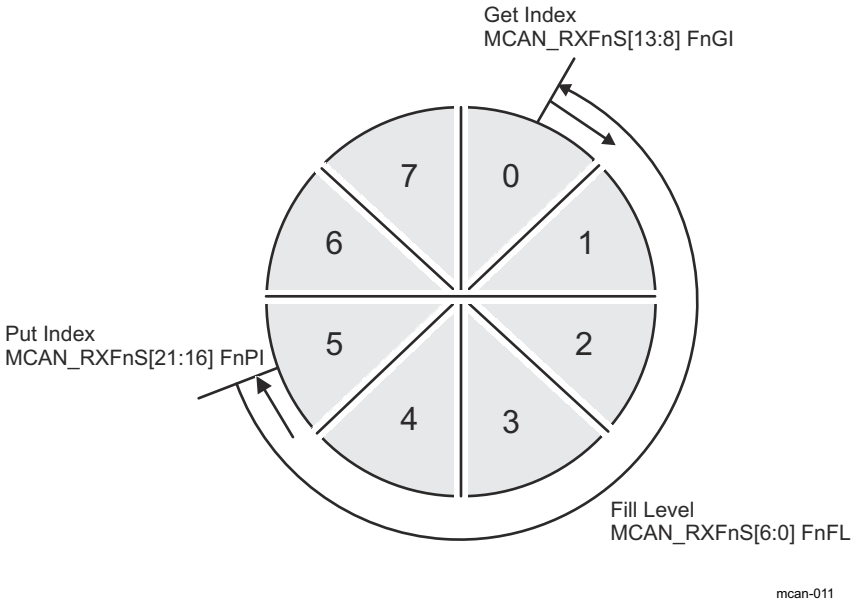


Figure 29-13. Rx FIFO Status

Rx FIFOs start address in the Message RAM (MCAN_RXFnC[15:2]FnSA field) has to be configured when reading from an Rx FIFO (Rx FIFO Get Index - MCAN_RXFnS[13:8] FnGI). Table 29-6 presents Rx Buffer/Rx FIFO Element Size for different Rx Buffer / Rx FIFO Data Field Size which is configured by way of the MCAN_RXESC register.

Table 29-6. Rx Buffer/Rx FIFO Element Size

MCAN_RXESC Register RBDS/F0DS/F1DS Bits	Data Field [bytes]	FIFO Element Size [RAM words]
000	8	4
001	12	5
010	16	6
011	20	7
100	24	8
101	32	10
110	48	14
111	64	18

29.5.13.2.1 Rx FIFO Blocking Mode

The Rx FIFO blocking mode is the default operation mode for the Rx FIFOs. It is configured by the MCAN_RXFnC[31] FnOM = 0.

If an Rx FIFO full condition is reached (MCAN_RXFnS[21:16] FnPI = MCAN_RXFnS[13:8] FnGI), no further messages are written to the corresponding Rx FIFO until at least one message has been read out and the Rx FIFO Get Index has been incremented. An Rx FIFO full condition is signalled by the MCAN_RXFnS[24] FnF = 1 and interrupt flag MCAN_IR.RF0F/MCAN_IR.RF1F is set.

In case a message is received while the corresponding Rx FIFO is full, this message is rejected and the message lost condition is signalled by MCAN_RXFnS[25] RFnL = 1 and interrupt flag MCAN_IR.RF0L/ MCAN_IR.RF1L is set.

29.5.13.2.2 Rx FIFO Overwrite Mode

The Rx FIFO overwrite mode is configured by the MCAN_RXFnC[31] FnOM = 1. When an Rx FIFO full condition is reached (MCAN_RXFnS[21:16] FnPI = MCAN_RXFnS[13:8] FnGI) signalled by MCAN_RXFnS[24] FnF = 1, the next accepted message for the FIFO will overwrite the oldest FIFO message. Put index/Get index are both incremented by one.

In overwrite mode, if an Rx FIFO full condition is signalled, reading of the Rx FIFO elements should start at least at get index + 1. The reason for that is, that it might happen, that a received message is written to the Message RAM (Put index) while the Host CPU is reading from the Message RAM (Get index). In this case inconsistent data may be read from the respective Rx FIFO element. The problem is solved by adding an offset to the Get index when reading from the Rx FIFO. The offset depends on how fast the Host CPU accesses the Rx FIFO. Figure 29-14 shows an offset of two with respect to the Get index when reading the Rx FIFO. In this case the two messages stored in element 1 and 2 are lost.

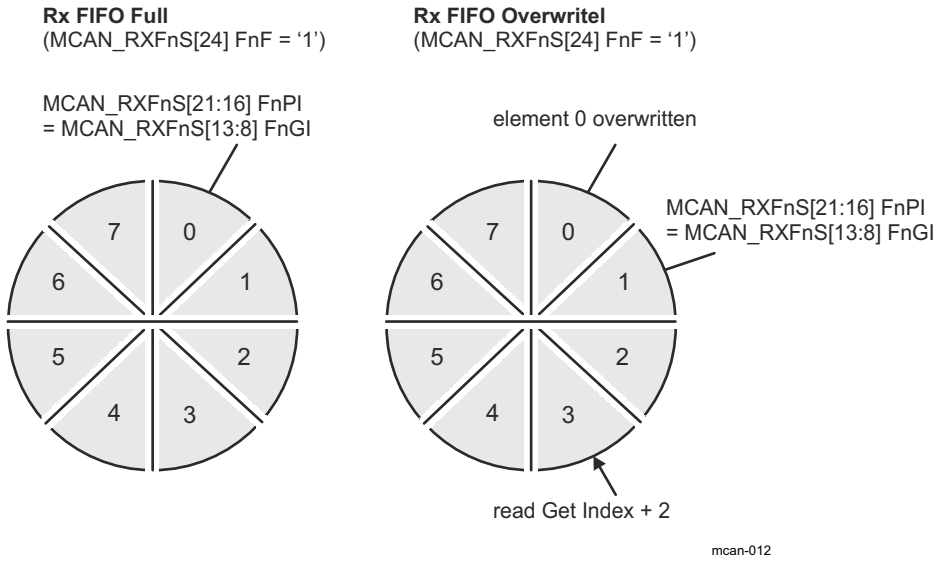


Figure 29-14. Rx FIFO Overflow Handling

After reading from the Rx FIFO, the number of the last element read has to be written to the Rx FIFO Acknowledge Index MCAN_RXFnA[5:0] FnAI. This increments the get index to that element number. In case the Put index has not been incremented to this Rx FIFO element, the Rx FIFO full condition is reset (MCAN_RXFnS[24] FnF = 0).

29.5.13.3 Dedicated Rx Buffers

The MCAN supports up to 64 dedicated Rx buffers. The start address of the Rx buffers section in the Message RAM is configured by way of the MCAN_RXBC.RBSA field. To store in an Rx Buffer a Standard or Extended Message ID Filter Element with SFEC/EFEC = 111 and SFID2/EFID2[10:9] = 00 has to be configured (see Section 29.5.16.5 and Section 29.5.16.6).

After a received message has been accepted by a filter element, the message is stored into the Rx Buffer in the Message RAM referenced by the filter element (the format is the same as for an Rx FIFO element). In addition, the flag MCAN_IR.DRX (message stored in Dedicated Rx Buffer) is set.

Table 29-7 shows an example filter configuration for Rx buffers.

Table 29-7. Example Filter Configuration for Rx Buffers

Filter Element	SFID1[10:0] EFID1[28:0]	SFID2[10:9] EFID2[10:9]	SFID2[5:0] EFID2[5:0]
0	ID message 1	00	00 0000
1	ID message 2	00	00 0001
2	ID message 3	00	00 0010

After the last word of a matching received message has been written to the Message RAM, the respective New Data flag in register MCAN_NDAT1/MCAN_NDAT1 is set. As long as the New Data flag is set, the respective Rx Buffer is locked against updates from received matching frames. The New Data flags have to be reset by the Host CPU by writing a 1 to the respective bit position.

While an Rx buffer's New Data flag is set, a Message ID Filter Element referencing this specific Rx Buffer will not match, causing the acceptance filtering to continue. Following Message ID Filter Elements may cause the received message to be stored into another Rx Buffer, or into an Rx FIFO, or the message may be rejected, depending on filter configuration.

29.5.13.3.1 Rx Buffer Handling

Rx Buffer Handling include the following steps:

- Reset interrupt flag MCAN_IR.DRX
- Read New Data registers
- Read messages from Message RAM
- Reset New Data flags of processed messages

29.5.14 Tx Handling

The Tx handler is used to handle the Tx requests. It controls the transfer of transmit messages from the dedicated Tx buffers, the Tx FIFO, and the Tx Queue to the CAN Core, the Tx Event FIFO, and the Put and Get Index operations. The MCAN module supports up to 32 Tx buffers. These Tx buffers can be configured as dedicated Tx buffers, Tx FIFO, or Tx Queue and as combination of dedicated Tx buffers/Tx FIFO or dedicated Tx buffers/Tx Queue. For each Tx Buffer element Classical CAN or CAN FD transmission mode can be configured. Section 29.5.16.3 describes the Tx Buffer Element. Table 29-8 shows the possible configurations for message transmission.

Table 29-8. Possible Configurations for Message Transmission

MCAN_CCCR Register		Tx Buffer Element		Frame Transmission
BRSE	FDOE	FDF	BRS	
ignored	0	ignored	ignored	Classic CAN
0	1	0	ignored	Classic CAN
0	1	1	ignored	CAN FD without bit rate switching
1	1	0	ignored	Classic CAN
1	1	1	0	CAN FD without bit rate switching
1	1	1	1	CAN FD with bit rate switching

When the Tx Buffer Request Pending (MCAN_TXBRP) register is updated, or when a transmission has been started the Tx Handler starts scanning to check for the highest priority pending Tx request. The Tx Buffer with the lowest Message ID has highest priority.

Note

AUTOSAR requires at least three Tx Queue buffers and support of transmit cancellation.

29.5.14.1 Transmit Pause

The transmit pause feature is intended for use in CAN networks where the CAN Message IDs are specific and cannot easily be changed. These Message IDs may have a higher priority than other defined Message IDs, while in a specific application their relative priority should be inverse. This allows for a case where one ECU sends a burst of CAN messages that cause another ECU's CAN messages to be delayed (paused).

The transmit pause feature is enabled by the MCAN_CCCR.TXP bit. By default this bit is disabled (MCAN_CCCR.TXP = 0). Each time after successfully transmitted message, a pause for two CAN bit times occurs before the start of the next transmission. This allows the other CAN nodes in the network to transmit messages even if their Message IDs have lower priority.

29.5.14.2 Dedicated Tx Buffers

Dedicated Tx buffers are intended for message transmission under complete control of the Host CPU.

There are two options:

- Each dedicated Tx Buffer is configured with a specific Message ID.
- Two or more dedicated Tx buffers are configured with the same Message ID. In this case the Tx Buffer with the lowest buffer number is transmitted first.

After the data section has been updated, a transmission is requested by an Add Request. This is done using the MCAN_TXBAR[x]ARn bit (where x = 0 - 31). The requested messages arbitrate internally with messages from an optional Tx FIFO or Tx Queue and externally with messages on the CAN bus, and are sent out according to their Message ID.

Table 29-9 shows Tx Buffer/Tx FIFO/Tx Queue Element Size. A Dedicated Tx Buffer allocates element size 32-bit words in the Message RAM. The start address of a dedicated Tx Buffer in the Message RAM is calculated by adding transmit buffer index from 0 to 31 (MCAN_TXFQS.TFQP) × Element size to the Tx Buffer start address MCAN_TXBC.TBSA field.

Table 29-9. Tx Buffer/Tx FIFO/Tx Queue Element Size

MCAN_TXESC.TBDS	Data Field [bytes]	Element Size [RAM words]
000	8	4
001	12	5
010	16	6
011	20	7
100	24	8
101	32	10
110	48	14
111	64	18

29.5.14.3 Tx FIFO

Tx FIFO mode is configured by setting bit MCAN_TXBC.TFQM = 0. The stored in the Tx FIFO messages are transmitted starting with the message referenced by the Get Index MCAN_TXFQS.TFGI field. After each transmission the Get Index is incremented until the Tx FIFO is empty. The Tx FIFO Free Level MCAN_TXFQS.TFFL field indicates the number of the available free Tx FIFO elements. The Tx FIFO allows transmission of messages with the same Message ID from different Tx buffers in the order these messages have been written to the Tx FIFO.

New transmit messages have to be written to the Tx FIFO starting with the Tx Buffer referenced by the Put Index MCAN_TXFQS.TFQP field. After each Add Request (MCAN_TXBAR[x] ARn = 1) the Put Index is incremented to the next free Tx FIFO element. When the Put Index reaches the Get Index (MCAN_TXFQS.TFQP = MCAN_TXFQS.TFGI), Tx FIFO Full condition is signaled by bit MCAN_TXFQS.TFQF = 1. In this case no further messages should be written to the Tx FIFO until the next message has been transmitted and the Get Index has been incremented.

The number of requested Tx buffers should not exceed the number of free Tx buffers as indicated by the Tx FIFO Free Level MCAN_TXFQS.TFFL field.

In case a transmission request for the Tx Buffer referenced by the Get Index is canceled, the Get Index is incremented to the next Tx Buffer with pending transmission request and the Tx FIFO Free Level MCAN_TXFQS.TFFL field is recalculated. In case transmission cancellation is applied to any other Tx Buffer - the Get Index and the FIFO Free Level remain unchanged.

A Tx FIFO element allocates element size 32-bit words in the Message RAM (see Table 29-9). The start address of the next available (free) Tx FIFO Buffer is calculated by adding Tx FIFO/Queue Put Index MCAN_TXFQS.TFQP (from 0 to 31) × Element Size to the Tx Buffer Start Address MCAN_TXBC.TBSA field.

29.5.14.4 Tx Queue

Tx Queue mode is configured by setting bit MCAN_TXBC.TFQM = 1. The stored in the Tx Queue messages are transmitted starting with the highest priority message (lowest Message ID). In case two or more Queue buffers are configured with the same Message ID, the Queue Buffer with the lowest buffer number is transmitted first.

New transmit messages have to be written to the Tx FIFO starting with the Tx Buffer referenced by the Put Index MCAN_TXFQS.TFQP field. Each Add Request cyclically increments the Put Index to the next free Tx Buffer. In case of Tx Queue Full condition (MCAN_TXFQS.TFQF = 1), the Put Index is not valid and no further message should be written to the Tx Queue until at least one of the requested messages has been sent out or a pending transmission request has been canceled.

The application may use the MCAN_TXBRP register instead of the Put Index and may place messages to any Tx Buffer without pending transmission request.

A Tx Queue Buffer allocates element size 32-bit words in the Message RAM (see Table 29-9). The start address of the next available (free) Tx Queue Buffer is calculated by adding Tx FIFO/Queue Put Index MCAN_TXFQS.TFQP (from 0 to 31) × Element Size to the Tx Buffer Start Address MCAN_TXBC.TBSA field.

29.5.14.5 Mixed Dedicated Tx Buffers/Tx FIFO

For this combination the Tx buffers section in the Message RAM is separated in two parts:

- Dedicated Tx buffers: the number of Dedicated Tx buffers is configured by the MCAN_TXBC.NDTB field
- Tx FIFO: the number of Tx buffers assigned to the Tx FIFO is configured by the MCAN_TXBC.TFQS field

If the MCAN_TXBC.TFQS field is empty (zero) - only Dedicated Tx buffers are used.

Tx prioritization:

- Scan Dedicated Tx buffers and oldest pending Tx FIFO Buffer (referenced by the MCAN_TXFQS.TFGI field)
- Buffer with lowest Message ID gets highest priority and is transmitted next

Figure 29-15 shows Mixed Dedicated Tx buffers/Tx FIFO example.

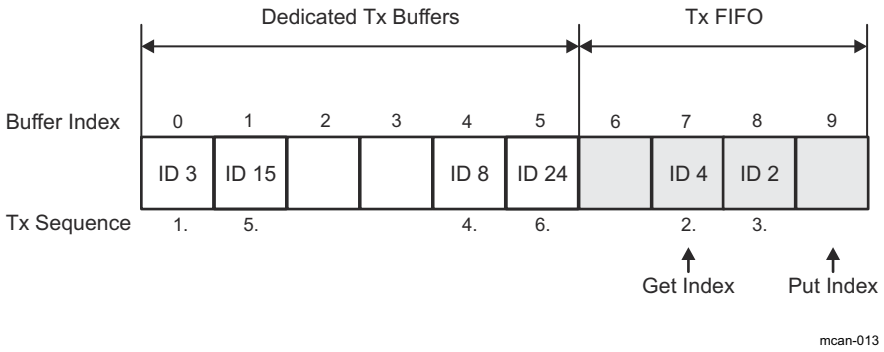


Figure 29-15. Mixed Dedicated Tx Buffers /Tx FIFO (example)

29.5.14.6 Mixed Dedicated Tx Buffers/Tx Queue

For this combination the Tx buffers section in the Message RAM is separated in two parts:

- Dedicated Tx buffers: the number of Dedicated Tx buffers is configured by the MCAN_TXBC.NDTB field
- Tx Queue: the number of Tx buffers assigned to the Tx Queue is configured by the MCAN_TXBC.TFQS field

If MCAN_TXBC.TFQS field is empty (zero) - only Dedicated Tx buffers are used.

Tx prioritization:

- Scan all Tx buffers with activated transmission request
- Tx Buffer with lowest Message ID gets highest priority and is transmitted next

Figure 29-16 shows Mixed Dedicated Tx buffers/Tx Queue example.

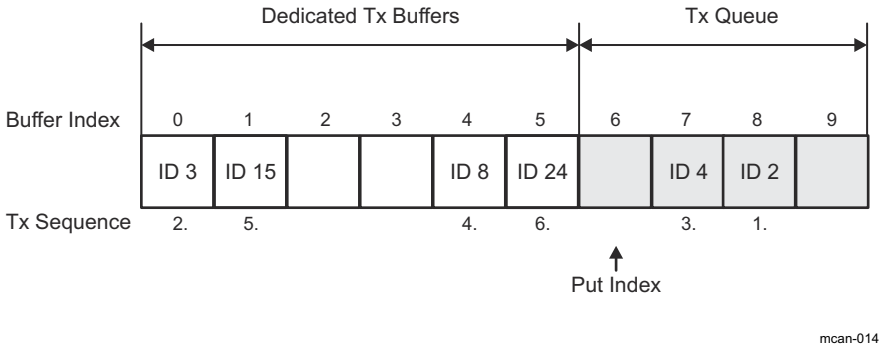


Figure 29-16. Mixed Dedicated Tx Buffers /Tx Queue (example)

29.5.14.7 Transmit Cancellation

This feature is especially intended for gateway and AUTOSAR based applications. The Host CPU can cancel a requested transmission from a dedicated Tx Buffer or a Tx Queue Buffer by setting bit MCAN_TXBCR[n] CRn = 1 (where n = 0 - 31). The corresponding bit position n is equivalent to the number of the Tx buffer.

Transmit cancellation is not intended for Tx FIFO operation.

Successful cancellation is signaled by setting the corresponding bit of the MCAN_TXBCF register (MCAN_TXBCF[n] CFn = 1).

If transmission from a Tx Buffer is already ongoing and a transmit cancellation is requested, the corresponding MCAN_TXBRP[n] TRPn bit remains set as long as the transmission is in progress. If the transmission was successful, the corresponding MCAN_TXBTO[n] TOn and MCAN_TXBCF[n] CFn bits are set. If the transmission was not successful, only the corresponding bit MCAN_TXBCF[n] CFn = 1.

Note

If pending transmission is canceled immediately before this transmission could have been started, a short time window occurs where no transmission is started even if another message is also pending in this node. This may enable another node to transmit a message that may have a lower priority than the second message in this node.

29.5.14.8 Tx Event Handling

To support Tx Event Handling, the Message RAM has implemented a Tx Event FIFO section. Up to 32 Tx Event FIFO elements can be configured. Section 29.5.16.4 describes the Tx Event FIFO element. After message transmission on the CAN bus, Message ID and Timestamp are stored in a Tx Event FIFO element. To link a Tx Event to a Tx Event FIFO element, the Message Marker from the transmitted Tx Buffer is copied into the Tx Event FIFO element.

A Tx Event FIFO full condition is signaled by the MCAN_IR.TEFF bit. In this case no further elements are written to the Tx Event FIFO until at least one element has been read out and the Tx Event FIFO Get Index has been incremented (MCAN_TXEFS.EFGI). In case a Tx Event occurs while the Tx Event FIFO is full, this event is rejected and interrupt flag MCAN_IR.TEFL bit is set.

The Tx Event FIFO watermark can be configured to avoid a Tx Event FIFO overflow. When the Tx Event FIFO fill level reaches the Tx Event FIFO watermark configured by the MCAN_TXEFC.EFWM field, interrupt flag MCAN_IR.TEFW is set. When reading from the Tx Event FIFO, two times the Tx Event FIFO Get Index MCAN_TXEFS.EFGI field has to be added to the Tx Event FIFO start address MCAN_TXEFC.EFSA field.

29.5.15 FIFO Acknowledge Handling

The Get Indices of the two Rx FIFOs (Rx FIFO 0 or Rx FIFO 1) and the Tx Event FIFO are controlled by writing to the corresponding FIFO Acknowledge Index (see MCAN_RXF0A, MCAN_RXF1A, and MCAN_TXEFA). Writing to the FIFO Acknowledge Index will set the FIFO Get Index to the FIFO Acknowledge Index plus one and thereby updates the FIFO Fill Level.

There are two use cases:

- A single element has been read from the FIFO: the Get Index value is written to the FIFO Acknowledge Index.
- A sequence of elements has been read from the FIFO: the Get Index value (Index of the last element read) is written to the FIFO Acknowledge Index at the end of that read sequence.

The Host CPU has free access to the Message RAM. The special care has to be taken when reading FIFO elements in an arbitrary order (Get Index not considered). This can be useful when reading a High Priority Message from one of the two Rx FIFOs. In this case the FIFO's Acknowledge Index should not be written because this would set the Get Index to a wrong position and also changes the FIFO's Fill Level. In this case some of the older FIFO elements would be lost.

Note

The application has to ensure that a valid value is written to the FIFO Acknowledge Index. The MCAN module does not check for erroneous values.

29.5.16 Message RAM

The MCAN module has a Message RAM. The main purpose of the Message RAM is to store:

- Received Messages
- Transmit Messages
- Tx Event Elements
- Message ID Filter Elements

29.5.16.1 Message RAM Configuration

The MCAN module is configured to allocate up to 2048 words in the Message RAM. The Message RAM has a width of 32 bits.

The address range of the Message RAM is from 0x0005 8000 to 0x0005 9FFF.

The Message RAM is capable to include each of the sections listed in Figure 29-17. It is not necessary to configure each of the sections (a section in the Message RAM may be 0) and there is no restriction with respect to the sequence of the sections. For parity checking or ECC, a respective number of bits has to be added to each word. When the MCAN module addresses the Message RAM, it addresses 32-bit words. The start addresses are configurable and they are 32-bit word addresses.

The element size can be configured for:

- Rx FIFO 0, by way of the MCAN_RXESC.F0DS field
- Rx FIFO, 1 by way of the MCAN_RXESC.F1DS field
- Rx buffers, by way of the MCAN_RXESC.RBDS field
- Tx buffers, by way of the MCAN_TXESC.TBDS field

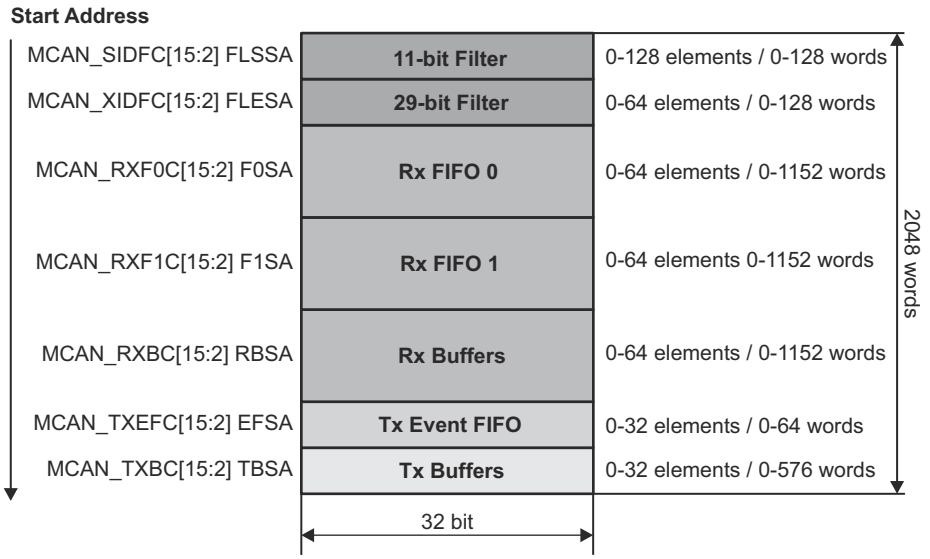


Figure 29-17. Message RAM Configuration

The Host CPU configures the following information in the Message RAM:

- Start addresses of the memory sections
- Number of elements in each section
- The size of the elements in some sections

Note

The MCAN module does not check for errors in the Message RAM configuration. The configuration of the start addresses of the different sections and the number of elements of each section has to be done carefully. This prevents falsification or loss of data.

29.5.16.2 Rx Buffer and FIFO Element

Up to 64 Rx buffers and two Rx FIFOs can be configured in the Message RAM. Each Rx FIFO section can be configured to store up to 64 received messages. The element size can be configured for storage of CAN FD messages with up to 64 bytes data field by way of the MCAN_RXESC register.

Figure 29-18 shows Rx Buffer/Rx FIFO element structure.

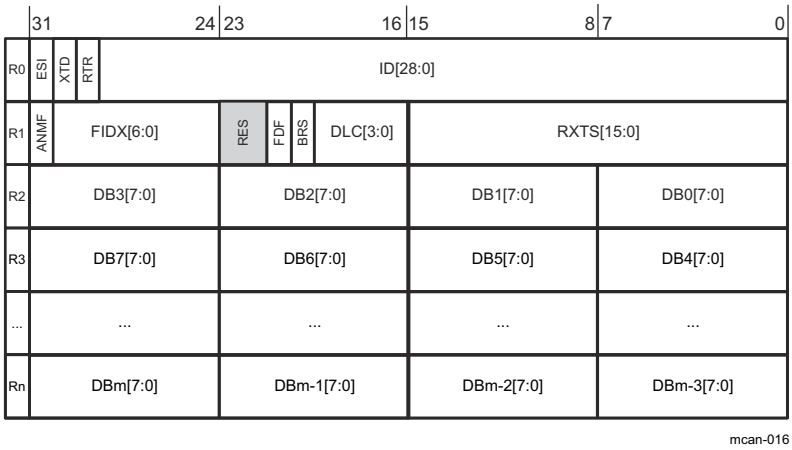


Figure 29-18. Rx Buffer/Rx FIFO Element Structure

Table 29-10 shows Rx Buffer/Rx FIFO element field descriptions.

Table 29-10. Rx Buffer/Rx FIFO Element Field Descriptions

Word	Bits	Field Name	Description
R0	31	ESI	Error State Indicator • 0x0: Transmitting node is error active • 0x1: Transmitting node is error passive
	30	XTD	Extended Identifier Signals to the Host CPU whether the received frame has a standard or extended identifier. • 0x0: 11-bit standard identifier • 0x1: 29-bit extended identifier
	29	RTR	Remote Transmission Request Signals to the Host CPU whether the received frame is a data frame or a remote frame. • 0x0: Received frame is a data frame • 0x1: Received frame is a remote frame **Note:** There are no remote frames in CAN FD format. In case a CAN FD frame was received (FDF = 1), RTR bit reflects the state of the reserved r1 bit (RES[23]). In CAN FD frames (FDF=1), the dominant RRS (Remote Request Substitution) bit replaces the RTR (Remote Transmission Request) bit.
	28:0	ID[28:0]	Identifier Standard or extended identifier depending on XTD bit. A standard identifier is stored into ID[28:18].

Table 29-10. Rx Buffer/Rx FIFO Element Field Descriptions (continued)

Word	Bits	Field Name	Description
R1	31	ANMF	Accepted Non-matching Frame Acceptance of non-matching frames may be enabled using the MCAN_GFC.ANFS and MCAN_GFC.ANFE fields. • 0x0: Received frame matching filter index FIDX field • 0x1: Received frame did not match any Rx filter element
	30:24	FIDX[6:0]	Filter Index 0x0-0x7F (0-127): Index of matching Rx acceptance filter element (invalid if ANMF = 1). Range is 0 to MCAN_SIDFC.LSS - 1 respectively MCAN_XIDFC.LSE - 1.
	23:22	RES	Reserved
	21	FDF	FD Format • 0x0: Standard frame format • 0x1: CAN FD frame format (new DLC-coding and CRC)
	20	BRS	Bit Rate Switch • 0x0: Frame received without bit rate switching • 0x1: Frame received with bit rate switching
	19:16	DLC[3:0]	Data Length Code • 0x0-0x8 (0-8): CAN + CAN FD: received frame has 0-8 data bytes • 0x9-0xF (9-15): CAN: received frame has 8 data bytes • 0x9-0xF (9-15): CAN FD: received frame has 12/16/20/24/32/48/64 data bytes
	15:0	RXTS[15:0]	Rx Timestamp Timestamp Counter value captured on start of frame reception. Resolution depending on configuration of the Timestamp Counter Prescaler MCAN_TSCC.TCP.
R2	31:24	DB3[7:0]	Data Byte 3
	23:16	DB2[7:0]	Data Byte 2
	15:8	DB1[7:0]	Data Byte 1
	7:0	DB0[7:0]	Data Byte 0
R3	31:24	DB7[7:0]	Data Byte 7
	23:16	DB6[7:0]	Data Byte 6
	15:8	DB5[7:0]	Data Byte 5
	7:0	DB4[7:0]	Data Byte 4
...	...	...	...
Rn	31:24	DBm[7:0]	Data Byte m
	23:16	DBm-1[7:0]	Data Byte m-1
	15:8	DBm-2[7:0]	Data Byte m-2
	7:0	DBm-3[7:0]	Data Byte m-3

Note: Depending on the configuration of the element size (MCAN_RXESC), between two and sixteen 32-bit words (Rn = 3-17) are used for storage of a CAN message's data field.

29.5.16.3 Tx Buffer Element

The Tx buffers section can be configured to hold dedicated Tx buffers as well as a Tx FIFO/Tx Queue. In case that the Tx buffers section is shared by dedicated Tx buffers and a Tx FIFO/Tx Queue, the dedicated Tx buffers start at the beginning of the Tx buffers section followed by the buffers assigned to the Tx FIFO or Tx Queue. The Tx Handler makes difference between dedicated Tx buffers and Tx FIFO/Tx Queue by way of the MCAN_TXBC.TFQS and MCAN_TXBC.NDTB fields. The element size can be configured for storage of CAN FD messages with up to 64 bytes data field by way of the MCAN_TXESC register.

Figure 29-19 shows Tx Buffer element structure.

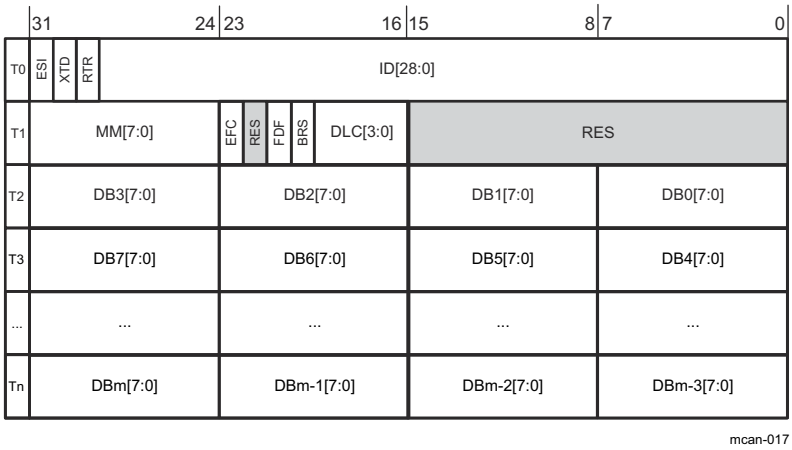


Figure 29-19. Tx Buffer Element Structure

Table 29-11 shows Tx Buffer element field descriptions.

Table 29-11. Tx Buffer Element Field Descriptions

Word	Bits	Field Name	Description
T0	31	ESI	Error State Indicator • 0x0: ESI bit in CAN FD format depends only on error passive flag • 0x1: ESI bit in CAN FD format transmitted recessive **Note:** The ESI bit of the transmit buffer is ORed with the error passive flag to decide the value of the ESI bit in the transmitted CAN FD frame. As required by the CAN FD protocol specification, an error active node can optionally transmit the ESI bit recessive, but an error passive node will always transmit the ESI bit recessive.
	30	XTD	Extended Identifier • 0x0: 11-bit standard identifier • 0x1: 29-bit extended identifier
	29	RTR	Remote Transmission Request • 0x0: Transmit data frame • 0x1: Transmit remote frame **Note:** When RTR = 1, the MCAN module transmits a remote frame according to ISO11898-1:2015, even if the MCAN_CCCR.FDOE bit enables the transmission in CAN FD format.
	28:0	ID[28:0]	Identifier Standard or extended identifier depending on XTD bit. A standard identifier has to be written to ID[28:18].

Table 29-11. Tx Buffer Element Field Descriptions (continued)

Word	Bits	Field Name	Description
T1	31:24	MM[7:0]	Message Marker Written by Host CPU during Tx Buffer configuration. Copied into Tx Event FIFO element for identification of Tx message status (see also MM[7:0] field in Table 29-12).
	23	EFC	Event FIFO Control • 0x0: Don't store Tx events • 0x1: Store Tx events
	22	RES	Reserved
	21	FDF	FD Format • 0x0: Frame transmitted in Classic CAN format • 0x1: Frame transmitted in CAN FD format
	20	BRS	Bit Rate Switch • 0x0: CAN FD frames transmitted without bit rate switching • 0x1: CAN FD frames transmitted with bit rate switching **Note:** ESI, FDF, and BRS bits are only evaluated when CAN FD operation is enabled using the MCAN_CCCR.FDOE bit. BRS bit is only evaluated when MCAN_CCCR.BRSE = 1.
	19:16	DLC[3:0]	Data Length Code • 0x0-0x8 (0-8): CAN + CAN FD: transmit frame has 0-8 data bytes • 0x9-0xF (9-15): CAN: transmit frame has 8 data bytes • 0x9-0xF (9-15): CAN FD: transmit frame has 12/16/20/24/32/48/64 data bytes
	15:0	RES	Reserved
T2	31:24	DB3[7:0]	Data Byte 3
	23:16	DB2[7:0]	Data Byte 2
	15:8	DB1[7:0]	Data Byte 1
	7:0	DB0[7:0]	Data Byte 0
T3	31:24	DB7[7:0]	Data Byte 7
	23:16	DB6[7:0]	Data Byte 6
	15:8	DB5[7:0]	Data Byte 5
	7:0	DB4[7:0]	Data Byte 4
...	...	...	...
Tn	31:24	DBm[7:0]	Data Byte m
	23:16	DBm-1[7:0]	Data Byte m-1
	15:8	DBm-2[7:0]	Data Byte m-2
	7:0	DBm-3[7:0]	Data Byte m-3

Note

Depending on the configuration of the element size (MCAN_TXESC), between two and sixteen 32-bit words (Tn = 3-17) are used for storage of a CAN message's data field.

29.5.16.4 Tx Event FIFO Element

Each element stores information about transmitted messages. By reading the Tx Event FIFO the Host CPU gets this information in the order the messages were transmitted. Status information about the Tx Event FIFO can be obtained from the MCAN_TXEFS register.

Figure 29-20 shows Tx Event FIFO element structure.

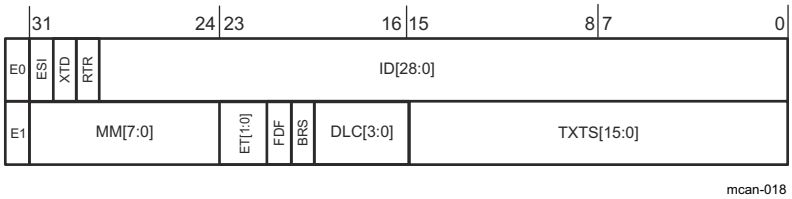


Figure 29-20. Tx Event FIFO Element Structure

Table 29-12 shows Tx Event FIFO element field descriptions.

Table 29-12. Tx Event FIFO Element Field Descriptions

Word	Bits	Field Name	Description
E0	31	ESI	Error State Indicator • 0x0: Transmitting node is error active • 0x1: Transmitting node is error passive
	30	XTD	Extended Identifier • 0x0: 11-bit standard identifier • 0x1: 29-bit extended identifier
	29	RTR	Remote Transmission Request • 0x0: Data frame transmitted • 0x1: Remote frame transmitted
	28:0	ID[28:0]	Identifier Standard or extended identifier depending on XTD bit. A standard identifier has to be written to ID[28:18].

Table 29-12. Tx Event FIFO Element Field Descriptions (continued)

Word	Bits	Field Name	Description
E1	31:24	MM[7:0]	Message Marker Copied from Tx Buffer into Tx Event FIFO element for identification of Tx message status (see also MM[7:0] field in Table 29-11).
	23:22	ET[1:0]	Event Type • 0x0: Reserved • 0x1: Tx event • 0x2: Transmission in spite of cancellation (always set for transmissions in DAR mode) • 0x3: Reserved
	21	FDF	FD Format • 0x0: Standard frame format • 0x1: CAN FD frame format (new DLC-coding and CRC)
	20	BRS	Bit Rate Switch • 0x0: Frame transmitted without bit rate switching • 0x1: Frame transmitted with bit rate switching
	19:16	DLC[3:0]	Data Length Code • 0x0-0x8 (0-8): CAN + CAN FD: frame with 0-8 data bytes transmitted • 0x9-0xF (9-15): CAN: frame with 8 data bytes transmitted • 0x9-0xF (9-15): CAN FD: frame with 12/16/20/24/32/48/64 data bytes transmitted
	15:0	TXTS[15:0]	Tx Timestamp Timestamp Counter value captured on start of frame transmission. Resolution depending on configuration of the Timestamp Counter Prescaler MCAN_TSCC.TCP filed.

29.5.16.5 Standard Message ID Filter Element

Up to 128 filter elements can be configured for 11-bit standard IDs. When accessing a Standard Message ID Filter element, its address is the Filter List Standard Start Address MCAN_SIDFC.FLSSA field plus the index of the filter element (0-127).

Figure 29-21 shows Standard Message ID Filter element structure.

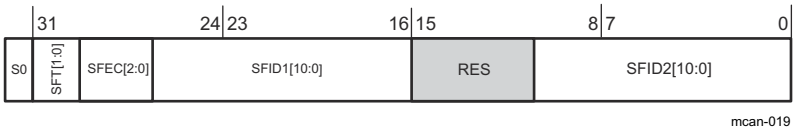


Figure 29-21. Standard Message ID Filter Element Structure

Table 29-13 shows Standard Message ID Filter element field descriptions.

Table 29-13. Standard Message ID Filter Element Field Descriptions

Word	Bits	Field Name	Description
S0	31:30	SFT[1:0]	Standard Filter Type • 0x0: Range filter from SFID1 to SFID2 (SFID2 ≥ SFID1) • 0x1: Dual ID filter for SFID1 or SFID2 • 0x2: Classic filter: SFID1 = filter; SFID2 = mask • 0x3: Filter element disabled **Note:** With SFT = 11 the filter element is disabled and the acceptance filtering continues (same behavior as with SFEC = 000)
	29:27	SFEC[2:0]	Standard Filter Element Configuration All enabled filter elements are used for acceptance filtering of standard frames. Acceptance filtering stops at the first matching enabled filter element or when the end of the filter list is reached. If SFEC = 100, 101, or 110 match sets interrupt flag MCAN_IR.HPM and, if enabled, an interrupt is generated. In this case, the MCAN_HPMS register is updated with the status of the priority match. • 0x0: Disable filter element • 0x1: Store in Rx FIFO 0 if filter matches • 0x2: Store in Rx FIFO 1 if filter matches • 0x3: Reject ID if filter matches • 0x4: Set priority if filter matches • 0x5: Set priority and store in FIFO 0 if filter matches • 0x6: Set priority and store in FIFO 1 if filter matches • 0x7: Store into Rx Buffer , configuration of SFT[1:0] ignored
	26:16	SFID1[10:0]	Standard Filter ID 1 When filtering for Rx buffers this field defines the ID of a standard message to be stored. The received identifiers must match exactly, no masking mechanism is used.
	15:11	RES	Reserved
	10:0	SFID2[10:0]	Standard Filter ID 2 This bit field has a different meaning depending on the configuration of SFEC: • SFEC = 001 - 110: Second ID of standard ID filter element • SFEC = 111: Filter for Rx buffers
		SFID2[10:9]	This filed is decides whether the received message is stored into an Rx Buffer or treated as message A, B, or C of the debug message sequence. • 0x0: Store message into an Rx Buffer • 0x1: Debug Message A • 0x2: Debug Message B • 0x3: Debug Message C **Note:** Debug feature is not supported.
		SFID2[8:6]	This field is used to control the filter event pins at the Extension Interface. A one at the respective bit position enables generation of a pulse at the related filter event pin with the duration of one MCAN_ICKL period in case the filter matches. **Note:** Only two filter event pins are supported.
		SFID2[5:0]	This field defines the offset to the Rx Buffer Start Address MCAN_RXBC.RBSA field for storage of a matching message.

29.5.16.6 Extended Message ID Filter Element

Up to 64 filter elements can be configured for 29-bit extended IDs. When accessing an Extended Message ID Filter element, its address is the Filter List Extended Start Address MCAN_XIDFC.FLESA field plus two times the index of the filter element (0-63).

Figure 29-22 shows Extended Message ID Filter element structure.

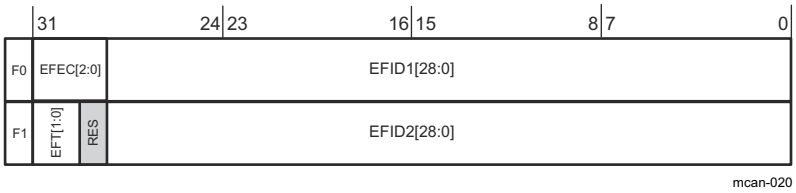


Figure 29-22. Extended Message ID Filter Element Structure

Table 29-14 shows Extended Message ID Filter element field descriptions.

Table 29-14. Extended Message ID Filter Element Field Descriptions

Word	Bits	Field Name	Description
F0	31:29	EFEC[2:0]	Extended Filter Element Configuration All enabled filter elements are used for acceptance filtering of extended frames. Acceptance filtering stops at the first matching enabled filter element or when the end of the filter list is reached. If EFEC = 100, 101, or 110 match sets interrupt flag MCAN_IR.HPM and, if enabled, an interrupt is generated. In this case, the MCAN_HPMS register is updated with the status of the priority match. • 0x0: Disable filter element • 0x1: Store in Rx FIFO 0 if filter matches • 0x2: Store in Rx FIFO 1 if filter matches • 0x3: Reject ID if filter matches • 0x4: Set priority if filter matches • 0x5: Set priority and store in FIFO 0 if filter matches • 0x6: Set priority and store in FIFO 1 if filter matches • 0x7: Store into Rx Buffer or as debug message, configuration of EFT[1:0] ignored
	28:0	EFID1[28:0]	Extended Filter ID 1 First ID of extended ID filter element. When filtering for Rx buffers this field defines the ID of an extended message to be stored. The received identifiers must match exactly, only XIDAM masking mechanism (see Section 29.5.13.1.5) is used.

Table 29-14. Extended Message ID Filter Element Field Descriptions (continued)

Word	Bits	Field Name	Description
F1	31:30	EFT[1:0]	Extended Filter Type • 0x0: Range filter from EFID1 to EFID2 (EFID2 ≥ EFID1) • 0x1: Dual ID filter for EFID1 or EFID2 • 0x2: Classic filter: EFID1 = filter, EFID2 = mask • 0x3: Range filter from EFID1 to EFID2 (EFID2 ≥ EFID1), XIDAM mask not applied
	29	RES	Reserved
	28:0	EFID2[28:0]	Extended Filter ID 2 This bit field has a different meaning depending on the configuration of EFEC: • EFEC = 001 - 110: Second ID of extended ID filter element • EFEC = 111: Filter for Rx buffers
		EFID2[10:9]	This field decides whether the received message is stored into an Rx Buffer or treated as message A, B, or C of the debug message sequence. • 0x0: Store message into an Rx Buffer • 0x1: Debug Message A • 0x2: Debug Message B • 0x3: Debug Message C **Note:** Debug feature is not supported.
		EFID2[8:6]	This field is used to control the filter event pins at the Extension Interface. A one at the respective bit position enables generation of a pulse at the related filter event pin with the duration of one MCAN_ICKL period in case the filter matches. **Note:** Only two filter event pins are supported.
		EFID2[5:0]	This field defines the offset to the Rx Buffer Start Address MCAN_RXBC.RBSA field for storage of a matching message.

29.6 Software

29.6.1 MCAN Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/mcan

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

29.6.1.1 MCAN Internal Loopback with Interrupt

FILE: mcan_ex1_loopback.c

This example shows the MCAN Loopback functionality. The internal loopback mode is entered. The sent message should be received by the node. Use the last address of memory for Rx buffer.

External Connections
- None.

Watch Variables
- error - Checks if there is an error that occurred when the data was sent using internal loopback.

29.6.1.2 MCAN Loopback with Interrupts Example Using SYSCONFIG Tool

FILE: mcan_ex3_loopback_syscfg.c

This example illustrates the MCAN Loopback functionality. The internal loopback mode is entered. The message transmitted would be received by the node. The last address of memory is used for the Rx buffer. Peripheral configuration is done through SYSCONFIG

External Connections
- None.

Watch Variables
- error - Checks if there is an error that occurred when the data was sent using internal loopback.

29.6.1.3 MCAN receive using Rx Buffer

FILE: mcan_ex4_receive.c

This example demonstrates the MCAN receive function. Communication is done between two CAN nodes. The transmitting node could be another MCU or a CAN bus analysis tool capable of transmitting CAN FD frames. The transmit and receive pins of the MCAN module should be connected to a CAN transceiver. Nominal Bit Rate of 500 kbps & Data bit rate of 1 Mbps is used

Only Standard frame with message ID 0x4 is received.

If another C2000 MCU is used as the transmitter, mcan_ex3_transmit.c can be run on it for the transmit function.

Hardware Required
- A C2000 board with CAN transceiver

External Connections
Both nodes should communicate through CAN FD capable transceivers.
- MCAN is on DEVICE_GPIO_PIN_CANRXA (MCANRXA)
- and DEVICE_GPIO_PIN_CANTXA (MCANTXA)

Watch Variables
- rxMsg

29.6.1.4 MCAN External Reception (with mask filter) into RX-FIFO1

FILE: mcan_ex5_mask_filter_receive.c

This example demonstrates Receiving, with mask filter configuration. The transmitting node could be a CAN FD capable controller or a CAN bus analysis tool capable of transmitting CAN FD frames. Bits 0, 1 & 3 of the identifier are masked. So these bits can have any value. This is achieved by using stdFiltelem.sfid1 = 00000001111 and stdFiltelem.sfid2 (mask 0 for X) = 11111110100, which means any message with an ID of 0b0000000X1XX are received and stored into the FIFO. Following STD IDs are received: 0x004, 0x005, 0x006, 0x007, 0x00C, 0x00D, 0x00E, 0x00F. All other IDs are not received. Classic bit-mask filter is used. This example may be used in conjunction with mcan_ex3_transmit.

The transmit and receive pins of the MCAN module should be connected to a CAN Transceiver. Nominal Bit Rate of 500 kbps and Data bit rate of 1 Mbps is used.

Hardware Required

- A C2000 board with CAN transceiver

External Connections

Both nodes should communicate through CAN FD capable transceivers.

- MCAN is on DEVICE_GPIO_PIN_CANRXA (MCANRXA)
- and DEVICE_GPIO_PIN_CANTXA (MCANTXA)

Watch Variables

- rxMsg

29.6.1.5 MCAN Classic frames transmission using Tx Buffer

FILE: mcan_ex7_classic_transmit.c

This simple example shows external communication between the MCAN module and another CAN node. It shows how to transmit classic CAN frames. The GPIOs of MCAN should be connected to a CAN Transceiver. Bit Rate is 500 kbps. Extended Identifier 0x15A5A5A5 is transmitted with 8 data bytes.

Hardware Required

- A C2000 board with CAN transceiver

External Connections

Both nodes should communicate through CAN transceivers.

- MCAN is on DEVICE_GPIO_PIN_CANRXA (MCANRXA)
- and DEVICE_GPIO_PIN_CANTXA (MCANTXA)

Watch Variables

- txMsg

29.6.1.6 MCAN External Reception (with RANGE filter) into RX-FIFO1

FILE: mcan_ex8_range_filter.c

This example demonstrates Receiving, with RANGE filter configuration. The transmitting node could be a CAN FD capable controller or a CAN bus analysis tool capable of transmitting CAN FD frames. Only Extended IDs from 0x1FFFFF23 to 0x1FFFFF46 are received. Other IDs are not received. RXFIFO1 starts at an offset of 748 (2EC). MCAN Message RAM starts at 0x58000, so received messages will be copied starting at address 0x582EC. Note that as long as the ID matches, "classic" CAN frames will also be received.

The transmit and receive pins of the MCAN module should be connected to a CAN Transceiver. Nominal Bit Rate of 500 kbps and Data bit rate of 1 Mbps is used.

Hardware Required

- A C2000 board with CAN transceiver

External Connections

Both nodes should communicate through CAN FD capable transceivers.

- MCAN is on DEVICE_GPIO_PIN_CANRXA (MCANRXA)
- and DEVICE_GPIO_PIN_CANTXA (MCANTXA)

Watch Variables

- rxMsg

29.6.1.7 MCAN External Transmit using Tx Buffer

FILE: mcan_ex9_transmit.c

This example demonstrates the MCAN External Transmit function. External communication is done between two CAN nodes. The receiving node could be another MCU or a CAN bus analysis tool capable of Receiving/ ACKnowledging transmitted frames. The transmit and receive pins of the MCAN module should be connected to a CAN Transceiver. Nominal Bit Rate of 500 kbps and Data bit rate of 1 Mbps is used. Standard Identifier (STD ID) 0x4 is transmitted with 64 data bytes. #defines that are not required for this test case have been commented out. However, they have been left in the code should the scope of this code be expanded to include Receive and FIFO functions.

If another C2000 MCU is used as the receiver, mcan_ex4_receive.c can be run on it for the receive function.

Hardware Required

- A C2000 board with CAN transceiver

External Connections

Both nodes should communicate through CAN FD capable transceivers.

- MCAN is on DEVICE_GPIO_PIN_CANRXA (MCANRXA)
- and DEVICE_GPIO_PIN_CANTXA (MCANTXA)

Watch Variables

- txMsg

29.7 MCAN Registers

This section describes the MCAN module registers.

29.7.1 MCAN Base Address Table

Table 29-15. MCAN Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
McanaSsRegs	MCANASS_REGS	MCANASS_BASE	0x0005_C400	YES	-	YES	-	YES
McanaRegs	MCANA_REGS	MCANA_BASE	0x0005_C600	YES	-	YES	-	YES
McanaErrorRegs	MCANA_ERROR_REGS	MCANA_ERROR_BASE	0x0005_C800	YES	-	YES	-	YES

29.7.2 MCANSS_REGS Registers

Table 29-16 lists the memory-mapped registers for the MCANSS_REGS registers. All register offset addresses not listed in Table 29-16 should be considered as reserved locations and the register contents should not be modified.

Table 29-16. MCANSS_REGS Registers

Offset (x8)	Offset (x16)	Acronym	Register Name	Write Protection	Section
0h	0h	MCANSS_PID	MCAN Subsystem Revision Register		Go
4h	2h	MCANSS_CTRL	MCAN Subsystem Control Register		Go
8h	4h	MCANSS_STAT	MCAN Subsystem Status Register		Go
Ch	6h	MCANSS_ICS	MCAN Subsystem Interrupt Clear Shadow Register		Go
10h	8h	MCANSS_IRS	MCAN Subsystem Interrupt Raw Satus Register		Go
14h	Ah	MCANSS_IECS	MCAN Subsystem Interrupt Enable Clear Shadow Register		Go
18h	Ch	MCANSS_IE	MCAN Subsystem Interrupt Enable Register		Go
1Ch	Eh	MCANSS_IES	MCAN Subsystem Interrupt Enable Status		Go
20h	10h	MCANSS_EOI	MCAN Subsystem End of Interrupt		Go
24h	12h	MCANSS_EXT_TS_PRESCALER	MCAN Subsystem External Timestamp Prescaler 0		Go
28h	14h	MCANSS_EXT_TS_UNSERVICED_INTR_CNTR	MCAN Subsystem External Timestamp Unserviced Interrupts Counter		Go

Complex bit access types are encoded to fit into small table cells. Table 29-17 shows the codes that are used for access types in this section.

Table 29-17. MCANSS_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.

Table 29-17. MCANSS_REGS Access Type Codes (continued)

Access Type	Code	Description
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

29.7.2.1 MCANSS_PID Register (Offset (x8) = 0h, Offset (x16) = 0h) [Reset = 68E05101h]

MCANSS_PID is shown in Figure 29-23 and described in Table 29-18.

Return to the Summary Table.

MCAN Subsystem Revision Register

Figure 29-23. MCANSS_PID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SCHEME		RESERVED		MODULE_ID											
R-1h		R-2h		R-8E0h											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED					MAJOR			RESERVED		MINOR					
R-Ah					R-1h			R-0h		R-1h					

Table 29-18. MCANSS_PID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	SCHEME	R	1h	PID Register Scheme Reset type: SYSRSn
29-28	RESERVED	R	2h	Reserved
27-16	MODULE_ID	R	8E0h	Module Identification Number Reset type: SYSRSn
15-11	RESERVED	R	Ah	Reserved
10-8	MAJOR	R	1h	Major Revision of the MCAN Subsystem Reset type: SYSRSn
7-6	RESERVED	R	0h	Reserved
5-0	MINOR	R	1h	Minor Revision of the MCAN Subsystem Reset type: SYSRSn

29.7.2.2 MCANSS_CTRL Register (Offset (x8) = 4h, Offset (x16) = 2h) [Reset = 8h]

MCANSS_CTRL is shown in Figure 29-24 and described in Table 29-19.

Return to the Summary Table.

MCAN Subsystem Control Register

Figure 29-24. MCANSS_CTRL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	EXT_TS_CNTR_EN	AUTOWAKEUP	WAKEUPREQEN	DBGSUSP_FREE	RESERVED		
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-1h	R-0h		

Table 29-19. MCANSS_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-7	RESERVED	R	0h	Reserved
6	EXT_TS_CNTR_EN	R/W	0h	External Timestamp Counter Enable. 0 External timestamp counter disabled 1 External timestamp counter enabled Reset type: SYSRSn
5	AUTOWAKEUP	R/W	0h	Automatic Wakeup Enable. Enables the MCANSS to automatically clear the MCAN CCCR.INIT bit, fully waking the MCAN up, on an enabled wakeup request. 0 Disable the automatic write to CCCR.INIT 1 Enable the automatic write to CCCR.INIT Reset type: SYSRSn
4	WAKEUPREQEN	R/W	0h	Wakeup Request Enable. Enables the MCANSS to wakeup on CAN RXD activity. 0 Disable wakeup request 1 Enables wakeup request Reset type: SYSRSn
3	DBGSUSP_FREE	R/W	1h	Debug Suspend Free Bit. Enables debug suspend. 0 Disable debug suspend 1 Enable debug suspend Reset type: SYSRSn
2-0	RESERVED	R	0h	Reserved

29.7.2.3 MCANSS_STAT Register (Offset (x8) = 8h, Offset (x16) = 4h) [Reset = X]

MCANSS_STAT is shown in Figure 29-25 and described in Table 29-20.

Return to the Summary Table.

MCAN Subsystem Status Register

Figure 29-25. MCANSS_STAT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					ENABLE_FDOE	MEM_INIT_DONE	RESET
R-0h					R-X	R-0h	R-0h

Table 29-20. MCANSS_STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reserved
2	ENABLE_FDOE	R	X	Flexible Datarate Operation Enable. Determines whether CAN FD operation may be enabled via the MCAN core CCCR.FDOE bit (bit 8) or if only standard CAN operation is possible with this instance of the MCAN. 0 MCAN is only capable of standard CAN communication 1 MCAN may be configured to perform CAN FD communication Reset type: SYSRSn
1	MEM_INIT_DONE	R	0h	Memory Initialization Done. 0 Message RAM initialization is in progress 1 Message RAM is initialized for use Reset type: SYSRSn
0	RESET	R	0h	Soft Reset Status. 0 Not in reset 1 Reset is in progress Reset type: SYSRSn

29.7.2.4 MCANSS_ICS Register (Offset (x8) = Ch, Offset (x16) = 6h) [Reset = 0h]

MCANSS_ICS is shown in Figure 29-26 and described in Table 29-21.

Return to the Summary Table.

MCAN Subsystem Interrupt Clear Shadow Register

Figure 29-26. MCANSS_ICS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EXT_TS_CNTR_OVFL
R-0h							R-0/W1C-0h

Table 29-21. MCANSS_ICS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	EXT_TS_CNTR_OVFL	R-0/W1C	0h	External Timestamp Counter Overflow Interrupt Status Clear. Reads always return a 0. 0 Write of '0' has no effect 1 Write of '1' clears the MCANSS_IRS.EXT_TS_CNTR_OVFL bit Reset type: SYSRSn

29.7.2.5 MCANSS_IRS Register (Offset (x8) = 10h, Offset (x16) = 8h) [Reset = 0h]

MCANSS_IRS is shown in Figure 29-27 and described in Table 29-22.

Return to the Summary Table.

MCAN Subsystem Interrupt Raw Satus Register

Figure 29-27. MCANSS_IRS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EXT_TS_CNTR_OVFL
R-0h							R/W1S-0h

Table 29-22. MCANSS_IRS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	EXT_TS_CNTR_OVFL	R/W1S	0h	External Timestamp Counter Overflow Interrupt Status. This bit is set by HW or by a SW write of '1'. To clear, use the MCANSS_ICS.EXT_TS_CNTR_OVFL bit. 0 External timestamp counter has not overflowed 1 External timestamp counter has overflowed When this bit is set to '1' by HW or SW, the MCANSS_EXT_TS_UNSERVICED_INTR_CNTR.EXT_TS_INTR_CNTR bit field will increment by 1. Reset type: SYSRSn

29.7.2.6 MCANSS_IECS Register (Offset (x8) = 14h, Offset (x16) = Ah) [Reset = 0h]

MCANSS_IECS is shown in Figure 29-28 and described in Table 29-23.

Return to the Summary Table.

MCAN Subsystem Interrupt Enable Clear Shadow Register

Figure 29-28. MCANSS_IECS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EXT_TS_CNTR_OVFL
R-0h							R-0/W1C-0h

Table 29-23. MCANSS_IECS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	EXT_TS_CNTR_OVFL	R-0/W1C	0h	External Timestamp Counter Overflow Interrupt Enable Clear. Reads always return a 0. 0 Write of '0' has no effect 1 Write of '1' clears the MCANSS_IES.EXT_TS_CNTR_OVFL bit Reset type: SYSRSn

29.7.2.7 MCANSS_IE Register (Offset (x8) = 18h, Offset (x16) = Ch) [Reset = 0h]

MCANSS_IE is shown in Figure 29-29 and described in Table 29-24.

Return to the Summary Table.

MCAN Subsystem Interrupt Enable Register

Figure 29-29. MCANSS_IE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EXT_TS_CNTR_OVFL
R-0h							R/W1S-0h

Table 29-24. MCANSS_IE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	EXT_TS_CNTR_OVFL	R/W1S	0h	External Timestamp Counter Overflow Interrupt Enable. A write of '0' has no effect. A write of '1' sets the MCANSS_IES.EXT_TS_CNTR_OVFL bit. Reset type: SYSRSn

29.7.2.8 MCANSS_IES Register (Offset (x8) = 1Ch, Offset (x16) = Eh) [Reset = 0h]

MCANSS_IES is shown in Figure 29-30 and described in Table 29-25.

Return to the Summary Table.

MCAN Subsystem Interrupt Enable Status

Figure 29-30. MCANSS_IES Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EXT_TS_CNTR_OVFL
R-0h							R-0h

Table 29-25. MCANSS_IES Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	EXT_TS_CNTR_OVFL	R	0h	External Timestamp Counter Overflow Interrupt Enable Status. To set, use the CANSS_IE.EXT_TS_CNTR_OVFL bit. To clear, use the MCANSS_IECS.EXT_TS_CNTR_OVFL bit. 0 External timestamp counter overflow interrupt is not enabled 1 External timestamp counter overflow interrupt is enabled Reset type: SYSRSn

29.7.2.9 MCANSS_EOI Register (Offset (x8) = 20h, Offset (x16) = 10h) [Reset = 0h]

MCANSS_EOI is shown in Figure 29-31 and described in Table 29-26.

Return to the Summary Table.

MCAN Subsystem End of Interrupt

Figure 29-31. MCANSS_EOI Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								EOI							
R-0h																								R-0/W1S-0h							

Table 29-26. MCANSS_EOI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	EOI	R-0/W1S	0h	End of Interrupt. A write to this register will clear the associated interrupt. If the unserviced interrupt counter is > 1, another interrupt is generated. 0x00 External TS Interrupt is cleared 0x01 MCAN[0] interrupt is cleared 0x02 MCAN[1] interrupt is cleared Other writes are ignored. Reset type: SYSRSn

29.7.2.10 MCANSS_EXT_TS_PRESCALER Register (Offset (x8) = 24h, Offset (x16) = 12h) [Reset = 0h]

MCANSS_EXT_TS_PRESCALER is shown in Figure 29-32 and described in Table 29-27.

Return to the Summary Table.

MCAN Subsystem External Timestamp Prescaler 0

Figure 29-32. MCANSS_EXT_TS_PRESCALER Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								PRESCALER																							
R-0h								R/W-0h																							

Table 29-27. MCANSS_EXT_TS_PRESCALER Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-0	PRESCALER	R/W	0h	External Timestamp Prescaler Reload Value. The external timestamp count rate is the host (system) clock rate divided by this value, except in the case of 0. A zero value in this bit field will act identically to a value of 0x000001. Reset type: SYSRSn

29.7.2.11 MCANSS_EXT_TS_UNSERVICED_INTR_CNTR Register (Offset (x8) = 28h, Offset (x16) = 14h) [Reset = 0h]

MCANSS_EXT_TS_UNSERVICED_INTR_CNTR is shown in Figure 29-33 and described in Table 29-28.

Return to the Summary Table.

MCAN Subsystem External Timestamp Unserviced Interrupts Counter

Figure 29-33. MCANSS_EXT_TS_UNSERVICED_INTR_CNTR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			EXT_TS_INTR_CNTR				
R-0h			R-0h				

Table 29-28. MCANSS_EXT_TS_UNSERVICED_INTR_CNTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reserved
4-0	EXT_TS_INTR_CNTR	R	0h	External Timestamp Counter Unserviced Rollover Interrupts. If this value is > 1, an MCANSS_EOI write of '1' to bit 0 will issue another interrupt. The status of this bit field is affected by the MCANSS_IRS.EXT_TS_CNTR_OVFL bit field. Reset type: SYSRSn

29.7.3 MCAN_REGS Registers

Table 29-29 lists the memory-mapped registers for the MCAN_REGS registers. All register offset addresses not listed in Table 29-29 should be considered as reserved locations and the register contents should not be modified.

Table 29-29. MCAN_REGS Registers

Offset (x8)	Offset (x16)	Acronym	Register Name	Write Protection	Section
0h	0h	MCAN_CREL	MCAN Core Release Register		Go
4h	2h	MCAN_ENDN	MCAN Endian Register		Go
Ch	6h	MCAN_DBTP	MCAN Data Bit Timing and Prescaler Register		Go
10h	8h	MCAN_TEST	MCAN Test Register		Go
14h	Ah	MCAN_RWD	MCAN RAM Watchdog		Go
18h	Ch	MCAN_CCCR	MCAN CC Control Register		Go
1Ch	Eh	MCAN_NBTP	MCAN Nominal Bit Timing and Prescaler Register		Go
20h	10h	MCAN_TSCC	MCAN Timestamp Counter Configuration		Go
24h	12h	MCAN_TSCV	MCAN Timestamp Counter Value		Go
28h	14h	MCAN_TOCC	MCAN Timeout Counter Configuration		Go
2Ch	16h	MCAN_TOCV	MCAN Timeout Counter Value		Go
40h	20h	MCAN_ECR	MCAN Error Counter Register		Go
44h	22h	MCAN_PSR	MCAN Protocol Status Register		Go
48h	24h	MCAN_TDCR	MCAN Transmitter Delay Compensation Register		Go
50h	28h	MCAN_IR	MCAN Interrupt Register		Go
54h	2Ah	MCAN_IE	MCAN Interrupt Enable		Go
58h	2Ch	MCAN_ILS	MCAN Interrupt Line Select		Go
5Ch	2Eh	MCAN_ILE	MCAN Interrupt Line Enable		Go
80h	40h	MCAN_GFC	MCAN Global Filter Configuration		Go
84h	42h	MCAN_SIDFC	MCAN Standard ID Filter Configuration		Go
88h	44h	MCAN_XIDFC	MCAN Extended ID Filter Configuration		Go
90h	48h	MCAN_XIDAM	MCAN Extended ID and Mask		Go
94h	4Ah	MCAN_HPMS	MCAN High Priority Message Status		Go
98h	4Ch	MCAN_NDAT1	MCAN New Data 1		Go
9Ch	4Eh	MCAN_NDAT2	MCAN New Data 2		Go
A0h	50h	MCAN_RXF0C	MCAN Rx FIFO 0 Configuration		Go
A4h	52h	MCAN_RXF0S	MCAN Rx FIFO 0 Status		Go
A8h	54h	MCAN_RXF0A	MCAN Rx FIFO 0 Acknowledge		Go
ACh	56h	MCAN_RXBC	MCAN Rx Buffer Configuration		Go
B0h	58h	MCAN_RXF1C	MCAN Rx FIFO 1 Configuration		Go
B4h	5Ah	MCAN_RXF1S	MCAN Rx FIFO 1 Status		Go
B8h	5Ch	MCAN_RXF1A	MCAN Rx FIFO 1 Acknowledge		Go
BCh	5Eh	MCAN_RXESC	MCAN Rx Buffer / FIFO Element Size Configuration		Go
C0h	60h	MCAN_TXBC	MCAN Tx Buffer Configuration		Go
C4h	62h	MCAN_TXFQS	MCAN Tx FIFO / Queue Status		Go
C8h	64h	MCAN_TXESC	MCAN Tx Buffer Element Size Configuration		Go
CCh	66h	MCAN_TXBRP	MCAN Tx Buffer Request Pending		Go

Table 29-29. MCAN_REGS Registers (continued)

Offset (x8)	Offset (x16)	Acronym	Register Name	Write Protection	Section
D0h	68h	MCAN_TXBAR	MCAN Tx Buffer Add Request		Go
D4h	6Ah	MCAN_TXBCR	MCAN Tx Buffer Cancellation Request		Go
D8h	6Ch	MCAN_TXBTO	MCAN Tx Buffer Transmission Occurred		Go
DCh	6Eh	MCAN_TXBCF	MCAN Tx Buffer Cancellation Finished		Go
E0h	70h	MCAN_TXBTIE	MCAN Tx Buffer Transmission Interrupt Enable		Go
E4h	72h	MCAN_TXBCIE	MCAN Tx Buffer Cancellation Finished Interrupt Enable		Go
F0h	78h	MCAN_TXEFC	MCAN Tx Event FIFO Configuration		Go
F4h	7Ah	MCAN_TXEFS	MCAN Tx Event FIFO Status		Go
F8h	7Ch	MCAN_TXEFA	MCAN Tx Event FIFO Acknowledge		Go

Complex bit access types are encoded to fit into small table cells. Table 29-30 shows the codes that are used for access types in this section.

Table 29-30. MCAN_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
RC	R C	Read to Clear
RS	R S	Read to Set
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1SQ	W 1S Q	Write 1 to set Qualified. A condition must be met for this operation to occur.
WQ	W Q	Write Qualified. A condition must be met for this operation to occur.
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

29.7.3.1 MCAN_CREL Register (Offset (x8) = 0h, Offset (x16) = 0h) [Reset = 32380608h]

MCAN_CREL is shown in Figure 29-34 and described in Table 29-31.

Return to the Summary Table.

MCAN Core Release Register

Figure 29-34. MCAN_CREL Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
REL				STEP				SUBSTEP				YEAR			
R-3h				R-2h				R-3h				R-8h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MON								DAY							
R-6h								R-8h							

Table 29-31. MCAN_CREL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	REL	R	3h	Core Release. One digit, BCD-coded. Reset type: SYSRSn
27-24	STEP	R	2h	Step of Core Release. One digit, BCD-coded. Reset type: SYSRSn
23-20	SUBSTEP	R	3h	Sub-Step of Core Release. One digit, BCD-coded. Reset type: SYSRSn
19-16	YEAR	R	8h	Time Stamp Year. One digit, BCD-coded. Reset type: SYSRSn
15-8	MON	R	6h	Time Stamp Month. Two digits, BCD-coded. Reset type: SYSRSn
7-0	DAY	R	8h	Time Stamp Day. Two digits, BCD-coded. Reset type: SYSRSn

29.7.3.2 MCAN_ENDN Register (Offset (x8) = 4h, Offset (x16) = 2h) [Reset = 87654321h]

MCAN_ENDN is shown in Figure 29-35 and described in Table 29-32.

Return to the Summary Table.

MCAN Endian Register

Figure 29-35. MCAN_ENDN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ETV																															
R-87654321h																															

Table 29-32. MCAN_ENDN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ETV	R	87654321h	Endianess Test Value. Reading the constant value maintained in this register allows software to determine the endianess of the host CPU. Reset type: SYSRSn

29.7.3.3 MCAN_DBTP Register (Offset (x8) = Ch, Offset (x16) = 6h) [Reset = A33h]

MCAN_DBTP is shown in Figure 29-36 and described in Table 29-33.

Return to the Summary Table.

This register is only writable if bits CCCR.CCE and CCCR.INIT are set. The CAN bit time may be programed in the range of 4 to 49 time quanta. The CAN time quantum may be programmed in the range of 1 to 32 m_can_cclk periods. tq = (DBRP + 1) mtq.
DTSEG1 is the sum of Prop_Seg and Phase_Seg1. DTSEG2 is Phase_Seg2.
Therefore the length of the bit time is (programmed values) [DTSEG1 + DTSEG2 + 3] tq or (functional values) [Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] tq.
The Information Processing Time (IPT) is zero, meaning the data for the next bit is available at the first clock edge after the sample point.

Figure 29-36. MCAN_DBTP Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
TDC	RESERVED		DBRP				
R/WQ-0h	R-0h		R/WQ-0h				
15	14	13	12	11	10	9	8
RESERVED			DTSEG1				
R-0h			R/WQ-Ah				
7	6	5	4	3	2	1	0
DTSEG2				DSJW			
R/WQ-3h				R/WQ-3h			

Table 29-33. MCAN_DBTP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23	TDC	R/WQ	0h	Transmitter Delay Compensation 0 Transmitter Delay Compensation disabled 1 Transmitter Delay Compensation enabled +I107 Reset type: SYSRSn
22-21	RESERVED	R	0h	Reserved
20-16	DBRP	R/WQ	0h	Data Bit Rate Prescaler. The value by which the oscillator frequency is divided for generating the bit time quanta. The bit time is built up from a multiple of this quanta. Valid values for the Bit Rate Prescaler are 0 to 31. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
15-13	RESERVED	R	0h	Reserved
12-8	DTSEG1	R/WQ	Ah	Data Time Segment Before Sample Point. Valid values are 0 to 31. The actual interpretation by the hardware of this value is such that one more than the programmed value is used. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

Table 29-33. MCAN_DBTP Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7-4	DTSEG2	R/WQ	3h	Data Time Segment After Sample Point. Valid values are 0 to 15. The actual interpretation by the hardware of this value is such that one more than the programmed value is used. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
3-0	DSJW	R/WQ	3h	Data Resynchronization Jump Width. Valid values are 0 to 15. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

29.7.3.4 MCAN_TEST Register (Offset (x8) = 10h, Offset (x16) = 8h) [Reset = X]

MCAN_TEST is shown in Figure 29-37 and described in Table 29-34.

Return to the Summary Table.

Write access to the Test Register has to be enabled by setting bit CCCR.TEST to '1'. All Test Register functions are set to their reset values when bit CCCR.TEST is reset.
Loop Back Mode and software control of the internal CAN TX pin are hardware test modes. Programming of TX ≠ "00" may disturb the message transfer on the CAN bus.

Figure 29-37. MCAN_TEST Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RX	TX		LBCK	RESERVED			
R-X	R/WQ-0h		R/WQ-0h	R-0h			

Table 29-34. MCAN_TEST Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	RX	R	X	Receive Pin. Monitors the actual value of the CAN receive pin. 0 The CAN bus is dominant (CAN RX pin = '0') 1 The CAN bus is recessive (CAN RX pin = '1') Reset type: SYSRSn
6-5	TX	R/WQ	0h	Control of Transmit Pin 00 CAN TX pin controlled by the CAN Core, updated at the end of the CAN bit time 01 Sample Point can be monitored at CAN TX pin 10 Dominant ('0') level at CAN TX pin 11 Recessive ('1') at CAN TX pin Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
4	LBCK	R/WQ	0h	Loop Back Mode 0 Reset value, Loop Back Mode is disabled 1 Loop Back Mode is enabled Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
3-0	RESERVED	R	0h	Reserved

29.7.3.5 MCAN_RWD Register (Offset (x8) = 14h, Offset (x16) = Ah) [Reset = 0h]

MCAN_RWD is shown in Figure 29-38 and described in Table 29-35.

Return to the Summary Table.

MCAN RAM Watchdog

Figure 29-38. MCAN_RWD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																WDV								WDC							
R-0h																R-0h								R/WQ-0h							

Table 29-35. MCAN_RWD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	WDV	R	0h	Watchdog Value. Actual Message RAM Watchdog Counter Value. The RAM Watchdog monitors the READY output of the Message RAM. A Message RAM access via the MCAN's Generic Controller Interface starts the Message RAM Watchdog Counter with the value configured by the WDC field. The counter is reloaded with WDC when the Message RAM signals successful completion by activating its READY output. In case there is no response from the Message RAM until the counter has counted down to zero, the counter stops and interrupt flag MCAN_IR.WDI is set. The RAM Watchdog Counter is clocked by the host (system) clock. Reset type: SYSRSn
7-0	WDC	R/WQ	0h	Watchdog Configuration. Start value of the Message RAM Watchdog Counter. With the reset value of "00" the counter is disabled. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

29.7.3.6 MCAN_CCCR Register (Offset (x8) = 18h, Offset (x16) = Ch) [Reset = 1h]

MCAN_CCCR is shown in Figure 29-39 and described in Table 29-36.

Return to the Summary Table.

MCAN CC Control Register

Figure 29-39. MCAN_CCCR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
NISO	TXP	EFBI	PXHD	RESERVED		BRSE	FDOE
R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R-0h		R/WQ-0h	R/WQ-0h
7	6	5	4	3	2	1	0
TEST	DAR	MON	CSR	CSA	ASM	CCE	INIT
R/W1SQ-0h	R/WQ-0h	R/W1SQ-0h	R/W-0h	R-0h	R/W1SQ-0h	R/WQ-0h	R/W-1h

Table 29-36. MCAN_CCCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	NISO	R/WQ	0h	Non ISO Operation. If this bit is set, the MCAN uses the CAN FD frame format as specified by the Bosch CAN FD Specification V1.0. 0 CAN FD frame format according to ISO 11898-1:2015 1 CAN FD frame format according to Bosch CAN FD Specification V1.0 Reset type: SYSRSn
14	TXP	R/WQ	0h	Transmit Pause. If this bit is set, the MCAN pauses for two CAN bit times before starting the next transmission after itself has successfully transmitted a frame. 0 Transmit pause disabled 1 Transmit pause enabled Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
13	EFBI	R/WQ	0h	Edge Filtering during Bus Integration 0 Edge filtering disabled 1 Two consecutive dominant tq required to detect an edge for hard synchronization Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
12	PXHD	R/WQ	0h	Protocol Exception Handling Disable 0 Protocol exception handling enabled 1 Protocol exception handling disabled Note: When protocol exception handling is disabled, the MCAN will transmit an error frame when it detects a protocol exception condition. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
11-10	RESERVED	R	0h	Reserved

Table 29-36. MCAN_CCCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	BRSE	R/WQ	0h	Bit Rate Switch Enable 0 Bit rate switching for transmissions disabled 1 Bit rate switching for transmissions enabled Note: When CAN FD operation is disabled FDOE = '0', BRSE is not evaluated. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
8	FDOE	R/WQ	0h	Flexible Datarate Operation Enable 0 FD operation disabled 1 FD operation enabled Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
7	TEST	R/W1SQ	0h	Test Mode Enable 0 Normal operation, register TEST holds reset values 1 Test Mode, write access to register TEST enabled Qualified Write 1 to Set is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
6	DAR	R/WQ	0h	Disable Automatic Retransmission 0 Automatic retransmission of messages not transmitted successfully enabled 1 Automatic retransmission disabled Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
5	MON	R/W1SQ	0h	Bus Monitoring Mode. Bit MON can only be set by SW when both CCE and INIT are set to '1'. The bit can be reset by SW at any time. 0 Bus Monitoring Mode is disabled 1 Bus Monitoring Mode is enabled Qualified Write 1 to Set is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
4	CSR	R/W	0h	Clock Stop Request 0 No clock stop is requested 1 Clock stop requested. When clock stop is requested, first INIT and then CSA will be set after all pending transfer requests have been completed and the CAN bus reached idle. Reset type: SYSRSn
3	CSA	R	0h	Clock Stop Acknowledge 0 No clock stop acknowledged 1 MCAN may be set in power down by stopping the Host and CAN clocks Reset type: SYSRSn
2	ASM	R/W1SQ	0h	Restricted Operation Mode. Bit ASM can only be set by SW when both CCE and INIT are set to '1'. The bit can be reset by SW at any time. 0 Normal CAN operation 1 Restricted Operation Mode active Qualified Write 1 to Set is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
1	CCE	R/WQ	0h	Configuration Change Enable 0 The CPU has no write access to the protected configuration registers 1 The CPU has write access to the protected configuration registers (while CCCR.INIT = '1') Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

Table 29-36. MCAN_CCCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INIT	R/W	1h	Initialization 0 Normal Operation 1 Initialization is started Note: Due to the synchronization mechanism between the two clock domains, there may be a delay until the value written to INIT can be read back. Therefore the programmer has to assure that the previous value written to INIT has been accepted by reading INIT before setting INIT to a new value. Reset type: SYSRSn

29.7.3.7 MCAN_NBTP Register (Offset (x8) = 1Ch, Offset (x16) = Eh) [Reset = 06000A03h]

MCAN_NBTP is shown in Figure 29-40 and described in Table 29-37.

Return to the Summary Table.

This register is only writable if bits CCCR.CCE and CCCR.INIT are set. The CAN bit time may be programed in the range of 4 to 385 time quanta. The CAN time quantum may be programmed in the range of 1 to 512 m_can_cclk periods. tq = (NBRP + 1) mtq.
NTSEG1 is the sum of Prop_Seg and Phase_Seg1. NTSEG2 is Phase_Seg2.
Therefore the length of the bit time is (programmed values) [NTSEG1 + NTSEG2 + 3] tq or (functional values) [Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2] tq.
The Information Processing Time (IPT) is zero, meaning the data for the next bit is available at the first clock edge after the sample point.
Note: With a CAN clock of 8 MHz, the reset value of 0x06000A03 configures the MCAN for a bit rate of 500 kBit/s.

Figure 29-40. MCAN_NBTP Register

31	30	29	28	27	26	25	24
NSJW							NBRP
R/WQ-3h							R/WQ-0h
23	22	21	20	19	18	17	16
NBRP							
R/WQ-0h							
15	14	13	12	11	10	9	8
NTSEG1							
R/WQ-Ah							
7	6	5	4	3	2	1	0
RESERVED	NTSEG2						
R-0h	R/WQ-3h						

Table 29-37. MCAN_NBTP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	NSJW	R/WQ	3h	Nominal (Re)Synchronization Jump Width. Valid values are 0 to 127. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
24-16	NBRP	R/WQ	0h	Nominal Bit Rate Prescaler. The value by which the oscillator frequency is divided for generating the bit time quanta. The bit time is built up from a multiple of this quanta. Valid values for the Bit Rate Prescaler are 0 to 511. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
15-8	NTSEG1	R/WQ	Ah	Nominal Time Segment Before Sample Point. Valid values are 1 to 255. The actual interpretation by the hardware of this value is such that one more than the programmed value is used. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
7	RESERVED	R	0h	Reserved

Table 29-37. MCAN_NBTP Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-0	NTSEG2	R/WQ	3h	Nominal Time Segment After Sample Point. Valid values are 1 to 127. The actual interpretation by the hardware of this value is such that one more than the programmed value is used. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

29.7.3.8 MCAN_TSCC Register (Offset (x8) = 20h, Offset (x16) = 10h) [Reset = 0h]

MCAN_TSCC is shown in Figure 29-41 and described in Table 29-38.

Return to the Summary Table.

MCAN Timestamp Counter Configuration

Figure 29-41. MCAN_TSCC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED												TCP			
R-0h												R/WQ-0h			
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED														TSS	
R-0h														R/WQ-0h	

Table 29-38. MCAN_TSCC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-20	RESERVED	R	0h	Reserved
19-16	TCP	R/WQ	0h	Timestamp Counter Prescaler. Configures the timestamp and timeout counters time unit in multiples of CAN bit times. Valid values are 0 to 15. The actual interpretation by the hardware of this value is such that one more than the value programmed here is used. Note: With CAN FD an external counter is required for timestamp generation (TSS = "10"). Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
15-2	RESERVED	R	0h	Reserved
1-0	TSS	R/WQ	0h	Timestamp Select 00 Timestamp counter value always 0x0000 01 Timestamp counter value incremented according to TCP 10 External timestamp counter value used 11 Same as "00" Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

29.7.3.9 MCAN_TSCV Register (Offset (x8) = 24h, Offset (x16) = 12h) [Reset = 0h]

MCAN_TSCV is shown in Figure 29-42 and described in Table 29-39.

Return to the Summary Table.

MCAN Timestamp Counter Value

Figure 29-42. MCAN_TSCV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																TSC															
R-0h																R/W-0h															

Table 29-39. MCAN_TSCV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	TSC	R/W	0h	Timestamp Counter. The internal/external Timestamp Counter value is captured on start of frame (both Rx and Tx). When TSCC.TSS = "01", the Timestamp Counter is incremented in multiples of CAN bit times, [1...16], depending on the configuration of TSCC.TCP. A wrap around sets interrupt flag IR.TSW. Write access resets the counter to zero. When TSCC.TSS = "10", TSC reflects the External Timestamp Counter value, and a write access has no impact. Note: A "wrap around" is a change of the Timestamp Counter value from non-zero to zero not caused by write access to MCAN_TSCV. Reset type: SYSRSn

29.7.3.10 MCAN_TOCC Register (Offset (x8) = 28h, Offset (x16) = 14h) [Reset = FFFF0000h]

MCAN_TOCC is shown in Figure 29-43 and described in Table 29-40.

Return to the Summary Table.

MCAN Timeout Counter Configuration

Figure 29-43. MCAN_TOCC Register

31	30	29	28	27	26	25	24
TOP							
R/WQ-FFFFh							
23	22	21	20	19	18	17	16
TOP							
R/WQ-FFFFh							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					TOS		ETOC
R-0h					R/WQ-0h		R/WQ-0h

Table 29-40. MCAN_TOCC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	TOP	R/WQ	FFFFh	Timeout Period. Start value of the Timeout Counter (down-counter). Configures the Timeout Period. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
15-3	RESERVED	R	0h	Reserved
2-1	TOS	R/WQ	0h	Timeout Select. When operating in Continuous mode, a write to TOCV presets the counter to the value configured by TOCC.TOP and continues down-counting. When the Timeout Counter is controlled by one of the FIFOs, an empty FIFO presets the counter to the value configured by TOCC.TOP. Down-counting is started when the first FIFO element is stored. 00 Continuous operation 01 Timeout controlled by Tx Event FIFO 10 Timeout controlled by Rx FIFO 0 11 Timeout controlled by Rx FIFO 1 Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
0	ETOC	R/WQ	0h	Enable Timeout Counter 0 Timeout Counter disabled 1 Timeout Counter enabled Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

29.7.3.11 MCAN_TOCV Register (Offset (x8) = 2Ch, Offset (x16) = 16h) [Reset = FFFFh]

MCAN_TOCV is shown in Figure 29-44 and described in Table 29-41.

Return to the Summary Table.

MCAN Timeout Counter Value

Figure 29-44. MCAN_TOCV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																TOC															
R-0h																R/W-FFFFh															

Table 29-41. MCAN_TOCV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	TOC	R/W	FFFFh	Timeout Counter. The Timeout Counter is decremented in multiples of CAN bit times, [1...16], depending on the configuration of TSCC.TCP. When decremented to zero, interrupt flag IR.TOO is set and the Timeout Counter is stopped. Start and reset/restart conditions are configured via TOCC.TOS. Reset type: SYSRSn

29.7.3.12 MCAN_ECR Register (Offset (x8) = 40h, Offset (x16) = 20h) [Reset = 0h]

MCAN_ECR is shown in Figure 29-45 and described in Table 29-42.

Return to the Summary Table.

MCAN Error Counter Register

Figure 29-45. MCAN_ECR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED								CEL							
R-0h								RC-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RP	REC							TEC							
R-0h	R-0h							R-0h							

Table 29-42. MCAN_ECR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	CEL	RC	0h	CAN Error Logging. The counter is incremented each time when a CAN protocol error causes the Transmit Error Counter or the Receive Error Counter to be incremented. It is reset by read access to CEL. The counter stops at 0xFF the next increment of TEC or REC sets interrupt flag IR.ELO. Note: When CCCR.ASM is set, the CAN protocol controller does not increment TEC and REC when a CAN protocol error is detected, but CEL is still incremented. Reset type: SYSRSn
15	RP	R	0h	Receive Error Passive 0 The Receive Error Counter is below the error passive level of 128 1 The Receive Error Counter has reached the error passive level of 128 Reset type: SYSRSn
14-8	REC	R	0h	Receive Error Counter. Actual state of the Receive Error Counter, values between 0 and 127. Note: When CCCR.ASM is set, the CAN protocol controller does not increment TEC and REC when a CAN protocol error is detected, but CEL is still incremented. Reset type: SYSRSn
7-0	TEC	R	0h	Transmit Error Counter. Actual state of the Transmit Error Counter, values between 0 and 255. Note: When CCCR.ASM is set, the CAN protocol controller does not increment TEC and REC when a CAN protocol error is detected, but CEL is still incremented. Reset type: SYSRSn

29.7.3.13 MCAN_PSR Register (Offset (x8) = 44h, Offset (x16) = 22h) [Reset = 707h]

MCAN_PSR is shown in Figure 29-46 and described in Table 29-43.

Return to the Summary Table.

MCAN Protocol Status Register

Figure 29-46. MCAN_PSR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED	TDCV						
R-0h	R-0h						
15	14	13	12	11	10	9	8
RESERVED	PXE	RFDF	RBRS	RESI	DLEC		
R-0h	RC-0h	RC-0h	RC-0h	RC-0h	RS-7h		
7	6	5	4	3	2	1	0
BO	EW	EP	ACT		LEC		
R-0h	R-0h	R-0h	R-0h		RS-7h		

Table 29-43. MCAN_PSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R	0h	Reserved
22-16	TDCV	R	0h	Transmitter Delay Compensation Value. Position of the secondary sample point, defined by the sum of the measured delay from the internal CAN TX signal to the internal CAN RX signal and TDCR.TDCO. The SSP position is, in the data phase, the number of mtq between the start of the transmitted bit and the secondary sample point. Valid values are 0 to 127 mtq. Reset type: SYSRSn
15	RESERVED	R	0h	Reserved
14	PXE	RC	0h	Protocol Exception Event 0 No protocol exception event occurred since last read access 1 Protocol exception event occurred Reset type: SYSRSn
13	RFDF	RC	0h	Received a CAN FD Message. This bit is set independent of acceptance filtering. 0 Since this bit was reset by the CPU, no CAN FD message has been received 1 Message in CAN FD format with FDF flag set has been received Reset type: SYSRSn
12	RBRS	RC	0h	BRS Flag of Last Received CAN FD Message. This bit is set together with RFDF, independent of acceptance filtering. 0 Last received CAN FD message did not have its BRS flag set 1 Last received CAN FD message had its BRS flag set Reset type: SYSRSn
11	RESI	RC	0h	ESI Flag of Last Received CAN FD Message. This bit is set together with RFDF, independent of acceptance filtering. 0 Last received CAN FD message did not have its ESI flag set 1 Last received CAN FD message had its ESI flag set Reset type: SYSRSn

Table 29-43. MCAN_PSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10-8	DLEC	RS	7h	Data Phase Last Error Code. Type of last error that occurred in the data phase of a CAN FD format frame with its BRS flag set. Coding is the same as for LEC. This field will be cleared to zero when a CAN FD format frame with its BRS flag set has been transferred (reception or transmission) without error. Reset type: SYSRSn
7	BO	R	0h	Bus_Off Status 0 The M_CAN is not Bus_Off 1 The M_CAN is in Bus_Off state Reset type: SYSRSn
6	EW	R	0h	Warning Status 0 Both error counters are below the Error_Warning limit of 96 1 At least one of error counter has reached the Error_Warning limit of 96 Reset type: SYSRSn
5	EP	R	0h	Error Passive 0 The M_CAN is in the Error_Active state. It normally takes part in bus communication and sends an active error flag when an error has been detected 1 The M_CAN is in the Error_Passive state Reset type: SYSRSn
4-3	ACT	R	0h	Node Activity. Monitors the module's CAN communication state. 00 Synchronizing - node is synchronizing on CAN communication 01 Idle - node is neither receiver nor transmitter 10 Receiver - node is operating as receiver 11 Transmitter - node is operating as transmitter Note: ACT is set to "00" by a Protocol Exception Event. Reset type: SYSRSn

Table 29-43. MCAN_PSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2-0	LEC	RS	7h	Last Error Code. The LEC indicates the type of the last error to occur on the CAN bus. This field will be cleared to '0' when a message has been transferred (reception or transmission) without error. 0 No Error: No error occurred since LEC has been reset by successful reception or transmission. 1 Stuff Error: More than 5 equal bits in a sequence have occurred in a part of a received message where this is not allowed. 2 Form Error: A fixed format part of a received frame has the wrong format. 3 AckError: The message transmitted by the MCAN was not acknowledged by another node. 4 Bit1Error: During the transmission of a message (with the exception of the arbitration field), the device wanted to send a recessive level (bit of logical value '1'), but the monitored bus value was dominant. 5 Bit0Error: During the transmission of a message (or acknowledge bit, or active error flag, or overload flag), the device wanted to send a dominant level (data or identifier bit logical value '0'), but the monitored bus value was recessive. During Bus_Off recovery this status is set each time a sequence of 11 recessive bits has been monitored. This enables the CPU to monitor the proceeding of the Bus_Off recovery sequence (indicating the bus is not stuck at dominant or continuously disturbed). 6 CRCError: The CRC check sum of a received message was incorrect. The CRC of an incoming message does not match with the CRC calculated from the received data. 7 NoChange: Any read access to the Protocol Status Register re-initializes the LEC to '7'. When the LEC shows the value '7', no CAN bus event was detected since the last CPU read access to the Protocol Status Register. Note: When a frame in CAN FD format has reached the data phase with BRS flag set, the next CAN event (error or valid frame) will be shown in DLEC instead of LEC. An error in a fixed stuff bit of a CAN FD CRC sequence will be shown as a Form Error, not Stuff Error. Note: The Bus_Off recovery sequence (see ISO 11898-1:2015) cannot be shortened by setting or resetting CCCR.INIT. If the device goes Bus_Off, it will set CCCR.INIT of its own accord, stopping all bus activities. Once CCCR.INIT has been cleared by the CPU, the device will then wait for 129 occurrences of Bus Idle (129 * 11 consecutive recessive bits) before resuming normal operation. At the end of the Bus_Off recovery sequence, the Error Management Counters will be reset. During the waiting time after the resetting of CCCR.INIT, each time a sequence of 11 recessive bits has been monitored, a Bit0Error code is written to PSR.LEC, enabling the CPU to readily check up whether the CAN bus is stuck at dominant or continuously disturbed and to monitor the Bus_Off recovery sequence. ECR.REC is used to count these sequences. Reset type: SYSRSn

29.7.3.14 MCAN_TDCR Register (Offset (x8) = 48h, Offset (x16) = 24h) [Reset = 0h]

MCAN_TDCR is shown in Figure 29-47 and described in Table 29-44.

Return to the Summary Table.

MCAN Transmitter Delay Compensation Register

Figure 29-47. MCAN_TDCR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	TDCO						
R-0h	R/WQ-0h						
7	6	5	4	3	2	1	0
RESERVED	TDCF						
R-0h	R/WQ-0h						

Table 29-44. MCAN_TDCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	RESERVED	R	0h	Reserved
14-8	TDCO	R/WQ	0h	Transmitter Delay Compensation Offset. Offset value defining the distance between the measured delay from the internal CAN TX signal to the internal CAN RX signal and the secondary sample point. Valid values are 0 to 127 mtq. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6-0	TDCF	R/WQ	0h	Transmitter Delay Compensation Filter Window Length. Defines the minimum value for the SSP position, dominant edges on the internal CAN RX signal that would result in an earlier SSP position are ignored for transmitter delay measurement. The feature is enabled when TDCF is configured to a value greater than TDCO. Valid values are 0 to 127 mtq. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

29.7.3.15 MCAN_IR Register (Offset (x8) = 50h, Offset (x16) = 28h) [Reset = 80000000h]

MCAN_IR is shown in Figure 29-48 and described in Table 29-45.

Return to the Summary Table.

The flags are set when one of the listed conditions is detected (edge-sensitive). The flags remain set until the Host clears them. Aflag is cleared by writing a '1' to the corresponding bit position. Writing a '0' has no effect. Ahard reset will clear the register. The configuration of IE controls whether an interrupt is generated. The configuration of ILS controls on which interrupt line an interrupt is signalled.

Figure 29-48. MCAN_IR Register

31	30	29	28	27	26	25	24
RESERVED	RESERVED	ARA	PED	PEA	WDI	BO	EW
R-0h	R-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
23	22	21	20	19	18	17	16
EP	ELO	BEU	RESERVED	DRX	TOO	MRAF	TSW
R/W1C-0h	R/W1C-0h	R/W1C-0h	R-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
15	14	13	12	11	10	9	8
TEFL	TEFF	TEFW	TEFN	TFE	TCF	TC	HPM
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
7	6	5	4	3	2	1	0
RF1L	RF1F	RF1W	RF1N	RF0L	RF0F	RF0W	RF0N
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

Table 29-45. MCAN_IR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved
30	RESERVED	R	0h	Reserved
29	ARA	R/W1C	0h	Access to Reserved Address 0 No access to reserved address occurred 1 Access to reserved address occurred Reset type: SYSRSn
28	PED	R/W1C	0h	Protocol Error in Data Phase (Data Bit Time is used) 0 No protocol error in data phase 1 Protocol error in data phase detected (PSR.DLEC ≠ 0,7) Reset type: SYSRSn
27	PEA	R/W1C	0h	Protocol Error in Arbitration Phase (Nominal Bit Time is used) 0 No protocol error in arbitration phase 1 Protocol error in arbitration phase detected (PSR.LEC ≠ 0,7) Reset type: SYSRSn
26	WDI	R/W1C	0h	Watchdog Interrupt 0 No Message RAM Watchdog event occurred 1 Message RAM Watchdog event due to missing READY Reset type: SYSRSn
25	BO	R/W1C	0h	Bus_Off Status 0 Bus_Off status unchanged 1 Bus_Off status changed Reset type: SYSRSn
24	EW	R/W1C	0h	Warning Status 0 Error_Warning status unchanged 1 Error_Warning status changed Reset type: SYSRSn

Table 29-45. MCAN_IR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23	EP	R/W1C	0h	Error Passive 0 Error_Passive status unchanged 1 Error_Passive status changed Reset type: SYSRSn
22	ELO	R/W1C	0h	Error Logging Overflow 0 CAN Error Logging Counter did not overflow 1 Overflow of CAN Error Logging Counter occurred Reset type: SYSRSn
21	BEU	R/W1C	0h	Bit Error Uncorrected. Message RAM bit error detected, uncorrected. This bit is set when a double bit error is detected by the ECC aggregator attached to the Message RAM. An uncorrected Message RAM bit error sets CCCR.INIT to '1'. This is done to avoid transmission of corrupted data. 0 No bit error detected when reading from Message RAM 1 Bit error detected, uncorrected (e.g. parity logic) Reset type: SYSRSn
20	RESERVED	R	0h	Reserved
19	DRX	R/W1C	0h	Message Stored to Dedicated Rx Buffer. The flag is set whenever a received message has been stored into a dedicated Rx Buffer. 0 No Rx Buffer updated 1 At least one received message stored into an Rx Buffer Reset type: SYSRSn
18	TOO	R/W1C	0h	Timeout Occurred 0 No timeout 1 Timeout reached Reset type: SYSRSn
17	MRAF	R/W1C	0h	Message RAM Access Failure. The flag is set, when the Rx Handler: - has not completed acceptance filtering or storage of an accepted message until the arbitration field of the following message has been received. In this case acceptance filtering or message storage is aborted and the Rx Handler starts processing of the following message. - was not able to write a message to the Message RAM. In this case message storage is aborted. In both cases the FIFO put index is not updated resp. the New Data flag for a dedicated Rx Buffer is not set, a partly stored message is overwritten when the next message is stored to this location. The flag is also set when the Tx Handler was not able to read a message from the Message RAM in time. In this case message transmission is aborted. In case of a Tx Handler access failure the MCAN is switched into Restricted Operation Mode. To leave Restricted Operation Mode, the Host CPU has to reset CCCR.ASM. 0 No Message RAM access failure occurred 1 Message RAM access failure occurred Reset type: SYSRSn
16	TSW	R/W1C	0h	Timestamp Wraparound 0 No timestamp counter wrap-around 1 Timestamp counter wrapped around Reset type: SYSRSn
15	TEFL	R/W1C	0h	Tx Event FIFO Element Lost 0 No Tx Event FIFO element lost 1 Tx Event FIFO element lost, also set after write attempt to Tx Event FIFO of size zero Reset type: SYSRSn
14	TEFF	R/W1C	0h	Tx Event FIFO Full 0 Tx Event FIFO not full 1 Tx Event FIFO full Reset type: SYSRSn

Table 29-45. MCAN_IR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	TEFW	R/W1C	0h	Tx Event FIFO Watermark Reached 0 Tx Event FIFO fill level below watermark 1 Tx Event FIFO fill level reached watermark Reset type: SYSRSn
12	TEFN	R/W1C	0h	Tx Event FIFO New Entry 0 Tx Event FIFO unchanged 1 Tx Handler wrote Tx Event FIFO element Reset type: SYSRSn
11	TFE	R/W1C	0h	Tx FIFO Empty 0 Tx FIFO non-empty 1 Tx FIFO empty Reset type: SYSRSn
10	TCF	R/W1C	0h	Transmission Cancellation Finished 0 No transmission cancellation finished 1 Transmission cancellation finished Reset type: SYSRSn
9	TC	R/W1C	0h	Transmission Completed 0 No transmission completed 1 Transmission completed Reset type: SYSRSn
8	HPM	R/W1C	0h	High Priority Message 0 No high priority message received 1 High priority message received Reset type: SYSRSn
7	RF1L	R/W1C	0h	Rx FIFO 1 Message Lost 0 No Rx FIFO 1 message lost 1 Rx FIFO 1 message lost, also set after write attempt to Rx FIFO 1 of size zero Reset type: SYSRSn
6	RF1F	R/W1C	0h	Rx FIFO 1 Full 0 Rx FIFO 1 not full 1 Rx FIFO 1 full Reset type: SYSRSn
5	RF1W	R/W1C	0h	Rx FIFO 1 Watermark Reached 0 Rx FIFO 1 fill level below watermark 1 Rx FIFO 1 fill level reached watermark Reset type: SYSRSn
4	RF1N	R/W1C	0h	Rx FIFO 1 New Message 0 No new message written to Rx FIFO 1 1 New message written to Rx FIFO 1 Reset type: SYSRSn
3	RF0L	R/W1C	0h	Rx FIFO 0 Message Lost 0 No Rx FIFO 0 message lost 1 Rx FIFO 0 message lost, also set after write attempt to Rx FIFO 0 of size zero Reset type: SYSRSn
2	RF0F	R/W1C	0h	Rx FIFO 0 Full 0 Rx FIFO 0 not full 1 Rx FIFO 0 full Reset type: SYSRSn
1	RF0W	R/W1C	0h	Rx FIFO 0 Watermark Reached 0 Rx FIFO 0 fill level below watermark 1 Rx FIFO 0 fill level reached watermark Reset type: SYSRSn
0	RF0N	R/W1C	0h	Rx FIFO 0 New Message 0 No new message written to Rx FIFO 0 1 New message written to Rx FIFO 0 Reset type: SYSRSn

29.7.3.16 MCAN_IE Register (Offset (x8) = 54h, Offset (x16) = 2Ah) [Reset = 0h]

MCAN_IE is shown in Figure 29-49 and described in Table 29-46.

Return to the Summary Table.

MCAN Interrupt Enable

Figure 29-49. MCAN_IE Register

31	30	29	28	27	26	25	24
RESERVED		ARAE	PEDE	PEAE	WDIE	BOE	EWE
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
EPE	ELOE	BEUE	BECE	DRXE	TOOE	MRAFE	TSWE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
TEFLE	TEFFE	TEFWE	TEFNE	TFEE	TCFE	TCE	HPME
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RF1LE	RF1FE	RF1WE	RF1NE	RF0LE	RF0FE	RF0WE	RF0NE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 29-46. MCAN_IE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29	ARAE	R/W	0h	Access to Reserved Address Enable Reset type: SYSRSn
28	PEDE	R/W	0h	Protocol Error in Data Phase Enable Reset type: SYSRSn
27	PEAE	R/W	0h	Protocol Error in Arbitration Phase Enable Reset type: SYSRSn
26	WDIE	R/W	0h	Watchdog Interrupt Enable Reset type: SYSRSn
25	BOE	R/W	0h	Bus_Off Status Enable Reset type: SYSRSn
24	EWE	R/W	0h	Warning Status Enable Reset type: SYSRSn
23	EPE	R/W	0h	Error Passive Enable Reset type: SYSRSn
22	ELOE	R/W	0h	Error Logging Overflow Enable Reset type: SYSRSn
21	BEUE	R/W	0h	Bit Error Uncorrected Enable Reset type: SYSRSn
20	BECE	R/W	0h	Bit Error Corrected Enable A separate interrupt line reserved for corrected bit errors is provided via the MCAN_ERROR_REGS. It advised for the user to use these registers and leave this bit cleared to '0'. Reset type: SYSRSn
19	DRXE	R/W	0h	Message Stored to Dedicated Rx Buffer Enable Reset type: SYSRSn
18	TOOE	R/W	0h	Timeout Occurred Enable Reset type: SYSRSn
17	MRAFE	R/W	0h	Message RAM Access Failure Enable Reset type: SYSRSn

Table 29-46. MCAN_IE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	TSWE	R/W	0h	Timestamp Wraparound Enable Reset type: SYSRSn
15	TEFLE	R/W	0h	Tx Event FIFO Element Lost Enable Reset type: SYSRSn
14	TEFFE	R/W	0h	Tx Event FIFO Full Enable Reset type: SYSRSn
13	TEFWE	R/W	0h	Tx Event FIFO Watermark Reached Enable Reset type: SYSRSn
12	TEFNE	R/W	0h	Tx Event FIFO New Entry Enable Reset type: SYSRSn
11	TFEE	R/W	0h	Tx FIFO Empty Enable Reset type: SYSRSn
10	TCFE	R/W	0h	Transmission Cancellation Finished Enable Reset type: SYSRSn
9	TCE	R/W	0h	Transmission Completed Enable Reset type: SYSRSn
8	HPME	R/W	0h	High Priority Message Enable Reset type: SYSRSn
7	RF1LE	R/W	0h	Rx FIFO 1 Message Lost Enable Reset type: SYSRSn
6	RF1FE	R/W	0h	Rx FIFO 1 Full Enable Reset type: SYSRSn
5	RF1WE	R/W	0h	Rx FIFO 1 Watermark Reached Enable Reset type: SYSRSn
4	RF1NE	R/W	0h	Rx FIFO 1 New Message Enable Reset type: SYSRSn
3	RF0LE	R/W	0h	Rx FIFO 0 Message Lost Enable Reset type: SYSRSn
2	RF0FE	R/W	0h	Rx FIFO 0 Full Enable Reset type: SYSRSn
1	RF0WE	R/W	0h	Rx FIFO 0 Watermark Reached Enable Reset type: SYSRSn
0	RF0NE	R/W	0h	Rx FIFO 0 New Message Enable Reset type: SYSRSn

29.7.3.17 MCAN_ILS Register (Offset (x8) = 58h, Offset (x16) = 2Ch) [Reset = 0h]

MCAN_ILS is shown in Figure 29-50 and described in Table 29-47.

Return to the Summary Table.

The Interrupt Line Select register assigns an interrupt generated by a specific interrupt flag from the Interrupt Register to one of the two module interrupt lines. For interrupt generation the respective interrupt line has to be enabled via ILE.EINT0 and ILE.EINT1.

Figure 29-50. MCAN_ILS Register

31	30	29	28	27	26	25	24
RESERVED		ARAL	PEDL	PEAL	WDIL	BOL	EWL
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
EPL	ELOL	BEUL	BECL	DRXL	TOOL	MRAFL	TSWL
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
TEFLL	TEFFL	TEFWL	TEFNL	TFEL	TCFL	TCL	HPML
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RF1LL	RF1FL	RF1WL	RF1NL	RF0LL	RF0FL	RF0WL	RF0NL
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 29-47. MCAN_ILS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29	ARAL	R/W	0h	Access to Reserved Address Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
28	PEDL	R/W	0h	Protocol Error in Data Phase Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
27	PEAL	R/W	0h	Protocol Error in Arbitration Phase Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
26	WDIL	R/W	0h	Watchdog Interrupt Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
25	BOL	R/W	0h	Bus_Off Status Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
24	EWL	R/W	0h	Warning Status Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
23	EPL	R/W	0h	Error Passive Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn

Table 29-47. MCAN_ILS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
22	ELOL	R/W	0h	Error Logging Overflow Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
21	BEUL	R/W	0h	Bit Error Uncorrected Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
20	BECL	R/W	0h	Bit Error Corrected Line A separate interrupt line reserved for corrected bit errors is provided via the MCAN_ERROR_REGS. It advised for the user to use these registers and leave the MCAN_IE.BECE bit cleared to '0' (disabled), thereby relegating this bit to not applicable. Reset type: SYSRSn
19	DRXL	R/W	0h	Message Stored to Dedicated Rx Buffer Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
18	TOOL	R/W	0h	Timeout Occurred Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
17	MRAFL	R/W	0h	Message RAM Access Failure Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
16	TSWL	R/W	0h	Timestamp Wraparound Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
15	TEFLL	R/W	0h	Tx Event FIFO Element Lost Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
14	TEFFL	R/W	0h	Tx Event FIFO Full Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
13	TEFWL	R/W	0h	Tx Event FIFO Watermark Reached Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
12	TEFNL	R/W	0h	Tx Event FIFO New Entry Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
11	TFEL	R/W	0h	Tx FIFO Empty Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
10	TCFL	R/W	0h	Transmission Cancellation Finished Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
9	TCL	R/W	0h	Transmission Completed Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn

Table 29-47. MCAN_ILS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	HPML	R/W	0h	High Priority Message Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
7	RF1LL	R/W	0h	Rx FIFO 1 Message Lost Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
6	RF1FL	R/W	0h	Rx FIFO 1 Full Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
5	RF1WL	R/W	0h	Rx FIFO 1 Watermark Reached Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
4	RF1NL	R/W	0h	Rx FIFO 1 New Message Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
3	RF0LL	R/W	0h	Rx FIFO 0 Message Lost Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
2	RF0FL	R/W	0h	Rx FIFO 0 Full Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
1	RF0WL	R/W	0h	Rx FIFO 0 Watermark Reached Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn
0	RF0NL	R/W	0h	Rx FIFO 0 New Message Line 0 Interrupt source is assigned to Interrupt Line 0 1 Interrupt source is assigned to Interrupt Line 1 Reset type: SYSRSn

29.7.3.18 MCAN_ILE Register (Offset (x8) = 5Ch, Offset (x16) = 2Eh) [Reset = 0h]

MCAN_ILE is shown in Figure 29-51 and described in Table 29-48.

Return to the Summary Table.

MCAN Interrupt Line Enable

Figure 29-51. MCAN_ILE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						EINT1	EINT0
R-0h						R/W-0h	R/W-0h

Table 29-48. MCAN_ILE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	EINT1	R/W	0h	Enable Interrupt Line 1 0 Interrupt Line 1 is disabled 1 Interrupt Line 1 is enabled Reset type: SYSRSn
0	EINT0	R/W	0h	Enable Interrupt Line 0 0 Interrupt Line 0 is disabled 1 Interrupt Line 0 is enabled Reset type: SYSRSn

29.7.3.19 MCAN_GFC Register (Offset (x8) = 80h, Offset (x16) = 40h) [Reset = 0h]

MCAN_GFC is shown in Figure 29-52 and described in Table 29-49.

Return to the Summary Table.

MCAN Global Filter Configuration

Figure 29-52. MCAN_GFC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		ANFS		ANFE		RRFS	RRFE
R-0h		R/WQ-0h		R/WQ-0h		R/WQ-0h	R/WQ-0h

Table 29-49. MCAN_GFC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5-4	ANFS	R/WQ	0h	Accept Non-matching Frames Standard. Defines how received messages with 11-bit IDs that do not match any element of the filter list are treated. 00 Accept in Rx FIFO 0 01 Accept in Rx FIFO 1 10 Reject 11 Reject Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
3-2	ANFE	R/WQ	0h	Accept Non-matching Frames Extended. Defines how received messages with 29-bit IDs that do not match any element of the filter list are treated. 00 Accept in Rx FIFO 0 01 Accept in Rx FIFO 1 10 Reject 11 Reject Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
1	RRFS	R/WQ	0h	Reject Remote Frames Standard 0 Filter remote frames with 11-bit standard IDs 1 Reject all remote frames with 11-bit standard IDs Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
0	RRFE	R/WQ	0h	Reject Remote Frames Extended 0 Filter remote frames with 29-bit extended IDs 1 Reject all remote frames with 29-bit extended IDs Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

29.7.3.20 MCAN_SIDFC Register (Offset (x8) = 84h, Offset (x16) = 42h) [Reset = 0h]

MCAN_SIDFC is shown in Figure 29-53 and described in Table 29-50.

Return to the Summary Table.

MCAN Standard ID Filter Configuration

Figure 29-53. MCAN_SIDFC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
LSS							
R/WQ-0h							
15	14	13	12	11	10	9	8
FLSSA							
R/WQ-0h							
7	6	5	4	3	2	1	0
FLSSA						RESERVED	
R/WQ-0h						R-0h	

Table 29-50. MCAN_SIDFC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	LSS	R/WQ	0h	List Size Standard 0 No standard Message ID filter 1-128 Number of standard Message ID filter elements >128 Values greater than 128 are interpreted as 128 Reset type: SYSRSn
15-2	FLSSA	R/WQ	0h	Filter List Standard Start Address. Start address of standard Message ID filter list (32-bit word address). Reset type: SYSRSn
1-0	RESERVED	R	0h	Reserved

29.7.3.21 MCAN_XIDFC Register (Offset (x8) = 88h, Offset (x16) = 44h) [Reset = 0h]

MCAN_XIDFC is shown in Figure 29-54 and described in Table 29-51.

Return to the Summary Table.

MCAN Extended ID Filter Configuration

Figure 29-54. MCAN_XIDFC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED	LSE						
R-0h	R/WQ-0h						
15	14	13	12	11	10	9	8
FLESA							
R/WQ-0h							
7	6	5	4	3	2	1	0
FLESA						RESERVED	
R/WQ-0h						R-0h	

Table 29-51. MCAN_XIDFC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R	0h	Reserved
22-16	LSE	R/WQ	0h	List Size Extended 0 No extended Message ID filter 1-64 Number of extended Message ID filter elements >64 Values greater than 64 are interpreted as 64 Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
15-2	FLESA	R/WQ	0h	Filter List Extended Start Address. Start address of extended Message ID filter list (32-bit word address). Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
1-0	RESERVED	R	0h	Reserved

29.7.3.22 MCAN_XIDAM Register (Offset (x8) = 90h, Offset (x16) = 48h) [Reset = 1FFFFFFFh]

MCAN_XIDAM is shown in Figure 29-55 and described in Table 29-52.

Return to the Summary Table.

MCAN Extended ID and Mask

Figure 29-55. MCAN_XIDAM Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED			EIDM												
R-0h			R/WQ-1FFFFFFFh												
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EIDM															
R/WQ-1FFFFFFFh															

Table 29-52. MCAN_XIDAM Register Field Descriptions

Bit	Field	Type	Reset	Description
31-29	RESERVED	R	0h	Reserved
28-0	EIDM	R/WQ	1FFFFFFFh	Extended ID Mask. For acceptance filtering of extended frames the Extended ID AND Mask is ANDed with the Message ID of a received frame. Intended for masking of 29-bit IDs in SAE J1939. With the reset value of all bits set to one the mask is not active. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

29.7.3.23 MCAN_HPMS Register (Offset (x8) = 94h, Offset (x16) = 4Ah) [Reset = 0h]

MCAN_HPMS is shown in Figure 29-56 and described in Table 29-53.

Return to the Summary Table.

This register is updated every time a Message ID filter element configured to generate a priority event matches. This can be used to monitor the status of incoming high priority messages and to enable fast access to these messages.

Figure 29-56. MCAN_HPMS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
FLST	FIDX						
R-0h	R-0h						
7	6	5	4	3	2	1	0
MSI		BIDX					
R-0h		R-0h					

Table 29-53. MCAN_HPMS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	FLST	R	0h	Filter List. Indicates the filter list of the matching filter element. 0 Standard Filter List 1 Extended Filter List Reset type: SYSRSn
14-8	FIDX	R	0h	Filter Index. Index of matching filter element. Range is 0 to SIDFC.LSS - 1 resp. XIDFC.LSE - 1. Reset type: SYSRSn
7-6	MSI	R	0h	Message Storage Indicator 00 No FIFO selected 01 FIFO message lost 10 Message stored in FIFO 0 11 Message stored in FIFO 1 Reset type: SYSRSn
5-0	BIDX	R	0h	Buffer Index. Index of Rx FIFO element to which the message was stored. Only valid when MSI[1] = '1'. Reset type: SYSRSn

29.7.3.24 MCAN_NDAT1 Register (Offset (x8) = 98h, Offset (x16) = 4Ch) [Reset = 0h]

MCAN_NDAT1 is shown in Figure 29-57 and described in Table 29-54.

Return to the Summary Table.

MCAN New Data 1

Figure 29-57. MCAN_NDAT1 Register

31	30	29	28	27	26	25	24
ND31	ND30	ND29	ND28	ND27	ND26	ND25	ND24
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
23	22	21	20	19	18	17	16
ND23	ND22	ND21	ND20	ND19	ND18	ND17	ND16
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
15	14	13	12	11	10	9	8
ND15	ND14	ND13	ND12	ND11	ND10	ND9	ND8
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
7	6	5	4	3	2	1	0
ND7	ND6	ND5	ND4	ND3	ND2	ND1	ND0
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

Table 29-54. MCAN_NDAT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ND31	R/W1C	0h	New Data RX Buffer 31 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
30	ND30	R/W1C	0h	New Data RX Buffer 30 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
29	ND29	R/W1C	0h	New Data RX Buffer 29 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
28	ND28	R/W1C	0h	New Data RX Buffer 28 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
27	ND27	R/W1C	0h	New Data RX Buffer 27 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
26	ND26	R/W1C	0h	New Data RX Buffer 26 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
25	ND25	R/W1C	0h	New Data RX Buffer 25 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
24	ND24	R/W1C	0h	New Data RX Buffer 24 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn

Table 29-54. MCAN_NDAT1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23	ND23	R/W1C	0h	New Data RX Buffer 23 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
22	ND22	R/W1C	0h	New Data RX Buffer 22 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
21	ND21	R/W1C	0h	New Data RX Buffer 21 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
20	ND20	R/W1C	0h	New Data RX Buffer 20 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
19	ND19	R/W1C	0h	New Data RX Buffer 19 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
18	ND18	R/W1C	0h	New Data RX Buffer 18 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
17	ND17	R/W1C	0h	New Data RX Buffer 17 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
16	ND16	R/W1C	0h	New Data RX Buffer 16 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
15	ND15	R/W1C	0h	New Data RX Buffer 15 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
14	ND14	R/W1C	0h	New Data RX Buffer 14 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
13	ND13	R/W1C	0h	New Data RX Buffer 13 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
12	ND12	R/W1C	0h	New Data RX Buffer 12 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
11	ND11	R/W1C	0h	New Data RX Buffer 11 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
10	ND10	R/W1C	0h	New Data RX Buffer 10 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn

Table 29-54. MCAN_NDAT1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	ND9	R/W1C	0h	New Data RX Buffer 9 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
8	ND8	R/W1C	0h	New Data RX Buffer 8 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
7	ND7	R/W1C	0h	New Data RX Buffer 7 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
6	ND6	R/W1C	0h	New Data RX Buffer 6 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
5	ND5	R/W1C	0h	New Data RX Buffer 5 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
4	ND4	R/W1C	0h	New Data RX Buffer 4 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
3	ND3	R/W1C	0h	New Data RX Buffer 3 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
2	ND2	R/W1C	0h	New Data RX Buffer 2 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
1	ND1	R/W1C	0h	New Data RX Buffer 1 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
0	ND0	R/W1C	0h	New Data RX Buffer 0 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn

29.7.3.25 MCAN_NDAT2 Register (Offset (x8) = 9Ch, Offset (x16) = 4Eh) [Reset = 0h]

MCAN_NDAT2 is shown in Figure 29-58 and described in Table 29-55.

Return to the Summary Table.

MCAN New Data 2

Figure 29-58. MCAN_NDAT2 Register

31	30	29	28	27	26	25	24
ND63	ND62	ND61	ND60	ND59	ND58	ND57	ND56
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
23	22	21	20	19	18	17	16
ND55	ND54	ND53	ND52	ND51	ND50	ND49	ND48
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
15	14	13	12	11	10	9	8
ND47	ND46	ND45	ND44	ND43	ND42	ND41	ND40
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
7	6	5	4	3	2	1	0
ND39	ND38	ND37	ND36	ND35	ND34	ND33	ND32
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h

Table 29-55. MCAN_NDAT2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	ND63	R/W1C	0h	New Data RX Buffer 63 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
30	ND62	R/W1C	0h	New Data RX Buffer 62 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
29	ND61	R/W1C	0h	New Data RX Buffer 61 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
28	ND60	R/W1C	0h	New Data RX Buffer 60 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
27	ND59	R/W1C	0h	New Data RX Buffer 59 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
26	ND58	R/W1C	0h	New Data RX Buffer 58 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
25	ND57	R/W1C	0h	New Data RX Buffer 57 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
24	ND56	R/W1C	0h	New Data RX Buffer 56 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn

Table 29-55. MCAN_NDAT2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23	ND55	R/W1C	0h	New Data RX Buffer 55 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
22	ND54	R/W1C	0h	New Data RX Buffer 54 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
21	ND53	R/W1C	0h	New Data RX Buffer 53 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
20	ND52	R/W1C	0h	New Data RX Buffer 52 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
19	ND51	R/W1C	0h	New Data RX Buffer 51 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
18	ND50	R/W1C	0h	New Data RX Buffer 50 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
17	ND49	R/W1C	0h	New Data RX Buffer 49 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
16	ND48	R/W1C	0h	New Data RX Buffer 48 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
15	ND47	R/W1C	0h	New Data RX Buffer 47 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
14	ND46	R/W1C	0h	New Data RX Buffer 46 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
13	ND45	R/W1C	0h	New Data RX Buffer 45 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
12	ND44	R/W1C	0h	New Data RX Buffer 44 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
11	ND43	R/W1C	0h	New Data RX Buffer 43 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
10	ND42	R/W1C	0h	New Data RX Buffer 42 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn

Table 29-55. MCAN_NDAT2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	ND41	R/W1C	0h	New Data RX Buffer 41 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
8	ND40	R/W1C	0h	New Data RX Buffer 40 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
7	ND39	R/W1C	0h	New Data RX Buffer 39 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
6	ND38	R/W1C	0h	New Data RX Buffer 38 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
5	ND37	R/W1C	0h	New Data RX Buffer 37 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
4	ND36	R/W1C	0h	New Data RX Buffer 36 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
3	ND35	R/W1C	0h	New Data RX Buffer 35 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
2	ND34	R/W1C	0h	New Data RX Buffer 34 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
1	ND33	R/W1C	0h	New Data RX Buffer 33 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn
0	ND32	R/W1C	0h	New Data RX Buffer 32 0 Rx Buffer not updated 1 Rx Buffer updated from new message Reset type: SYSRSn

29.7.3.26 MCAN_RXF0C Register (Offset (x8) = A0h, Offset (x16) = 50h) [Reset = 0h]

MCAN_RXF0C is shown in Figure 29-59 and described in Table 29-56.

Return to the Summary Table.

MCAN Rx FIFO 0 Configuration

Figure 29-59. MCAN_RXF0C Register

31	30	29	28	27	26	25	24
F0OM	F0WM						
R/WQ-0h	R/WQ-0h						
23	22	21	20	19	18	17	16
RESERVED	F0S						
R-0h	R/WQ-0h						
15	14	13	12	11	10	9	8
F0SA							
R/WQ-0h							
7	6	5	4	3	2	1	0
F0SA						RESERVED	
R/WQ-0h						R-0h	

Table 29-56. MCAN_RXF0C Register Field Descriptions

Bit	Field	Type	Reset	Description
31	F0OM	R/WQ	0h	FIFO 0 Operation Mode. FIFO 0 can be operated in blocking or in overwrite mode. 0 FIFO 0 blocking mode 1 FIFO 0 overwrite mode Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
30-24	F0WM	R/WQ	0h	Rx FIFO 0 Watermark 0 Watermark interrupt disabled 1-64 Level for Rx FIFO 0 watermark interrupt (IR.RF0W) >64 Watermark interrupt disabled Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
23	RESERVED	R	0h	Reserved
22-16	F0S	R/WQ	0h	Rx FIFO 0 Size. The Rx FIFO 0 elements are indexed from 0 to F0S-1. 0 No Rx FIFO 0 1-64 Number of Rx FIFO 0 elements >64 Values greater than 64 are interpreted as 64 Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
15-2	F0SA	R/WQ	0h	Rx FIFO 0 Start Address. Start address of Rx FIFO 0 in Message RAM (32-bit word address). Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
1-0	RESERVED	R	0h	Reserved

29.7.3.27 MCAN_RXF0S Register (Offset (x8) = A4h, Offset (x16) = 52h) [Reset = 0h]

MCAN_RXF0S is shown in Figure 29-60 and described in Table 29-57.

Return to the Summary Table.

MCAN Rx FIFO 0 Status

Figure 29-60. MCAN_RXF0S Register

31	30	29	28	27	26	25	24
RESERVED						RF0L	F0F
R-0h						R-0h	R-0h
23	22	21	20	19	18	17	16
RESERVED		F0PI					
R-0h		R-0h					
15	14	13	12	11	10	9	8
RESERVED		F0GI					
R-0h		R-0h					
7	6	5	4	3	2	1	0
RESERVED	F0FL						
R-0h	R-0h						

Table 29-57. MCAN_RXF0S Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved
25	RF0L	R	0h	Rx FIFO 0 Message Lost. This bit is a copy of interrupt flag IR.RF0L. When IR.RF0L is reset, this bit is also reset. 0 No Rx FIFO 0 message lost 1 Rx FIFO 0 message lost, also set after write attempt to Rx FIFO 0 of size zero Note: Overwriting the oldest message when RXF0C.F0OM = '1' will not set this flag. Reset type: SYSRSn
24	F0F	R	0h	Rx FIFO 0 Full 0 Rx FIFO 0 not full 1 Rx FIFO 0 full Reset type: SYSRSn
23-22	RESERVED	R	0h	Reserved
21-16	F0PI	R	0h	Rx FIFO 0 Put Index. Rx FIFO 0 write index pointer, range 0 to 63. Reset type: SYSRSn
15-14	RESERVED	R	0h	Reserved
13-8	F0GI	R	0h	Rx FIFO 0 Get Index. Rx FIFO 0 read index pointer, range 0 to 63. Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6-0	F0FL	R	0h	Rx FIFO 0 Fill Level. Number of elements stored in Rx FIFO 0, range 0 to 64. Reset type: SYSRSn

29.7.3.28 MCAN_RXF0A Register (Offset (x8) = A8h, Offset (x16) = 54h) [Reset = 0h]

MCAN_RXF0A is shown in Figure 29-61 and described in Table 29-58.

Return to the Summary Table.

MCAN Rx FIFO 0 Acknowledge

Figure 29-61. MCAN_RXF0A Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																										F0AI					
R-0h																										R/W-0h					

Table 29-58. MCAN_RXF0A Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5-0	F0AI	R/W	0h	Rx FIFO 0 Acknowledge Index. After the Host has read a message or a sequence of messages from Rx FIFO 0 it has to write the buffer index of the last element read from Rx FIFO 0 to F0AI. This will set the Rx FIFO 0 Get Index RXF0S.F0GI to F0AI + 1 and update the FIFO 0 Fill Level RXF0S.F0FL. Reset type: SYSRSn

29.7.3.29 MCAN_RXBC Register (Offset (x8) = ACh, Offset (x16) = 56h) [Reset = 0h]

MCAN_RXBC is shown in Figure 29-62 and described in Table 29-59.

Return to the Summary Table.

MCAN Rx Buffer Configuration

Figure 29-62. MCAN_RXBC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RBSA							
R/WQ-0h							
7	6	5	4	3	2	1	0
RBSA						RESERVED	
R/WQ-0h						R-0h	

Table 29-59. MCAN_RXBC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-2	RBSA	R/WQ	0h	Rx Buffer Start Address. Configures the start address of the Rx Buffers section in the Message RAM (32-bit word address). +I466 Reset type: SYSRSn
1-0	RESERVED	R	0h	Reserved

29.7.3.30 MCAN_RXF1C Register (Offset (x8) = B0h, Offset (x16) = 58h) [Reset = 0h]

MCAN_RXF1C is shown in Figure 29-63 and described in Table 29-60.

Return to the Summary Table.

MCAN Rx FIFO 1 Configuration

Figure 29-63. MCAN_RXF1C Register

31	30	29	28	27	26	25	24
F1OM	F1WM						
R/WQ-0h	R/WQ-0h						
23	22	21	20	19	18	17	16
RESERVED	F1S						
R-0h	R/WQ-0h						
15	14	13	12	11	10	9	8
F1SA							
R/WQ-0h							
7	6	5	4	3	2	1	0
F1SA						RESERVED	
R/WQ-0h						R-0h	

Table 29-60. MCAN_RXF1C Register Field Descriptions

Bit	Field	Type	Reset	Description
31	F1OM	R/WQ	0h	FIFO 1 Operation Mode. FIFO 1 can be operated in blocking or in overwrite mode. 0 FIFO 1 blocking mode 1 FIFO 1 overwrite mode Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
30-24	F1WM	R/WQ	0h	Rx FIFO 1 Watermark 0 Watermark interrupt disabled 1-64 Level for Rx FIFO 1 watermark interrupt (IR.RF1W) >64 Watermark interrupt disabled Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
23	RESERVED	R	0h	Reserved
22-16	F1S	R/WQ	0h	Rx FIFO 1 Size. The Rx FIFO 1 elements are indexed from 0 to F1S - 1. 0 No Rx FIFO 1 1-64 Number of Rx FIFO 1 elements >64 Values greater than 64 are interpreted as 64 Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
15-2	F1SA	R/WQ	0h	Rx FIFO 1 Start Address Start address of Rx FIFO 1 in Message RAM (32-bit word address). Reset type: SYSRSn
1-0	RESERVED	R	0h	Reserved

29.7.3.31 MCAN_RXF1S Register (Offset (x8) = B4h, Offset (x16) = 5Ah) [Reset = 0h]

MCAN_RXF1S is shown in Figure 29-64 and described in Table 29-61.

Return to the Summary Table.

MCAN Rx FIFO 1 Status

Figure 29-64. MCAN_RXF1S Register

31	30	29	28	27	26	25	24
DMS		RESERVED				RF1L	F1F
R-0h		R-0h				R-0h	R-0h
23	22	21	20	19	18	17	16
RESERVED		F1PI					
R-0h		R-0h					
15	14	13	12	11	10	9	8
RESERVED		F1GI					
R-0h		R-0h					
7	6	5	4	3	2	1	0
RESERVED	F1FL						
R-0h	R-0h						

Table 29-61. MCAN_RXF1S Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	DMS	R	0h	Debug Message Status 00 Idle state, wait for reception of debug messages, DMA request is cleared 01 Debug message A received 10 Debug messages A, B received 11 Debug messages A, B, C received, DMA request is set Reset type: SYSRSn
29-26	RESERVED	R	0h	Reserved
25	RF1L	R	0h	Rx FIFO 1 Message Lost. This bit is a copy of interrupt flag IR.RF1L. When IR.RF1L is reset, this bit is also reset. 0 No Rx FIFO 1 message lost 1 Rx FIFO 1 message lost, also set after write attempt to Rx FIFO 1 of size zero Note: Overwriting the oldest message when RXF1C.F1OM = '1' will not set this flag. Reset type: SYSRSn
24	F1F	R	0h	Rx FIFO 1 Full 0 Rx FIFO 1 not full 1 Rx FIFO 1 full Reset type: SYSRSn
23-22	RESERVED	R	0h	Reserved
21-16	F1PI	R	0h	Rx FIFO 1 Put Index. Rx FIFO 1 write index pointer, range 0 to 63. Reset type: SYSRSn
15-14	RESERVED	R	0h	Reserved
13-8	F1GI	R	0h	Rx FIFO 1 Get Index. Rx FIFO 1 read index pointer, range 0 to 63. Reset type: SYSRSn
7	RESERVED	R	0h	Reserved
6-0	F1FL	R	0h	Rx FIFO 1 Fill Level. Number of elements stored in Rx FIFO 1, range 0 to 64. Reset type: SYSRSn

29.7.3.32 MCAN_RXF1A Register (Offset (x8) = B8h, Offset (x16) = 5Ch) [Reset = 0h]

MCAN_RXF1A is shown in Figure 29-65 and described in Table 29-62.

Return to the Summary Table.

MCAN Rx FIFO 1 Acknowledge

Figure 29-65. MCAN_RXF1A Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																										F1AI					
R-0h																										R/W-0h					

Table 29-62. MCAN_RXF1A Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5-0	F1AI	R/W	0h	Rx FIFO 1 Acknowledge Index. After the Host has read a message or a sequence of messages from Rx FIFO 1 it has to write the buffer index of the last element read from Rx FIFO 1 to F1AI. This will set the Rx FIFO 1 Get Index RXF1S.F1GI to F1AI + 1 and update the FIFO 1 Fill Level RXF1S.F1FL. Reset type: SYSRSn

29.7.3.33 MCAN_RXESC Register (Offset (x8) = BCh, Offset (x16) = 5Eh) [Reset = 0h]

MCAN_RXESC is shown in Figure 29-66 and described in Table 29-63.

Return to the Summary Table.

Configures the number of data bytes belonging to an Rx Buffer / Rx FIFO element. Data field sizes >8 bytes are intended for CAN FD operation only.

Figure 29-66. MCAN_RXESC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED					RBDS		
R-0h					R/WQ-0h		
7	6	5	4	3	2	1	0
RESERVED	F1DS			RESERVED	F0DS		
R-0h	R/WQ-0h			R-0h	R/WQ-0h		

Table 29-63. MCAN_RXESC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-11	RESERVED	R	0h	Reserved
10-8	RBDS	R/WQ	0h	Rx Buffer Data Field Size 000 8 byte data field 001 12 byte data field 010 16 byte data field 011 20 byte data field 100 24 byte data field 101 32 byte data field 110 48 byte data field 111 64 byte data field Note: In case the data field size of an accepted CAN frame exceeds the data field size configured for the matching Rx Buffer or Rx FIFO, only the number of bytes as configured by RXESC are stored to the Rx Buffer resp. Rx FIFO element. The rest of the frame's data field is ignored. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
7	RESERVED	R	0h	Reserved

Table 29-63. MCAN_RXESC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6-4	F1DS	R/WQ	0h	Rx FIFO 1 Data Field Size 000 8 byte data field 001 12 byte data field 010 16 byte data field 011 20 byte data field 100 24 byte data field 101 32 byte data field 110 48 byte data field 111 64 byte data field Note: In case the data field size of an accepted CAN frame exceeds the data field size configured for the matching Rx Buffer or Rx FIFO, only the number of bytes as configured by RXESC are stored to the Rx Buffer resp. Rx FIFO element. The rest of the frame's data field is ignored. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
3	RESERVED	R	0h	Reserved
2-0	F0DS	R/WQ	0h	Rx FIFO 0 Data Field Size 000 8 byte data field 001 12 byte data field 010 16 byte data field 011 20 byte data field 100 24 byte data field 101 32 byte data field 110 48 byte data field 111 64 byte data field Note: In case the data field size of an accepted CAN frame exceeds the data field size configured for the matching Rx Buffer or Rx FIFO, only the number of bytes as configured by RXESC are stored to the Rx Buffer resp. Rx FIFO element. The rest of the frame's data field is ignored. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

29.7.3.34 MCAN_TXBC Register (Offset (x8) = C0h, Offset (x16) = 60h) [Reset = 0h]

MCAN_TXBC is shown in Figure 29-67 and described in Table 29-64.

Return to the Summary Table.

MCAN Tx Buffer Configuration

Figure 29-67. MCAN_TXBC Register

31	30	29	28	27	26	25	24
RESERVED	TFQM	TFQS					
R-0h	R/WQ-0h	R/WQ-0h					
23	22	21	20	19	18	17	16
RESERVED		NDTB					
R-0h		R/WQ-0h					
15	14	13	12	11	10	9	8
TBSA							
R/WQ-0h							
7	6	5	4	3	2	1	0
TBSA						RESERVED	
R/WQ-0h						R-0h	

Table 29-64. MCAN_TXBC Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved
30	TFQM	R/WQ	0h	Tx FIFO/Queue Mode 0 Tx FIFO operation 1 Tx Queue operation Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
29-24	TFQS	R/WQ	0h	Transmit FIFO/Queue Size 0 No Tx FIFO/Queue 1-32 Number of Tx Buffers used for Tx FIFO/Queue >32 Values greater than 32 are interpreted as 32 Note: Be aware that the sum of TFQS and NDTB may be not greater than 32. There is no check for erroneous configurations. The Tx Buffers section in the Message RAM starts with the dedicated Tx Buffers. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
23-22	RESERVED	R	0h	Reserved
21-16	NDTB	R/WQ	0h	Number of Dedicated Transmit Buffers 0 No Dedicated Tx Buffers 1-32 Number of Dedicated Tx Buffers >32 Values greater than 32 are interpreted as 32 Note: Be aware that the sum of TFQS and NDTB may be not greater than 32. There is no check for erroneous configurations. The Tx Buffers section in the Message RAM starts with the dedicated Tx Buffers. Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

Table 29-64. MCAN_TXBC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-2	TBSA	R/WQ	0h	Tx Buffers Start Address. Start address of Tx Buffers section in Message RAM (32-bit word address). Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn
1-0	RESERVED	R	0h	Reserved

29.7.3.35 MCAN_TXFQS Register (Offset (x8) = C4h, Offset (x16) = 62h) [Reset = 0h]

MCAN_TXFQS is shown in Figure 29-68 and described in Table 29-65.

Return to the Summary Table.

The Tx FIFO/Queue status is related to the pending Tx requests listed in register TXBRP. Therefore the effect of Add/Cancellation requests may be delayed due to a running Tx scan (TXBRP not yet updated).

Figure 29-68. MCAN_TXFQS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED		TFQF	TFQP				
R-0h		R-0h	R-0h				
15	14	13	12	11	10	9	8
RESERVED			TFGI				
R-0h			R-0h				
7	6	5	4	3	2	1	0
RESERVED		TFFL					
R-0h		R-0h					

Table 29-65. MCAN_TXFQS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-22	RESERVED	R	0h	Reserved
21	TFQF	R	0h	Tx FIFO/Queue Full 0 Tx FIFO/Queue not full 1 Tx FIFO/Queue full Reset type: SYSRSn
20-16	TFQP	R	0h	Tx FIFO/Queue Put Index. Tx FIFO/Queue write index pointer, range 0 to 31. Note: In case of mixed configurations where dedicated Tx Buffers are combined with a Tx FIFO or a Tx Queue, the Put and Get Indices indicate the number of the Tx Buffer starting with the first dedicated Tx Buffers. Example: For a configuration of 12 dedicated Tx Buffers and a Tx FIFO of 20 Buffers a Put Index of 15 points to the fourth buffer of the Tx FIFO. Reset type: SYSRSn
15-13	RESERVED	R	0h	Reserved
12-8	TFGI	R	0h	Tx FIFO Get Index. Tx FIFO read index pointer, range 0 to 31. Read as zero when Tx Queue operation is configured (TXBC.TFQM = '1'). Note: In case of mixed configurations where dedicated Tx Buffers are combined with a Tx FIFO or a Tx Queue, the Put and Get Indices indicate the number of the Tx Buffer starting with the first dedicated Tx Buffers. Example: For a configuration of 12 dedicated Tx Buffers and a Tx FIFO of 20 Buffers a Put Index of 15 points to the fourth buffer of the Tx FIFO. Reset type: SYSRSn
7-6	RESERVED	R	0h	Reserved
5-0	TFFL	R	0h	Tx FIFO Free Level. Number of consecutive free Tx FIFO elements starting from TFGI, range 0 to 32. Read as zero when Tx Queue operation is configured (TXBC.TFQM = '1'). Reset type: SYSRSn

29.7.3.36 MCAN_TXESC Register (Offset (x8) = C8h, Offset (x16) = 64h) [Reset = 0h]

MCAN_TXESC is shown in Figure 29-69 and described in Table 29-66.

Return to the Summary Table.

Configures the number of data bytes belonging to a Tx Buffer element. Data field sizes > 8 bytes are intended for CAN FD operation only.

Figure 29-69. MCAN_TXESC Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													TBDS		
R-0h													R/WQ-0h		

Table 29-66. MCAN_TXESC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-3	RESERVED	R	0h	Reserved
2-0	TBDS	R/WQ	0h	Tx Buffer Data Field Size 000 8 byte data field 001 12 byte data field 010 16 byte data field 011 20 byte data field 100 24 byte data field 101 32 byte data field 110 48 byte data field 111 64 byte data field Note: In case the data length code DLC of a Tx Buffer element is configured to a value higher than the Tx Buffer data field size TXESC.TBDS, the bytes not defined by the Tx Buffer are transmitted as "0xCC" (padding bytes). Qualified Write is possible only with CCCR.CCE='1' and CCCR.INIT='1'. Reset type: SYSRSn

29.7.3.37 MCAN_TXBRP Register (Offset (x8) = CCh, Offset (x16) = 66h) [Reset = 0h]

MCAN_TXBRP is shown in Figure 29-70 and described in Table 29-67.

Return to the Summary Table.

MCAN Tx Buffer Request Pending

Figure 29-70. MCAN_TXBRP Register

31	30	29	28	27	26	25	24
TRP31	TRP30	TRP29	TRP28	TRP27	TRP26	TRP25	TRP24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
TRP23	TRP22	TRP21	TRP20	TRP19	TRP18	TRP17	TRP16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
TRP15	TRP14	TRP13	TRP12	TRP11	TRP10	TRP9	TRP8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
TRP7	TRP6	TRP5	TRP4	TRP3	TRP2	TRP1	TRP0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 29-67. MCAN_TXBRP Register Field Descriptions

Bit	Field	Type	Reset	Description
31	TRP31	R	0h	Transmission Request Pending 31. See description for bit 0. Reset type: SYSRSn
30	TRP30	R	0h	Transmission Request Pending 30. See description for bit 0. Reset type: SYSRSn
29	TRP29	R	0h	Transmission Request Pending 29. See description for bit 0. Reset type: SYSRSn
28	TRP28	R	0h	Transmission Request Pending 28. See description for bit 0. Reset type: SYSRSn
27	TRP27	R	0h	Transmission Request Pending 27. See description for bit 0. Reset type: SYSRSn
26	TRP26	R	0h	Transmission Request Pending 26. See description for bit 0. Reset type: SYSRSn
25	TRP25	R	0h	Transmission Request Pending 25. See description for bit 0. Reset type: SYSRSn
24	TRP24	R	0h	Transmission Request Pending 24. See description for bit 0. Reset type: SYSRSn
23	TRP23	R	0h	Transmission Request Pending 23. See description for bit 0. Reset type: SYSRSn
22	TRP22	R	0h	Transmission Request Pending 22. See description for bit 0. Reset type: SYSRSn
21	TRP21	R	0h	Transmission Request Pending 21. See description for bit 0. Reset type: SYSRSn
20	TRP20	R	0h	Transmission Request Pending 20. See description for bit 0. Reset type: SYSRSn
19	TRP19	R	0h	Transmission Request Pending 19. See description for bit 0. Reset type: SYSRSn
18	TRP18	R	0h	Transmission Request Pending 18. See description for bit 0. Reset type: SYSRSn
17	TRP17	R	0h	Transmission Request Pending 17. See description for bit 0. Reset type: SYSRSn

Table 29-67. MCAN_TXBRP Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	TRP16	R	0h	Transmission Request Pending 16. See description for bit 0. Reset type: SYSRSn
15	TRP15	R	0h	Transmission Request Pending 15. See description for bit 0. Reset type: SYSRSn
14	TRP14	R	0h	Transmission Request Pending 14. See description for bit 0. Reset type: SYSRSn
13	TRP13	R	0h	Transmission Request Pending 13. See description for bit 0. Reset type: SYSRSn
12	TRP12	R	0h	Transmission Request Pending 12. See description for bit 0. Reset type: SYSRSn
11	TRP11	R	0h	Transmission Request Pending 11. See description for bit 0. Reset type: SYSRSn
10	TRP10	R	0h	Transmission Request Pending 10. See description for bit 0. Reset type: SYSRSn
9	TRP9	R	0h	Transmission Request Pending 9. See description for bit 0. Reset type: SYSRSn
8	TRP8	R	0h	Transmission Request Pending 8. See description for bit 0. Reset type: SYSRSn
7	TRP7	R	0h	Transmission Request Pending 7. See description for bit 0. Reset type: SYSRSn
6	TRP6	R	0h	Transmission Request Pending 6. See description for bit 0. Reset type: SYSRSn
5	TRP5	R	0h	Transmission Request Pending 5. See description for bit 0. Reset type: SYSRSn
4	TRP4	R	0h	Transmission Request Pending 4. See description for bit 0. Reset type: SYSRSn
3	TRP3	R	0h	Transmission Request Pending 3. See description for bit 0. Reset type: SYSRSn
2	TRP2	R	0h	Transmission Request Pending 2. See description for bit 0. Reset type: SYSRSn
1	TRP1	R	0h	Transmission Request Pending 1. See description for bit 0. Reset type: SYSRSn

Table 29-67. MCAN_TXBRP Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	TRP0	R	0h	Transmission Request Pending 0. Each Tx Buffer has its own Transmission Request Pending bit. The bits are set via register TXBAR. The bits are reset after a requested transmission has completed or has been cancelled via register TXBCR. TXBRP bits are set only for those Tx Buffers configured via TXBC. After a TXBRP bit has been set, a Tx scan is started to check for the pending Tx request with the highest priority (Tx Buffer with lowest Message ID). A cancellation request resets the corresponding transmission request pending bit of register TXBRP. In case a transmission has already been started when a cancellation is requested, this is done at the end of the transmission, regardless whether the transmission was successful or not. The cancellation request bits are reset directly after the corresponding TXBRP bit has been reset. After a cancellation has been requested, a finished cancellation is signalled via TXBCF - after successful transmission together with the corresponding TXBTO bit - when the transmission has not yet been started at the point of cancellation - when the transmission has been aborted due to lost arbitration - when an error occurred during frame transmission In DAR mode all transmissions are automatically cancelled if they are not successful. The corresponding TXBCF bit is set for all unsuccessful transmissions. 0 No transmission request pending 1 Transmission request pending Note: TXBRP bits which are set while a Tx scan is in progress are not considered during this particular Tx scan. In case a cancellation is requested for such a Tx Buffer, this Add Request is cancelled immediately, the corresponding TXBRP bit is reset. Reset type: SYSRSn

29.7.3.38 MCAN_TXBAR Register (Offset (x8) = D0h, Offset (x16) = 68h) [Reset = 0h]

MCAN_TXBAR is shown in Figure 29-71 and described in Table 29-68.

Return to the Summary Table.

MCAN Tx Buffer Add Request

Figure 29-71. MCAN_TXBAR Register

31	30	29	28	27	26	25	24
AR31	AR30	AR29	AR28	AR27	AR26	AR25	AR24
R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h
23	22	21	20	19	18	17	16
AR23	AR22	AR21	AR20	AR19	AR18	AR17	AR16
R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h
15	14	13	12	11	10	9	8
AR15	AR14	AR13	AR12	AR11	AR10	AR9	AR8
R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h
7	6	5	4	3	2	1	0
AR7	AR6	AR5	AR4	AR3	AR2	AR1	AR0
R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h

Table 29-68. MCAN_TXBAR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	AR31	R/WQ	0h	Add Request 31. See description for bit 0. Reset type: SYSRSn
30	AR30	R/WQ	0h	Add Request 30. See description for bit 0. Reset type: SYSRSn
29	AR29	R/WQ	0h	Add Request 29. See description for bit 0. Reset type: SYSRSn
28	AR28	R/WQ	0h	Add Request 28. See description for bit 0. Reset type: SYSRSn
27	AR27	R/WQ	0h	Add Request 27. See description for bit 0. Reset type: SYSRSn
26	AR26	R/WQ	0h	Add Request 26. See description for bit 0. Reset type: SYSRSn
25	AR25	R/WQ	0h	Add Request 25. See description for bit 0. Reset type: SYSRSn
24	AR24	R/WQ	0h	Add Request 24. See description for bit 0. Reset type: SYSRSn
23	AR23	R/WQ	0h	Add Request 23. See description for bit 0. Reset type: SYSRSn
22	AR22	R/WQ	0h	Add Request 22. See description for bit 0. Reset type: SYSRSn
21	AR21	R/WQ	0h	Add Request 21. See description for bit 0. Reset type: SYSRSn
20	AR20	R/WQ	0h	Add Request 20. See description for bit 0. Reset type: SYSRSn
19	AR19	R/WQ	0h	Add Request 19. See description for bit 0. Reset type: SYSRSn
18	AR18	R/WQ	0h	Add Request 18. See description for bit 0. Reset type: SYSRSn
17	AR17	R/WQ	0h	Add Request 17. See description for bit 0. Reset type: SYSRSn

Table 29-68. MCAN_TXBAR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	AR16	R/WQ	0h	Add Request 16. See description for bit 0. Reset type: SYSRSn
15	AR15	R/WQ	0h	Add Request 15. See description for bit 0. Reset type: SYSRSn
14	AR14	R/WQ	0h	Add Request 14. See description for bit 0. Reset type: SYSRSn
13	AR13	R/WQ	0h	Add Request 13. See description for bit 0. Reset type: SYSRSn
12	AR12	R/WQ	0h	Add Request 12. See description for bit 0. Reset type: SYSRSn
11	AR11	R/WQ	0h	Add Request 11. See description for bit 0. Reset type: SYSRSn
10	AR10	R/WQ	0h	Add Request 10. See description for bit 0. Reset type: SYSRSn
9	AR9	R/WQ	0h	Add Request 9. See description for bit 0. Reset type: SYSRSn
8	AR8	R/WQ	0h	Add Request 8. See description for bit 0. Reset type: SYSRSn
7	AR7	R/WQ	0h	Add Request 7. See description for bit 0. Reset type: SYSRSn
6	AR6	R/WQ	0h	Add Request 6. See description for bit 0. Reset type: SYSRSn
5	AR5	R/WQ	0h	Add Request 5. See description for bit 0. Reset type: SYSRSn
4	AR4	R/WQ	0h	Add Request 4. See description for bit 0. Reset type: SYSRSn
3	AR3	R/WQ	0h	Add Request 3. See description for bit 0. Reset type: SYSRSn
2	AR2	R/WQ	0h	Add Request 2. See description for bit 0. Reset type: SYSRSn
1	AR1	R/WQ	0h	Add Request 1. See description for bit 0. Reset type: SYSRSn
0	AR0	R/WQ	0h	Add Request 0. Each Tx Buffer has its own Add Request bit. Writing a '1' will set the corresponding Add Request bit writing a '0' has no impact. This enables the Host to set transmission requests for multiple Tx Buffers with one write to TXBAR. TXBAR bits are set only for those Tx Buffers configured via TXBC. When no Tx scan is running, the bits are reset immediately, else the bits remain set until the Tx scan process has completed. 0 No transmission request added 1 Transmission requested added Note: If an add request is applied for a Tx Buffer with pending transmission request (corresponding TXBRP bit already set), this add request is ignored. Qualified Write is possible only with CCCR.CCE='0' Reset type: SYSRSn

29.7.3.39 MCAN_TXBCR Register (Offset (x8) = D4h, Offset (x16) = 6Ah) [Reset = 0h]

MCAN_TXBCR is shown in Figure 29-72 and described in Table 29-69.

Return to the Summary Table.

MCAN Tx Buffer Cancellation Request

Figure 29-72. MCAN_TXBCR Register

31	30	29	28	27	26	25	24
CR31	CR30	CR29	CR28	CR27	CR26	CR25	CR24
R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h
23	22	21	20	19	18	17	16
CR23	CR22	CR21	CR20	CR19	CR18	CR17	CR16
R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h
15	14	13	12	11	10	9	8
CR15	CR14	CR13	CR12	CR11	CR10	CR9	CR8
R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h
7	6	5	4	3	2	1	0
CR7	CR6	CR5	CR4	CR3	CR2	CR1	CR0
R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h	R/WQ-0h

Table 29-69. MCAN_TXBCR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CR31	R/WQ	0h	Cancellation Request 31. See description for bit 0. Reset type: SYSRSn
30	CR30	R/WQ	0h	Cancellation Request 30. See description for bit 0. Reset type: SYSRSn
29	CR29	R/WQ	0h	Cancellation Request 29. See description for bit 0. Reset type: SYSRSn
28	CR28	R/WQ	0h	Cancellation Request 28. See description for bit 0. Reset type: SYSRSn
27	CR27	R/WQ	0h	Cancellation Request 27. See description for bit 0. Reset type: SYSRSn
26	CR26	R/WQ	0h	Cancellation Request 26. See description for bit 0. Reset type: SYSRSn
25	CR25	R/WQ	0h	Cancellation Request 25. See description for bit 0. Reset type: SYSRSn
24	CR24	R/WQ	0h	Cancellation Request 24. See description for bit 0. Reset type: SYSRSn
23	CR23	R/WQ	0h	Cancellation Request 23. See description for bit 0. Reset type: SYSRSn
22	CR22	R/WQ	0h	Cancellation Request 22. See description for bit 0. Reset type: SYSRSn
21	CR21	R/WQ	0h	Cancellation Request 21. See description for bit 0. Reset type: SYSRSn
20	CR20	R/WQ	0h	Cancellation Request 20. See description for bit 0. Reset type: SYSRSn
19	CR19	R/WQ	0h	Cancellation Request 19. See description for bit 0. Reset type: SYSRSn
18	CR18	R/WQ	0h	Cancellation Request 18. See description for bit 0. Reset type: SYSRSn
17	CR17	R/WQ	0h	Cancellation Request 17. See description for bit 0. Reset type: SYSRSn

Table 29-69. MCAN_TXBCR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	CR16	R/WQ	0h	Cancellation Request 16. See description for bit 0. Reset type: SYSRSn
15	CR15	R/WQ	0h	Cancellation Request 15. See description for bit 0. Reset type: SYSRSn
14	CR14	R/WQ	0h	Cancellation Request 14. See description for bit 0. Reset type: SYSRSn
13	CR13	R/WQ	0h	Cancellation Request 13. See description for bit 0. Reset type: SYSRSn
12	CR12	R/WQ	0h	Cancellation Request 12. See description for bit 0. Reset type: SYSRSn
11	CR11	R/WQ	0h	Cancellation Request 11. See description for bit 0. Reset type: SYSRSn
10	CR10	R/WQ	0h	Cancellation Request 10. See description for bit 0. Reset type: SYSRSn
9	CR9	R/WQ	0h	Cancellation Request 9. See description for bit 0. Reset type: SYSRSn
8	CR8	R/WQ	0h	Cancellation Request 8. See description for bit 0. Reset type: SYSRSn
7	CR7	R/WQ	0h	Cancellation Request 7. See description for bit 0. Reset type: SYSRSn
6	CR6	R/WQ	0h	Cancellation Request 6. See description for bit 0. Reset type: SYSRSn
5	CR5	R/WQ	0h	Cancellation Request 5. See description for bit 0. Reset type: SYSRSn
4	CR4	R/WQ	0h	Cancellation Request 4. See description for bit 0. Reset type: SYSRSn
3	CR3	R/WQ	0h	Cancellation Request 3. See description for bit 0. Reset type: SYSRSn
2	CR2	R/WQ	0h	Cancellation Request 2. See description for bit 0. Reset type: SYSRSn
1	CR1	R/WQ	0h	Cancellation Request 1. See description for bit 0. Reset type: SYSRSn
0	CR0	R/WQ	0h	Cancellation Request 0. Each Tx Buffer has its own Cancellation Request bit. Writing a '1' will set the corresponding Cancellation Request bit writing a '0' has no impact. This enables the Host to set cancellation requests for multiple Tx Buffers with one write to TXBCR. TXBCR bits are set only for those Tx Buffers configured via TXBC. The bits remain set until the corresponding bit of TXBRP is reset. 0 No cancellation pending 1 Cancellation pending Qualified Write is possible only with CCCR.CCE='0' Reset type: SYSRSn

29.7.3.40 MCAN_TXBTO Register (Offset (x8) = D8h, Offset (x16) = 6Ch) [Reset = 0h]

MCAN_TXBTO is shown in Figure 29-73 and described in Table 29-70.

Return to the Summary Table.

MCAN Tx Buffer Transmission Occurred

Figure 29-73. MCAN_TXBTO Register

31	30	29	28	27	26	25	24
TO31	TO30	TO29	TO28	TO27	TO26	TO25	TO24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
TO23	TO22	TO21	TO20	TO19	TO18	TO17	TO16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
TO15	TO14	TO13	TO12	TO11	TO10	TO9	TO8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
TO7	TO6	TO5	TO4	TO3	TO2	TO1	TO0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 29-70. MCAN_TXBTO Register Field Descriptions

Bit	Field	Type	Reset	Description
31	TO31	R	0h	Transmission Occurred 31. See description for bit 0. Reset type: SYSRSn
30	TO30	R	0h	Transmission Occurred 30. See description for bit 0. Reset type: SYSRSn
29	TO29	R	0h	Transmission Occurred 29. See description for bit 0. Reset type: SYSRSn
28	TO28	R	0h	Transmission Occurred 28. See description for bit 0. Reset type: SYSRSn
27	TO27	R	0h	Transmission Occurred 27. See description for bit 0. Reset type: SYSRSn
26	TO26	R	0h	Transmission Occurred 26. See description for bit 0. Reset type: SYSRSn
25	TO25	R	0h	Transmission Occurred 25. See description for bit 0. Reset type: SYSRSn
24	TO24	R	0h	Transmission Occurred 24. See description for bit 0. Reset type: SYSRSn
23	TO23	R	0h	Transmission Occurred 23. See description for bit 0. Reset type: SYSRSn
22	TO22	R	0h	Transmission Occurred 22. See description for bit 0. Reset type: SYSRSn
21	TO21	R	0h	Transmission Occurred 21. See description for bit 0. Reset type: SYSRSn
20	TO20	R	0h	Transmission Occurred 20. See description for bit 0. Reset type: SYSRSn
19	TO19	R	0h	Transmission Occurred 19. See description for bit 0. Reset type: SYSRSn
18	TO18	R	0h	Transmission Occurred 18. See description for bit 0. Reset type: SYSRSn
17	TO17	R	0h	Transmission Occurred 17. See description for bit 0. Reset type: SYSRSn

Table 29-70. MCAN_TXBTO Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	TO16	R	0h	Transmission Occurred 16. See description for bit 0. Reset type: SYSRSn
15	TO15	R	0h	Transmission Occurred 15. See description for bit 0. Reset type: SYSRSn
14	TO14	R	0h	Transmission Occurred 14. See description for bit 0. Reset type: SYSRSn
13	TO13	R	0h	Transmission Occurred 13. See description for bit 0. Reset type: SYSRSn
12	TO12	R	0h	Transmission Occurred 12. See description for bit 0. Reset type: SYSRSn
11	TO11	R	0h	Transmission Occurred 11. See description for bit 0. Reset type: SYSRSn
10	TO10	R	0h	Transmission Occurred 10. See description for bit 0. Reset type: SYSRSn
9	TO9	R	0h	Transmission Occurred 9. See description for bit 0. Reset type: SYSRSn
8	TO8	R	0h	Transmission Occurred 8. See description for bit 0. Reset type: SYSRSn
7	TO7	R	0h	Transmission Occurred 7. See description for bit 0. Reset type: SYSRSn
6	TO6	R	0h	Transmission Occurred 6. See description for bit 0. Reset type: SYSRSn
5	TO5	R	0h	Transmission Occurred 5. See description for bit 0. Reset type: SYSRSn
4	TO4	R	0h	Transmission Occurred 4. See description for bit 0. Reset type: SYSRSn
3	TO3	R	0h	Transmission Occurred 3. See description for bit 0. Reset type: SYSRSn
2	TO2	R	0h	Transmission Occurred 2. See description for bit 0. Reset type: SYSRSn
1	TO1	R	0h	Transmission Occurred 1. See description for bit 0. Reset type: SYSRSn
0	TO0	R	0h	Transmission Occurred 0. Each Tx Buffer has its own Transmission Occurred bit. The bits are set when the corresponding TXBRP bit is cleared after a successful transmission. The bits are reset when a new transmission is requested by writing a '1' to the corresponding bit of register TXBAR. 0 No transmission occurred 1 Transmission occurred Reset type: SYSRSn

29.7.3.41 MCAN_TXBCF Register (Offset (x8) = DCh, Offset (x16) = 6Eh) [Reset = 0h]

MCAN_TXBCF is shown in Figure 29-74 and described in Table 29-71.

Return to the Summary Table.

MCAN Tx Buffer Cancellation Finished

Figure 29-74. MCAN_TXBCF Register

31	30	29	28	27	26	25	24
CF31	CF30	CF29	CF28	CF27	CF26	CF25	CF24
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
CF23	CF22	CF21	CF20	CF19	CF18	CF17	CF16
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
CF15	CF14	CF13	CF12	CF11	CF10	CF9	CF8
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
CF7	CF6	CF5	CF4	CF3	CF2	CF1	CF0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 29-71. MCAN_TXBCF Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CF31	R	0h	Cancellation Finished 31. See description for bit 0. Reset type: SYSRSn
30	CF30	R	0h	Cancellation Finished 30. See description for bit 0. Reset type: SYSRSn
29	CF29	R	0h	Cancellation Finished 29. See description for bit 0. Reset type: SYSRSn
28	CF28	R	0h	Cancellation Finished 28. See description for bit 0. Reset type: SYSRSn
27	CF27	R	0h	Cancellation Finished 27. See description for bit 0. Reset type: SYSRSn
26	CF26	R	0h	Cancellation Finished 26. See description for bit 0. Reset type: SYSRSn
25	CF25	R	0h	Cancellation Finished 25. See description for bit 0. Reset type: SYSRSn
24	CF24	R	0h	Cancellation Finished 24. See description for bit 0. Reset type: SYSRSn
23	CF23	R	0h	Cancellation Finished 23. See description for bit 0. Reset type: SYSRSn
22	CF22	R	0h	Cancellation Finished 22. See description for bit 0. Reset type: SYSRSn
21	CF21	R	0h	Cancellation Finished 21. See description for bit 0. Reset type: SYSRSn
20	CF20	R	0h	Cancellation Finished 20. See description for bit 0. Reset type: SYSRSn
19	CF19	R	0h	Cancellation Finished 19. See description for bit 0. Reset type: SYSRSn
18	CF18	R	0h	Cancellation Finished 18. See description for bit 0. Reset type: SYSRSn
17	CF17	R	0h	Cancellation Finished 17. See description for bit 0. Reset type: SYSRSn

Table 29-71. MCAN_TXBCF Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	CF16	R	0h	Cancellation Finished 16. See description for bit 0. Reset type: SYSRSn
15	CF15	R	0h	Cancellation Finished 15. See description for bit 0. Reset type: SYSRSn
14	CF14	R	0h	Cancellation Finished 14. See description for bit 0. Reset type: SYSRSn
13	CF13	R	0h	Cancellation Finished 13. See description for bit 0. Reset type: SYSRSn
12	CF12	R	0h	Cancellation Finished 12. See description for bit 0. Reset type: SYSRSn
11	CF11	R	0h	Cancellation Finished 11. See description for bit 0. Reset type: SYSRSn
10	CF10	R	0h	Cancellation Finished 10. See description for bit 0. Reset type: SYSRSn
9	CF9	R	0h	Cancellation Finished 9. See description for bit 0. Reset type: SYSRSn
8	CF8	R	0h	Cancellation Finished 8. See description for bit 0. Reset type: SYSRSn
7	CF7	R	0h	Cancellation Finished 7. See description for bit 0. Reset type: SYSRSn
6	CF6	R	0h	Cancellation Finished 6. See description for bit 0. Reset type: SYSRSn
5	CF5	R	0h	Cancellation Finished 5. See description for bit 0. Reset type: SYSRSn
4	CF4	R	0h	Cancellation Finished 4. See description for bit 0. Reset type: SYSRSn
3	CF3	R	0h	Cancellation Finished 3. See description for bit 0. Reset type: SYSRSn
2	CF2	R	0h	Cancellation Finished 2. See description for bit 0. Reset type: SYSRSn
1	CF1	R	0h	Cancellation Finished 1. See description for bit 0. Reset type: SYSRSn
0	CF0	R	0h	Cancellation Finished 0. Each Tx Buffer has its own Cancellation Finished bit. The bits are set when the corresponding TXBRP bit is cleared after a cancellation was requested via TXBCR. In case the corresponding TXBRP bit was not set at the point of cancellation, CF is set immediately. The bits are reset when a new transmission is requested by writing a '1' to the corresponding bit of register TXBAR. 0 No transmit buffer cancellation 1 Transmit buffer cancellation finished Reset type: SYSRSn

29.7.3.42 MCAN_TXBTIE Register (Offset (x8) = E0h, Offset (x16) = 70h) [Reset = 0h]

MCAN_TXBTIE is shown in Figure 29-75 and described in Table 29-72.

Return to the Summary Table.

MCAN Tx Buffer Transmission Interrupt Enable

Figure 29-75. MCAN_TXBTIE Register

31	30	29	28	27	26	25	24
TIE31	TIE30	TIE29	TIE28	TIE27	TIE26	TIE25	TIE24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
TIE23	TIE22	TIE21	TIE20	TIE19	TIE18	TIE17	TIE16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
TIE15	TIE14	TIE13	TIE12	TIE11	TIE10	TIE9	TIE8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
TIE7	TIE6	TIE5	TIE4	TIE3	TIE2	TIE1	TIE0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 29-72. MCAN_TXBTIE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	TIE31	R/W	0h	Transmission Interrupt Enable 31. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
30	TIE30	R/W	0h	Transmission Interrupt Enable 30. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
29	TIE29	R/W	0h	Transmission Interrupt Enable 29. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
28	TIE28	R/W	0h	Transmission Interrupt Enable 28. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
27	TIE27	R/W	0h	Transmission Interrupt Enable 27. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
26	TIE26	R/W	0h	Transmission Interrupt Enable 26. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn

Table 29-72. MCAN_TXBTIE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	TIE25	R/W	0h	Transmission Interrupt Enable 25. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
24	TIE24	R/W	0h	Transmission Interrupt Enable 24. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
23	TIE23	R/W	0h	Transmission Interrupt Enable 23. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
22	TIE22	R/W	0h	Transmission Interrupt Enable 22. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
21	TIE21	R/W	0h	Transmission Interrupt Enable 21. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
20	TIE20	R/W	0h	Transmission Interrupt Enable 20. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
19	TIE19	R/W	0h	Transmission Interrupt Enable 19. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
18	TIE18	R/W	0h	Transmission Interrupt Enable 18. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
17	TIE17	R/W	0h	Transmission Interrupt Enable 17. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
16	TIE16	R/W	0h	Transmission Interrupt Enable 16. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
15	TIE15	R/W	0h	Transmission Interrupt Enable 15. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
14	TIE14	R/W	0h	Transmission Interrupt Enable 14. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn

Table 29-72. MCAN_TXBTIE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	TIE13	R/W	0h	Transmission Interrupt Enable 13. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
12	TIE12	R/W	0h	Transmission Interrupt Enable 12. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
11	TIE11	R/W	0h	Transmission Interrupt Enable 11. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
10	TIE10	R/W	0h	Transmission Interrupt Enable 10. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
9	TIE9	R/W	0h	Transmission Interrupt Enable 9. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
8	TIE8	R/W	0h	Transmission Interrupt Enable 8. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
7	TIE7	R/W	0h	Transmission Interrupt Enable 7. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
6	TIE6	R/W	0h	Transmission Interrupt Enable 6. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
5	TIE5	R/W	0h	Transmission Interrupt Enable 5. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
4	TIE4	R/W	0h	Transmission Interrupt Enable 4. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
3	TIE3	R/W	0h	Transmission Interrupt Enable 3. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
2	TIE2	R/W	0h	Transmission Interrupt Enable 2. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn

Table 29-72. MCAN_TXBTIE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	TIE1	R/W	0h	Transmission Interrupt Enable 1. Each Tx Buffer has its own Transmission Interrupt Enable bit. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn
0	TIE0	R/W	0h	Transmission Interrupt Enable 0. 0 Transmission interrupt disabled 1 Transmission interrupt enable Reset type: SYSRSn

29.7.3.43 MCAN_TXBCIE Register (Offset (x8) = E4h, Offset (x16) = 72h) [Reset = 0h]

MCAN_TXBCIE is shown in Figure 29-76 and described in Table 29-73.

Return to the Summary Table.

MCAN Tx Buffer Cancellation Finished Interrupt Enable

Figure 29-76. MCAN_TXBCIE Register

31	30	29	28	27	26	25	24
CFIE31	CFIE30	CFIE29	CFIE28	CFIE27	CFIE26	CFIE25	CFIE24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
CFIE23	CFIE22	CFIE21	CFIE20	CFIE19	CFIE18	CFIE17	CFIE16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
CFIE15	CFIE14	CFIE13	CFIE12	CFIE11	CFIE10	CFIE9	CFIE8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CFIE7	CFIE6	CFIE5	CFIE4	CFIE3	CFIE2	CFIE1	CFIE0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 29-73. MCAN_TXBCIE Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CFIE31	R/W	0h	Cancellation Finished Interrupt Enable 31. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
30	CFIE30	R/W	0h	Cancellation Finished Interrupt Enable 30. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
29	CFIE29	R/W	0h	Cancellation Finished Interrupt Enable 29. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
28	CFIE28	R/W	0h	Cancellation Finished Interrupt Enable 28. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
27	CFIE27	R/W	0h	Cancellation Finished Interrupt Enable 27. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
26	CFIE26	R/W	0h	Cancellation Finished Interrupt Enable 26. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn

Table 29-73. MCAN_TXBCIE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
25	CFIE25	R/W	0h	Cancellation Finished Interrupt Enable 25. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
24	CFIE24	R/W	0h	Cancellation Finished Interrupt Enable 24. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
23	CFIE23	R/W	0h	Cancellation Finished Interrupt Enable 23. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
22	CFIE22	R/W	0h	Cancellation Finished Interrupt Enable 22. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
21	CFIE21	R/W	0h	Cancellation Finished Interrupt Enable 21. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
20	CFIE20	R/W	0h	Cancellation Finished Interrupt Enable 20. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
19	CFIE19	R/W	0h	Cancellation Finished Interrupt Enable 19. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
18	CFIE18	R/W	0h	Cancellation Finished Interrupt Enable 18. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
17	CFIE17	R/W	0h	Cancellation Finished Interrupt Enable 17. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
16	CFIE16	R/W	0h	Cancellation Finished Interrupt Enable 16. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
15	CFIE15	R/W	0h	Cancellation Finished Interrupt Enable 15. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
14	CFIE14	R/W	0h	Cancellation Finished Interrupt Enable 14. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn

Table 29-73. MCAN_TXBCIE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	CFIE13	R/W	0h	Cancellation Finished Interrupt Enable 13. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
12	CFIE12	R/W	0h	Cancellation Finished Interrupt Enable 12. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
11	CFIE11	R/W	0h	Cancellation Finished Interrupt Enable 11. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
10	CFIE10	R/W	0h	Cancellation Finished Interrupt Enable 10. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
9	CFIE9	R/W	0h	Cancellation Finished Interrupt Enable 9. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
8	CFIE8	R/W	0h	Cancellation Finished Interrupt Enable 8. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
7	CFIE7	R/W	0h	Cancellation Finished Interrupt Enable 7. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
6	CFIE6	R/W	0h	Cancellation Finished Interrupt Enable 6. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
5	CFIE5	R/W	0h	Cancellation Finished Interrupt Enable 5. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
4	CFIE4	R/W	0h	Cancellation Finished Interrupt Enable 4. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
3	CFIE3	R/W	0h	Cancellation Finished Interrupt Enable 3. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
2	CFIE2	R/W	0h	Cancellation Finished Interrupt Enable 2. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn

Table 29-73. MCAN_TXBCIE Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	CFIE1	R/W	0h	Cancellation Finished Interrupt Enable 1. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn
0	CFIE0	R/W	0h	Cancellation Finished Interrupt Enable 0. Each Tx Buffer has its own Cancellation Finished Interrupt Enable bit. 0 Cancellation finished interrupt disabled 1 Cancellation finished interrupt enabled Reset type: SYSRSn

29.7.3.44 MCAN_TXEFC Register (Offset (x8) = F0h, Offset (x16) = 78h) [Reset = 0h]

MCAN_TXEFC is shown in Figure 29-77 and described in Table 29-74.

Return to the Summary Table.

MCAN Tx Event FIFO Configuration

Figure 29-77. MCAN_TXEFC Register

31	30	29	28	27	26	25	24
RESERVED		EFWM					
R-0h		R/WQ-0h					
23	22	21	20	19	18	17	16
RESERVED		EFS					
R-0h		R/WQ-0h					
15	14	13	12	11	10	9	8
EFSA							
R/WQ-0h							
7	6	5	4	3	2	1	0
EFSA						RESERVED	
R/WQ-0h						R-0h	

Table 29-74. MCAN_TXEFC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	RESERVED	R	0h	Reserved
29-24	EFWM	R/WQ	0h	Event FIFO Watermark 0 Watermark interrupt disabled 1-32 Level for Tx Event FIFO watermark interrupt (IR.TEFW) >32 Watermark interrupt disabled Reset type: SYSRSn
23-22	RESERVED	R	0h	Reserved
21-16	EFS	R/WQ	0h	Event FIFO Size. The Tx Event FIFO elements are indexed from 0 to EFS - 1. 0 Tx Event FIFO disabled 1-32 Number of Tx Event FIFO elements >32 Values greater than 32 are interpreted as 32 Reset type: SYSRSn
15-2	EFSA	R/WQ	0h	Event FIFO Start Address. Start address of Tx Event FIFO in Message RAM (32-bit word address). Reset type: SYSRSn
1-0	RESERVED	R	0h	Reserved

29.7.3.45 MCAN_TXEFS Register (Offset (x8) = F4h, Offset (x16) = 7Ah) [Reset = 0h]

MCAN_TXEFS is shown in Figure 29-78 and described in Table 29-75.

Return to the Summary Table.

MCAN Tx Event FIFO Status

Figure 29-78. MCAN_TXEFS Register

31	30	29	28	27	26	25	24
RESERVED						TEFL	EFF
R-0h						R-0h	R-0h
23	22	21	20	19	18	17	16
RESERVED			EFPI				
R-0h			R-0h				
15	14	13	12	11	10	9	8
RESERVED			EFGI				
R-0h			R-0h				
7	6	5	4	3	2	1	0
RESERVED		EFFL					
R-0h		R-0h					

Table 29-75. MCAN_TXEFS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved
25	TEFL	R	0h	Tx Event FIFO Element Lost. This bit is a copy of interrupt flag IR.TEFL. When IR.TEFL is reset, this bit is also reset. 0 No Tx Event FIFO element lost 1 Tx Event FIFO element lost, also set after write attempt to Tx Event FIFO of size zero. Reset type: SYSRSn
24	EFF	R	0h	Event FIFO Full 0 Tx Event FIFO not full 1 Tx Event FIFO full Reset type: SYSRSn
23-21	RESERVED	R	0h	Reserved
20-16	EFPI	R	0h	Event FIFO Put Index.Tx Event FIFO write index pointer, range 0 to 31. Reset type: SYSRSn
15-13	RESERVED	R	0h	Reserved
12-8	EFGI	R	0h	Event FIFO Get Index. Tx Event FIFO read index pointer, range 0 to 31. Reset type: SYSRSn
7-6	RESERVED	R	0h	Reserved
5-0	EFFL	R	0h	Event FIFO Fill Level. Number of elements stored in Tx Event FIFO, range 0 to 32. Reset type: SYSRSn

29.7.3.46 MCAN_TXEFA Register (Offset (x8) = F8h, Offset (x16) = 7Ch) [Reset = 0h]

MCAN_TXEFA is shown in Figure 29-79 and described in Table 29-76.

Return to the Summary Table.

MCAN Tx Event FIFO Acknowledge

Figure 29-79. MCAN_TXEFA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																											EFAI				
R-0h																											R/W-0h				

Table 29-76. MCAN_TXEFA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reserved
4-0	EFAI	R/W	0h	Event FIFO Acknowledge Index. After the Host has read an element or a sequence of elements from the Tx Event FIFO it has to write the index of the last element read from Tx Event FIFO to EFAI. This will set the Tx Event FIFO Get Index TXEFS.EFGI to EFAI + 1 and update the Event FIFO Fill Level TXEFS.EFFL. Reset type: SYSRSn

29.7.4 MCAN_ERROR_REGS Registers

Table 29-77 lists the memory-mapped registers for the MCAN_ERROR_REGS registers. All register offset addresses not listed in Table 29-77 should be considered as reserved locations and the register contents should not be modified.

Table 29-77. MCAN_ERROR_REGS Registers

Offset (x8)	Offset (x16)	Acronym	Register Name	Write Protection	Section
0h	0h	MCANERR_REV	MCAN Error Aggregator Revision Register		Go
8h	4h	MCANERR_VECTOR	MCAN ECC Vector Register		Go
Ch	6h	MCANERR_STAT	MCAN Error Misc Status		Go
10h	8h	MCANERR_WRAP_REV	MCAN ECC Wrapper Revision Register		Go
14h	Ah	MCANERR_CTRL	MCAN ECC Control		Go
18h	Ch	MCANERR_ERR_CTRL1	MCAN ECC Error Control 1 Register		Go
1Ch	Eh	MCANERR_ERR_CTRL2	MCAN ECC Error Control 2 Register		Go
20h	10h	MCANERR_ERR_STAT1	MCAN ECC Error Status 1 Register		Go
24h	12h	MCANERR_ERR_STAT2	MCAN ECC Error Status 2 Register		Go
28h	14h	MCANERR_ERR_STAT3	MCAN ECC Error Status 3 Register		Go
3Ch	1Eh	MCANERR_SEC_EOI	MCAN Single Error Corrected End of Interrupt Register		Go
40h	20h	MCANERR_SEC_STATUS	MCAN Single Error Corrected Interrupt Status Register		Go
80h	40h	MCANERR_SEC_ENABLE_SET	MCAN Single Error Corrected Interrupt Enable Set Register		Go
C0h	60h	MCANERR_SEC_ENABLE_CLR	MCAN Single Error Corrected Interrupt Enable Clear Register		Go
13Ch	9Eh	MCANERR_DED_EOI	MCAN Double Error Detected End of Interrupt Register		Go
140h	A0h	MCANERR_DED_STATUS	MCAN Double Error Detected Interrupt Status Register		Go
180h	C0h	MCANERR_DED_ENABLE_SET	MCAN Double Error Detected Interrupt Enable Set Register		Go
1C0h	E0h	MCANERR_DED_ENABLE_CLR	MCAN Double Error Detected Interrupt Enable Clear Register		Go
200h	100h	MCANERR_AGGR_ENABLE_SET	MCAN Error Aggregator Enable Set Register		Go
204h	102h	MCANERR_AGGR_ENABLE_CLR	MCAN Error Aggregator Enable Clear Register		Go
208h	104h	MCANERR_AGGR_STATUS_SET	MCAN Error Aggregator Status Set Register		Go
20Ch	106h	MCANERR_AGGR_STATUS_CLR	MCAN Error Aggregator Status Clear Register		Go

Complex bit access types are encoded to fit into small table cells. Table 29-78 shows the codes that are used for access types in this section.

Table 29-78. MCAN_ERROR_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s

Table 29-78. MCAN_ERROR_REGS Access Type Codes (continued)

Access Type	Code	Description
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
WD	W D	Write Decrement. Decrements the specified bit field by the amount written.
WI	W I	Write Increment. Increments the specified bit field by the amount written.
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

29.7.4.1 MCANERR_REV Register (Offset (x8) = 0h, Offset (x16) = 0h) [Reset = 66A0EA00h]

MCANERR_REV is shown in Figure 29-80 and described in Table 29-79.

Return to the Summary Table.

MCAN Error Aggregator Revision Register

Figure 29-80. MCANERR_REV Register

31	30	29	28	27	26	25	24
SCHEME		RESERVED		MODULE_ID			
R-1h		R-2h		R-6A0h			
23	22	21	20	19	18	17	16
MODULE_ID							
R-6A0h							
15	14	13	12	11	10	9	8
RESERVED						REVMAJ	
R-1Dh						R-2h	
7	6	5	4	3	2	1	0
RESERVED		REVMIN					
R-0h		R-0h					

Table 29-79. MCANERR_REV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	SCHEME	R	1h	PID Register Scheme Reset type: SYSRSn
29-28	RESERVED	R	2h	Reserved
27-16	MODULE_ID	R	6A0h	Module Identification Number Reset type: SYSRSn
15-11	RESERVED	R	1Dh	Reserved
10-8	REVMAJ	R	2h	Major Revision of the Error Aggregator Reset type: SYSRSn
7-6	RESERVED	R	0h	Reserved
5-0	REVMIN	R	0h	Minor Revision of the Error Aggregator Reset type: SYSRSn

29.7.4.2 MCANERR_VECTOR Register (Offset (x8) = 8h, Offset (x16) = 4h) [Reset = 0h]

MCANERR_VECTOR is shown in Figure 29-81 and described in Table 29-80.

Return to the Summary Table.

Each error detection and correction (EDC) controller has a bank of error registers (offsets 0x10 - 0x3B) associated with it. These registers are accessed via an internal serial bus (SVBUS). To access them through the ECC aggregator the controller ID desired must be written to the ECC_VECTOR field, together with the RD_SVBUS trigger and RD_SVBUS_ADDRESS bit field. This initiates the serial read which consummates by setting the RD_SVBUS_DONE bit. At this point the addressed register may be read by a normal CPU read of the appropriate offset address.

Figure 29-81. MCANERR_VECTOR Register

31	30	29	28	27	26	25	24
RESERVED							RD_SVBUS_DONE
R-0h							R-0h
23	22	21	20	19	18	17	16
RD_SVBUS_ADDRESS							
R/W-0h							
15	14	13	12	11	10	9	8
RD_SVBUS	RESERVED				ECC_VECTOR		
R-0/W1S-0h	R-0h				R/W-0h		
7	6	5	4	3	2	1	0
ECC_VECTOR							
R/W-0h							

Table 29-80. MCANERR_VECTOR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-25	RESERVED	R	0h	Reserved
24	RD_SVBUS_DONE	R	0h	Read Completion Flag Reset type: SYSRSn
23-16	RD_SVBUS_ADDRESS	R/W	0h	Read Address Offset Reset type: SYSRSn
15	RD_SVBUS	R-0/W1S	0h	Read Trigger Reset type: SYSRSn
14-11	RESERVED	R	0h	Reserved
10-0	ECC_VECTOR	R/W	0h	ECC RAM ID. Each error detection and correction (EDC) controller has a bank of error registers (offsets 0x10 - 0x3B) associated with it. These registers are accessed via an internal serial bus (SVBUS). To access them through the ECC aggregator the controller ID desired must be written to the ECC_VECTOR field, together with the RD_SVBUS trigger and RD_SVBUS_ADDRESS bit field. This initiates the serial read which consummates by setting the RD_SVBUS_DONE bit. At this point the addressed register may be read by a normal CPU read of the appropriate offset address. 0x000 Message RAM ECC controller is selected Others Reserved (do not use) Subsequent writes through the SVBUS (offsets 0x10 - 0x3B) have a delayed completion. To avoid conflicts, perform a read back of a register within this range after writing. Reset type: SYSRSn

29.7.4.3 MCANERR_STAT Register (Offset (x8) = Ch, Offset (x16) = 6h) [Reset = 2h]

MCANERR_STAT is shown in Figure 29-82 and described in Table 29-81.

Return to the Summary Table.

MCAN Error Misc Status

Figure 29-82. MCANERR_STAT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																					NUM_RAMS										
R-0h																					R-2h										

Table 29-81. MCANERR_STAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-11	RESERVED	R	0h	Reserved
10-0	NUM_RAMS	R	2h	Number of RAMs. Number of ECC RAMs serviced by the aggregator. Reset type: SYSRSn

29.7.4.4 MCANERR_WRAP_REV Register (Offset (x8) = 10h, Offset (x16) = 8h) [Reset = 66A42A02h]

MCANERR_WRAP_REV is shown in Figure 29-83 and described in Table 29-82.

Return to the Summary Table.

This register is accessed through the ECC aggregator via an internal serial bus. To access, the appropriate procedure must be first followed in the MCAN ECC Vector Register.

Figure 29-83. MCANERR_WRAP_REV Register

31	30	29	28	27	26	25	24
SCHEME		RESERVED		MODULE_ID			
R-1h		R-2h		R-6A4h			
23	22	21	20	19	18	17	16
MODULE_ID							
R-6A4h							
15	14	13	12	11	10	9	8
RESERVED					REVMAJ		
R-5h					R-2h		
7	6	5	4	3	2	1	0
RESERVED		REVMIN					
R-0h		R-2h					

Table 29-82. MCANERR_WRAP_REV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-30	SCHEME	R	1h	PID Register Scheme Reset type: SYSRSn
29-28	RESERVED	R	2h	Reserved
27-16	MODULE_ID	R	6A4h	Module Identification Number Reset type: SYSRSn
15-11	RESERVED	R	5h	Reserved
10-8	REVMAJ	R	2h	Major Revision of the Error Aggregator Reset type: SYSRSn
7-6	RESERVED	R	0h	Reserved
5-0	REVMIN	R	2h	Minor Revision of the Error Aggregator Reset type: SYSRSn

29.7.4.5 MCANERR_CTRL Register (Offset (x8) = 14h, Offset (x16) = Ah) [Reset = 187h]

MCANERR_CTRL is shown in Figure 29-84 and described in Table 29-83.

Return to the Summary Table.

This register is accessed through the ECC aggregator via an internal serial bus. To access, the appropriate procedure must be first followed in the MCAN ECC Vector Register.

Figure 29-84. MCANERR_CTRL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							CHECK_SVBUS_TIMEOUT
R-0h							R/W-1h
7	6	5	4	3	2	1	0
RESERVED	ERROR_ONCE	FORCE_N_ROW	FORCE_DED	FORCE_SEC	ENABLE_RMW	ECC_CHECK	ECC_ENABLE
R/W-1h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-1h	R/W-1h	R/W-1h

Table 29-83. MCANERR_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-9	RESERVED	R	0h	Reserved
8	CHECK_SVBUS_TIMEOUT	R/W	1h	Enables Serial VBUS timeout mechanism Reset type: SYSRSn
7	RESERVED	R/W	1h	Reserved
6	ERROR_ONCE	R/W	0h	If this bit is set, the FORCE_SEC/FORCE_DED will inject an error to the specified row only once. The FORCE_SEC bit will be cleared once a writeback happens. If writeback is not enabled, this error will be cleared the cycle following the read when the data is corrected. For double-bit errors, the FORCE_DED bit will be cleared the cycle following the double-bit error. Any subsequent reads will not force an error. Reset type: SYSRSn
5	FORCE_N_ROW	R/W	0h	Enable single/double-bit error on the next RAM read, regardless of the MCANERR_ERR_CTRL1.ECC_ROW setting. For write through mode, this applies to writes as well as reads. Reset type: SYSRSn
4	FORCE_DED	R/W	0h	Force double-bit error. Cleared the cycle following the error if ERROR_ONCE is asserted. For write through mode, this applies to writes as well as reads. MCANERR_ERR_CTRL1 and MCANERR_ERR_CTRL2 should be configured prior to setting this bit. Reset type: SYSRSn
3	FORCE_SEC	R/W	0h	Force single-bit error. Cleared on a writeback or the cycle following the error if ERROR_ONCE is asserted. For write through mode, this applies to writes as well as reads. MCANERR_ERR_CTRL1 and MCANERR_ERR_CTRL2 should be configured prior to setting this bit. Reset type: SYSRSn
2	ENABLE_RMW	R/W	1h	Enable read-modify-write on partial word writes Reset type: SYSRSn

Table 29-83. MCANERR_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	ECC_CHECK	R/W	1h	Enable ECC Check. ECC is completely bypassed if both ECC_ENABLE and ECC_CHECK are '0'. Reset type: SYSRSn
0	ECC_ENABLE	R/W	1h	Enable ECC Generation Reset type: SYSRSn

29.7.4.6 MCANERR_ERR_CTRL1 Register (Offset (x8) = 18h, Offset (x16) = Ch) [Reset = 0h]

MCANERR_ERR_CTRL1 is shown in Figure 29-85 and described in Table 29-84.

Return to the Summary Table.

This register is accessed through the ECC aggregator via an internal serial bus. To access, the appropriate procedure must be first followed in the MCAN ECC Vector Register.

Figure 29-85. MCANERR_ERR_CTRL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_ROW																															
R/W-0h																															

Table 29-84. MCANERR_ERR_CTRL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ECC_ROW	R/W	0h	Row address where FORCE_SEC or FORCE_DED needs to be applied. This is ignored if FORCE_N_ROW is set. Reset type: SYSRSn

29.7.4.7 MCANERR_ERR_CTRL2 Register (Offset (x8) = 1Ch, Offset (x16) = Eh) [Reset = 0h]

MCANERR_ERR_CTRL2 is shown in Figure 29-86 and described in Table 29-85.

Return to the Summary Table.

This register is accessed through the ECC aggregator via an internal serial bus. To access, the appropriate procedure must be first followed in the MCAN ECC Vector Register.

Figure 29-86. MCANERR_ERR_CTRL2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_BIT2																ECC_BIT1															
R/W-0h																R/W-0h															

Table 29-85. MCANERR_ERR_CTRL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	ECC_BIT2	R/W	0h	Second column/data bit that needs to be flipped when FORCE_DED is set Reset type: SYSRSn
15-0	ECC_BIT1	R/W	0h	Column/Data bit that needs to be flipped when FORCE_SEC or FORCE_DED is set Reset type: SYSRSn

29.7.4.8 MCANERR_ERR_STAT1 Register (Offset (x8) = 20h, Offset (x16) = 10h) [Reset = 0h]

MCANERR_ERR_STAT1 is shown in Figure 29-87 and described in Table 29-86.

Return to the Summary Table.

This register is accessed through the ECC aggregator via an internal serial bus. To access, the appropriate procedure must be first followed in the MCAN ECC Vector Register.

Figure 29-87. MCANERR_ERR_STAT1 Register

31	30	29	28	27	26	25	24
ECC_BIT1							
R-0h							
23	22	21	20	19	18	17	16
ECC_BIT1							
R-0h							
15	14	13	12	11	10	9	8
CLR_CTRL_REG_ERROR	RESERVED		CLR_ECC_OTHER	CLR_ECC_DED		CLR_ECC_SEC	
R/W1S-0h	R/WD-0h		R/W1C-0h	R/WD-0h		R/WD-0h	
7	6	5	4	3	2	1	0
CTRL_REG_ERROR	RESERVED		ECC_OTHER	ECC_DED		ECC_SEC	
R/W1S-0h	R/WI-0h		R/W1S-0h	R/WI-0h		R/WI-0h	

Table 29-86. MCANERR_ERR_STAT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	ECC_BIT1	R	0h	ECC Error Bit Position. Indicates the bit position in the RAM data that is in error on an SEC error. Only valid on an SEC error. 0 Bit 0 is in error 1 Bit 1 is in error 2 Bit 2 is in error 3 Bit 3 is in error ... 31 Bit 31 is in error >32 Invalid Reset type: SYSRSn
15	CLR_CTRL_REG_ERROR	R/W1S	0h	Writing a '1' clears the CTRL_REG_ERROR bit Reset type: SYSRSn
14-13	RESERVED	R/WD	0h	Reserved
12	CLR_ECC_OTHER	R/W1C	0h	Writing a '1' clears the ECC_OTHER bit. Reset type: SYSRSn
11-10	CLR_ECC_DED	R/WD	0h	Clear ECC_DED. A write of a non-zero value to this bit field decrements the ECC_DED bit field by the value provided. Reset type: SYSRSn
9-8	CLR_ECC_SEC	R/WD	0h	Clear ECC_SEC. A write of a non-zero value to this bit field decrements the ECC_SEC bit field by the value provided. Reset type: SYSRSn
7	CTRL_REG_ERROR	R/W1S	0h	Control Register Error. A bit field in the control register is in an ambiguous state. This means that the redundancy registers have detected a state where not all values are the same and has defaulted to the reset state. S/W needs to re-write these registers to a known state. A write of 1 will set this interrupt flag. Reset type: SYSRSn
6-5	RESERVED	R/WI	0h	Reserved

Table 29-86. MCANERR_ERR_STAT1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	ECC_OTHER	R/W1S	0h	SEC While Writeback Error Status 0 No SEC error while writeback pending 1 Indicates that successive single-bit errors have occurred while a writeback is still pending Reset type: SYSRSn
3-2	ECC_DED	R/WI	0h	Double Bit Error Detected Status. A 2-bit saturating counter of the number of DED errors that have occurred since last cleared. 0 No double-bit error detected 1 One double-bit error was detected 2 Two double-bit errors were detected 3 Three double-bit errors were detected A write of a non-zero value to this bit field increments it by the value provided. Reset type: SYSRSn
1-0	ECC_SEC	R/WI	0h	Single Bit Error Corrected Status. A 2-bit saturating counter of the number of SEC errors that have occurred since last cleared. 0 No single-bit error detected 1 One single-bit error was detected and corrected 2 Two single-bit errors were detected and corrected 3 Three single-bit errors were detected and corrected A write of a non-zero value to this bit field increments it by the value provided. Reset type: SYSRSn

29.7.4.9 MCANERR_ERR_STAT2 Register (Offset (x8) = 24h, Offset (x16) = 12h) [Reset = 0h]

MCANERR_ERR_STAT2 is shown in Figure 29-88 and described in Table 29-87.

Return to the Summary Table.

This register is accessed through the ECC aggregator via an internal serial bus. To access, the appropriate procedure must be first followed in the MCAN ECC Vector Register.

Figure 29-88. MCANERR_ERR_STAT2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ECC_ROW																															
R-0h																															

Table 29-87. MCANERR_ERR_STAT2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	ECC_ROW	R	0h	Indicates the row address where the single or double-bit error occurred. This value is address offset/4. Reset type: SYSRSn

29.7.4.10 MCANERR_ERR_STAT3 Register (Offset (x8) = 28h, Offset (x16) = 14h) [Reset = 0h]

MCANERR_ERR_STAT3 is shown in Figure 29-89 and described in Table 29-88.

Return to the Summary Table.

This register is accessed through the ECC aggregator via an internal serial bus. To access, the appropriate procedure must be first followed in the MCAN ECC Vector Register.

Figure 29-89. MCANERR_ERR_STAT3 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						CLR_SVBUS_TIMEOUT	RESERVED
R-0h						R-0/W1C-0h	R-0h
7	6	5	4	3	2	1	0
RESERVED						SVBUS_TIMEOUT	WB_PEND
R-0h						R-0/W1S-0h	R-0h

Table 29-88. MCANERR_ERR_STAT3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9	CLR_SVBUS_TIMEOUT	R-0/W1C	0h	Write 1 to clear the Serial VBUS Timeout Flag Reset type: SYSRSn
8-2	RESERVED	R	0h	Reserved
1	SVBUS_TIMEOUT	R-0/W1S	0h	Serial VBUS Timeout Flag. Write 1 to set. Reset type: SYSRSn
0	WB_PEND	R	0h	Delayed Write Back Pending Status 0 No write back pending 1 An ECC data correction write back is pending Reset type: SYSRSn

29.7.4.11 MCANERR_SEC_EOI Register (Offset (x8) = 3Ch, Offset (x16) = 1Eh) [Reset = 0h]

MCANERR_SEC_EOI is shown in Figure 29-90 and described in Table 29-89.

Return to the Summary Table.

MCAN Single Error Corrected End of Interrupt Register

Figure 29-90. MCANERR_SEC_EOI Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EOI_WR
R-0h							R-0/W1S-0h

Table 29-89. MCANERR_SEC_EOI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	EOI_WR	R-0/W1S	0h	Write to this register indicates that software has acknowledged the pending interrupt and the next interrupt can be sent to the host. Note that a write to the MCANERR_ERR_STAT1.CLR_ECC_SEC goes through the SVBUS and has a delayed completion. To avoid an additional interrupt, read the MCANERR_ERR_STAT1 register back prior to writing to this bit field. Reset type: SYSRSn

29.7.4.12 MCANERR_SEC_STATUS Register (Offset (x8) = 40h, Offset (x16) = 20h) [Reset = 0h]

MCANERR_SEC_STATUS is shown in Figure 29-91 and described in Table 29-90.

Return to the Summary Table.

MCAN Single Error Corrected Interrupt Status Register

Figure 29-91. MCANERR_SEC_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						RESERVED	MSGMEM_PEND
R-0h						R-0h	R-0/W1S-0h

Table 29-90. MCANERR_SEC_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved
0	MSGMEM_PEND	R-0/W1S	0h	Message RAM SEC Interrupt Pending 0 No SEC interrupt is pending 1 SEC interrupt is pending Reset type: SYSRSn

29.7.4.13 MCANERR_SEC_ENABLE_SET Register (Offset (x8) = 80h, Offset (x16) = 40h) [Reset = 0h]

MCANERR_SEC_ENABLE_SET is shown in Figure 29-92 and described in Table 29-91.

Return to the Summary Table.

MCAN Single Error Corrected Interrupt Enable Set Register

Figure 29-92. MCANERR_SEC_ENABLE_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						RESERVED	MSGMEM_ENABLE_SET
R-0h						R-0h	R/W1S-0h

Table 29-91. MCANERR_SEC_ENABLE_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved
0	MSGMEM_ENABLE_SET	R/W1S	0h	Message RAM SEC Interrupt Pending Enable Set. Writing a 1 to this bit enables the Message RAM SEC error interrupts. Writing a 0 has no effect. Reads return the corresponding enable bit's current value. Reset type: SYSRSn

29.7.4.14 MCANERR_SEC_ENABLE_CLR Register (Offset (x8) = C0h, Offset (x16) = 60h) [Reset = 0h]

MCANERR_SEC_ENABLE_CLR is shown in Figure 29-93 and described in Table 29-92.

Return to the Summary Table.

MCAN Single Error Corrected Interrupt Enable Clear Register

Figure 29-93. MCANERR_SEC_ENABLE_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						RESERVED	MSGMEM_ENABLE_CLR
R-0h						R-0h	R/W1C-0h

Table 29-92. MCANERR_SEC_ENABLE_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	RESERVED	R	0h	Reserved
0	MSGMEM_ENABLE_CLR	R/W1C	0h	Message RAM SEC Interrupt Pending Enable Clear. Writing a 1 to this bit disables the Message RAM SEC error interrupts. Writing a 0 has no effect. Reads return the corresponding enable bit's current value. Reset type: SYSRSn

29.7.4.15 MCANERR_DED_EOI Register (Offset (x8) = 13Ch, Offset (x16) = 9Eh) [Reset = 0h]

MCANERR_DED_EOI is shown in Figure 29-94 and described in Table 29-93.

Return to the Summary Table.

MCAN Double Error Detected End of Interrupt Register

Figure 29-94. MCANERR_DED_EOI Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							EOI_WR
R-0h							R-0/W1S-0h

Table 29-93. MCANERR_DED_EOI Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	EOI_WR	R-0/W1S	0h	Write to this register indicates that software has acknowledged the pending interrupt and the next interrupt can be sent to the host. Note that a write to the MCANERR_ERR_STAT1.CLR_ECC_DED goes through the SVBUS and has a delayed completion. To avoid an additional interrupt, read the MCANERR_ERR_STAT1 register back prior to writing to this bit field. Reset type: SYSRSn

29.7.4.16 MCANERR_DED_STATUS Register (Offset (x8) = 140h, Offset (x16) = A0h) [Reset = 0h]

MCANERR_DED_STATUS is shown in Figure 29-95 and described in Table 29-94.

Return to the Summary Table.

MCAN Double Error Detected Interrupt Status Register

Figure 29-95. MCANERR_DED_STATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						RESERVED	MSGMEM_PEND
R-0h						R-0/W1S-0h	R-0/W1S-0h

Table 29-94. MCANERR_DED_STATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	RESERVED	R-0/W1S	0h	Reserved
0	MSGMEM_PEND	R-0/W1S	0h	Message RAM DED Interrupt Pending 0 No DED interrupt is pending 1 DED interrupt is pending Reset type: SYSRSn

29.7.4.17 MCANERR_DED_ENABLE_SET Register (Offset (x8) = 180h, Offset (x16) = C0h) [Reset = 0h]

MCANERR_DED_ENABLE_SET is shown in Figure 29-96 and described in Table 29-95.

Return to the Summary Table.

MCAN Double Error Detected Interrupt Enable Set Register

Figure 29-96. MCANERR_DED_ENABLE_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						RESERVED	MSGMEM_ENABLE_SET
R-0h						R/W1S-0h	R/W1S-0h

Table 29-95. MCANERR_DED_ENABLE_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	RESERVED	R/W1S	0h	Reserved
0	MSGMEM_ENABLE_SET	R/W1S	0h	Message RAM DED Interrupt Pending Enable Set. Writing a 1 to this bit enables the Message RAM DED error interrupts. Writing a 0 has no effect. Reads return the corresponding enable bit's current value. Reset type: SYSRSn

29.7.4.18 MCANERR_DED_ENABLE_CLR Register (Offset (x8) = 1C0h, Offset (x16) = E0h) [Reset = 0h]

MCANERR_DED_ENABLE_CLR is shown in Figure 29-97 and described in Table 29-96.

Return to the Summary Table.

MCAN Double Error Detected Interrupt Enable Clear Register

Figure 29-97. MCANERR_DED_ENABLE_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						RESERVED	MSGMEM_ENABLE_CLR
R-0h						R/W1C-0h	R/W1C-0h

Table 29-96. MCANERR_DED_ENABLE_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	RESERVED	R/W1C	0h	Reserved
0	MSGMEM_ENABLE_CLR	R/W1C	0h	Message RAM DED Interrupt Pending Enable Clear. Writing a 1 to this bit disables the Message RAM DED error interrupts. Writing a 0 has no effect. Reads return the corresponding enable bit's current value. Reset type: SYSRSn

29.7.4.19 MCANERR_AGGR_ENABLE_SET Register (Offset (x8) = 200h, Offset (x16) = 100h) [Reset = 0h]

MCANERR_AGGR_ENABLE_SET is shown in Figure 29-98 and described in Table 29-97.

Return to the Summary Table.

MCAN Error Aggregator Enable Set Register

Figure 29-98. MCANERR_AGGR_ENABLE_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						ENABLE_TIMEOUT_SET	ENABLE_PARITY_SET
R-0h						R/W1S-0h	R/W1S-0h

Table 29-97. MCANERR_AGGR_ENABLE_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	ENABLE_TIMEOUT_SET	R/W1S	0h	Write 1 to enable timeout errors. Reads return the corresponding enable bit's current value. Reset type: SYSRSn
0	ENABLE_PARITY_SET	R/W1S	0h	Write 1 to enable parity errors. Reads return the corresponding enable bit's current value. Reset type: SYSRSn

29.7.4.20 MCANERR_AGGR_ENABLE_CLR Register (Offset (x8) = 204h, Offset (x16) = 102h) [Reset = 0h]

MCANERR_AGGR_ENABLE_CLR is shown in Figure 29-99 and described in Table 29-98.

Return to the Summary Table.

MCAN Error Aggregator Enable Clear Register

Figure 29-99. MCANERR_AGGR_ENABLE_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						ENABLE_TIMEOUT_CLR	ENABLE_PARITY_CLR
R-0h						R/W1C-0h	R/W1C-0h

Table 29-98. MCANERR_AGGR_ENABLE_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	ENABLE_TIMEOUT_CLR	R/W1C	0h	Write 1 to disable timeout errors. Reads return the corresponding enable bit's current value. Reset type: SYSRSn
0	ENABLE_PARITY_CLR	R/W1C	0h	Write 1 to disable parity errors. Reads return the corresponding enable bit's current value. Reset type: SYSRSn

29.7.4.21 MCANERR_AGGR_STATUS_SET Register (Offset (x8) = 208h, Offset (x16) = 104h) [Reset = 0h]

MCANERR_AGGR_STATUS_SET is shown in Figure 29-100 and described in Table 29-99.

Return to the Summary Table.

MCAN Error Aggregator Status Set Register

Figure 29-100. MCANERR_AGGR_STATUS_SET Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				SVBUS_TIMEOUT		AGGR_PARITY_ERR	
R-0h				R/WI-0h		R/WI-0h	

Table 29-99. MCANERR_AGGR_STATUS_SET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3-2	SVBUS_TIMEOUT	R/WI	0h	Aggregator Serial VBUS Timeout Error Status 2-bit saturating counter of the number of SVBUS timeout errors that have occurred since last cleared. 0 No timeout errors have occurred 1 One timeout error has occurred 2 Two timeout errors have occurred 3 Three timeout errors have occurred A write of a non-zero value to this bit field increments it by the value provided. Reset type: SYSRSn
1-0	AGGR_PARITY_ERR	R/WI	0h	Aggregator Parity Error Status 2-bit saturating counter of the number of parity errors that have occurred since last cleared. 0 No parity errors have occurred 1 One parity error has occurred 2 Two parity errors have occurred 3 Three parity errors have occurred A write of a non-zero value to this bit field increments it by the value provided. Reset type: SYSRSn

29.7.4.22 MCANERR_AGGR_STATUS_CLR Register (Offset (x8) = 20Ch, Offset (x16) = 106h) [Reset = 0h]

MCANERR_AGGR_STATUS_CLR is shown in Figure 29-101 and described in Table 29-100.

Return to the Summary Table.

MCAN Error Aggregator Status Clear Register

Figure 29-101. MCANERR_AGGR_STATUS_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				SVBUS_TIMEOUT		AGGR_PARITY_ERR	
R-0h				R/WD-0h		R/WD-0h	

Table 29-100. MCANERR_AGGR_STATUS_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3-2	SVBUS_TIMEOUT	R/WD	0h	Aggregator Serial VBUS Timeout Error Status 2-bit saturating counter of the number of SVBUS timeout errors that have occurred since last cleared. 0 No timeout errors have occurred 1 One timeout error has occurred 2 Two timeout errors have occurred 3 Three timeout errors have occurred A write of a non-zero value to this bit field decrements it by the value provided. Reset type: SYSRSn
1-0	AGGR_PARITY_ERR	R/WD	0h	Aggregator Parity Error Status 2-bit saturating counter of the number of parity errors that have occurred since last cleared. 0 No parity errors have occurred 1 One parity error has occurred 2 Two parity errors have occurred 3 Three parity errors have occurred A write of a non-zero value to this bit field decrements it by the value provided. Reset type: SYSRSn

29.7.5 MCAN Registers to Driverlib Functions

Table 29-101. MCAN Registers to Driverlib Functions

File	Driverlib Function
SS_PID	
-	
SS_CTRL	
-	
SS_STAT	
-	

Table 29-101. MCAN Registers to Driverlib Functions (continued)

File	Driverlib Function
SS_ICS	
-	
SS_IRS	
-	
SS_IECS	
-	
SS_IE	
-	
SS_IES	
-	
SS_EOI	
-	
SS_EXT_TS_PRESCALER	
-	
SS_EXT_TS_UNSERVICED_INTR_CNTR	
-	
CREL	
-	
ENDN	
-	
DBTP	
-	
TEST	
-	
RWD	
-	
CCCR	
-	
NBTP	
-	
TSCC	
-	
TSCV	
-	
TOCC	
-	
TOCV	
-	
ECR	
-	
PSR	
-	
TDCR	
-	
IR	

Table 29-101. MCAN Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
IE	
-	
ILS	
-	
ILE	
-	
GFC	
-	
SIDFC	
-	
XIDFC	
-	
XIDAM	
-	
HPMS	
-	
NDAT1	
-	
NDAT2	
-	
RXF0C	
-	
RXF0S	
-	
RXF0A	
-	
RXBC	
-	
RXF1C	
-	
RXF1S	
-	
RXF1A	
-	
RXESC	
-	
TXBC	
-	
TXFQS	
-	
TXESC	
-	
TXBRP	
-	

Table 29-101. MCAN Registers to Driverlib Functions (continued)

File	Driverlib Function
TXBAR	
-	
TXBCR	
-	
TXBTO	
-	
TXBCF	
-	
TXBTIE	
-	
TXBCIE	
-	
TXEFC	
-	
TXEFS	
-	
TXEFA	
-	
ERR_REV	
-	
ERR_VECTOR	
-	
ERR_STAT	
-	
ERR_WRAP_REV	
-	
ERR_CTRL	
-	
ERR_ERR_CTRL1	
-	
ERR_ERR_CTRL2	
-	
ERR_ERR_STAT1	
-	
ERR_ERR_STAT2	
-	
ERR_ERR_STAT3	
-	
ERR_SEC_EOI	
-	
ERR_SEC_STATUS	
-	
ERR_SEC_ENABLE_SET	
-	
ERR_SEC_ENABLE_CLR	

Table 29-101. MCAN Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
ERR_DED_EOI	
-	
ERR_DED_STATUS	
-	
ERR_DED_ENABLE_SET	
-	
ERR_DED_ENABLE_CLR	
-	
ERR_AGGR_ENABLE_SET	
-	
ERR_AGGR_ENABLE_CLR	
-	
ERR_AGGR_STATUS_SET	
-	
ERR_AGGR_STATUS_CLR	
-	

This page intentionally left blank.

Chapter 30

Local Interconnect Network (LIN)

This chapter describes the local interconnect network (LIN) module. Since this module can also operate like a conventional serial communications interface (SCI) port, it is referred to as the SCI/LIN module in this document. In SCI compatibility mode, it is functionally compatible to the standalone SCI module. However, since the SCI/LIN module uses a different register/bit structure, code written for this module cannot be directly ported to the standalone SCI module and vice versa.

This module can be configured to operate in either SCI (UART) or LIN mode.

30.1 Introduction

The SCI/LIN is compliant to the LIN protocol specified in the *LIN Specification Package*.The SCI/LIN module can be programmed to work either as an SCI or as a LIN. The SCI's hardware features are augmented to achieve LIN compatibility.

The SCI module is a universal asynchronous receiver-transmitter (UART) that implements the standard non-return to zero format.

The LIN standard is based on the SCI (UART) serial data link format. The communication concept is single-master/multiple-slave with a message identification for multi-cast transmission between any network nodes.

Throughout the chapter, Compatibility Mode refers to SCI Mode functionality of the SCI/LIN Module. Section 30.2 explains about the SCI functionality and Section 30.3 explains about the LIN functionality. Though the registers are common for LIN and SCI, the register descriptions has notes to identify the register/bit usage in different modes.

30.1.1 SCI Features

The following are the features of the SCI module:

- Standard universal asynchronous receiver-transmitter (UART) communication
- Supports full- or half-duplex operation
- Standard non-return to zero (NRZ) format
- Double-buffered receive and transmit functions in compatibility mode
- Supports two individually enabled interrupt lines: level 0 and level 1
- Configurable frame format of 3 to 13 bits per character based on the following:
  - Data word length programmable from one to eight bits
  - Additional address bit in address-bit mode
  - Parity programmable for zero or one parity bit, odd or even parity
  - Stop programmable for one or two stop bits
- Asynchronous communication mode
- Two multiprocessor communication formats allow communication between more than two devices
- Sleep mode is available to free CPU resources during multiprocessor communication and then wake up to receive an incoming message
- The 24-bit programmable baud rate supports 2^{24} different baud rates provide high accuracy baud rate selection
- At 100-MHz peripheral clock, 3.125 Mbits/s is the Max Baud Rate achievable
- Five error flags and Seven status flags provide detailed information regarding SCI events
- Two external pins: LINRX and LINTX
- Multi-buffered receive and transmit units

Note

The SCI/LIN module is functionally compatible with the C2000™ SCI modules, but not directly software compatible due to different register control structures.

The SCI/LIN module does not support UART hardware flow control. This feature can be implemented in software using a general-purpose I/O pin.

The SCI/LIN module does not support isosynchronous mode as there is no SCICLK pin.

30.1.2 LIN Features

The following are the features of the LIN module:

- Compatibility with LIN 1.3 protocols
- Configurable Baud Rate up to 20 kpbs
- Two external pins: LINRX and LINTX.
- Multi-buffered receive and transmit units
- Identification masks for message filtering
- Automatic master header generation
  - Programmable synchronization break field
  - Synchronization field
  - Identifier field
- Slave automatic synchronization
  - Synchronization break detection
  - Optional baud rate update
  - Synchronization validation
- Wake up on LINRX dominant level from transceiver
- Automatic wake up support
  - Wakeup signal generation
  - Expiration times on wakeup signals
- Automatic bus idle detection
- Error detection
  - Bit error
  - Bus error
  - No-response error
  - Checksum error
  - Synchronization field error
  - Parity error
- 2 Interrupt lines with priority encoding for:
  - Receive
  - Transmit
  - ID, error, and status
- Support for LIN 2.0 checksum
- Enhanced synchronizer finite state machine (FSM) support for frame processing
- Enhanced handling of extended frames
- Enhanced baud rate generator
- Update wakeup/go to sleep

30.1.3 Block Diagram

The SCI/LIN module contains the core SCI block with added sub-blocks to support LIN protocol.

The three major components of the SCI Module are:

- **Transmitter** (TX) contains two major registers to perform the double-buffering:
  - The transmitter data buffer register (SCITD) contains data loaded by the CPU to be transferred to the shift register for transmission.
  - The transmitter shift register (SCITXSHF) loads data from the data buffer (SCITD) and shifts data onto the LINTX pin, one bit at a time.
- **Baud Clock Generator**
  - A programmable baud generator produces a baud clock scaled from the input clock VCLK
- **Receiver** (RX) contains two major registers to perform the double-buffering:
  - The receiver shift register (SCIRXSHF) shifts data in from the LINRX pin one bit at a time and transfers completed data into the receive data buffer.
  - The receiver data buffer register (SCIRD) contains received data transferred from the receiver shift register

The SCI receiver and transmitter are double-buffered, and each has its own separate enable and interrupt bits. The receiver and transmitter may each be operated independently or simultaneously in full duplex mode.

To ensure data integrity, the SCI checks the data it receives for breaks, parity, overrun, and framing errors. The bit rate (baud) is programmable to over 16 million different rates through a 24-bit baud-select register. Figure 30-1 shows the detailed SCI block diagram.

The SCI/LIN module is based on the standalone SCI with the addition of an error detector (parity calculator, checksum calculator, and bit monitor), a mask filter, a synchronizer, and a multi-buffered receiver and transmitter. The SCI interface, the DMA control subblocks and the baud generator are modified as part of the hardware enhancements for LIN compatibility. Figure 30-2 shows the SCI/LIN block diagram.

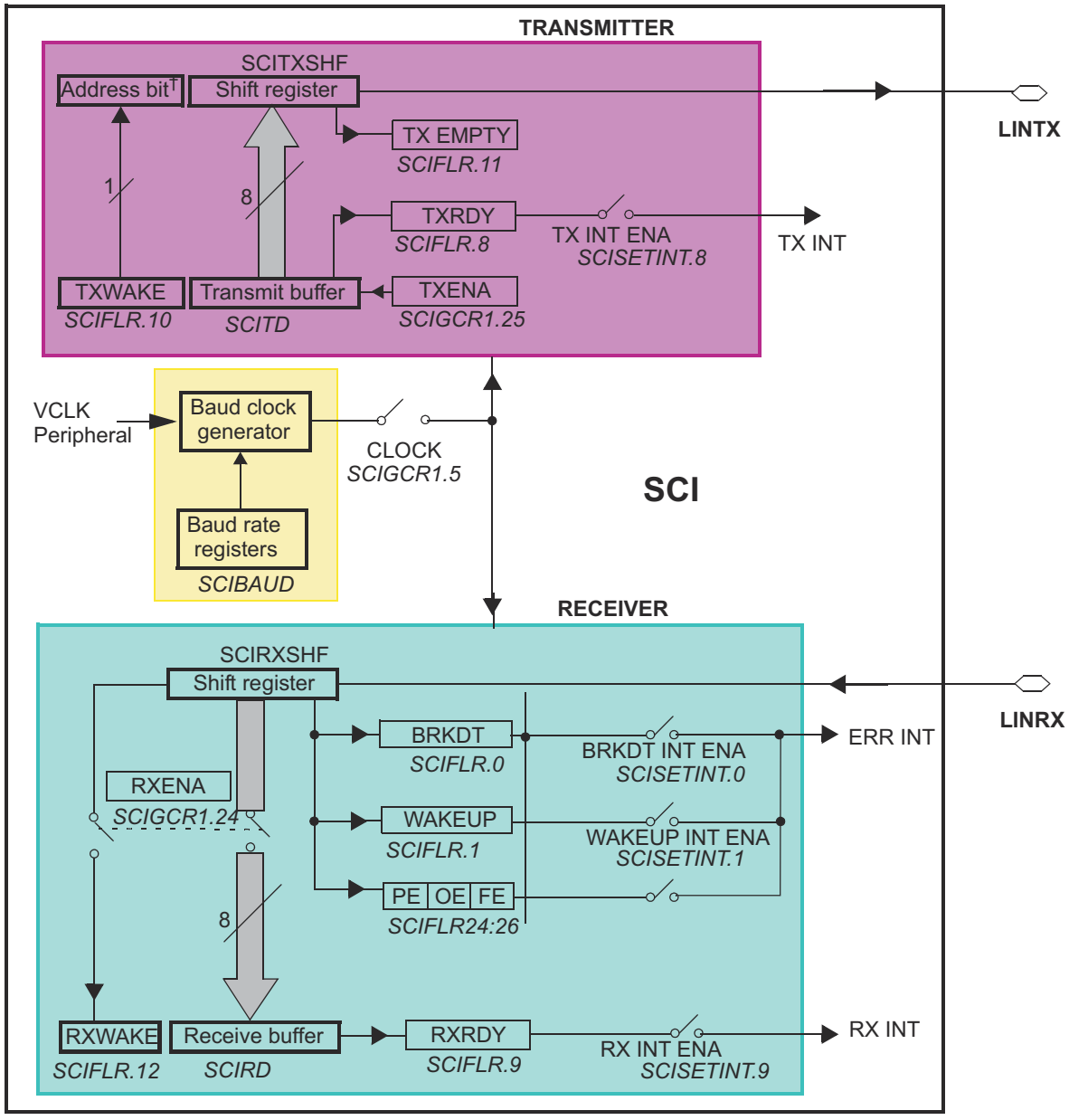


Figure 30-1. SCI Block Diagram

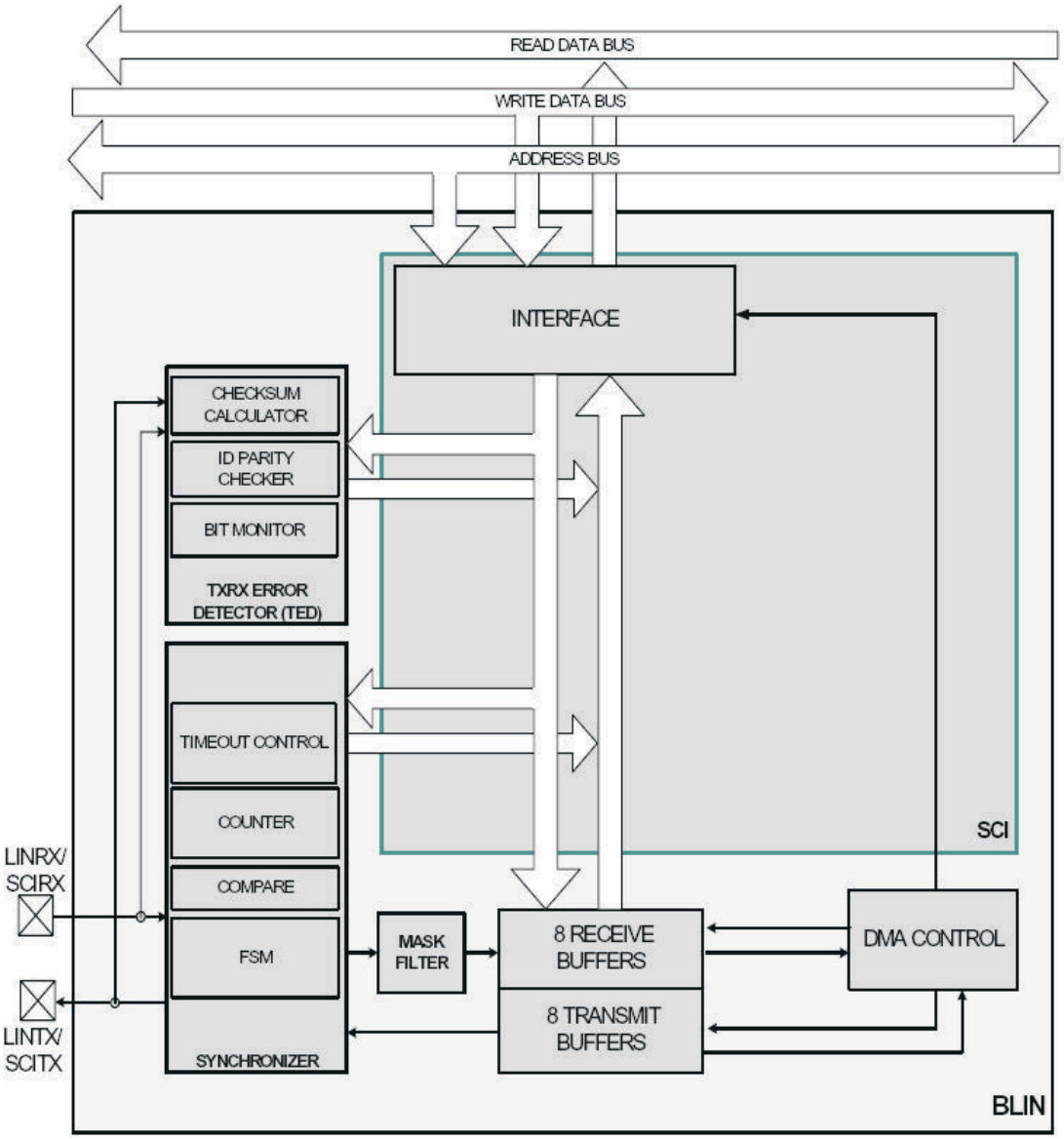


Figure 30-2. SCI/LIN Block Diagram

30.2 Serial Communications Interface Module

30.2.1 SCI Communication Formats

The SCI module can be configured to meet the requirements of many applications. Because communication formats vary depending on the specific application, many attributes of the SCI/LIN are user configurable. The configuration options are:

- SCI Frame format
- SCI Timing modes
- SCI Baud rate
- SCI Multiprocessor modes

30.2.1.1 SCI Frame Formats

The SCI uses a programmable frame format. All frames consist of the following:

- One start bit
- One to eight data bits
- Zero or one address bit
- Zero or one parity bit
- One or two stop bits

The frame format for both the transmitter and receiver is programmable through the bits in the SCIGCR1 register. Both receive and transmit data is in nonreturn to zero (NRZ) format, which means that the transmit and receive lines are at logic high when idle. Each frame transmission begins with a start bit, in which the transmitter pulls the SCI line low (logic low). Following the start bit, the frame data is sent and received least significant bit first (LSB).

An address bit is present in each frame if the SCI is configured to be in address-bit mode but is not present in any frame if the SCI is configured for idle-line mode. The format of frames with and without the address bit is illustrated in Figure 30-3.

A parity bit is present in every frame when the PARITY ENA bit is set. The value of the parity bit depends on the number of one bits in the frame and whether odd or even parity has been selected via the PARITY ENA bit. Both examples in Figure 30-3 have parity enabled.

All frames include one stop bit, which is always a high level. This high level at the end of each frame is used to indicate the end of a frame to ensure synchronization between communicating devices. Two stop bits are transmitted if the STOP bit in SCIGCR1 register is set. The examples shown in Figure 30-3 use one stop bit per frame.

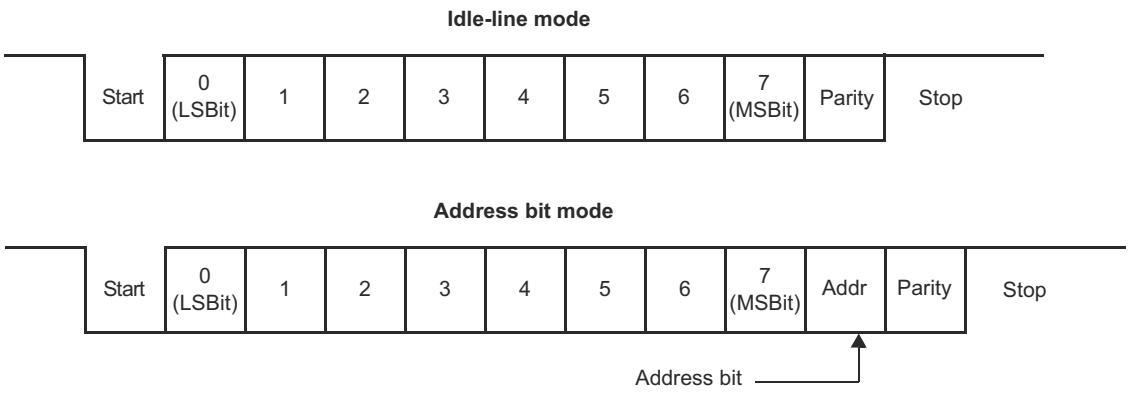


Figure 30-3. Typical SCI Data Frame Formats

30.2.1.2 SCI Asynchronous Timing Mode

The SCI can be configured to use the asynchronous timing mode using TIMING MODE bit in SCIGCR1 register.

The asynchronous timing mode uses only the receive and transmit data lines to interface with devices using the standard universal asynchronous receiver- transmitter (UART) protocol.

In the asynchronous timing mode, each bit in a frame has a duration of 16 SCI baud clock periods. Each bit therefore consists of 16 samples (one for each clock period). When the SCI is using asynchronous mode, the baud rates of all communicating devices must match as closely as possible. Receive errors result from devices communicating at different baud rates.

With the receiver in the asynchronous timing mode, the SCI detects a valid start bit if the first four samples after a falling edge on the LINRX pin are of logic level 0. As soon as a falling edge is detected on LINRX, the SCI assumes that a frame is being received and synchronizes itself to the bus.

To prevent interpreting noise as Start bit SCI expects LINRX line to be low for at least four contiguous SCI baud clock periods to detect a valid start bit. The bus is considered idle if this condition is not met. When a valid start bit is detected, the SCI determines the value of each bit by sampling the LINRX line value during the seventh, eighth, and ninth SCI baud clock periods. A majority vote of these three samples is used to determine the value stored in the SCI receiver shift register. By sampling in the middle of the bit, the SCI reduces errors caused by propagation delays and rise and fall times and data line noises. Figure 30-4 illustrates how the receiver samples a start bit and a data bit in asynchronous timing mode.

The transmitter transmits each bit for a duration of 16 SCI baud clock periods. During the first clock period for a bit, the transmitter shifts the value of that bit onto the LINTX pin. The transmitter then holds the current bit value on LINTX for 16 SCI baud clock periods.

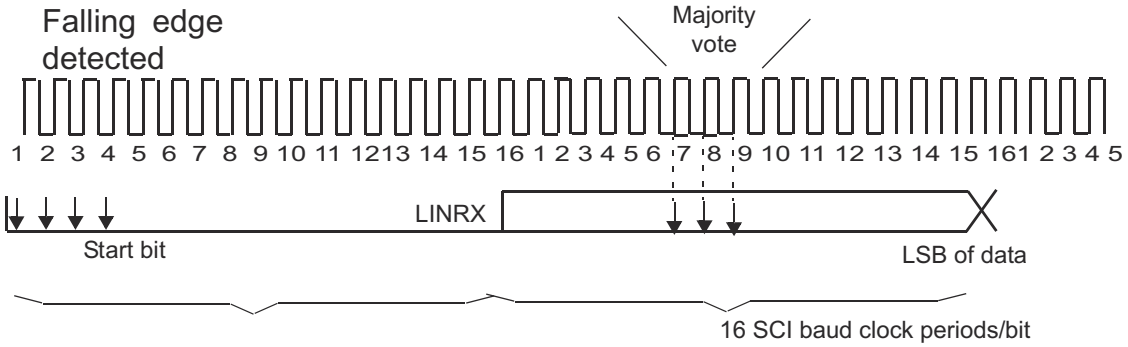


Figure 30-4. Asynchronous Communication Bit Timing

30.2.1.3 SCI Baud Rate

The SCI/LIN has an internally generated serial clock determined by the peripheral VCLK and the prescalers P and M in this register. The SCI uses the 24-bit integer prescaler P value in the BRS register to select the required baud rates. The additional 4-bit fractional divider M refines the baud rate selection.

In asynchronous timing mode, the SCI generates a baud clock according to the following formula:

$$SCICLK\ Frequency = \frac{VCLK\ Frequency}{P + 1 + \frac{M}{16}}$$

$$Asynchronous\ baud\ value = \frac{SCICLK\ Frequency}{16}$$

For P = 0,

$$Asynchronous\ baud\ value = \frac{VCLK\ Frequency}{32} \quad (30)$$

Table 30-1. P and M Values for Standard Bit Rates

100 MHz Device, LIN Module Input Clock = 50 MHz				
Desired Bit Rate	**Actual Bit Rate**	**Percentage Error**	**P Value**	**M Value**
115200	115207.4	0.0064	26	2
57600	57603.69	0.0064	53	4
38400	38402.46	0.0064	80	6
19200	19201.23	0.0064	161	12
9600	9600.61	0.0064	324	8
4800	4799.85	0.0032	650	1
2400	2400.04	0.0016	1301	1
1200	1199.99	0.0008	2603	1

30.2.1.3.1 Superfractional Divider, SCI Asynchronous Mode

The superfractional divider is available in SCI asynchronous mode (idle-line and address-bit mode). Building on the 4-bit fractional divider M (BRS[27:24]), the superfractional divider uses an additional 3-bit modulating value (see Table 30-3). The bits with a 1 in the table will have an additional VCLK period added to their T_{bit}. If the character length is more than 10, then the modulation table will be a rolled-over version of the original table (Table 30-2), as shown in Table 30-3.

The baud rate will vary over a data field to average according to the BRS[30:28] value by a "**d**" fraction of the peripheral internal clock: **0<d<1.** See Figure 30-5 for a simple Average "d' calculation based on "U" value (BRS[30:28]).

The instantaneous bit time is expressed in terms of T_{VCLK} as follows:

For all P other than 0, and all M and d (0 or 1),

$$T^{i}bit = \left[16\left(P + 1 + \frac{M}{16}\right) + d\right]T_{VCLK} \tag{31}$$

For P = 0 $T_{bit} = 32T_{VCLK}$

The averaged bit time is expressed in terms of T_{VCLK} as follows:

For all P other than 0, and all M and d (0<d<1),

$$T^{a}bit = \left[16\left(P + 1 + \frac{M}{16}\right) + d\right]T_{VCLK} \tag{32}$$

For P = 0 $T_{bit} = 32T_{VCLK}$

Table 30-2. Superfractional Bit Modulation for SCI Mode (Normal Configuration)

Normal Configuration = Start Bit + 8 Data Bits + Stop Bit										
BRS[30:28]	**Start Bit**	**D[0]**	**D[1]**	**D[2]**	**D[3]**	**D[4]**	**D[5]**	**D[6]**	**D[7]**	**Stop Bit**
0h	0	0	0	0	0	0	0	0	0	0
1h	1	0	0	0	0	0	0	0	1	0
2h	1	0	0	0	1	0	0	0	1	0
3h	1	0	1	0	1	0	0	0	1	0
4h	1	0	1	0	1	0	1	0	1	0
5h	1	1	1	0	1	0	1	0	1	1
6h	1	1	1	0	1	1	1	0	1	1
7h	1	1	1	1	1	1	1	0	1	1

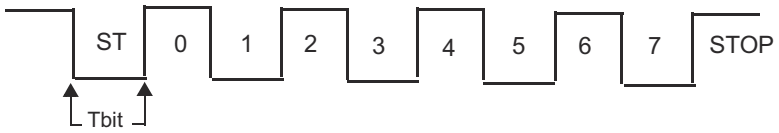
Table 30-3. Superfractional Bit Modulation for SCI Mode (Maximum Configuration)

Maximum Configuration = Start Bit + 8 Data Bits + Addr Bit + Parity Bit + Stop Bit 0 + Stop Bit 1													
BRS[30:28]	Start Bit	D[0]	D[1]	D[2]	D[3]	D[4]	D[5]	D[6]	D[7]	Addr	Parity	Stop0	Stop1
0h	0	0	0	0	0	0	0	0	0	0	0	0	0
1h	1	0	0	0	0	0	0	0	1	0	0	0	0
2h	1	0	0	0	1	0	0	0	1	0	0	0	1
3h	1	0	1	0	1	0	0	0	1	0	1	0	1
4h	1	0	1	0	1	0	1	0	1	0	1	0	1
5h	1	1	1	0	1	0	1	0	1	1	1	0	1
6h	1	1	1	0	1	1	1	0	1	1	1	0	1
7h	1	1	1	1	1	1	1	0	1	1	1	1	1

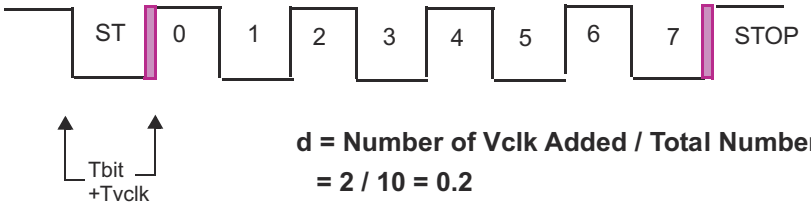
Table 30-4. SCI Mode (Minimum Configuration)

Minimum Configuration = Start Bit + 1 Data Bit + Stop Bit			
BRS[30:28]	Start Bit	D[0]	Stop Bit
0h	0	0	0
1h	1	0	0
2h	1	0	0
3h	1	0	1
4h	1	0	1
5h	1	1	1
6h	1	1	1
7h	1	1	1

Normal Data Frame with BRS[31:28] = 0



Normal Data Frame with BRS[31:28] = 1



d = Number of Vclk Added / Total Number of Bits
= 2 / 10 = 0.2

Figure 30-5. Superfractional Divider Example

Table 30-5. Comparative Baud Values for Different P Values, Asynchronous Mode

VCLK = 50 MHz				
24-Bit Register Value		Baud Selected		
Decimal	Hex	Ideal	Actual	Percent Error
26	00001A	115200	115740	0.47
53	000035	57600	57870	0.47
80	000050	38400	38580	0.47
162	0000A2	19200	19172	-0.15
299	00012B	10400	10417	0.16
325	000145	9600	9586	-0.15
399	00018F	7812.5	7812.5	0.00
650	00028A	4800	4800	0.00
15624	003BA0	200	200	0.00
624999	098967	5	5	0.00

30.2.1.4 SCI Multiprocessor Communication Modes

In some applications, the SCI may be connected to more than one serial communication device. In such a multiprocessor configuration, several frames of data may be sent to all connected devices or to an individual device. In the case of data sent to an individual device, the receiving devices must determine when they are being addressed. When a message is not intended for them, the devices can ignore the following data. When only two devices make up the SCI network, addressing is not needed, so multiprocessor communication schemes are not required.

SCI supports two multiprocessor communication modes which can be selected using COMM MODE bit:

- Idle-Line Mode
- Address Bit Mode

When the SCI is not used in a multiprocessor environment, software can consider all frames as data frames. In this case, the only distinction between the idle-line and address-bit modes is the presence of an extra bit (the address bit) in each frame sent with the address-bit protocol.

The SCI allows full-duplex communication where data can be sent and received via the transmit and receive pins simultaneously. However, the protocol used by the SCI assumes that only one device transmits data on the same bus line at any one time. No arbitration is done by the SCI.

30.2.1.4.1 Idle-Line Multiprocessor Modes

In idle-line multiprocessor mode, a frame that is preceded by an idle period (10 or more idle bits) is an address frame. A frame that is preceded by fewer than 10 idle bits is a data frame. Figure 30-6 illustrates the format of several blocks and frames with idle-line mode.

There are two ways to transmit an address frame using idle-line mode:

Method 1: In software, deliberately leave an idle period between the transmission of the last data frame of the previous block and the address frame of the new block.

Method 2: Configure the SCI to automatically send an idle period between the last data frame of the previous block and the address frame of the new block.

Although Method 1 is only accomplished by a delay loop in software, Method 2 can be implemented by using the transmit buffer and the TXWAKE bit in the following manner:

Step 1: Write a 1 to the TXWAKE bit.

Step 2: Write a dummy data value to the SCITD register. This triggers the SCI to begin the idle period as soon as the transmitter shift register is empty.

Step 3: Wait for the SCI to clear the TXWAKE flag.

Step 4: Write the address value to SCITD.

As indicated by Step 3, software should wait for the SCI to clear the TXWAKE bit. However, the SCI clears the TXWAKE bit at the same time it sets TXRDY (that is, transfers data from SCITD into SCITXSHF). Therefore, if the TX INT ENA bit is set, the transfer of data from SCITD to SCITXSHF causes an interrupt to be generated at the same time that the SCI clears the TXWAKE bit. If this interrupt method is used, software is not required to poll the TXWAKE bit waiting for the SCI to clear it.

When idle-line multiprocessor communications are used, software must ensure that the idle time exceeds 10 bit periods before addresses (using one of the methods mentioned above), and software must also ensure that data frames are written to the transmitter quickly enough to be sent without a delay of 10 bit periods between frames. Failure to comply with these conditions will result in data interpretation errors by other devices receiving the transmission.

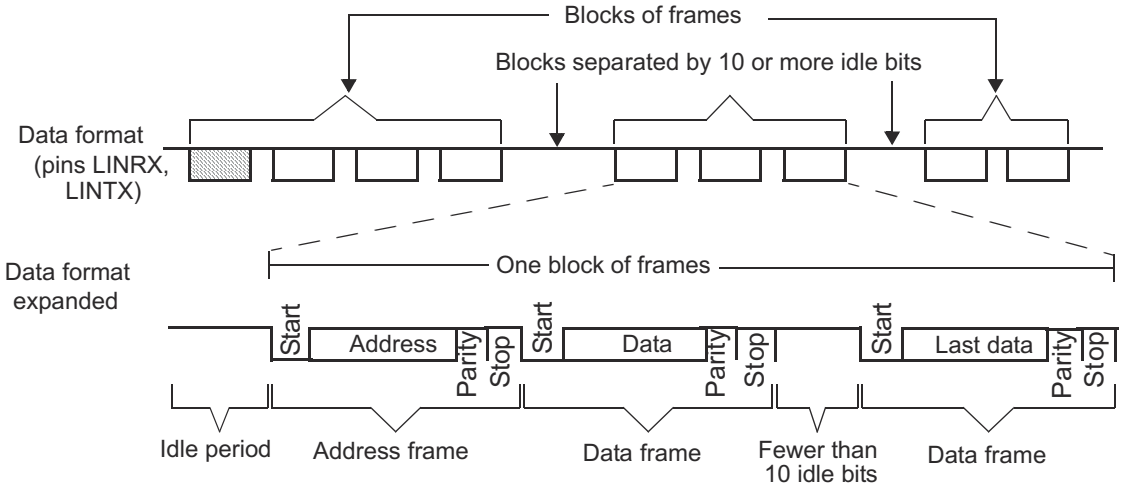


Figure 30-6. Idle-Line Multiprocessor Communication Format

30.2.1.4.2 Address-Bit Multiprocessor Mode

In the address-bit protocol, each frame has an extra bit immediately following the data field called an address bit. A frame with the address bit set to 1 is an address frame; a frame with the address bit set to 0 is a data frame. The idle period timing is irrelevant in this mode. Figure 30-7 illustrates the format of several blocks and frames with the address-bit mode.

When address-bit mode is used, the value of the TXWAKE bit is the value sent as the address bit. To send an address frame, software must set the TXWAKE bit. This bit is cleared as the contents of the SCITD are shifted from the TXWAKE register so that all frames sent are data except when the TXWAKE bit is written as a 1.

No dummy write to SCITD is required before an address frame is sent in address-bit mode. The first byte written to SCITD after the TXWAKE bit is written to 1 is transmitted with the address bit set when address-bit mode is used.

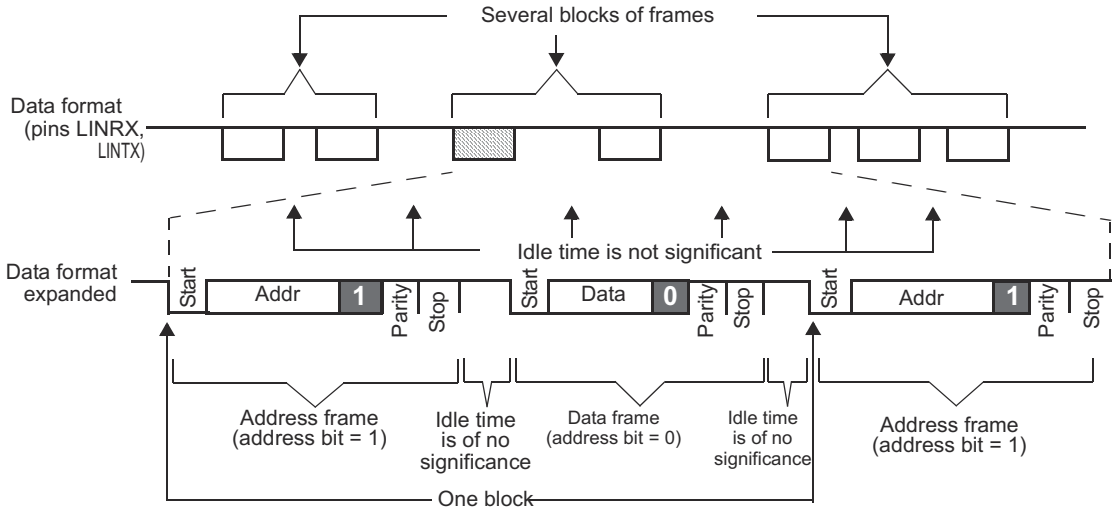


Figure 30-7. Address-Bit Multiprocessor Communication Format

30.2.1.5 SCI Multi Buffered Mode

To reduce CPU load when Receiving or Transmitting data in interrupt mode or DMA mode, the SCI/LIN module has eight separate receive and transmit buffers. Multi buffered mode is enabled by setting the MBUF MODE bit.

The multi-buffer 3-bit counter counts the data bytes transferred from the SCIRXSHF register to the RDy receive buffers and TDy transmit buffers register to SCITXSHF register. The 3-bit compare register contains the number of data bytes expected to be received or transmitted. the LENGTH value in SCIFORMAT register indicates the expected length and is used to load the 3-bit compare register.

A receive interrupt (RX interrupt; see the SCIINTVECT0 and SCIINTVECT1registers), and a receive ready RXRDY flag set in SCIFLR register, as well as a DMA request (RXDMA) could occur after receiving a response if there are no response receive errors for the frame (such as, there is, frame error, and overrun error).

A transmit interrupt (TX interrupt), and a transmit ready flag (TXRDY flag in SCIFLR register), and a DMA request (TXDMA) could occur after transmitting a response.

Figure 30-8 and Figure 30-9 show the receive and transmit multi-buffer functional block diagram, respectively.

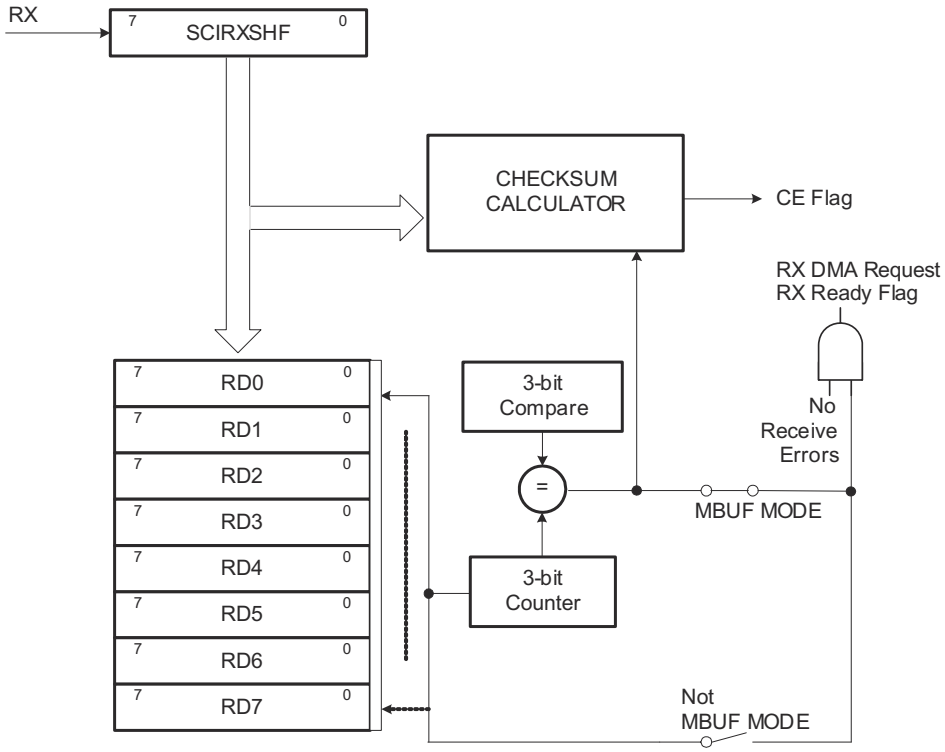


Figure 30-8. Receive Buffers

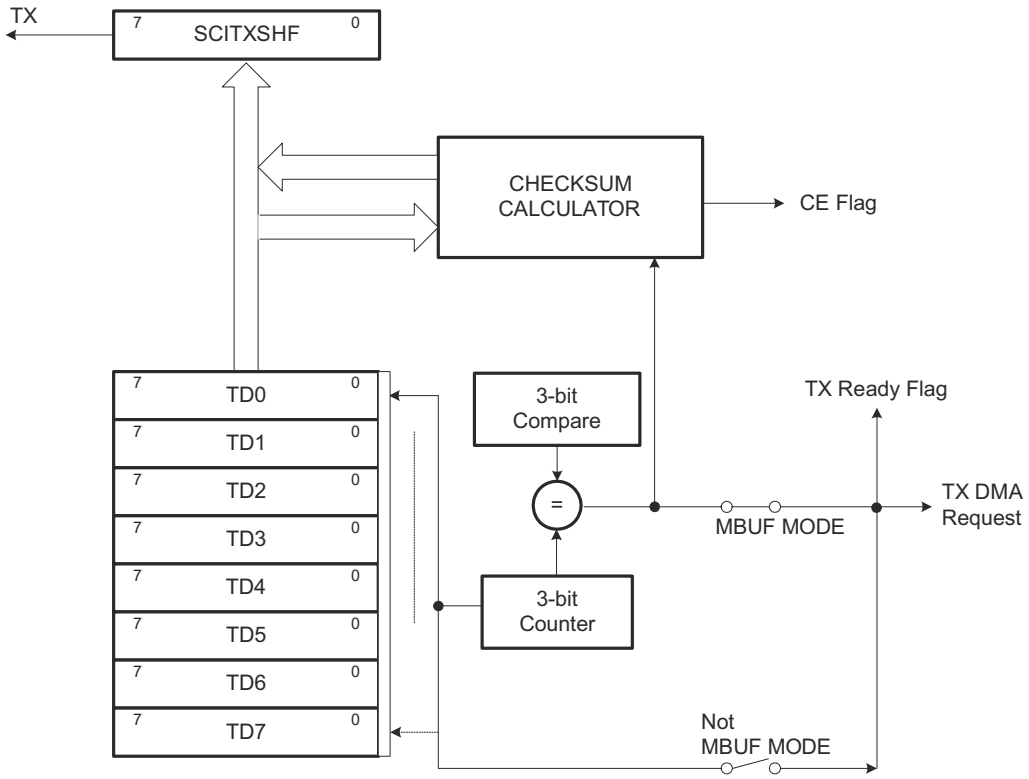


Figure 30-9. Transmit Buffers

30.2.2 SCI Interrupts

The SCI/LIN module has two interrupt lines, level 0 and level 1, to the vectored interrupt manager (VIM) module (see Figure 30-10). Two offset registers SCIINTVECT0 and SCIINTVECT1 determine which flag triggered the interrupt according to the respective priority encoders. Each interrupt condition has a bit to enable/disable the interrupt in the SCISETINT and SCICLRINT registers, respectively.

Each interrupt also has a bit that can be set as interrupt level 0(INT0) or as interrupt level 1(INT1). By default, interrupts are in interrupt level 0. SCISETINTLVL sets a given interrupt to level1. SCICLEARINTLVL resets a given interrupt level to the default level 0.

The interrupt vector registers SCIINTVECT0 and SCIINTVECT1 return the vector of the pending interrupt line INT0 or INT1. If more than one interrupt is pending, the interrupt vector register holds the highest priority interrupt.

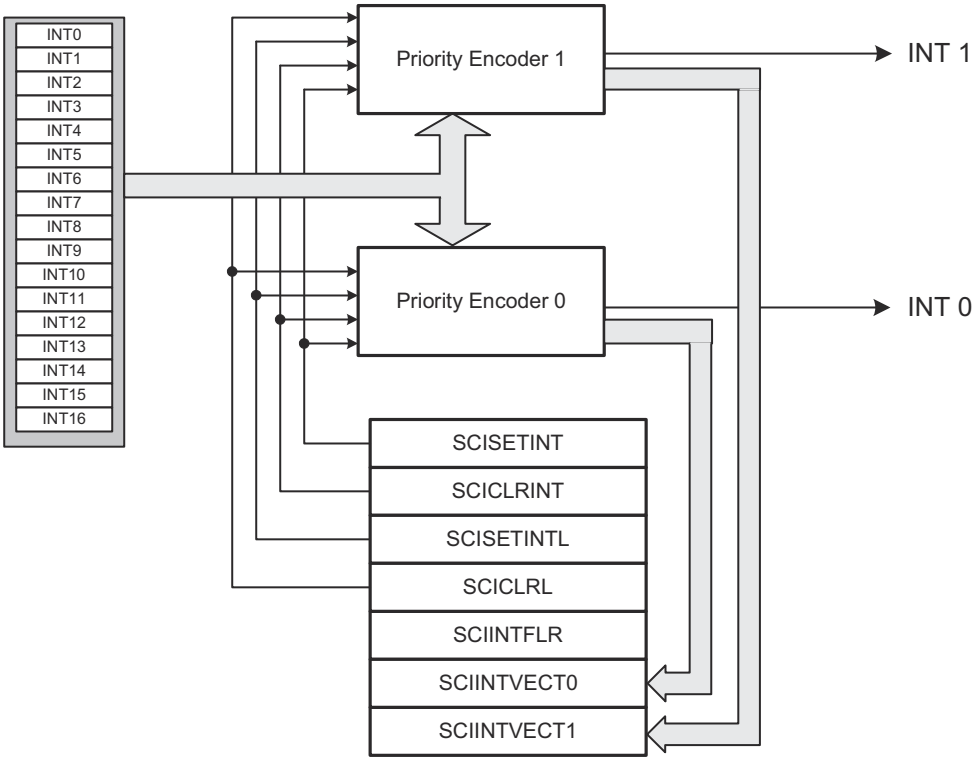


Figure 30-10. General Interrupt Scheme

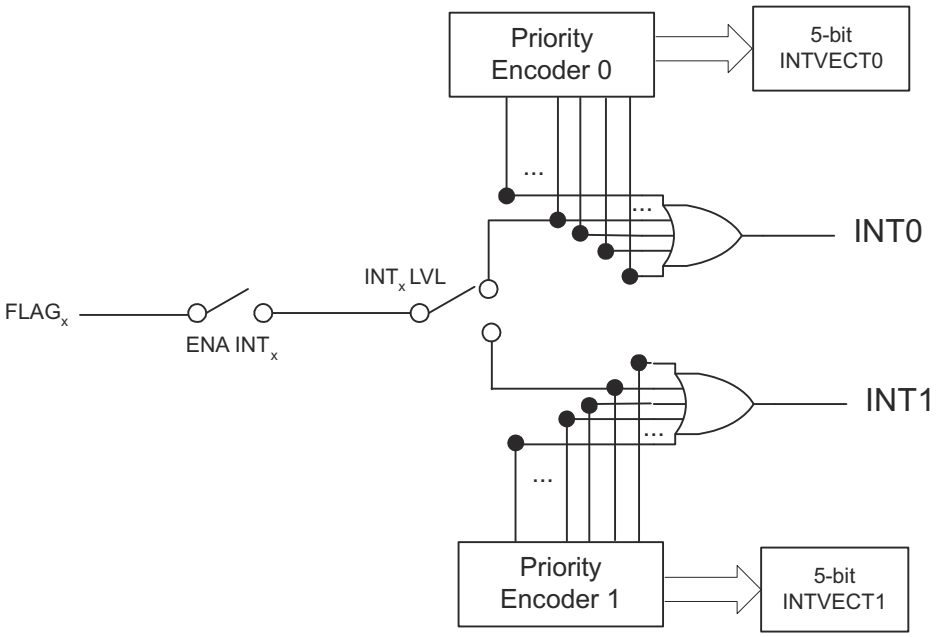


Figure 30-11. Interrupt Generation for Given Flags

30.2.2.1 Transmit Interrupt

To use transmit interrupt functionality, SET TX INT bit must be enabled . The transmit ready (TXRDY) flag is set when the SCI transfers the contents of SCITD to the shift register, SCITXSHF. The TXRDY flag indicates that SCITD is ready to be loaded with more data. In addition, the SCI sets the TX EMPTY bit if both the SCITD and SCITXSHF registers are empty. If the SET TX INT bit is set, then a transmit interrupt is generated when the TXRDY flag goes high. Transmit Interrupt is not generated immediately after setting the SET TX INT . Transmit Interrupt is generated only after the first transfer from SCITD to SCITXSHF, that is first data has to be written to SCITD before any interrupt gets generated. To transmit further data, data can be written to SCITD in the transmit Interrupt service routine.

Writing data to the SCITD register clears the TXRDY bit. When this data has been moved to the SCITXSHF register, the TXRDY bit is set again. The interrupt request can be suspended by setting the CLR TX INT bit; however, when the SET TX INT bit is again set to 1, the TXRDY interrupt is asserted again. The transmit interrupt request can be eliminated until the next series of values is written to SCITD, by disabling the transmitter via the TXENA bit, by a software reset SWnRST, or by a device hardware reset.

30.2.2.2 Receive Interrupt

The receive ready (RXRDY) flag is set when the SCI transfers newly received data from SCIRXSHF to SCIRD. The RXRDY flag therefore indicates that the SCI has new data to be read. Receive interrupts are enabled by the SET RX INT bit. If the SET RX INT is set when the SCI sets the RXRDY flag, then a receive interrupt is generated. The received data can be read in the Interrupt Service routine.

30.2.2.3 WakeUp Interrupt

SCI sets the WAKEUP flag if bus activity on the RX line either prevents power-down mode from being entered, or RX line activity causes an exit from power-down mode. If enabled (SET WAKEUP INT), wakeup interrupt is triggered once WAKEUP flag is set.

30.2.2.4 Error Interrupts

The following error detections are supported with an interrupt by the SCI module:

- Parity errors (PE)
- Frame errors (FE)
- Break Detect errors (BRKDT)
- Overrun errors (OE)
- Bit errors (BE)

If all of these errors (PE, FE, BRKDT, OE, BE) are flagged, an interrupt for the flagged errors will be generated if enabled. A message is valid for both the transmitter and the receiver, if there is no error detected until the end of the frame. Each of these flags is located in the receiver status (SCIFLR) register.

There are 16 interrupt sources in the SCI/LIN module. In SCI mode, 8 interrupts are supported, as listed in Table 30-6.

Table 30-6. SCI/LIN Interrupts

Offset(1)	Interrupt	Applicable to SCI	Applicable to LIN
0	No interrupt	-	-
1	Wakeup	Yes	Yes
2	Inconsistent-sync-field error	No	Yes
3	Parity error	Yes	Yes
4	ID	No	Yes
5	Physical bus error	No	Yes
6	Frame error	Yes	Yes
7	Break detect	Yes	No
8	Checksum error	No	Yes
9	Overrun error	Yes	Yes
10	Bit error (BE)	Yes	Yes
11	Receive	Yes	Yes
12	Transmit	Yes	Yes
13	No-response error	No	Yes
14	Timeout after wakeup signal (150 ms)	No	Yes
15	Timeout after three wakeup signals (1.5 s)	No	Yes
16	Timeout (Bus Idle, 4s)	No	Yes

(1) Offset 1 is the highest priority. Offset 16 is the lowest priority.

Table 30-7. SCI Receiver Status Flags

SCI Flag	Register	Bit	Value After Reset(1)
CE	SCIFLR	29	0
ISFE	SCIFLR	28	0
NRE	SCIFLR	27	0
FE	SCIFLR	26	0
OE	SCIFLR	25	0
PE	SCIFLR	24	0
RXWAKE	SCIFLR	12	0
RXRDY	SCIFLR	9	0
BUSY	SCIFLR	3	0
IDLE	SCIFLR	2	1
WAKEUP	SCIFLR	1	0
BRKDT	SCIFLR	0	0

(1) The flags are frozen with their reset value while SWnRST = 0.

Table 30-8. SCI Transmitter Status Flags

SCI Flag	Register	Bit	Value After Reset(1)
BE	SCIFLR	31	0
PBE	SCIFLR	30	0
TXWAKE	SCIFLR	10	0
TXEMPTY	SCIFLR	11	1
TXRDY	SCIFLR	8	1

(1) The flags are frozen with their reset value while SWnRST = 0.

30.2.3 SCI DMA Interface

DMA requests for receive (RXDMA request) and transmit (TXDMA request) are available for the SCI/LIN module. The DMA transfers depending on whether multi-buffer mode bit (MBUF MODE) is enabled or not enabled.

30.2.3.1 Receive DMA Requests

This DMA functionality is enabled/disabled by the CPU using the SET RX DMA/CLR RX DMA bits, respectively.

In Multi-Buffered SCI mode with DMA enabled, the receiver loads the RDy buffers for each received character. RX DMA request is triggered once the last character of the programmed number of characters (LENGTH) are received and copied to the corresponding RDy buffer successfully.

If the multi-buffer option is disabled, then DMA requests will be generated on a byte-per-byte basis.

In multiprocessor mode, the SCI can generate receiver interrupts for address frames and DMA requests for data frames or DMA requests for both. This is controlled by the SET RX DMA ALL bit.

In multiprocessor mode with the SLEEP bit set, no DMA is generated for received data frames. The software must clear the SLEEP bit before data frames can be received.

30.2.3.2 Transmit DMA Requests

DMA functionality is enabled/disabled by the CPU with SET TX DMA/CLR TX DMA bits, respectively.

In Multi-Buffered SCI mode once TXRDY bit is set or after a transmission of programmed number of characters (LENGTH) (up to eight data bytes stored in the transmit buffer(s) TDy in the LINTD0 and LINTD1 registers), a DMA request is generated in order to reload the transmit buffer for the next transmission. If the multi-buffer option is disabled, then DMA requests will be generated on a byte-per-byte basis.

30.2.4 SCI Configurations

Before the SCI sends or receives data, its registers should be properly configured. Upon power-up or a system-level reset, each bit in the SCI registers is set to a default state. The registers are writable only after the RESET bit in the SCIGCR0 register is set to 1. Of particular importance is the SWnRST bit in the SCIGCR1 register. The SWnRST is an active-low bit initialized to 0 and keeps the SCI in a reset state until it is programmed to 1. Therefore, all SCI configuration should be completed before a 1 is written to the SWnRST bit.

The following list details the configuration steps that software should perform prior to the transmission or reception of data. As long as the SWnRST bit is cleared to 0 the entire time that the SCI is being configured, the order in which the registers are programmed is not important.

- Enable SCI by setting the RESET bit to 1.
- Clear the SWnRST bit to 0 before SCI is configured.
- Select the desired frame format by programming the SCIGCR1 register.
- Set both the RX FUNC and TX FUNC bits in SCIPIO0 to 1 to configure the LINRX and LINTX pins for SCI functionality.
- Select the baud rate to be used for communication by programming the BRS register.
- Set the CLOCK bit in SCIGCR1 to 1 to select the internal clock.
- Set the CONT bit in SCIGCR1 to 1 to make SCI not halt for an emulation breakpoint until its current reception or transmission is complete (this bit is used only in an emulation environment).
- Set the LOOP BACK bit in SCIGCR1 to 1 to connect the transmitter to the receiver internally (this feature is used to perform a self-test).
- Set the RXENA bit in SCIGCR1 to 1, if data is to be received.
- Set the TXENA bitin SCIGCR1 to 1, if data is to be transmitted.
- Set the SWnRST bit to 1 after SCI is configured.
- Perform receiving or transmitting data (see Section 30.2.4.1 or Section 30.2.4.2).

30.2.4.1 Receiving Data

The SCI receiver is enabled to receive messages if both the RX FUNC bit and the RXENA bit are set to 1. If the RX FUNC bit is not set, the LINRX pin functions as a general-purpose I/O pin rather than as an SCI function pin.

SCI module can receive data in one of the following modes:

- Single-Buffer (Normal) Mode
- Multi-Buffer Mode

After a valid idle period is detected, data is automatically received as it arrives on the LINRX pin.

30.2.4.1.1 Receiving Data in Single-Buffer Mode

Single-buffer mode is selected when the MBUF MODE bit in SCIGCR1 is cleared to 0. In this mode, SCI sets the RXRDY bit when it transfers newly received data from SCIRXSHF to SCIRD. The SCI clears the RXRDY bit after the new data in SCIRD has been read. Also, as data is transferred from SCIRXSHF to SCIRD, the SCI sets the FE, OE, or PE flags if any of these error conditions were detected in the received data. These error conditions are supported with configurable interrupt capability. The wake-up and break-detect status bits are also set if one of these errors occurs, but they do not necessarily occur at the same time that new data is being loaded into SCIRD.

You can receive data by:

1. Polling Receive Ready Flag
2. Receive Interrupt
3. DMA

In polling method, software can poll for the RXRDY bit and read the data from the SCIRD register once the RXRDY bit is set high. The CPU is unnecessarily overloaded by selecting the polling method. To avoid this, you can use either the interrupt or DMA method. To use the interrupt method, the SET RX INT bit is set. To use the DMA method, the SET RX DMA bit is set. Either an interrupt or a DMA request is generated the moment the RXRDY bit is set.

30.2.4.1.2 Receiving Data in Multi-Buffer Mode

Multi-buffer mode is selected when the MBUFMODE bit in SCIGCR1 is set to 1. In this mode, SCI sets the RXRDY bit after receiving the programmed number of data in the receive buffer, the complete frame. The error condition detection logic is similar to the single-buffer mode, except that it monitors for the complete frame. Like single-buffer mode, you can use the polling or interrupt method to read the data. The SCI clears the RXRDY bit after the new data in SCIRD has been read.

30.2.4.2 Transmitting Data

The SCI transmitter is enabled if both the TX FUNC bit and the TXENA bit are set to 1. If the TX FUNC bit is not set, the LINTX pin functions as a general-purpose I/O pin rather than as an SCI function pin. Any value written to the SCITD before TXENA is set to 1 is not transmitted. Both of these control bits allow for the SCI transmitter to be held inactive independently of the receiver.

SCI module can transmit data in one of the following modes:

- Single-Buffer (Normal) Mode
- Multi-Buffered or Buffered SCI Mode

30.2.4.2.1 Transmitting Data in Single-Buffer Mode

Single-buffer mode is selected when the MBUF MODE bit in SCIGCR1 is cleared to 0. In this mode, SCI waits for data to be written to SCITD, transfers it to SCITXSHF, and transmits the data. The TXRDY and TX EMPTY bits indicate the status of the transmit buffers. That is, when the transmitter is ready for data to be written to SCITD, the TXRDY bit is set. Additionally, if both SCITD and SCITXSHF are empty, then the TX EMPTY bit is also set.

You can transmit data by:

1. Polling Transmit Ready Flag
2. Transmit Interrupt

In polling method, software can poll for the TXRDY bit to go high before writing the data to the SCITD register. The CPU is unnecessarily overloaded by selecting the polling method. To avoid this, you can use the interrupt or DMA method. To use the interrupt method, the SET TX INT bit is set. To use the DMA method, the SET TX DMA bit is set. Either an interrupt or a DMA request is generated the moment the TXRDY bit is set. When the SCI has completed transmission of all pending frames, the SCITXSHF register and SCITD are empty, the TXRDY bit is set, and an interrupt/DMA request is generated, if enabled. Because all data has been transmitted, the interrupt/DMA request should be halted. This can either be done by disabling the transmit interrupt (CLR TX INT) / DMA request (CLR TX DMA bit), or by disabling the transmitter (clear TXENA bit).

Note

The TXRDY flag cannot be cleared by reading the corresponding interrupt offset in the SCIINTVECT0 or SCIINTVECT1 register.

30.2.4.2.2 Transmitting Data in Multi-Buffer Mode

Multi-buffer mode is selected when the MBUF MODE bit in SCIGCR1 is set to 1. Like single-buffer mode, you can use the polling or interrupt method to write the data to be transmitted. The transmitted data has to be written to the SCITD registers. SCI waits for data to be written to the SCITD register and transfers the programmed number of bytes to SCITXSHF to transmit one by one automatically.

30.2.5 SCI Low-Power Mode

The SCI/LIN can be put in either local or global low-power mode. Global low-power mode is asserted by the system and is not controlled by the SCI/LIN. During global low-power mode, all clocks to the SCI/LIN are turned off so the module is completely inactive.

Local low-power mode is asserted by setting the POWERDOWN bit; setting this bit stops the clocks to the SCI/LIN internal logic and the module registers. Setting the POWERDOWN bit causes the SCI to enter local low-power mode and clearing the POWERDOWN bit causes SCI/LIN to exit from local low-power mode. All the registers are accessible during local power-down mode as any register access enables the clock to SCI for that particular access alone.

The wake-up interrupt is used to allow the SCI to exit low-power mode automatically when a low level is detected on the LINRX pin and also this clears the POWERDOWN bit. If wake-up interrupt is disabled, then the SCI/LIN immediately enters low-power mode whenever it is requested and also any activity on the LINRX pin does not cause the SCI to exit low-power mode.

Note

Enabling Local Low-Power Mode During Receive and Transmit

If the wake-up interrupt is enabled and low-power mode is requested while the receiver is receiving data, then the SCI immediately generates a wake-up interrupt to clear the powerdown bit and prevents the SCI from entering low-power mode and thus completes the current reception. Otherwise, if the wake-up interrupt is disabled, then the SCI completes the current reception and then enters the low-power mode.

30.2.5.1 Sleep Mode for Multiprocessor Communication

When the SCI receives data and transfers that data from SCIRXSHF to SCIRD, the RXRDY bit is set and if RX INT ENA is set, the SCI also generates an interrupt. The interrupt triggers the CPU to read the newly received frame before another one is received. In multiprocessor communication modes, this default behavior may be enhanced to provide selective indication of new data. When SCI receives an address frame that does not match its address, the device can ignore the data following this non-matching address until the next address frame by using sleep mode. Sleep mode can be used with both idle-line and address-bit multiprocessor modes.

If sleep mode is enabled by the SLEEP bit, then the SCI transfers data from SCIRXSHF to SCIRD only for address frames. Therefore, in sleep mode, all data frames are assembled in the SCIRXSHF register without being shifted into the SCIRD and without initiating a receive interrupt request. Upon reception of an address frame, the contents of the SCIRXSHF are moved into SCIRD, and the software must read SCIRD and determine if the SCI is being addressed by comparing the received address against the address previously set in the software and stored somewhere in memory (the SCI does not have hardware available for address comparison). If the SCI is being addressed, the software must clear the SLEEP bit so that the SCI will load SCIRD with the data of the data frames that follow the address frame.

When the SCI has been addressed and sleep mode has been disabled (in software) to allow the receipt of data, the SCI should check the RXWAKE bit (SCIFLR.12) to determine when the next address has been received. This bit is set to 1 if the current value in SCIRD is an address and set to 0 if SCIRD contains data. If the RXWAKE bit is set, then software should check the address in SCIRD against its own address. If it is still being addressed, then sleep mode should remain disabled. Otherwise, the SLEEP bit should be set again.

Following is a sequence of events typical of sleep mode operation:

- The SCI is configured and both sleep mode and receive actions are enabled.
- An address frame is received and a receive interrupt is generated.
- Software compares the received address frame against that set by software and determines that the SCI is not being addressed, so the value of the SLEEP bit is not changed.
- Several data frames are shifted into SCIRXSHF, but no data is moved to SCIRD and no receive interrupts are generated.
- A new address frame is received and a receive interrupt is generated.
- Software compares the received address frame against that set by software and determines that the SCI is being addressed and clears the SLEEP bit.
- Data shifted into SCIRXSHF is transferred to SCIRD, and a receive interrupt is generated after each data frame is received.
- In each interrupt routine, software checks RXWAKE to determine if the current frame is an address frame.
- Another address frame is received, RXWAKE is set, software determines that the SCI is not being addressed and sets the SLEEP bit back to 1. No receive interrupts are generated for the data frames following this address frame.

By ignoring data frames that are not intended for the device, fewer interrupts are generated. These interrupts would otherwise require CPU intervention to read data that is of no significance to this specific device. Using sleep mode can help free some CPU resources.

Except for the RXRDY flag, the SCI continues to update the receiver status flags (see Table 30-7) while sleep mode is active. In this way, if an error occurs on the receive line, an application can immediately respond to the error and take the appropriate corrective action.

Because the RXRDY bit is not updated for data frames when sleep mode is enabled, the SCI can enable sleep mode and use a polling algorithm if desired. In this case, when RXRDY is set, software knows that a new address has been received. If the SCI is not being addressed, then the software should not change the value of the SLEEP bit and should continue to poll RXRDY.

30.3 Local Interconnect Network Module

30.3.1 LIN Communication Formats

The SCI/LIN module can be used in LIN mode or SCI mode. The enhancements for baud generation, DMA controls, and additional receive/transmit buffers necessary for LIN mode operation are also part of the enhanced buffered SCI module. LIN mode is selected by enabling LIN MODE bit in SCIGCR1 register.

Note

The SCI/LIN is built around the SCI platform and uses a similar sampling scheme: 16 samples for each bit with majority vote on samples 8, 9, and 10. For the START bit, the first three samples are used.

The SCI/LIN control registers are located at the SCI/LIN base address. For a detailed description of each register, seeSection 30.7.

30.3.1.1 LIN Standards

For compatibility with LIN2.0 standard the following additional features are implemented over LIN1.3:

1. Support for LIN 2.0 checksum
2. Enhanced synchronizer FSM support for frame processing
3. Enhanced handling of extended frames
4. Enhanced baud rate generator
5. Update wakeup/go to sleep

The LIN module covers the CPU performance-consuming features, defined in the *LIN Specification Package* Revision 1.3 and 2.0 by hardware.

30.3.1.2 Message Frame

The LIN protocol defines a message frame format, illustrated in Figure 30-12. Each frame includes one master header, one response, one in-frame response space, and inter-byte spaces. In-frame-response and inter-byte spaces may be 0.

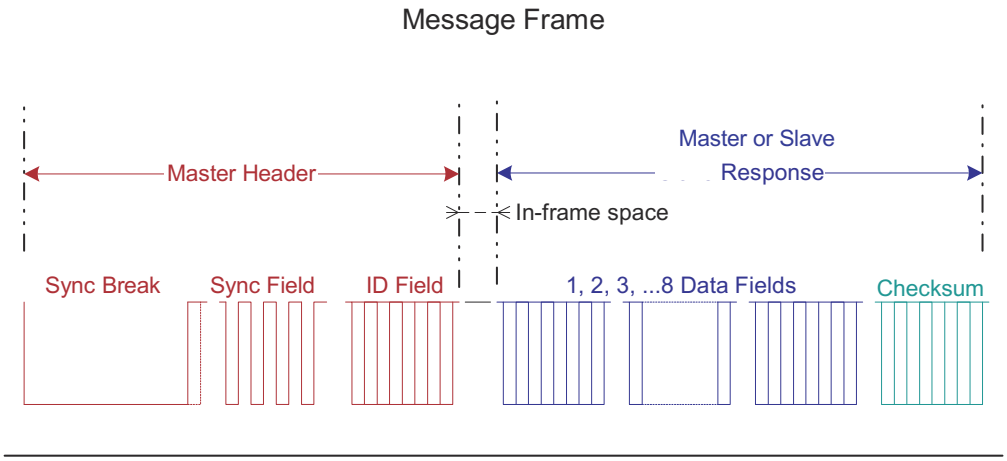


Figure 30-12. LIN Protocol Message Frame Format: Master Header and Slave Response

There is no arbitration in the definition of the LIN protocol; therefore, multiple slave nodes responding to a header might be detected as an error.

The LIN bus is a single channel wired-AND. The bus has a binary level: either dominant for a value of 0, or recessive for a value of 1.

30.3.1.2.1 Message Header

The header of a message is initiated by a master (see Figure 30-13) and consists of a three field-sequence:

- The sync break field signaling the beginning of a message
- The sync field conveying bit rate information of the LIN bus
- The ID field denoting the content of a message

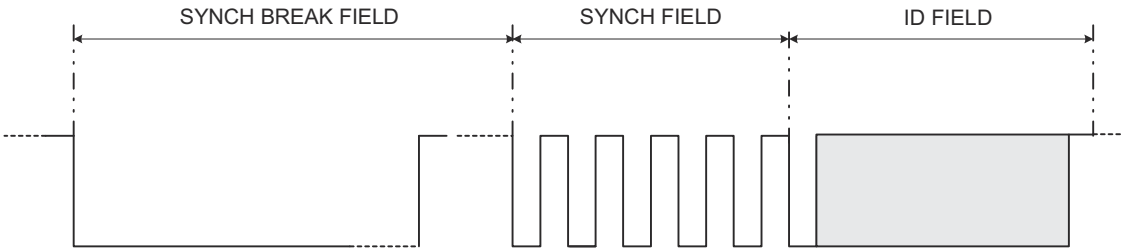


Figure 30-13. Header 3 Fields: Sync Break, Sync, and ID

30.3.1.2.2 Response

The format of the response is as illustrated in Figure 30-14. There are two types of fields in a response: data and checksum. The data field consists of exactly one data byte, one start bit, and one stop bit, for a total of 10 bits. The LSB is transmitted first. The checksum field consists of one checksum byte, one start bit and one stop bit. The checksum byte is the inverted modulo-256 sum over all data bytes in the data fields of the response.

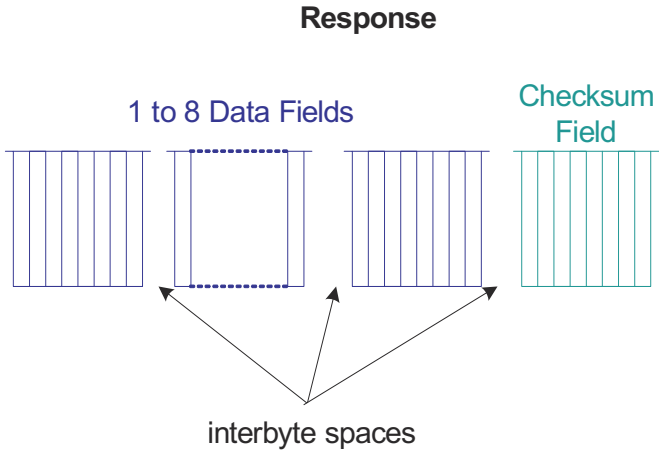


Figure 30-14. Response Format of LIN Message Frame

The format of the response is a stream of N data fields and one checksum field. Typically N is from 1 to 8, with the exception of the extended command frames (Section 30.3.1.6). The length N of the response is indicated either with the optional length control bits of the ID Field (this is used in standards earlier than LIN 1.x); see Table 30-9, or by LENGTH value in SCIFORMAT[18:16] register; see Table 30-10. The SCI/LIN module supports response lengths from 1 to 8 bytes in compliance with LIN 2.0.

Table 30-9. Response Length Info Using IDBYTE Field Bits [5:4] for LIN Standards Earlier than v1.3

ID5	ID4	Number of Data Bytes
0	0	2
0	1	2
1	0	4
1	1	8

Table 30-10. Response Length with SCIFORMAT[18:16] Programming

SCIFORMAT[18:16]	Number of Bytes
000	1
001	2
010	3
011	4
100	5
101	6
110	7
111	8

30.3.1.3 Synchronizer

The synchronizer has three major functions in the messaging between master and slave nodes. It generates the master header data stream, it synchronizes to the LIN bus for responding, and it locally detects timeouts. A bit rate is programmed using the prescalers in the BRSR register to match the indicated LIN_speed value in the LIN description file.

The LIN synchronizer will perform the following functions: master header signal generation, slave detection and synchronization to message header with optional baud rate adjustment, response transmission timing and timeout control.

The LIN synchronizer is capable of detecting an incoming break and initializing communication at all times.

30.3.1.4 Baud Rate

The transmission baud rate of any node is configured by the CPU at the beginning; this defines the bit time T_{bit}. The bit time is derived from the fields P and M in the baud rate selection register (BRSR).

The ranges for the prescaler values in the BRSR register are:

P = 0, 1, 2, 3, . . . , 2^{24} - 1

M = 0, 1, 2, . . . , 15

The P values in the BRSR register are user programmable. The P and M dividers could be used for both SCI mode and LIN mode to select a baud rate. If the ADAPT bit is set and the LIN slave is in adaptive baud rate mode, then all these divider values are automatically obtained during header reception when the synchronization field is measured.

The LIN protocol defines baud rate boundaries as follows:

1kHz ≤ F_{LINCLK} ≤ 20kHz

All transmitted bits are shifted in and out at T_{bit} periods.

30.3.1.4.1 Fractional Divider

The M field of the BRSR register modifies the integer prescaler P for fine tuning of the baud rate. The M value adds in increments of 1/16 of the P value.

The bit time, T_{bit} is expressed in terms of the VCLK period T_{VCLK} as follows:

For all P other than 0, and all M,

$$T_{bit} = 16\left(P + 1 + \frac{M}{16}\right)T_{VCLK}$$

For P= 0 : $T_{bit} = 32T_{VCLK}$

Therefore, the LINCLK frequency is given by:

$$F_{LINCLK} = \frac{F_{VCLK}}{16(P + 1 + \frac{M}{16})} \quad \text{For all P other than zero}$$

$$F_{LINCLK} = \frac{F_{VCLK}}{32} \quad \text{For } P = 0$$

30.3.1.4.2 Superfractional Divider

The superfractional divider scheme applies to the following modes:

- LIN master mode (sync field + identifier field + response field + checksum field)
- LIN slave mode (response field + checksum field)

30.3.1.4.2.1 Superfractional Divider In LIN Mode

Building on the 4-bit fractional divider M (BRSR[27:24], the superfractional divider uses an additional 3-bit modulating value, illustrated in Table 30-11. The sync field (0x55), the identifier field, and the response field can all be seen as 8-bit data bytes flanked by a start bit and a stop bit. The bits with a 1 in the table will have an additional VCLK period added to their T_{bit}. In LIN master mode, bit modulation applies to sync field + identifier field + response field. In LIN slave mode, bit modulation applies to identifier field + response field.

Table 30-11. Superfractional Bit Modulation for LIN Master Mode and Slave Mode

BRSR[30:28]	Start Bit	D[0]	D[1]	D[2]	D[3]	D[4]	D[5]	D[6]	D[7]	Stop Bit
0h	0	0	0	0	0	0	0	0	0	0
1h	1	0	0	0	0	0	0	0	1	0
2h	1	0	0	0	1	0	0	0	1	0
3h	1	0	1	0	1	0	0	0	1	0
4h	1	0	1	0	1	0	1	0	1	0
5h	1	1	1	0	1	0	1	0	1	1
6h	1	1	1	0	1	1	1	0	1	1
7h	1	1	1	1	1	1	1	0	1	1

The baud rate will vary over a LIN data field to average according to the BRSR[30:28] value by a *d* fraction of the peripheral internal clock: 0<d<1.

The instantaneous bit time is expressed in terms of T_{VCLK} as follows:

For all P other than 0, and all M and d (0 or 1),

$$T^{i}bit = \left[16\left(P + 1 + \frac{M}{16}\right) + d\right]T_{VCLK}$$

For P = 0, $T_{bit} = 32T_{VCLK}$

The averaged bit time is expressed in terms of T_{VCLK} as follows:

For all P other than 0, and all M and d (0<d<1),

$$T^{a}bit = \left[16\left(P + 1 + \frac{M}{16}\right) + d\right]T_{VCLK}$$

For P = 0, $T_{bit} = 32T_{VCLK}$

30.3.1.5 Header Generation

Automatic generation of the LIN protocol header data stream is supported without CPU interaction. The CPU or the DMA will trigger a message header generation and the LIN state machine will handle the generation itself. A master node initiates header generation on CPU or DMA writes to the IDBYTE in the LINID register. The header is always sent by the master to initiate a LIN communication and consists of three fields: break field, synchronization field, and identification field, as seen in Figure 30-15.

Note

The LIN protocol uses the parity bits in the identifier. The control length bits are optional to the LIN protocol.

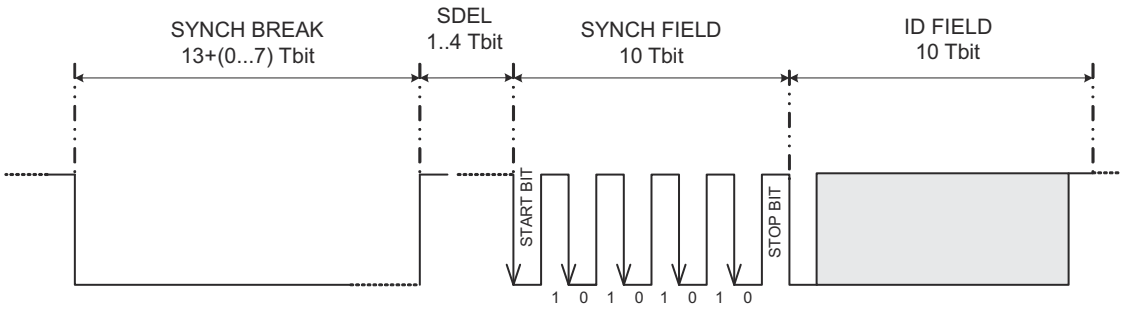


Figure 30-15. Message Header in Terms of T_{bit}

- The break field consists of two components:
  - The synchronization break (SYNC BREAK) consists of a minimum of 13 (dominant) low bits to a maximum of 20 dominant bits. The sync break length may be extended from the minimum with the 3-bit SBREAK value in the LINCOMP register.
  - The synchronization break delimiter (SDEL) consists of a minimum of 1 (recessive) high bit to a maximum of 4 recessive bits. The delimiter marks the end of the synchronization break field. The sync break delimiter length depends on the 2-bit SDEL value in the LINCOMP register.
- The synchronization field (SYNC FIELD) consists of one start bit, byte 0x55, and a stop bit. It is used to convey T_{bit} information and resynchronize LIN bus nodes.
- The identifier field's ID byte may use six bits as an identifier, with optional length control and two optional bits as parity of the identifier. The identifier parity is used and checked if the PARITY ENA bit is set. If length control bits are not used, then there can be a total of 64 identifiers plus parity. If neither length control or parity are used there can be up to 256 identifiers. See Figure 30-16 for an illustration of the ID field.

Note

Optional Control Length Bits

The control length bits only apply to LIN standards prior to LIN 1.3. IDBYTE field conveys response length information if compliant to standards earlier than LIN1.3. The SCIFORMAT register stores the length of the response for later versions of the LIN protocol.

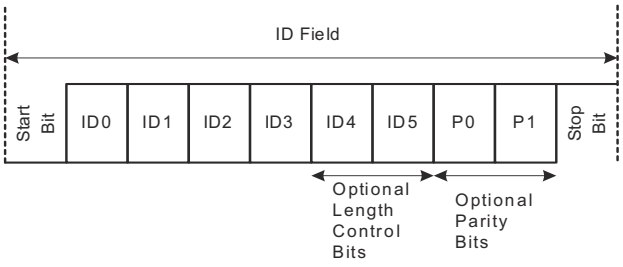


Figure 30-16. ID Field

Note

If the LIN module, configured as Slave in multi-buffer mode, is in the process of transmitting data while a new header comes in, the module might end up in responding with the data from the previous interrupted response (not the data corresponding to the new ID). To avoid this scenario the following procedure could be used:

1. Check for the Bit Error (BE) during the response transmission. If the BE flag is set, this indicates that a collision has happened on the LIN bus (here because of the new Sync Break).
2. In the Bit Error ISR, configure the TD0 and TD1 registers with the next set of data to be transmitted on a TX Match for the incoming ID. Before writing to TD0/TD1 make sure that there was not already an update because of a Bit Error; otherwise TD0/TD1 might be written twice for one ID.
3. Once the complete ID is received, based on the match, the newly configured data will be transmitted by the node.

30.3.1.5.1 Event Triggered Frame Handling

The LIN 2.0 protocol uses event-triggered frames that may occasionally cause collisions. Event-triggered frames have to be handled in software.

If no slave answers to an event triggered frame header, the master node will set the NRE flag, and a NRE interrupt will occur if enabled. If a collision occurs, a frame error and checksum error may arise before the NRE error. Those errors are flagged and the appropriate interrupts will occur, if enabled.

Frame errors and checksum errors depend on the behavior and synchronization of the responding slaves. If the slaves are totally synchronized and stop transmission once the collision occurred, it is possible that only the NRE error is flagged despite the occurrence of a collision. To detect if there has been a reception of one byte before the NRE error is flagged, the BUS BUSY flag can be used as an indicator.

The bus busy flag is set on the reception of the first bit of the header and remains set until the header reception is complete, and again is set on the reception of the first bit of the response. In the case of a collision the flag is cleared in the same cycle as the NRE flag is set.

Software could implement the following sequence:

- Once the reception of the header is done (poll for RXID flag), wait for the bus busy flag to get set or NRE flag to get set.
- If bus busy flag is not set before NRE flag, then it is a true no response case (no data has been transmitted onto the bus).
- If bus busy flag gets set, then wait for NRE flag to get set or for successful reception. If NRE flag is set, then in this case a collision has occurred on the bus.

Even in the case of a collision, the received (corrupted) data is accessible in the RX buffers; registers LINRD0 and LINRD1.

30.3.1.5.2 Header Reception and Adaptive Baud Rate

A slave node baud rate can optionally be adjusted to the detected bit rate as an option to the LIN module. The adaptive baud rate option is enabled by setting the ADAPT bit. During header reception, a slave measures the baud rate during detection of the sync field. If ADAPT bit is set, then the measured baud rate is compared to the slave node's programmed baud rate and adjusted to the LIN bus baud rate if necessary.

The LIN synchronizer determines two measurements: BRK_count and BAUD_count (Figure 30-17). These values are always calculated during the Header reception for sync field validation (Figure 30-18).

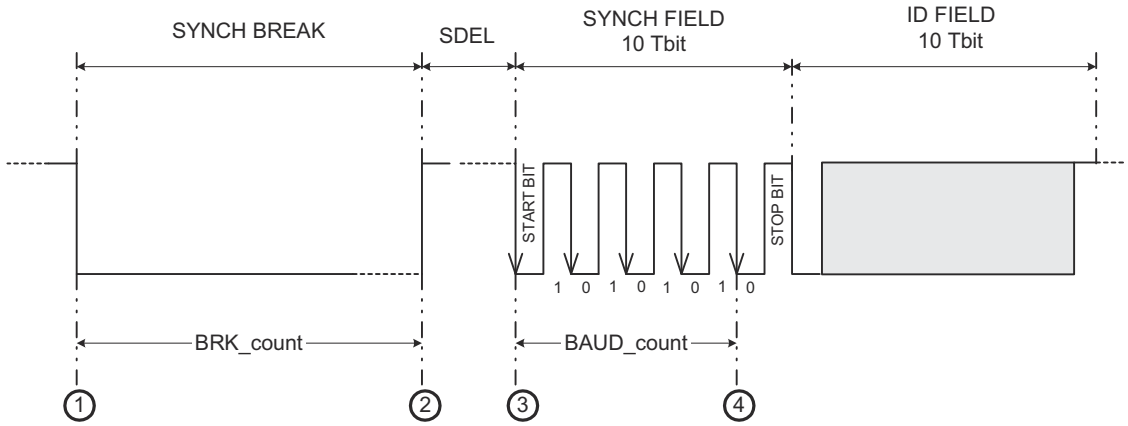


Figure 30-17. Measurements for Synchronization

By measuring the values BRK_count and BAUD_count, a valid sync break sequence can be detected as described in Figure 30-18. The four numbered events in Figure 30-17 signal the start/stop of the synchronizer counter. The synchronizer counter uses VCLK as the time base.

The synchronizer counter is used to measure the sync break relative to the detecting node T_{bit}. For a slave node receiving the sync break, a threshold of 11 T_{bit} is used as required by the LIN protocol. For detection of the dominant data stream of the sync break, the synchronizer counter is started on a falling edge and stopped on a rising edge of the LINRX. On detection of the sync break delimiter, the synchronizer counter value is saved and then reset.

On detection of five consecutive falling edges, the BAUD_count is measured. Bit timing calculation and consistency to required accuracy is implemented following the recommendations of LIN revision 2.0. A slave node can calculate a single T_{bit} time by division of BAUD_count by 8. In addition, for consistency between the detected edges the following is evaluated:

BAUD_count + BAUD_count » 2 + BAUD_count » 3 ≤ BRK_count

The BAUD_count value is shifted 3 times to the right and rounded using the first insignificant bit to obtain a T_{bit} unit. If the ADAPT bit is set, then the detected baud rate is compared to the programmed baud rate.

During the header reception processing as illustrated in Figure 30-18, if the measured BRK_count value is less than 11 T_{bit}, the sync break is not valid according to the protocol for a fixed rate. If the ADAPT bit is set, then the MBRS register is used for measuring BRK_count and BAUD_count values and automatically adjusts to any allowed LIN bus rate (refer to *LIN Specification Package 2.0*).

Note

In adaptive mode the MBRS divider should be set to allow a maximum baud rate that is not more than 10% above the expected operating baud rate in the LIN network. Otherwise, a 0x00 data byte could mistakenly be detected as a sync break.

The break-threshold relative to the slave node is 11 T_{bit}. The break is 13 T_{bit} as specified in LIN v1.3.

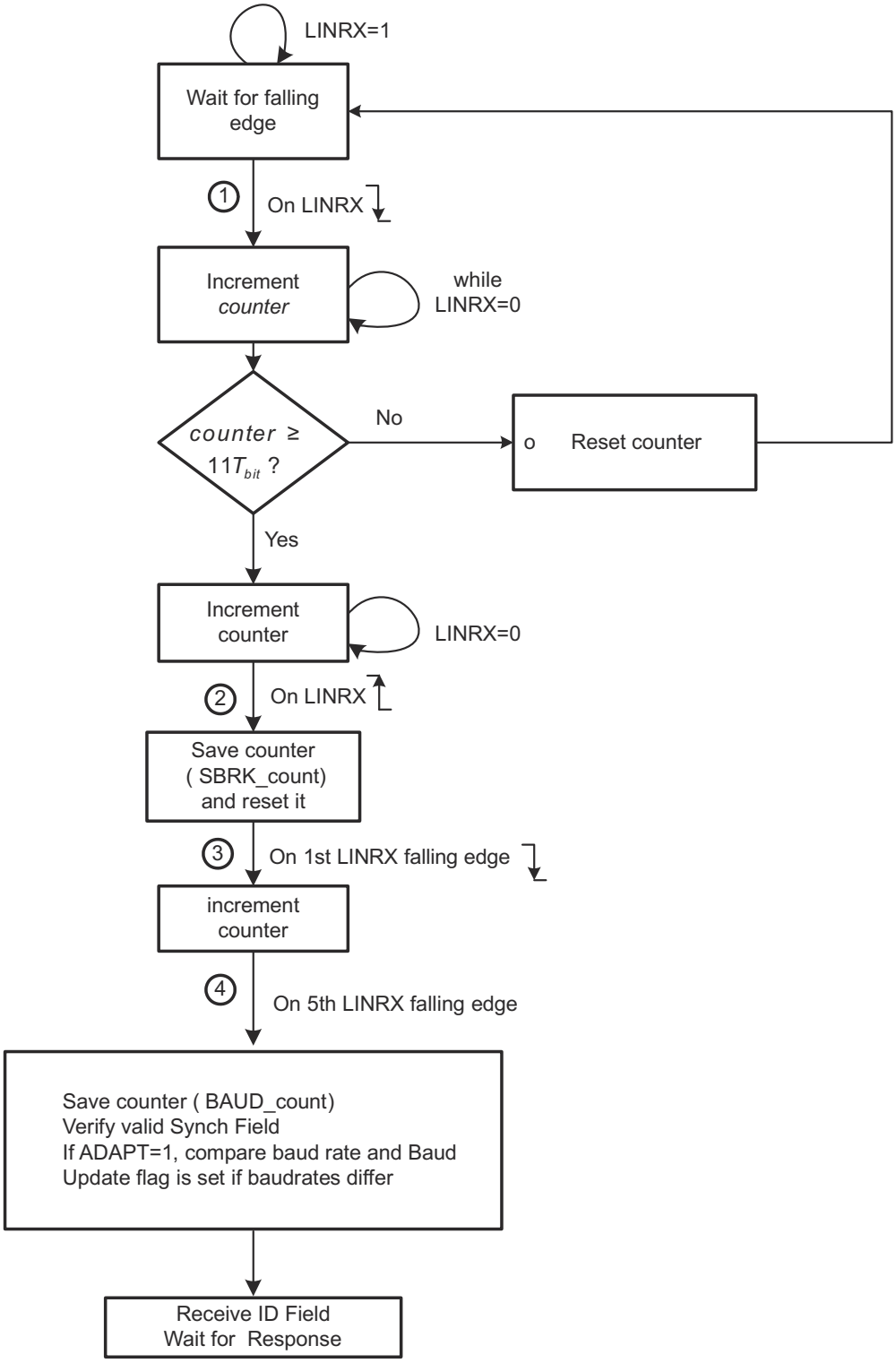


Figure 30-18. Synchronization Validation Process and Baud Rate Adjustment

If the sync field is not detected within the given tolerances, the inconsistent-sync-field-error (ISFE) flag will be set. An ISFE interrupt will be generated, if enabled by its respective bit in the SCISETINT register. The ID byte should be received after the sync field validation was successful. Any time a valid break (larger than 11 T_{bit}) is detected, the receiver's state machine should reset to reception of this new frame. This reset condition is only valid during response state, not if an additional sync break occurs during header reception.

Note

When an inconsistent sync field (ISFE) error occurs, suggested action for the application is to Reset the SWnRST bit and set the SWnRST bit to make sure that the internal state machines are back to their normal states.

30.3.1.6 Extended Frames Handling

The LIN protocol 2.0 and prior includes two extended frames with identifiers 62 (user-defined) and 63 (reserved extended). The response data length of the user-defined frame (ID 62, or 0x3E) is unlimited. The length for this identifier will be set at network configuration time to be shared with the LIN bus nodes.

Extended frame communication is triggered on reception of a header with identifier 0x3E; see Figure 30-19. Once the extended frame communication is triggered, unlike normal frames, this communication needs to be stopped before issuing another header. To stop the extended frame communication the STOP EXT FRAME bit must be set.

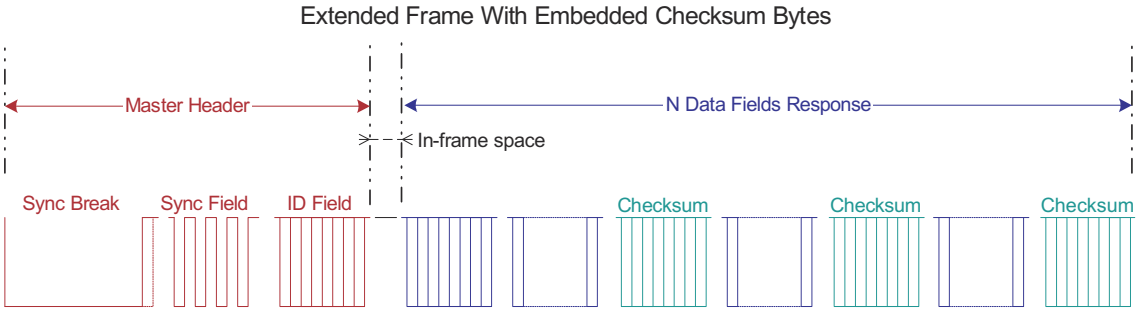


Figure 30-19. Optional Embedded Checksum in Response for Extended Frames

An ID interrupt will be generated (if enabled and there is a match) on reception of ID 62 (0x3E). This interrupt allows the CPU using a software counter to keep track of the bytes that are being sent out and decides when to calculate and insert a checksum byte (recommended at periodic rates). To handle this procedure, SC bit is used. A write to the send checksum bit SC will initiate an automatic send of the checksum byte. The last data field should always be a checksum in compliance with the LIN protocol.

The periodicity of the checksum insertion, defined at network configuration time, is used by the receiving node to evaluate the checksum of the ongoing message, and has the benefit of enhanced reliability.

For the sending node, the checksum is automatically embedded each time the send checksum bit SC is set. For the receiving node, the checksum is compared each time the compare checksum bit CC is set; see Figure 30-20.

Note

The LIN 2.0 enhanced checksum does not apply to the reserved identifiers. The reserved identifiers always use the classic checksum.

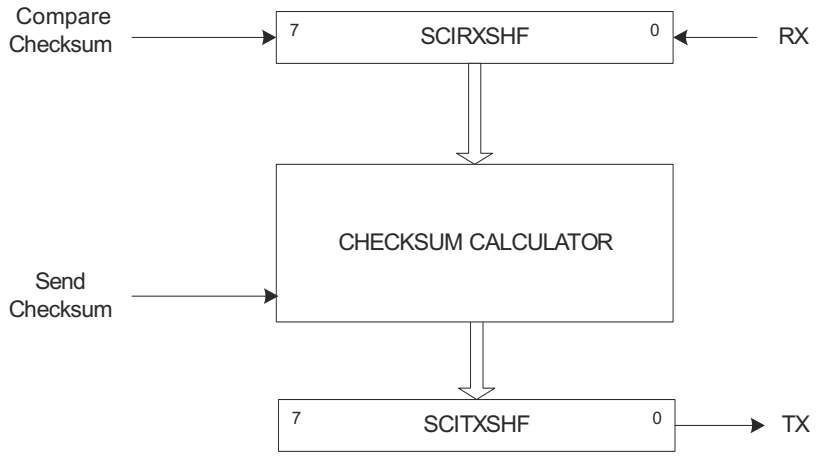


Figure 30-20. Checksum Compare and Send for Extended Frames

30.3.1.7 Timeout Control

Any LIN node listening to the bus and expecting a response initiated from a master node could flag a no-response error timeout event. The LIN protocol defines four types of timeout events, which are all handled by the hardware of the LIN module. The four LIN protocol events are:

- No-response timeout error
- Bus idle detection
- Timeout after wakeup signal
- Timeout after three wakeup signals

30.3.1.7.1 No-Response Error (NRE)

The no-response error will occur when any node expecting a response waits for T_{FRAME_MAX} time and the message frame is not fully completed within the maximum length allowed, T_{FRAME_MAX}. After this time a no-response error (NRE) is flagged in the NRE bit of the SCIFLR register. An interrupt is triggered if enabled.

As specified in the LIN 1.3 standard, the minimum time to transmit a frame is:

$T_{FRAME\_MIN} = T_{HEADER\_MIN} + T_{DATA\_FIELD} + T_{CHECKSUM\_FIELD} = 44 + 10N$

where N = number of data fields.

And the maximum time frame is given by:

$T_{FRAME\_MAX} = T_{FRAME\_MIN} * 1.4 = (44 + 10N) * 1.4$

The timeout value T_{FRAME_MAX} is derived from the *N* number of data fields value see Table 30-12. The *N* value is either embedded in the header's ID field for messages or is part of the description file. In the latter case, the 3-bit CHAR value in SCIFORMAT register, will indicate the value for *N*.

Note

The length coding of the ID field does not apply to two extended frame identifiers, ID fields of 0x3E (62) and 0x3F (63). In these cases, the ID field can be followed by an arbitrary number of data byte fields. Also, the LIN 2.0 protocol specification mentions that ID field 0x3F (63) cannot be used. For these two cases, the NRE will not be handled by the LIN controller hardware.

Table 30-12. Timeout Values in T_{bit} Units

N	T_{DATA_FIELD}	T_{FRAME_MIN}	T_{FRAME_MAX}
1	10	54	76
2	20	64	90
3	30	74	104
4	40	84	118
5	50	94	132
6	60	104	146
7	70	114	160
8	80	124	174

30.3.1.7.2 Bus Idle Detection

The second type of timeout can occur when a node detects an inactive LIN bus: no transitions between recessive and dominant values are detected on the bus. This happens after a minimum of 4 s (this is 80,000 F_{LINCLK} cycles with the fastest bus rate of 20 kbps). If a node detects no activity in the bus as the TIMEOUT bit is set, then it can be assumed that the LIN bus is in sleep mode. Application software can use the Timeout flag to determine when the LIN bus is inactive and put the LIN into sleep mode by writing the POWERDOWN bit.

Note

After the timeout was flagged, a SWnRESET should be asserted before entering Low-Power Mode. This is required to reset the receiver in case that an incomplete frame was on the bus before the idle period.

30.3.1.7.3 Timeout After Wakeup Signal and Timeout After Three Wakeup Signals

The third and fourth types of timeout are related to the wakeup signal. A node initiating a wakeup should expect a header from the master within a defined amount of time: timeout after wakeup signal. See Section 30.4.3 for more details.

30.3.1.8 TXRX Error Detector (TED)

The following sources of error are detected by the TXRX error detector logic (TED). The TED logic consists of a bit monitor, an ID parity checker, and a checksum error. The following errors are detected:

- Bit errors (BE)
- Physical bus errors (PBE)
- Identifier parity errors (PE)
- Checksum errors (CE)

All of these errors (BE, PBE, PE, CE) are flagged. An interrupt for the flagged errors will be generated if enabled. A message is valid for both the transmitter and the receiver if there is no error detected until the end of the frame.

30.3.1.8.1 Bit Errors

A bit error (BE) is detected at the bit time when the bit value that is monitored is different from the bit value that is sent. A bit error is indicated by the BE flag in SCIFLR. After signaling a BE, the transmission is aborted no later than the next byte. The bit monitor ensures that the transmitted bit in LINTX is the correct value on the LIN bus by reading back on the LINRX pin as shown in Figure 30-21.

Note

If a bit occurs due to receiving a header during a slave response, NRE/TIMEOUT flag will not be set for the new frame.

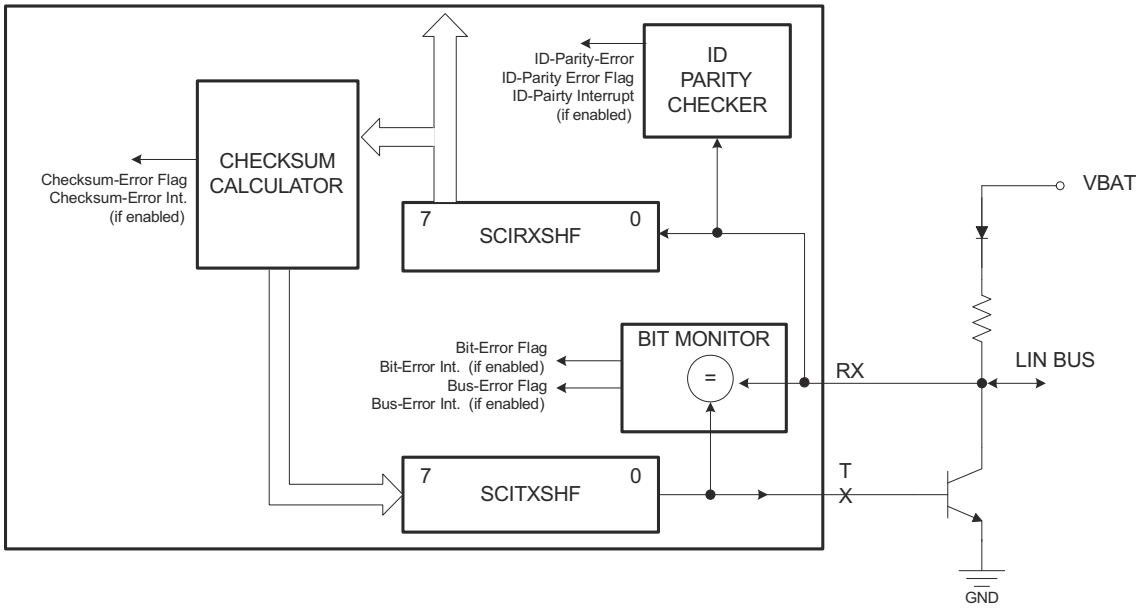


Figure 30-21. TXRX Error Detector

30.3.1.8.2 Physical Bus Errors

A Physical Bus Error (PBE) has to be detected by a master if no valid message can be generated on the bus (Bus shorted to GND or VBAT). The bit monitor detects a PBE during the header transmission, if no Sync Break can be generated (for example, because of a bus shortage to VBAT) or if no sync break delimiter can be generated (for example, because of a bus shortage to GND). Once the Sync Break Delimiter was validated, all other deviations between the monitored and the sent bit value are flagged as Bit Errors (BE) for this frame.

30.3.1.8.3 ID Parity Errors

If parity is enabled, an ID parity error (PE) is detected if any of the two parity bits of the sent ID byte are not equal to the calculated parity on the receiver node. The two parity bits are generated using the following mixed parity algorithm:

$$P0 = ID0 \oplus ID1 \oplus ID2 \oplus ID4 \ (even\ Parity)$$
$$P1 = ID1 \oplus ID3 \oplus ID4 \oplus ID5 \ (odd\ Parity)$$

If an ID-parity error is detected, the ID-parity error is flagged, and the received ID is not valid. See Section 30.3.1.9 for details.

30.3.1.8.4 Checksum Errors

A checksum error (CE) is detected and flagged at the receiving end if the calculated modulo-256 sum over all received data bytes (including the ID byte if it is the enhanced checksum type) plus the checksum byte does not result in 0xFF. The modulo-256 sum is calculated over each byte by adding with carry, where the carry bit of each addition is added to the LSB of its resulting sum.

For the transmitting node, the checksum byte sent at the end of a message is the inverted sum of all the data bytes (see Figure 30-22) for classic checksum implementation. The checksum byte is the inverted sum of the identifier byte and all the data bytes (see Figure 30-23) for the LIN 2.0 compliant enhanced checksum implementation. The classic checksum implementation should always be used for reserved identifiers 60 to 63; therefore, the CTYPE bit will be overridden in this case. For signal-carrying-frame identifiers (0 to 59) the type of checksum used depends on the CTYPE bit.

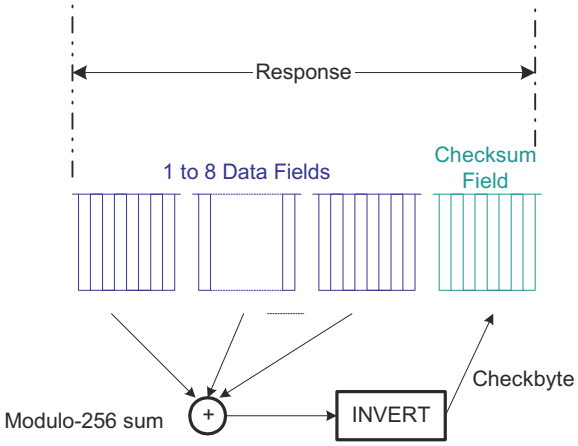


Figure 30-22. Classic Checksum Generation at Transmitting Node

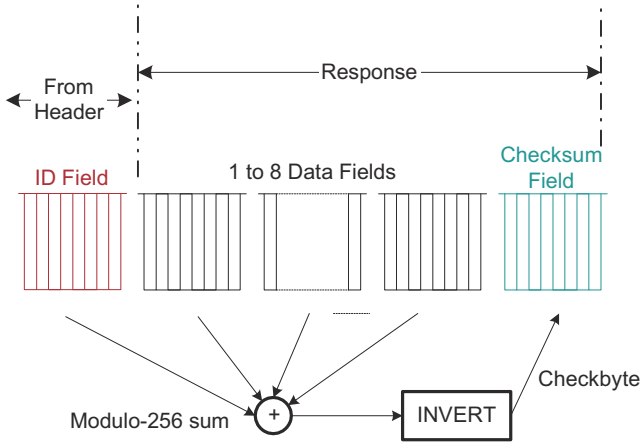


Figure 30-23. LIN 2.0-Compliant Checksum Generation at Transmitting Node

30.3.1.9 Message Filtering and Validation

Message filtering uses the entire identifier to determine which nodes will participate in a response, either receiving or transmitting a response. Therefore, two acceptance masks are used as shown in Figure 30-24. During header reception, all nodes filter the ID-Field (ID-Field is the part of the header explained in Figure 30-16) to determine whether they transmit a response or receive a response for the current message. There are two masks for message ID filtering: one to accept a response reception, the other to initiate a response transmission. See Figure 30-24. All nodes compare the received ID to the identifier stored in the ID-SlaveTask BYTE of the LINID register and use the RX ID MASK and the TX ID MASK fields in the LINMASK register to filter the bits of the identifier that should not be compared.

If there is an RX match with no parity error and the RXENA bit is set, there will be an ID RX flag and an interrupt will be triggered if enabled. If there is a TX match with no parity error and the TXENA bit is set, there will be an ID TX flag and an interrupt will be triggered if enabled in the SCISETINT register.

The masked bits become don't cares for the comparison. To build a mask for a set of identifiers, an XOR function could be used.

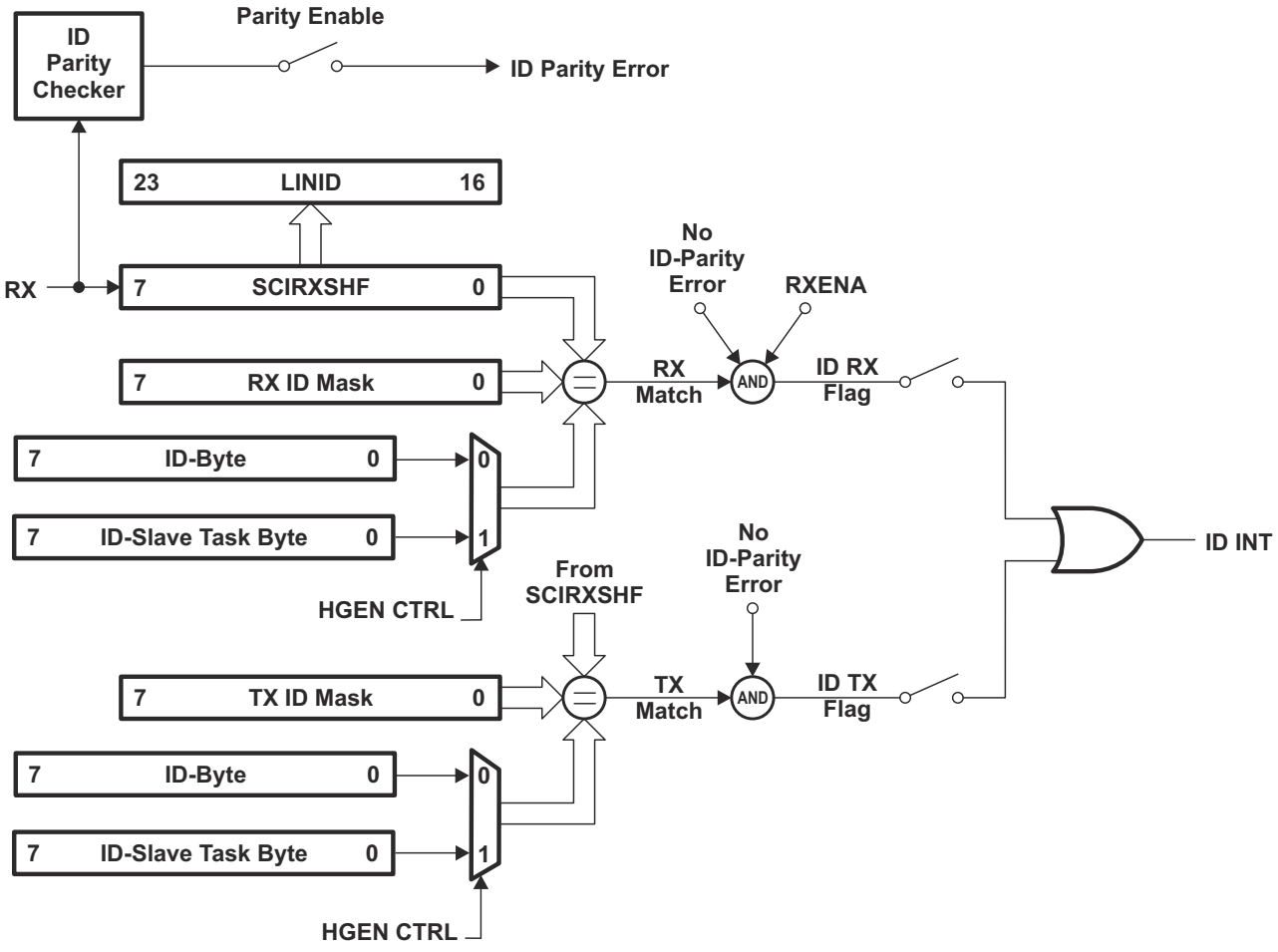


Figure 30-24. ID Reception, Filtering and Validation

For example, to build a mask to accept IDs 0x26 and 0x25 using LINID[7:0] = 0x20; that is, compare 5 most significant bits (MSBs) and filter 3 least significant bits (LSBs), the acceptance mask could be:

$$(0x26 + 0x25) \oplus 0x20 = 0x07$$

A mask of all zeros will compare all bits of the received identifier in the shift register with the ID-BYTE in LINID[7:0]. If HGEN CTRL is set to 1, a mask of 0xFF will always cause a match. A mask of all 1s will filter all bits of the received identifier, and thus there will be an ID match regardless of the content of the ID-SlaveTask BYTE field in the LINID register.

Note

When the HGEN CTRL bit = 0, the LIN nodes compare the received ID to the ID-BYTE field in the LINID register, and use the RX ID MASK and the TX ID MASK in the LINMASK register to filter the bits of the identifier that should not be compared.

If there is an RX match with no parity error and the RXENA bit is set, there will be an ID RX flag and an interrupt will be triggered if enabled. A mask of all 0s will compare all bits of the received identifier in the shift register with the ID-BYTE field in LINID[7:0]. A mask of all 1s will filter all bits of the received identifier and there will be no match.

If HGEN CTLR = 1:

- Received ID is compared with the ID-Slave-Task byte, using the RXID mask and the TXID mase.
- A mask of all ones will always result in a match.
- A mask of all zeroes means all the bits must be the same to result in a match.
- If a mask has some bits which are ones, then those bits will not be used for the filtering criterion.

If HGEN CTRL = 0:

- Received ID is compared with the ID byte, using the RXID mask and the TXID mask.
- A mask of all ones will result in no match.
- A mask of all zeroes means all the bits must be the same to result in a match.
- If a mask has some bits which are ones, then those bits will not be used for the filtering criterion.

During header reception, the received identifier is copied to the Received ID field LINID[23:16]. If there is no parity error and there is either a TX match or an RX match, then the corresponding TX or RX ID flag is set. If the ID interrupt is enabled, then an ID interrupt is generated.

After the ID interrupt is generated, the CPU may read the Received ID field LINID[23:16] and determine what response to load into the transmit buffers.

Note

When byte 0 is written to TD0 (LINTD0[31:24]), the response transmission is automatically generated.

In multi-buffer mode, the TXRDY flag will be set when all the response data bytes and checksum byte are copied to the shift register SCITXSHF. In non-multi-buffer mode, the TXRDY flag is set each time a byte is copied to the SCITXSHF register, and also for the last byte of the frame after the checksum byte is copied to the SCITXSHF register.

In multi-buffer mode, the TXEMPTY flag is set when both the transmit buffer(s) TDy and the SCITXSHF shift register are emptied and the checksum has been sent. In nonmulti-buffer mode, TXEMPTY is set each time TD0 and SCITXSHF are emptied, except for the last byte of the frame where the checksum byte must also be transmitted.

If parity is enabled, all slave receiving nodes will validate the identifier using all eight bits of the received ID byte. The SCI/LIN will flag a corrupted identifier if an ID-parity error is detected.

30.3.1.10 Receive Buffers

To reduce CPU load when receiving a LIN N-byte (with N = 1–8) response in interrupt mode or DMA mode, the SCI/LIN module has eight receive buffers. These buffers can store an entire LIN response in the RDy receive buffers. Figure 30-8 illustrates the receive buffers.

The checksum byte following the data bytes is validated by the internal checksum calculator. The checksum error (CE) flag indicates a checksum error and a CE interrupt will be generated if enabled in the SCISETINT register.

The multi-buffer 3-bit counter counts the data bytes transferred from the SCIRXSHF register to the RDy receive buffers if multi-buffer mode is enabled, or to RD0 if multi-buffer mode is disabled. The 3-bit compare register contains the number of data bytes expected to be received. In cases where the ID BYTE field does not convey message length (see *Note: Optional Control Length Bits* in Section 30.3.1.5), the LENGTH value, indicates the expected length and is used to load the 3-bit compare register. Whether the length control field or the LENGTH value is used is selectable with the COMM MODE bit.

A receive interrupt, and a receive ready RXRDY flag as well as a DMA request (RXDMA) could occur after receiving a response if there are no response receive errors for the frame (such as, there is no checksum error, frame error, and overrun error). The checksum byte will be compared before acknowledging a reception. A DMA request can be generated for each received byte or for the entire response depending on whether the multi-buffer mode is enabled or not (MBUF MODE bit).

Note

In multi-buffer mode following are the scenarios associated with clearing the "RXRDY" flag bit:

1. The RXRDY flag cannot be cleared by reading the corresponding interrupt offset in the SCIINTVECT0/1 register.
2. For LENGTH less than or equal to 4, Read to RD0 register will clear the “RXRDY” flag.
3. For LENGTH greater than 4, Read to RD1 register will clear the “RXRDY” flag.

30.3.1.11 Transmit Buffers

To reduce the CPU load when transmitting a LIN N-byte (with N = 1–8) response in interrupt mode or DMA mode, the SCI/LIN module has eight transmit buffers, TD0–TD7 in LINTD0 and LINTD1. With these transmit buffers, an entire LIN response field can be preloaded in the TXy transmit buffers. Optionally, a DMA transfer could be done on a byte-per-byte basis when multi-buffer mode is not enabled (MBUF MODE bit). Figure 30-9 illustrates the transmit buffers.

The multi-buffer 3-bit counter counts the data bytes transferred from the TDy transmit buffers register if multi-buffer mode is enabled, or from TD0 to SCITXSHF if multi-buffer mode is disabled. The 3-bit compare register contains the number of data bytes expected to be transmitted. If the ID field is not used to convey message length (see *Note: Optional Control Length Bits* in Section 30.3.1.5), the LENGTH value indicates the expected length and is used instead to load the 3-bit compare register. Whether the length control field or the LENGTH value is used is selectable with the COMM MODE bit.

A transmit interrupt (TX interrupt) and a transmit ready flag (TXRDY flag), as well as a DMA request (TXDMA) could occur after transmitting a response. A DMA request can be generated for each transmitted byte or for the entire response depending on whether multi-buffer mode is enabled or not (MBUF MODE bit).

The checksum byte will be automatically generated by the checksum calculator and sent after the data-fields transmission is finished. The multi-buffer 3-bit counter counts the data bytes transferred from the TDy buffers into the SCITXSHF register.

Note

The transmit interrupt request can be eliminated until the next series of data is written into the transmit buffers LINTD0 and LINTD1, by disabling the corresponding interrupt via the SCICLRINT register or by disabling the transmitter via the TXENA bit.

30.3.2 LIN Interrupts

LIN and SCI mode have a common Interrupt block as explained in Section 30.2.2. There are 16 interrupt sources in the SCI/LIN module, with 8 of them being LIN mode only, as seen in Table 30-6.

A LIN message frame indicating the timing and sequence of the LIN interrupts that could occur is shown in Figure 30-25.

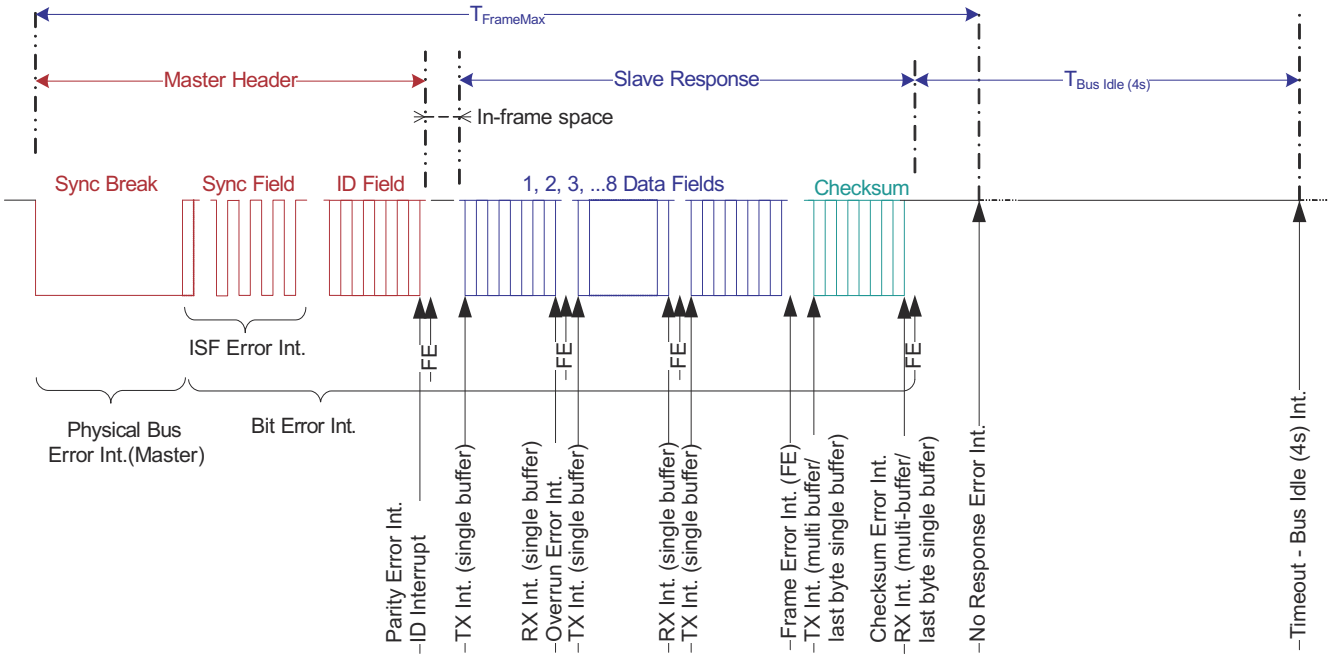


Figure 30-25. LIN Message Frame Showing LIN Interrupt Timing and Sequence

30.3.3 Servicing LIN Interrupts

When servicing an interrupt, clear the corresponding flag in the flag register (SCIFLR) before clearing the global interrupt flag (LIN_GLB_INT_CLR). The ISR should follow the guidelines below. This will prevent any spurious or duplicate interrupt from occurring.

- Clear the LIN interrupt flag in the SCIFLR register.
- Read the LIN interrupt status register to make sure the flag is cleared.
- Clear the global interrupt flag bit in LIN_GLB_INT_CLR.

Note

The transmit interrupt is generated before the LIN transmitter is ready to accept new data. Inside of the LIN transmit ISR, the software should wait until the buffer is completely empty before loading the next data. This can be done by polling for the Bus Busy Flag (SCIFLR.BUSY) to be 0.

30.3.4 LIN DMA Interface

LIN DMA Interface uses the SCI DMA interface logic. DMA requests for receive (RXDMA request) and transmit (TXDMA request) are available for the SCI/LIN module. There are two modes for DMA transfers depending on whether multi-buffer mode is enabled or not via the multi-buffer enable control bit (MBUF MODE).

Note

Do not use the DMA to transmit data to multiple slave IDs. Writing to the LINID register will initiate a new transmission. The DMA will write to the LINID register before the LIN state machine is ready to accept the new ID. Doing so will cause the LIN to miss this transmission.

30.3.4.1 LIN Receive DMA Requests

In LIN mode, when the multi-buffer option is enabled, if a received response (up to eight data bytes) is transferred to the receive buffers (RDy), then a DMA request is generated. If the multi-buffer option is disabled, then DMA requests will be generated on a byte-per-byte basis until all the expected response data fields are received. This DMA functionality is enabled and disabled using the SET RX DMA and CLR RX DMA bits, respectively.

30.3.4.2 LIN Transmit DMA Requests

In LIN mode with the multi-buffer option enabled, after a transmission (up to eight data bytes stored in the transmit buffer(s) TDy in the LINTD0 and LINTD1 registers), a DMA request is generated in order to reload the transmit buffer for the next transmission. If the multi-buffer option is disabled, then DMA requests will be generated on a byte-per-byte basis until all bytes are transferred. This DMA functionality is enabled and disabled using the SET TX DMA and CLR TX DMA bits, respectively.

30.3.5 LIN Configurations

The following list details the configuration steps that software should perform prior to the transmission or reception of data in LIN mode. As long as the SWnRST bit in the SCIGCR1 register is cleared to 0 the entire time that the LIN is being configured, the order in which the registers are programmed is not important.

- Enable LIN by setting RESET bit.
- Clear SWnRST to 0 before configuring the LIN.
- Enable the LINRX and LINTX pins by setting the RX FUNC and TX FUNC bits.
- Select LIN mode by programming LIN MODE bit.
- Select Master or Slave mode by programming the CLOCK bit.
- Select the desired frame format (checksum, parity, length control) by programming SCIGCR1.
- Select multi-buffer mode by programming MBUF MODE bit.
- Select the baud rate to be used for communication by programming BRSR.
- Set the maximum baud rate to be used for communication by programming MBRSR.
- Set the CONT bit to make LIN not halt for an emulation breakpoint until its current reception or transmission is complete (this bit is used only in an emulation environment).
- Set LOOP BACK bit to connect the transmitter to the receiver internally if needed (this feature is used to perform a self-test).
- Select the receiver enable RXENA bit if data is to be received.
- Select the transmit enable TXENA bit if data is to be transmitted.
- Select the RX ID MASK and the TX ID MASK fields in the LINMASK register.
- Set SWnRST to 1 after the LIN is configured.
- Perform Receive or Transmit data (see Section 30.3.1.9, Section 30.3.5.1, and Section 30.3.5.2).

Note

If TXENA is set and the SWnRST is released, the LIN will only generate a new DMA request. The LIN hardware will not generate a new Transmit interrupt request. If using interrupts, the first transmission must be started by software by writing the data to transmit followed by writing to LIN TX to initiate the transmission.

30.3.5.1 Receiving Data

The LIN receiver is enabled to receive messages if both the RX FUNC bit and the RXENA bit are set to 1. If the RX FUNC bit is not set, the LINRX pin functions as a general-purpose I/O pin rather than as a LIN function pin.

The ID RX FLAG is set after a valid LIN ID is received with RX Match. An ID interrupt is generated, if enabled.

30.3.5.1.1 Receiving Data in Single-Buffer Mode

Single-buffer mode is selected when the MBUF MODE bit is cleared to 0. In this mode, LIN sets the RXRDY bit when it transfers newly received data from SCIRXSHF to RD0. The SCI clears the RXRDY bit after the new data in RD0 has been read. Also, as data is transferred from SCIRXSHF to RD0, the LIN sets the FE, OE, or PE flags if any of these error conditions were detected in the received data. These error conditions are supported with configurable interrupt capability.

You can receive data by:

1. Polling Receive Ready Flag
2. Receive Interrupt

In polling method, software can poll for the RXRDY bit and read the data from RD0 byte of the LINRD0 register once the RXRDY bit is set high. The CPU is unnecessarily overloaded by selecting the polling method. To avoid this, you can use the interrupt method. To use the interrupt method, the SET RX INT bit is set. If the checksum scheme is enabled by setting the Compare Checksum (CC) bit to 1, the checksum will be compared on the byte that is currently being received, which is expected to be the checksum byte. The CC bit will be cleared once the checksum is received. A CE will immediately be flagged if there is a checksum error.

30.3.5.1.2 Receiving Data in Multi-Buffer Mode

Multi-buffer mode is selected when the MBUF MODE bit is set to 1. In this mode, LIN sets the RXRDY bit after receiving the programmed number of data in the receive buffer and the checksum field, the complete frame. The error condition detection logic is similar to the single-buffer mode, except that it monitors for the complete frame. Like single-buffer mode, you can use the polling, DMA, or interrupt method to read the data. The received data has to be read from the LINRD0 and LINRD1 registers, based on the number of bytes. For a LENGTH less than or equal to 4, a read from the LINRD0 register clears the RXRDY flag. For a LENGTH greater than 4, a read from the LINRD1 register clears the RXRDY flag. If the checksum scheme is enabled by setting the Compare Checksum (CC) bit to 1 during the reception of the data, then the byte that is received after the reception of the programmed number of data bytes indicated by the LENGTH field is treated as a checksum byte. The CC bit will be cleared once the checksum is received and compared.

30.3.5.2 Transmitting Data

The LIN transmitter is enabled if both the TX FUNC bit and the TXENA bit are set to 1. If the TX FUNC bit is not set, the LINTX pin functions as a general-purpose I/O pin rather than as a LIN function pin. Any value written to the TD0 before the TXENA bit is set to 1 is not transmitted. Both of these control bits allow for the LIN transmitter to be held inactive independently of the receiver.

The ID TX flag is set after a valid LIN ID is received with TX Match. An ID interrupt is generated, if enabled.

30.3.5.2.1 Transmitting Data in Single-Buffer Mode

Single-buffer mode is selected when the MBUF MODE bit is cleared to 0. In this mode, LIN waits for data to be written to TD0, transfers it to SCITXSHF, and transmits the data. The TXRDY and TX EMPTY bits indicate the status of the transmit buffers. That is, when the transmitter is ready for data to be written to TD0, the TXRDY bit is set. Additionally, if both TD0 and SCITXSHF are empty, then the TX EMPTY bit is also set.

You can transmit data by:

1. Polling Transmit Ready Flag
2. Transmit Interrupt

In polling method, software can poll for the TXRDY bit to go high before writing the data to the TD0. The CPU is unnecessarily overloaded by selecting the polling method. To avoid this, you can use the interrupt or DMA method. To use the interrupt method, the SET TX INT bit is set. To use the DMA method, the SET TX DMA bit is set. Either an interrupt or a DMA request is generated the moment the TXRDY bit is set. When the LIN has completed transmission of all pending frames, the SCITXSHF register and the TD0 are empty, the TXRDY bit is set, and an interrupt/DMA request is generated, if enabled. Because all data has been transmitted, the interrupt/DMA request should be halted. This can either be done by disabling the transmit interrupt (CLR TX

INT) / DMA request (CLR TX DMA bit) or by disabling the transmitter (clear TXENA bit). If the checksum scheme is enabled by setting the Send Checksum (SC) bit to 1, the checksum byte will be sent after the current byte transmission. The SC bit will be cleared after the checksum byte has been transmitted.

Note

The TXRDY flag cannot be cleared by reading the corresponding interrupt offset in the SCIINTVECT0 or SCIINTVECT1 register.

30.3.5.2.2 Transmitting Data in Multi-Buffer Mode

Multi-buffer mode is selected when the MBUF MODE bit is set to 1. Like single-buffer mode, you can use the polling or interrupt method to write the data to be transmitted. The transmitted data has to be written to the LINTD0 and LINTD1 registers, based on the number of bytes. LIN waits for data to be written to Byte 0 (TD0) of the LINTD0 register and transfers the programmed number of bytes to SCITXSHF to transmit one by one automatically. If the checksum scheme is enabled by setting the Send Checksum (SC) bit to 1, the checksum will be sent after transmission of the last byte of the programmed number of data bytes, indicated by the LENGTH field. The SC bit will be cleared after the checksum byte has been transmitted.

30.4 Low-Power Mode

The SCI/LIN module can be put in either local or global low-power mode. Global low-power mode is asserted by the system and is not controlled by the SCI/LIN module. During global low-power mode, all clocks to the SCI/LIN are turned off so the module is completely inactive. If global low-power mode is requested while the receiver is receiving data, then the SCI/LIN completes the current reception and then enters the low-power mode, that is, module enters low-power mode only when Busy bit (SCIFLR.3) is cleared.

The LIN module may enter low-power mode either when there was no activity on the LINRX pin for more than 4s (this can be either a constant recessive or dominant level) or when a Sleep Command frame was received. Once the Timeout flag (SCIFLR.4) was set or once a Sleep Command was received, the POWERDOWN bit (SCIGCR2.0) must be set by the application software to make the module enter local low-power mode. A wakeup signal will terminate the sleep mode of the LIN bus.

Note

Enabling Local Low-Power Mode During Receive and Transmit

If the wakeup interrupt is enabled and low-power mode is requested while the receiver is receiving data, then the SCI/LIN immediately generates a wake-up interrupt to clear the powerdown bit. Thus, the SCI/LIN is prevented from entering low-power mode and completes the current reception. Otherwise, if the wakeup interrupt is disabled, the SCI/LIN completes the current reception and then enters the low-power mode.

30.4.1 Entering Sleep Mode

In LIN protocol, a sleep command is used to broadcast the sleep mode to all nodes. The sleep command consists of a diagnostic master request frame with identifier 0x3C (60), with the first data field as 0x00. There should be no activity in the bus once all nodes receive the sleep command: the bus is in sleep mode.

Local low-power mode is asserted by setting the POWERDOWN bit; setting this bit stops the clocks to the SCI/LIN internal logic and registers. Clearing the POWERDOWN bit causes SCI/LIN to exit from local low-power mode. All the registers are accessible during local power-down mode. If a register is accessed in low-power mode, this access results in enabling the clock to the module for that particular access alone.

30.4.2 Wakeup

The wakeup interrupt is used to allow the SCI/LIN module to automatically exit a low-power mode. A SCI/LIN wakeup is triggered when a low level is detected on the receive RX pin, and this clears the POWERDOWN bit.

Note

If the wakeup interrupt is disabled, then the SCI/LIN enters low-power mode whenever it is requested to do so, but a low level on the receive RX pin does NOT cause the SCI/LIN to exit low-power mode.

In LIN mode, any node can terminate sleep mode by sending a wakeup signal; see Figure 30-26. A slave node that detects the bus in sleep mode, and with a wakeup request pending, will send a wakeup signal. The wakeup signal is a dominant value on the LIN bus for T_{WUSIG}; this is at least 5 T_{bits} for the LIN bus baud rates. The wakeup signal is generated by sending a 0xF0 byte containing 5 dominant T_{bits} and 5 recessive T_{bits}.

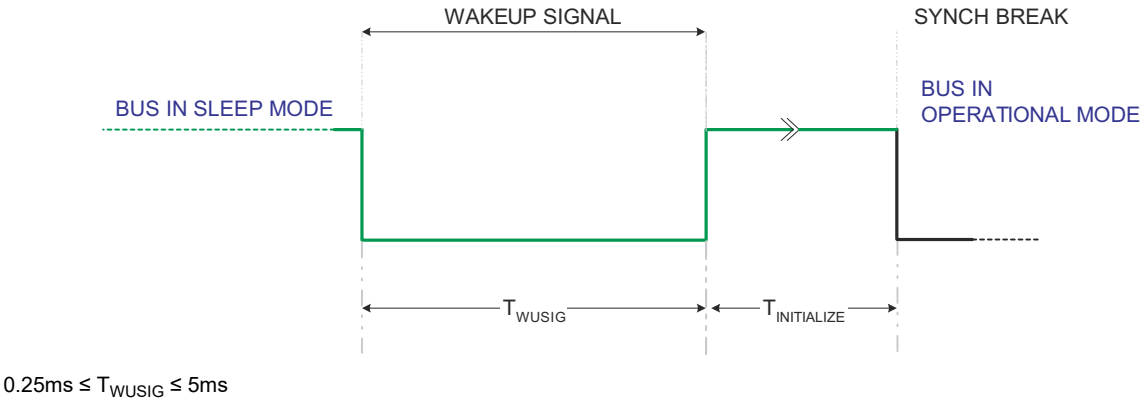


Figure 30-26. Wakeup Signal Generation

Assuming a perfect bus with no noise or loading effects, a write of 0xF0 to TD0 will load the transmitter to meet the wakeup signal timing requirement for T_{WUSIG}. Then, setting the GENWU bit will transmit the preloaded value in TD0 for a wakeup signal transmission.

Note

The GENWU bit can be set/reset only when SWnRST is set to'1' and the node is in power down mode. The bit will be cleared on a valid sync break detection. A master sending a wakeup request, will exit power down mode upon reception of the wakeup pulse. The bit will be cleared on a SWnRST. This can be used to stop a master from sending further wakeup requests.

The TI TPIC1021 LIN transceiver, upon receiving a wakeup signal, will translate it to the microcontroller for wakeup with a dominant level on the RX pin, or a signal to the voltage regulator. While the POWERDOWN bit is set, if the LIN module detects a recessive-to-dominant edge (falling edge) on the RX pin, it will generate a wakeup interrupt if enabled in the SCISETINT register.

According to LIN protocol 2.0, the TI TPIC1021 LIN transceiver detecting a dominant level on the bus longer than 150 ms will detect it as a wakeup request. The LIN controller's slave is ready to listen to the bus in less than 100 ms ($T_{INITIALIZE}$<100ms) after a dominant-to-recessive edge (end-of-wakeup signal).

30.4.3 Wakeup Timeouts

The LIN protocol defines the following timeouts for a wakeup sequence. After a wakeup signal has been sent to the bus, all nodes wait for the master to send a header. If no sync field is detected before 150 ms (3,000 cycles at 20 kHz) after a wakeup signal is transmitted, a new wakeup is sent by the same node that requested the first wakeup. This sequence is not repeated more than two times. After three attempts to wake up the LIN bus, wakeup signal generation is suspended for a 1.5 s (30,000 cycles at 20 kHz) period after three breaks.

Note

To achieve compatibility to LIN1.3 timeout conditions, the MBRS register must be set to assure that the LIN 2.0 (real-time-based) timings meet the LIN 1.3 bit time base. A node triggering the wakeup should set the MBRS register accordingly to meet the targeted time as 128 Tbits × programmed prescaler.

The LIN controller handles the wakeup expiration times defined by the LIN protocol with a hardware implementation.

30.5 Emulation Mode

In emulation mode, the CONT bit determines how the SCI/LIN operates when the program is suspended. The SCI/LIN counters are affected by this bit during debug mode. when set, the counters are not stopped and when cleared, the counters are stopped debug mode.

Any reads in emulation mode to a SCI/LIN register will not have any effect on the flags in the SCIFLR register.

Note

When emulation mode is entered during the Frame transmission or reception of the frame and CONT bit is not set, Communication is not expected to be successful. The suggested usage is to set CONT bit during emulation mode for successful communication.

30.6 Software

30.6.1 LIN Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/lin

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

30.6.1.1 LIN Internal Loopback with Interrupts

FILE: lin_ex1_loopback_interrupts.c

This example configures the LIN module in master mode for internal loopback with interrupts. The module is setup to perform 8 data transmissions with different transmit IDs and varying transmit data. Upon reception of an ID header, an interrupt is triggered on line 0 and an interrupt service routine (ISR) is called. The received data is then checked for accuracy.

The example can be adjusted to use interrupt line 1 instead of line 0 by uncommenting "LIN_setInterruptLevel1()".

External Connections

- None.

Watch Variables

- txData - An array with the data being sent
- rxData - An array with the data that was received
- result - The example completion status (PASS = 0xABCD, FAIL = 0xFFFF)
- level0Count - The number of line 0 interrupts
- level1Count - The number of line 1 interrupts

30.6.1.2 LIN SCI Mode Internal Loopback with Interrupts

FILE: lin_ex2_sci_loopback.c

This example configures the LIN module in SCI mode for internal loopback with interrupts. The LIN module performs as a SCI with a set character and frame length in a non-multi-buffer mode. The module is setup to continuously transmit a character, wait to receive that character, and repeat.

External Connections

- None.

Watch Variables

- rxCount - The number of RX interrupts
- transmitChar - The character being transmitted
- receivedChar - The character received

30.6.1.3 LIN SCI MODE Internal Loopback with DMA

FILE: lin_ex3_sci_dma.c

This example configures the LIN module in SCI mode for internal loopback with the use of the DMA. The LIN module performs as SCI with a set character and frame length in multi-buffer mode. When the transmit buffers in the LINTD0 and LINTD1 registers have enough space, the DMA will transfer data from global variable sData into those transmit registers. Once the received buffers in the LINRD0 and LINRD1 registers contain data,the DMA will transfer the data into the global variable rdata.

When all data has been placed into rData, a check of the validity of the data will be performed in one of the DMA channels' ISRs.

External Connections

- None

Watch Variables

- *sData* - Data to send
- *rData* - Received data

30.6.1.4 LIN Internal Loopback without Interrupts (polled mode)

FILE: lin_ex4_loopback_polling.c

This example configures the LIN module in master mode for internal loopback without interrupts. The module is setup to perform 8 data transmissions with different transmit IDs and varying transmit data. Waits for reception of an ID header. The received data is then checked for accuracy.

External Connections

- None.

Watch Variables

- txData - An array with the data being sent
- rxData - An array with the data that was received
- result - The example completion status (PASS = 0xABCD, FAIL = 0xFFFF)

30.6.1.5 LIN Internal Loopback with Interrupts using Sysconfig

FILE: lin_ex5_syscfg.c

This example is similar to ex1 but using syscfg tool to configure the LIN Module paramters. The file lin_ex5_syscfg.syscfg can be updated using the GUI tool to update the configuration parameters.
This example configures the LIN module in master mode for internal loopback with interrupts. The module is setup to perform 8 data transmissions with different transmit IDs and varying transmit data. Upon reception of an ID header, an interrupt is triggered on line 0 and an interrupt service routine (ISR) is called. The received data is then checked for accuracy.

External Connections

- None.

Watch Variables

- txData - An array with the data being sent
- rxData - An array with the data that was received
- result - The example completion status (PASS = 0xABCD, FAIL = 0xFFFF)
- level0Count - The number of line 0 interrupts
- level1Count - The number of line 1 interrupts

30.6.1.6 LIN Incomplete Header Detection

FILE: lin_ex6_detect_incomplete_header.c

This example demonstrates how an error in the header field of a LIN frame can be detected. It configures the LIN module in slave mode and waits till new frame is received. It configures an external interrupt to trigger by a falling edge in the LIN Rx pin confirming start of frame. In the ISR of XINT, a timer is configured to trigger after max time consumed by reception of a complete LIN frame. If a LIN frame is successfully received the timer is disabled before getting triggered in the LIN ISR else the timer ISR is called which indicates an error in header frame.

External Connections

- LINATX/RX to host node via transceiver.

Watch Variables

- linHeaderError - Number of times header error was detected
- rxData - An array with the data that was received if successfull reception
- xint1Count - Number of timer XINT triggered

30.6.1.7 LIN External Loopback without Interrupts (polled mode)

FILE: lin_ex7_external_loopback.c

This example configures the LINA module in master mode and LINB module in slave mode. Master transmits 8 byte data to the slave with a matching ID configured in slave Receive ID. The received data is then checked for accuracy.

External Connections

- Connect LINA TX/RX with LINB TX/RX via transceiver.

Watch Variables

- txData - An array with the data being sent
- rxData - An array with the data that was received
- result - The example completion status (PASS = 0xABCD, FAIL = 0xFFFF)

30.7 SCI/LIN Registers

The SCI/LIN module registers are based on the SCI registers, with added functionality registers enabled by the LIN MODE bit in the SCIGCR1 register.

These registers are accessible in 32-bit reads or writes. The SCI/LIN is controlled and accessed through the registers listed in the following sections. Among the features that can be programmed are the LIN protocol mode, communication and timing modes, baud rate value, frame format, DMA requests, and interrupt configuration.

30.7.1 LIN Base Address Table

Table 30-13. LIN Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
LinaRegs	LIN_REGS	LINA_BASE	0x0000_6A00	YES	YES	YES	YES	YES
LinbRegs	LIN_REGS	LINB_BASE	0x0000_6B00	YES	YES	YES	YES	YES

30.7.2 LIN_REGS Registers

Table 30-14 lists the memory-mapped registers for the LIN_REGS registers. All register offset addresses not listed in Table 30-14 should be considered as reserved locations and the register contents should not be modified.

Table 30-14. LIN_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	SCIGCR0	Global Control Register 0		Go
4h	SCIGCR1	Global Control Register 1		Go
8h	SCIGCR2	Global Control Register 2		Go
Ch	SCISETINT	Interrupt Enable Register		Go
10h	SCICLEARINT	Interrupt Disable Register		Go
14h	SCISETINTLVL	Set Interrupt Level Register		Go
18h	SCICLEARINTLVL	Clear Interrupt Level Register		Go
1Ch	SCIFLR	Flag Register		Go
20h	SCIINTVECT0	Interrupt Vector Offset Register 0		Go
24h	SCIINTVECT1	Interrupt Vector Offset Register 1		Go
28h	SCIFORMAT	Length Control Register		Go
2Ch	BRSR	Baud Rate Selection Register		Go
30h	SCIED	Emulation buffer Register		Go
34h	SCIRD	Receiver data buffer Register		Go
38h	SCITD	Transmit data buffer Register		Go
3Ch	SCIPIO0	Pin control Register 0		Go
44h	SCIPIO2	Pin control Register 2		Go
60h	LINCOMP	Compare register		Go
64h	LINRD0	Receive data register 0		Go
68h	LINRD1	Receive data register 1		Go
6Ch	LINMASK	Acceptance mask register		Go
70h	LINID	LIN ID Register		Go
74h	LINTD0	Transmit Data Register 0		Go
78h	LINTD1	Transmit Data Register 1		Go
7Ch	MBRSR	Maximum Baud Rate Selection Register		Go
90h	IODFTCTRL	IODFT for LIN		Go
E0h	LIN_GLB_INT_EN	LIN Global Interrupt Enable Register		Go
E4h	LIN_GLB_INT_FLG	LIN Global Interrupt Flag Register		Go
E8h	LIN_GLB_INT_CLR	LIN Global Interrupt Clear Register		Go

Complex bit access types are encoded to fit into small table cells. Table 30-15 shows the codes that are used for access types in this section.

Table 30-15. LIN_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear

Table 30-15. LIN_REGS Access Type Codes (continued)

Access Type	Code	Description
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

30.7.2.1 SCIGCR0 Register (Offset = 0h) [Reset = 0h]

SCIGCR0 is shown in Figure 30-26 and described in Table 30-16.

Return to the Summary Table.

The SCIGCR0 register defines the module reset.

Figure 30-26. SCIGCR0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							RESET
R-0h							R/W-0h

Table 30-16. SCIGCR0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-1	RESERVED	R	0h	Reserved
0	RESET	R/W	0h	This bit resets the SCI/LIN module. This bit is effective in LIN or SCI-compatible mode.. This bit affects the reset state of the SCI/LIN module. Reset type: SYSRSn 0h (R/W) = SCI/LIN module is in held in reset. 1h (R/W) = SCI/LIN module is out of reset.

30.7.2.2 SCIGCR1 Register (Offset = 4h) [Reset = 0h]

SCIGCR1 is shown in Figure 30-27 and described in Table 30-17.

Return to the Summary Table.

The SCIGCR1 register defines the frame format, protocol, and communication mode used by the SCI.

Figure 30-27. SCIGCR1 Register

31	30	29	28	27	26	25	24
RESERVED						TXENA	RXENA
R-0h						R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED						CONT	LOOPBACK
R-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED		STOPEXTFRAME	HGENCTRL	CTYPE	MBUFMODE	ADAPT	SLEEP
R-0h		R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
SWnRST	LINMODE	CLK_Master	STOP	PARITY	PARITYENA	TIMINGMODE	COMMMODE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 30-17. SCIGCR1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-26	RESERVED	R	0h	Reserved
25	TXENA	R/W	0h	Transmit enable. This bit is effective in LIN and SCI modes. Data is transferred from SCITD or the TDy (with y=0, 1,...7) buffers in LIN mode to the SCITXSHF shift out register only when the TXENA bit is set. Note: Data written to SCITD or the transmit multi-buffer before TXENA is set is not transmitted. If TXENA is cleared while transmission is ongoing, the data previously written to SCITD is sent (including the checksum byte in LIN mode). Reset type: SYSRSn 0h (R/W) = Disable transfers from SCITD or TDy to SCITXSHF 1h (R/W) = Enable transfers of data from SCITD or TDy to SCITXSHF

Table 30-17. SCIGCR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
24	RXENA	R/W	0h	Receive enable. This bit is effective in LIN or SCI-compatible mode. RXENA allows or prevents the transfer of data from SCIRXSHF to SCIRD or the receive multibuffers. Note: Clearing RXENA stops received characters from being transferred into the receive buffer or multi-buffers, prevents the RX status flags (see Table 7) from being updated by receive data, and inhibits both receive and error interrupts. However, the shift register continues to assemble data regardless of the state of RXENA. Note: If RXENA is cleared before the time the reception of a frame is complete, the data from the frame is not transferred into the receive buffer. Note: If RXENA is set before the time the reception of a frame is complete, the data from the frame is transferred into the receive buffer. If RXENA is set while SCIRXSHF is in the process of assembling a frame, the status flags are not guaranteed to be accurate for that frame. To ensure that the status flags correctly reflect what was detected on the bus during a particular frame, RXENA should be set before the detection of that frame Reset type: SYSRSn 0h (R/W) = Prevents the receiver from transferring data from the shift buffer to the receive buffer or multi-buffers 1h (R/W) = Allows the receiver to transfer data from the shift buffer to the receive buffer or multi-buffers
23-18	RESERVED	R	0h	Reserved
17	CONT	R/W	0h	Continue on suspend. This bit has an effect only when a program is being debugged with an emulator, and it determines how the SCI/LIN operates when the program is suspended. This bit affects the LIN counters. When this bit is set, the counters are not stopped during debug. When this bit is cleared, the counters are stopped during debug. Reset type: SYSRSn 0h (R/W) = When debug mode is entered, the SCI/LIN state machine is frozen. Transmissions and LIN counters are halted and resume when debug mode is exited. 1h (R/W) = When debug mode is entered, the SCI/LIN continues to operate until the current transmit and receive functions are complete.
16	LOOPBACK	R/W	0h	Loopback bit. This bit is effective in LIN or SCI-compatible mode. The self-checking option for the SCI/LIN can be selected with this bit. If the LINTX and LINRX pins are configured with SCI/LIN functionality, then the LINTX pin is internally connected to the LINRX pin. Externally, during loop back operation, the LINTX pin outputs a high value and the LINRX pin is in a high-impedance state. If this bit value is changed while the SCI/LIN is transmitting or receiving data, errors may result. Reset type: SYSRSn 0h (R/W) = Loopback mode is disabled. 1h (R/W) = Loopback mode is enabled.
15-14	RESERVED	R	0h	Reserved
13	STOPEXTFRAME	R/W	0h	Stop extended frame communication. This bit is effective in LIN mode only. This bit can be written only during extended frame communication. When the extended frame communication is stopped, this bit is cleared automatically. Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Extended frame communication will be stopped, once current frame transmission/reception is completed.

Table 30-17. SCIGCR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12	HGENCTRL	R/W	0h	HGEN control bit. This bit is effective in LIN mode only. This bit controls the type of mask filtering comparison. Reset type: SYSRSn 0h (R/W) = ID filtering using ID-Byte. RECEIVEDID and IDBYTE fields in the LINID register are used for detecting a match (using TX/RXMASK values). Mask of 0xFF in LINMASK register will result in NO match. 1h (R/W) = ID filtering using ID-SLAVETask byte (Recommended). RECEIVEDID and IDSLAVETASKBYTE fields in the LINID register are used for detecting a match (using TX/RXMASK values). Mask of 0xFF in LINMASK register will result in ALWAYS match
11	CTYPE	R/W	0h	Checksum type. This bit is effective in LIN mode only. This bit controls the type of checksum to be used: classic or enhanced. Reset type: SYSRSn 0h (R/W) = Classic checksum is used. This checksum is compatible with LIN 1.3 Slave nodes. The classic checksum contains the modulo-256 sum with carry over all data bytes. Frames sent with Identifier 60 (0x3C) to 63 (0x3F) must allways use the classic checksum. 1h (R/W) = Enhanced checksum is used. The enhanced checksum is compatible with LIN 2.0 and newer Slave nodes. The enhanced checksum contains the modulo-256 sum with carry over all data bytes AND the protected Identifier.
10	MBUFMODE	R/W	0h	Multibuffer mode. This bit is effective in LIN or SCI-compatible mode. This bit controls receive/transmit buffer usage, that is, whether the RX/TX multibuffers are used or a single register, RD0/TD0, is used. Reset type: SYSRSn 0h (R/W) = The multi-buffer mode is disabled. 1h (R/W) = The multi-buffer mode is enabled.
9	ADAPT	R/W	0h	Adapt mode enable. This mode is effective in LIN mode only. This bit has an effect during the detection of the Sync Field. There are two LIN protocol bit rate modes that could be enabled with this bit according to the Node capability file definition: automatic or select. Software and network configuration will decide which of the previous two modes. When this bit is cleared, the LIN 2.0 protocol fixed bit rate should be used. If the ADAPT bit is set, a LIN Slave node detecting the baudrate will compare it to the prescalers in BRSR register and update it if they are different. The BRSR register will be updated with the new value. If this bit is not set there will be no adjustment to the BRSR register. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Automatic baudrate adjustment is disabled. 1h (R/W) = Automatic baudrate adjustment is enabled.
8	SLEEP	R/W	0h	SCI sleep. SCI compatibility mode only. In a multiprocessor configuration, this bit controls the receive sleep function. Clearing this bit brings the SCI out of sleep mode. The receiver still operates when the SLEEP bit is set however, RXRDY is updated and SCIRD is loaded with new data only when an address frame is detected. The remaining receiver status flags are updated and an error interrupt is requested if the corresponding interrupt enable bit is set, regardless of the value of the SLEEP bit. In this way, if an error is detected on the receive data line while the SCI is asleep, software can promptly deal with the error condition. The SLEEP bit is not automatically cleared when an address byte is detected. This field is writable in SCI mode only. Reset type: SYSRSn 0h (R/W) = Sleep mode is disabled. 1h (R/W) = Sleep mode is enabled.

Table 30-17. SCIGCR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	SWnRST	R/W	0h	Software reset (active low). This bit is effective in LIN or SCI-compatible mode. The SCI/LIN should only be configured while SWnRST = 0. Only the following configuration bits can be changed in runtime (i.e., while SWnRESET = 1): - STOP EXT Frame (SCIGCR1[13]) - CC bit (SCIGCR2[17]) - SC bit (SCIGCR2[16]) Reset type: SYSRSn 0h (R/W) = The SCI/LIN is in its reset state no data will be transmitted or received. Writing a 0 to this bit intializes the SCI/LIN state machines and operating flags. All affected logic is held in the reset state until a 1 is written to this bit. 1h (R/W) = The SCI/LIN is in its ready state transmission and reception can occur. After this bit is set to 1, the configuration of the module should not change.
6	LINMODE	R/W	0h	LIN mode This bit controls the mode of operation of the module. Reset type: SYSRSn 0h (R/W) = LIN mode is disabled SCI compatibility mode is enabled. 1h (R/W) = LIN mode is enabled SCI compatibility mode is disabled.
5	CLK_Master	R/W	0h	SCI internal clock enable or LIN Master/Slave configuration. In the SCI mode, this bit enables the clock to the SCI module. In LIN mode, this bit determines whether a LIN node is a Slave or Master. Reset type: SYSRSn 0h (R/W) = SCI-compatible mode: Reserved. LIN mode: The module is in Slave mode. 1h (R/W) = SCI-compatible mode: Enable clock to the SCI module. LIN mode: The node is in Master mode.
4	STOP	R/W	0h	SCI number of stop bits. This bit is effective in SCI-compatible mode only. Note: The receiver checks for only one stop bit. However in idle-line mode, the receiver waits until the end of the second stop bit (if STOP = 1) to begin checking for an idle period. This field is writable in SCI mode only. Reset type: SYSRSn 0h (R/W) = One stop bit is used. 1h (R/W) = Two stop bits are used.
3	PARITY	R/W	0h	SCI parity odd/even selection. This bit is effective in SCI-compatible mode only. If the PARITY ENA bit (SCIGCR1.2) is set, PARITY designates odd or even parity. The parity bit is calculated based on the data bits in each frame and the address bit (in address-bit mode). The start and stop fields in the frame are not included in the parity calculation. This field is writable in SCI mode only. Reset type: SYSRSn 0h (R/W) = Odd parity is used. The SCI transmits and expects to receive a value in the parity bit that makes odd the total number of bits in the frame with the value of 1. 1h (R/W) = Even parity is used. The SCI transmits and expects to receive a value in the parity bit that makes even the total number of bits in the frame with the value of 1.

Table 30-17. SCIGCR1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	PARITYENA	R/W	0h	Parity enable. Enables or disables the parity function. Reset type: SYSRSn 0h (R/W) = SCI-compatible mode: Parity disabled no parity bit is generated during transmission or is expected during reception. LIN mode: ID-parity verification is disabled. 1h (R/W) = SCI compatible mode: Parity enabled. A parity bit is generated during transmission and is expected during reception. LIN mode: ID-parity verification is enabled.
1	TIMINGMODE	R/W	0h	SCI timing mode bit. This bit is effective in SCI-compatible mode only. It must be set to 1 when the SCI mode is used. This bit configures the SCI for asynchronous operation. Reset type: SYSRSn 0h (R/W) = Reserved. 1h (R/W) = Must be set to 1 when module is configured for SCI operation
0	COMMMODE	R/W	0h	SCI/LIN communication mode bit. In compatibility mode, it selects the SCI communication mode. In LIN mode it selects length control option for ID-field bits ID4 and ID5. Reset type: SYSRSn 0h (R/W) = SCI-compatible mode: Idle-line mode is used. LIN mode: ID4 and ID5 are not used for length control. 1h (R/W) = SCI-compatible mode: Address-bit mode is used. LIN mode: ID4 and ID5 are used for length control.

30.7.2.3 SCIGCR2 Register (Offset = 8h) [Reset = 0h]

SCIGCR2 is shown in Figure 30-28 and described in Table 30-18.

Return to the Summary Table.

The SCIGCR2 register is used to send or compare a checksum byte during extended frames, to generate a wakeup and for low-power mode control of the LIN module.

Figure 30-28. SCIGCR2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED						CC	SC
R-0h						R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							GENWU
R-0h							R/W-0h
7	6	5	4	3	2	1	0
RESERVED							POWERDOWN
R-0h							R/W-0h

Table 30-18. SCIGCR2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-18	RESERVED	R	0h	Reserved
17	CC	R/W	0h	Compare Checksum. This mode is effective in LIN mode only. This bit is used by the receiver for extended frames to trigger a checksum compare. The user will initiate this transaction by writing a one to this bit. In non multibuffer mode, once the CC bit is set, the checksum will be compared on the byte that is currently being received, expected to be the checkbyte. During Multi-buffer mode, following are the scenarios associated with the CC bit : - If CC bit is set during the reception of the data, then the byte that is received after the reception of the programmed no. of data bytes indicated by SCIFORMAT[18:16], is treated as a checksum byte. - If CC bit is set during the IDLE period (i.e. during inter-frame space), then the next immediate byte will be treated as a checksum byte. A CE will immediatly be flagged if there is a checksum error. This bit is automatically cleared once the checksum is successfully compared. Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Compare checksum on expected checkbyte

Table 30-18. SCIGCR2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	SC	R/W	0h	Send Checksum This mode is effective in LIN mode only. This bit is used by the transmitter with extended frames to send a checkbyte. In non multibuffer mode the checkbyte will be sent after the current byte transmission. In multibuffer mode the checkbyte will be sent after the last byte count, indicated by the SCIFORMAT[18:16]). This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = No checkbyte will be sent. 1h (R/W) = A checkbyte will be sent. This bit will automatically get cleared after the checkbyte is transmitted. The checksum will not be sent if this bit is set before transmitting the very first byte, that is, during interframe space.
15-9	RESERVED	R	0h	Reserved
8	GENWU	R/W	0h	Generate wakeup signal. This bit controls the generation of a wakeup signal, by transmitting the TDO buffer value. This bit is cleared on reception of a valid sync break. Reset type: SYSRSn 0h (R/W) = No effect 1h (R/W) = Transmit TDO for wakeup. This bit will be cleared on a SWnRST (SCIGCR1.7)
7-1	RESERVED	R	0h	Reserved
0	POWERDOWN	R/W	0h	Power down. This bit is effective in LIN or SCI-compatible mode. When the powerdown bit is set, the SCI/LIN module attempts to enter local low-power mode. If the POWERDOWN bit is set while the receiver is actively receiving data and the wakeup interrupt is disabled, then the SCI/LIN will delay low-power mode from being entered until completion of reception. In LIN mode the user may set the POWERDOWN bit on Sleep Command reception or on idle bus detection (more than 4 seconds, i.e. 80,000 cycles at 20kHz) Reset type: SYSRSn 0h (R/W) = Normal operation 1h (R/W) = Request local low-power mode

30.7.2.4 SCISETINT Register (Offset = Ch) [Reset = 0h]

SCISETINT is shown in Figure 30-29 and described in Table 30-19.

Return to the Summary Table.

The SCISETINT register is used to enable the various interrupts available in the LIN module.

Figure 30-29. SCISETINT Register

31	30	29	28	27	26	25	24
SETBEINT	SETPBEINT	SETCEINT	SETISFEINT	SETNREINT	SETFEINT	SETOEINT	SETPEINT
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
23	22	21	20	19	18	17	16
RESERVED					SET_RX_DMA_ALL	SET_RX_DMA	SET_TX_DMA
R-0h					R/W1S-0h	R/W1S-0h	R/W1S-0h
15	14	13	12	11	10	9	8
RESERVED		SETIDINT	RESERVED			SETRXINT	SETTXINT
R-0h		R/W1S-0h	R-0h			R/W1S-0h	R/W1S-0h
7	6	5	4	3	2	1	0
SETTOA3WUSINT	SETTOAWUSINT	RESERVED	SETTIMEOUTINT	RESERVED		SETWAKEUPINT	SETBRKDTINT
R/W1S-0h	R/W1S-0h	R-0h	R/W1S-0h	R-0h		R/W1S-0h	R/W1S-0h

Table 30-19. SCISETINT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SETBEINT	R/W1S	0h	Set bit error interrupt. This bit is effective in LIN mode only. Setting this bit enables the SCI/LIN module to generate an interrupt when there is a bit error. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled.
30	SETPBEINT	R/W1S	0h	Set physical bus error interrupt. This bit is effective in LIN mode only. Setting this bit enables the SCI/LIN module to generate an interrupt when a physical bus error occurs. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled.
29	SETCEINT	R/W1S	0h	Set checksum-error Interrupt. This bit is effective in LIN mode only. Setting this bit enables the SCI/LIN module to generate an interrupt when there is a checksum error. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled.
28	SETISFEINT	R/W1S	0h	Set inconsistent-sync-field-error interrupt. This bit is effective in LIN mode only. Setting this bit enables the SCI/LIN module to generate an interrupt when there is an inconsistent sync field error. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled.

Table 30-19. SCISETINT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	SETNREINT	R/W1S	0h	Set no-response-error interrupt. This bit is effective in LIN mode only. Setting this bit enables the SCI/LIN module to generate an interrupt when a no-response error occurs. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled.
26	SETFEINT	R/W1S	0h	Set framing-error interrupt. This bit is effective in LIN or SCI-compatible mode. Setting this bit enables the SCI/LIN module to generate an interrupt when a framing error occurs. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled.
25	SETOEINT	R/W1S	0h	Set overrun-error interrupt. This bit is effective in LIN or SCI-compatible mode. Setting this bit enables the SCI/LIN module to generate an interrupt when an overrun error occurs. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled.
24	SETPEINT	R/W1S	0h	Set parity interrupt. This bit is effective in LIN or SCI-compatible mode. Setting this bit enables the SCI/LIN module to generate an interrupt when a parity error occurs. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled.
23-19	RESERVED	R	0h	Reserved
18	SET_RX_DMA_ALL	R/W1S	0h	Set receiver DMA for Address & Data frames. This bit is effective in LIN or SCI-compatible mode. To enable RX DMA request for address and data frames this bit must be set. If it is cleared, RX interrupt request is generated for address frames and DMA requests are generated for data frames. Reset type: SYSRSn 0h (R/W) = Receiver DMA request is disabled for address frames (RX interrupt request is enabled for address frames). Writing a 0 to this bit has no effect. 1h (R/W) = Receiver DMA request is enabled for address and data frames
17	SET_RX_DMA	R/W1S	0h	Set receiver DMA. This bit is effective in LIN or SCI-compatible mode. To enable DMA requests for the receiver this bit must be set. If it is cleared, interrupt requests are generated depending on SETRXINT. Reset type: SYSRSn 0h (R/W) = Receiver DMA request is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Receiver DMA request is enabled.
16	SET_TX_DMA	R/W1S	0h	Set transmit DMA. This bit is effective in LIN or SCI-compatible mode. To enable DMA requests for the transmitter, this bit must be set. If it is cleared, interrupt requests are generated depending on SETTXINT. Reset type: SYSRSn 0h (R/W) = Transmit DMA request is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Transmit DMA request is enabled
15-14	RESERVED	R	0h	Reserved

Table 30-19. SCISETINT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
13	SETIDINT	R/W1S	0h	Set Identification interrupt. This bit is effective in LIN mode only. This bit is set to enable interrupt once a valid matching identifier is received. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled.
12-10	RESERVED	R	0h	Reserved
9	SETRXINT	R/W1S	0h	Set Receiver interrupt. Setting this bit enables the SCI/LIN to generate a receive interrupt after a frame has been completely received and the data is being transferred from SCIRXSHF to SCIRD. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled.
8	SETTXINT	R/W1S	0h	Set Transmitter interrupt. Setting this bit enables the SCI/LIN to generate a transmit interrupt as data is being transferred from SCITD to SCITXSHF and the TXRDY bit is being set. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled.
7	SETTOA3WUSINT	R/W1S	0h	Set Timeout After 3 Wakeup Signals interrupt. This bit is effective in LIN mode only. Setting this bit enables the SCI/LIN to generate an interrupt when there is a timeout after 3 wakeup signals have been sent. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled.
6	SETTOAWUSINT	R/W1S	0h	Set Timeout After Wakeup Signal interrupt. This bit is effective in LIN mode only. Setting this bit enables the SCI/LIN to generate an interrupt when there is a timeout after one wakeup signal has been sent. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled.
5	RESERVED	R	0h	Reserved
4	SETTIMEOUTINT	R/W1S	0h	Set timeout interrupt. This bit is effective in LIN mode only. Setting this bit enables the SCI/LIN to generate an interrupt when no LIN bus activity (bus idle) occurs for at least 4 seconds. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled.
3-2	RESERVED	R	0h	Reserved
1	SETWAKEUPINT	R/W1S	0h	Set wake-up interrupt. This bit is effective in LIN or SCI-compatible mode. Setting this bit enables the SCI/LIN to generate a wake-up interrupt and thereby exit low-power mode. The wake-up interrupt is asserted on falling edge of the wake-up pulse. If enabled, the wake-up interrupt is asserted when local low-power mode is requested while the receiver is busy or if a low level is detected on the SCIRX pin during low-power mode. Wake-up interrupt is not asserted upon a wakeup pulse if the module is not in power down mode. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled.

Table 30-19. SCISETINT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	SETBRKDTINT	R/W1S	0h	Set break-detect interrupt. This bit is effective in SCI-compatible mode only. Setting this bit enables the SCI/LIN to generate an interrupt if a break condition is detected on the LINRX pin. This field is writable in SCI mode only. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled.

30.7.2.5 SCICLEARINT Register (Offset = 10h) [Reset = 0h]

SCICLEARINT is shown in Figure 30-30 and described in Table 30-20.

Return to the Summary Table.

The SCICLEARINT register is used to disable the enabled interrupts without accessing the SCISETINT register.

Figure 30-30. SCICLEARINT Register

31	30	29	28	27	26	25	24
CLRBEINT	CLRPBEINT	CLRCEINT	CLRISFEINT	CLRNREINT	CLRFEINT	CLROEINT	CLRPEINT
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
23	22	21	20	19	18	17	16
RESERVED					RESERVED	SETRXDMA	CLRTXDMA
R-0h					R-0h	R/W1C-0h	R/W1C-0h
15	14	13	12	11	10	9	8
RESERVED		CLRIDINT	RESERVED			CLRRXINT	CLRTXINT
R-0h		R/W1C-0h	R-0h			R/W1C-0h	R/W1C-0h
7	6	5	4	3	2	1	0
CLRTOA3WUSINT	CLRTOAWUSINT	RESERVED	CLRTIMEOUTINT	RESERVED		CLRWAKEUPINT	CLRBRKDTINT
R/W1C-0h	R/W1C-0h	R-0h	R/W1C-0h	R-0h		R/W1C-0h	R/W1C-0h

Table 30-20. SCICLEARINT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CLRBEINT	R/W1C	0h	Clear Bit Error Interrupt. This bit is effective in LIN mode only. Setting this bit disables the bit error interrupt. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled. Writing a 1 to this bit will disable the interrupt and clear this bit.
30	CLRPBEINT	R/W1C	0h	Clear Physical Bus Error Interrupt. This bit is effective in LIN mode only. Setting this bit disables the physical-bus error interrupt. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled. Writing a 1 to this bit will disable the interrupt and clear this bit.
29	CLRCEINT	R/W1C	0h	Clear checksum-error Interrupt. This bit is effective in LIN mode only. Setting this bit disables the checksum-error interrupt. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled. Writing a 1 to this bit will disable the interrupt and clear this bit.
28	CLRISFEINT	R/W1C	0h	Clear Inconsistent-Sync-Field-Error Interrupt. This bit is effective in LIN mode only. Setting this bit disables the ISFE interrupt. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled. Writing a 1 to this bit will disable the interrupt and clear this bit.

Table 30-20. SCICLEARINT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	CLRNREINT	R/W1C	0h	Clear No-Reponse-Error Interrupt. This bit is effective in LIN mode only. Setting this bit disables the no-response error interrupt. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled. Writing a 1 to this bit will disable the interrupt and clear this bit.
26	CLRFEINT	R/W1C	0h	Clear Framing-Error Interrupt. This bit is effective in LIN or SCI-compatible mode. Setting this bit disables framing-error interrupt. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled. Writing a 1 to this bit will disable the interrupt and clear this bit.
25	CLROEINT	R/W1C	0h	Clear Overrun-Error Interrupt. This bit is effective in LIN or SCI-compatible mode. Setting this bit disables the overrun interrupt. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled. Writing a 1 to this bit will disable the interrupt and clear this bit.
24	CLRPEINT	R/W1C	0h	Clear Parity Interrupt. This bit is effective in LIN or SCI-compatible mode. Setting this bit disables the parity error interrupt. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled. Writing a 1 to this bit will disable the interrupt and clear this bit.
23-19	RESERVED	R	0h	Reserved
18	RESERVED	R	0h	Reserved
17	SETRXDMA	R/W1C	0h	Clear receiver DMA. This bit is effective in LIN or SCI-compatible mode. Setting this bit disables the receive DMA request. Reset type: SYSRSn 0h (R/W) = Receiver DMA request is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Receiver DMA request is enabled. Writing a 1 to this bit will disable the DMA request and clear this bit.
16	CLRTXDMA	R/W1C	0h	Clear transmit DMA. This bit is effective in LIN or SCI-compatible mode. Setting this bit disables the transmit DMA request. Reset type: SYSRSn 0h (R/W) = Transmit DMA request is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Transmit DMA request is enabled. Writing a 1 to this bit will disable the DMA request and clear this bit.
15-14	RESERVED	R	0h	Reserved
13	CLRIDINT	R/W1C	0h	Clear Identifier interrupt. This bit is effective in LIN mode only. Setting this bit disables the ID interrupt. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled. Writing a 1 to this bit will disable the interrupt and clear this bit.
12-10	RESERVED	R	0h	Reserved

Table 30-20. SCICLEARINT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	CLRRXINT	R/W1C	0h	Clear Receiver interrupt. This bit is effective in LIN or SCI-compatible mode. Setting this bit disables the receiver interrupt. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled. Writing a 1 to this bit will disable the interrupt and clear this bit.
8	CLRTXINT	R/W1C	0h	Clear Transmitter interrupt. This bit is effective in LIN or SCI-compatible mode. Setting this bit disables the transmitter interrupt. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled. Writing a 1 to this bit will disable the interrupt and clear this bit.
7	CLRTOA3WUSINT	R/W1C	0h	Clear Timeout After 3 Wakeup Signals interrupt. This bit is effective in LIN mode only. Setting this bit disables the timeout after 3 wakeup signals interrupt. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled. Writing a 1 to this bit will disable the interrupt and clear this bit.
6	CLRTOAWUSINT	R/W1C	0h	Clear Timeout After Wakeup Signal interrupt. This bit is effective in LIN mode only. Setting this bit disables the timeout after one wakeup signal interrupt. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled. Writing a 1 to this bit will disable the interrupt and clear this bit.
5	RESERVED	R	0h	Reserved
4	CLRTIMEOUTINT	R/W1C	0h	Clear Timeout interrupt. This bit is effective in LIN mode only. Setting this bit disables the timeout (LIN bus idle) interrupt. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled. Writing a 1 to this bit will disable the interrupt and clear this bit.
3-2	RESERVED	R	0h	Reserved
1	CLRWAKEUPINT	R/W1C	0h	Clear Wake-up interrupt. This bit is effective in LIN or SCI-compatible mode. Setting this bit disables the wake-up interrupt. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled. Writing a 1 to this bit will disable the interrupt and clear this bit.
0	CLRBRKDTINT	R/W1C	0h	Clear Break-detect interrupt. This bit is effective in SCI-compatible mode only. Setting this bit disables the Break-detect interrupt. This field is writable in SCI mode only. Reset type: SYSRSn 0h (R/W) = Interrupt is disabled. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt is enabled. Writing a 1 to this bit will disable the interrupt and clear this bit.

30.7.2.6 SCISETINTLVL Register (Offset = 14h) [Reset = 0h]

SCISETINTLVL is shown in Figure 30-31 and described in Table 30-21.

Return to the Summary Table.

The SCISETINTLVL register is used to map individual interrupt sources to the INT1 interrupt line.

Figure 30-31. SCISETINTLVL Register

31	30	29	28	27	26	25	24
SETBEINTLVL	SETPBEINTLVL	SETCEINTLVL	SETISFEINTLVL	SETNREINTLVL	SETFEINTLVL	SETOEINTLVL	SETPEINTLVL
R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h
23	22	21	20	19	18	17	16
RESERVED					RESERVED	RESERVED	
R-0h					R-0h	R-0h	
15	14	13	12	11	10	9	8
RESERVED		SETIDINTLVL	RESERVED			SETRXINTOVO	SETTXINTLVL
R-0h		R/W1S-0h	R-0h			R/W1S-0h	R/W1S-0h
7	6	5	4	3	2	1	0
SETTOA3WUSINTLVL	SETTOAWUSINTLVL	RESERVED	SETTIMEOUTINTLVL	RESERVED		SETWAKEUPINTLVL	SETBRKDTINTLVL
R/W1S-0h	R/W1S-0h	R-0h	R/W1S-0h	R-0h		R/W1S-0h	R/W1S-0h

Table 30-21. SCISETINTLVL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SETBEINTLVL	R/W1S	0h	Set Bit Error interrupt level. This bit is effective in LIN mode only. Writing to this bit maps the Bit Error interrupt level to the INT1 line. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line.
30	SETPBEINTLVL	R/W1S	0h	Set Physical Bus Error interrupt level. This bit is effective in LIN mode only. Writing to this bit maps the Physical Bus Error interrupt level to the INT1 line. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line.
29	SETCEINTLVL	R/W1S	0h	Set Checksum-error interrupt level. This bit is effective in LIN mode only. Writing to this bit maps the Checksum-error interrupt level to the INT1 line. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line.
28	SETISFEINTLVL	R/W1S	0h	Set Inconsistent-Sync-Field-Error interrupt level. This bit is effective in LIN mode only. Writing to this bit maps the Inconsistent-Sync-Field-Error interrupt level to the INT1 line. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line.

Table 30-21. SCISETINTLVL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	SETNREINTLVL	R/W1S	0h	Set No-Reponse-Error interrupt level. This bit is effective in LIN mode only. Writing to this bit maps the No-Response-Error interrupt level to the INT1 line. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line.
26	SETFEINTLVL	R/W1S	0h	Set Framing-Error interrupt level. This bit is effective in LIN or SCI-compatible mode. Writing to this bit maps the Framing-Error interrupt level to the INT1 line. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line.
25	SETOEINTLVL	R/W1S	0h	Set Overrun-Error Interrupt Level. This bit is effective in LIN or SCI-compatible mode. Writing to this bit maps the Overrun-Error interrupt level to the INT1 line. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line.
24	SETPEINTLVL	R/W1S	0h	Set Parity Error interrupt level. This bit is effective in LIN or SCI-compatible mode. Writing to this bit maps the Parity error interrupt level to the INT1 line. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line.
23-19	RESERVED	R	0h	Reserved
18	RESERVED	R	0h	Reserved
17-16	RESERVED	R	0h	Reserved
15-14	RESERVED	R	0h	Reserved
13	SETIDINTLVL	R/W1S	0h	Set ID interrupt level. This bit is effective in LIN mode only. Writing to this bit maps the ID interrupt level to the INT1 line. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line.
12-10	RESERVED	R	0h	Reserved
9	SETRXINTOVO	R/W1S	0h	Set Receiver interrupt level. This bit is effective in LIN or SCI-compatible mode. Writing to this bit maps the receiver interrupt level to the INT1 line. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line.
8	SETTXINTLVL	R/W1S	0h	Set Transmitter interrupt level. This bit is effective in LIN or SCI-compatible mode. Writing to this bit maps the transmitter interrupt level to the INT1 line. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line.
7	SETTOA3WUSINTLVL	R/W1S	0h	Set Timeout After 3 Wakeup Signals interrupt level. This bit is effective in LIN mode only. Writing to this bit maps the timeout after 3 wakeup signals interrupt level to the INT1 line. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line.

Table 30-21. SCISETINTLVL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	SETTOAWUSINTLVL	R/W1S	0h	Set Timeout After Wakeup Signal interrupt level. This bit is effective in LIN mode only. Writing to this bit maps the the timeout after wakeup interrupt level to the INT1 line. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line.
5	RESERVED	R	0h	Reserved
4	SETTIMEOUTINTLVL	R/W1S	0h	Set Timeout interrupt level. This bit is effective in LIN mode only. Writing to this bit maps the timeout interrupt level to the INT1 line. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line.
3-2	RESERVED	R	0h	Reserved
1	SETWAKEUPINTLVL	R/W1S	0h	Set Wake-up interrupt level. This bit is effective in LIN or SCI-compatible mode. Writing to this bit maps the Wake-up interrupt level to the INT1 line. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line.
0	SETBRKDTINTLVL	R/W1S	0h	Set Break-detect interrupt level. This bit is effective in SCI-compatible mode only. Writing to this bit maps the Break-detect interrupt level to the INT1 line. This field is writable in SCI mode only. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line.

30.7.2.7 SCICLEARINTLVL Register (Offset = 18h) [Reset = 0h]

SCICLEARINTLVL is shown in Figure 30-32 and described in Table 30-22.

Return to the Summary Table.

The SCICLEARINTLVL register is used to map individual interrupt sources to the INT0 line.

Figure 30-32. SCICLEARINTLVL Register

31	30	29	28	27	26	25	24
CLRBEINTLVL	CLRPBEINTLVL	CLRCEINTLVL	CLRISFEINTLVL	CLRNREINTLVL	CLRFEINTLVL	CLROEINTLVL	CLRPEINTLVL
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
23	22	21	20	19	18	17	16
RESERVED					RESERVED	RESERVED	
R-0h					R-0h	R-0h	
15	14	13	12	11	10	9	8
RESERVED		CLRIDINTLVL	RESERVED			CLRRXINTLVL	CLRTXINTLVL
R-0h		R/W1C-0h	R-0h			R/W1C-0h	R/W1C-0h
7	6	5	4	3	2	1	0
CLRTOA3WUSINTLVL	CLRTOAWUSINTLVL	RESERVED	CLRTIMEOUTINTLVL	RESERVED		CLRWAKEUPINTLVL	CLRBRKDTINTLVL
R/W1C-0h	R/W1C-0h	R-0h	R/W1C-0h	R-0h		R/W1C-0h	R/W1C-0h

Table 30-22. SCICLEARINTLVL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CLRBEINTLVL	R/W1C	0h	Clear Bit Error interrupt level. This bit is effective in LIN mode only. Writing to this bit maps the Bit Error interrupt level to the INT0 line. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line. Writing a 1 to this bit will map the interrupt to INT0 and clear this bit.
30	CLRPBEINTLVL	R/W1C	0h	Clear Physical Bus Error interrupt level. This bit is effective in LIN mode only. Writing to this bit maps the Physical Bus Error interrupt level to the INT0 line. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line. Writing a 1 to this bit will map the interrupt to INT0 and clear this bit.
29	CLRCEINTLVL	R/W1C	0h	Clear Checksum-error interrupt level. This bit is effective in LIN mode only. Writing to this bit maps the Checksum-error interrupt level to the INT0 line. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line. Writing a 1 to this bit will map the interrupt to INT0 and clear this bit.
28	CLRISFEINTLVL	R/W1C	0h	Clear Inconsistent-Sync-Field-Error interrupt level. This bit is effective in LIN mode only. Writing to this bit maps the Inconsistent-Sync-Field-Error interrupt level to the INT0 line. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line. Writing a 1 to this bit will map the interrupt to INT0 and clear this bit.

Table 30-22. SCICLEARINTLVL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
27	CLRNREINTLVL	R/W1C	0h	Clear No-Reponse-Error interrupt level. This bit is effective in LIN mode only. Writing to this bit maps the No-Response-Error interrupt level to the INT0 line. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line. Writing a 1 to this bit will map the interrupt to INT0 and clear this bit.
26	CLRFEINTLVL	R/W1C	0h	Clear Framing-Error interrupt level. This bit is effective in LIN or SCI-compatible mode. Writing to this bit maps the Framing-Error interrupt level to the INT0 line. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line. Writing a 1 to this bit will map the interrupt to INT0 and clear this bit.
25	CLROEINTLVL	R/W1C	0h	Clear Overrun-Error Interrupt Level. This bit is effective in LIN or SCI-compatible mode. Writing to this bit maps the Overrun-Error interrupt level to the INT0 line. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line. Writing a 1 to this bit will map the interrupt to INT0 and clear this bit.
24	CLRPEINTLVL	R/W1C	0h	Clear Parity Error interrupt level. This bit is effective in LIN or SCI-compatible mode. Writing to this bit maps the Parity Error interrupt level to the INT0 line. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line. Writing a 1 to this bit will map the interrupt to INT0 and clear this bit.
23-19	RESERVED	R	0h	Reserved
18	RESERVED	R	0h	Reserved
17-16	RESERVED	R	0h	Reserved
15-14	RESERVED	R	0h	Reserved
13	CLRIDINTLVL	R/W1C	0h	Clear ID interrupt level. This bit is effective in LIN mode only. Writing to this bit maps the ID interrupt level to the INT0 line. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line. Writing a 1 to this bit will map the interrupt to INT0 and clear this bit.
12-10	RESERVED	R	0h	Reserved
9	CLRRXINTLVL	R/W1C	0h	Clear Receiver interrupt level. This bit is effective in LIN or SCI-compatible mode. Writing to this bit maps the receiver interrupt level to the INT0 line. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line. Writing a 1 to this bit will map the interrupt to INT0 and clear this bit.
8	CLRTXINTLVL	R/W1C	0h	Clear Transmitter interrupt level. This bit is effective in LIN or SCI-compatible mode. Writing to this bit maps the transmitter interrupt level to the INT0 line. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line. Writing a 1 to this bit will map the interrupt to INT0 and clear this bit.

Table 30-22. SCICLEARINTLVL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	CLRTOA3WUSINTLVL	R/W1C	0h	Clear Timeout After 3 Wakeup Signals interrupt level. This bit is effective in LIN mode only. Writing to this bit maps the timeout after 3 wakeup signals interrupt level to the INT0 line. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line. Writing a 1 to this bit will map the interrupt to INT0 and clear this bit.
6	CLRTOAWUSINTLVL	R/W1C	0h	Clear Timeout After Wakeup Signal interrupt level. This bit is effective in LIN mode only. Writing to this bit maps the the timeout after wakeup interrupt level to the INT0 line. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line. Writing a 1 to this bit will map the interrupt to INT0 and clear this bit.
5	RESERVED	R	0h	Reserved
4	CLRTIMEOUTINTLVL	R/W1C	0h	Clear Timeout interrupt level. This bit is effective in LIN mode only. Writing to this bit maps the timeout interrupt level to the INT0 line. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line. Writing a 1 to this bit will map the interrupt to INT0 and clear this bit.
3-2	RESERVED	R	0h	Reserved
1	CLRWAKEUPINTLVL	R/W1C	0h	Clear Wake-up interrupt level. This bit is effective in LIN or SCI-compatible mode. Writing to this bit maps the Wake-up interrupt level to the INT0 line. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. Writing a 0 to this bit has no effect. 1h (R/W) = Interrupt level mapped to INT1 line. Writing a 1 to this bit will map the interrupt to INT0 and clear this bit.
0	CLRBRKDTINTLVL	R/W1C	0h	Clear Break-detect interrupt level. This bit is effective in SCI-compatible mode only. Writing to this bit maps the Break-detect interrupt level to the INT0 line. This field is writable in SCI mode only. Reset type: SYSRSn 0h (R/W) = Interrupt level mapped to INT0 line. 1h (R/W) = Interrupt level mapped to INT1 line. Writing a 1 to this bit will map the interrupt to INT0 and clear this bit.

30.7.2.8 SCIFLR Register (Offset = 1Ch) [Reset = 904h]

SCIFLR is shown in Figure 30-33 and described in Table 30-23.

Return to the Summary Table.

The SCIFLR register indicates the current status of the various interrupt sources of the LIN module.

Figure 30-33. SCIFLR Register

31	30	29	28	27	26	25	24
BE	PBE	CE	ISFE	NRE	FE	OE	PE
R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	IDRXFLAG	IDTXFLAG	RXWAKE	TXEMPTY	TXWAKE	RXRDY	TXRDY
R-0h	R/W1C-0h	R/W1C-0h	R-0h	R-1h	R/W-0h	R/W1C-0h	R-1h
7	6	5	4	3	2	1	0
TOA3WUS	TOAWUS	RESERVED	TIMEOUT	BUSY	IDLE	WAKEUP	BRKDT
R/W1C-0h	R/W1C-0h	R-0h	R/W1C-0h	R-0h	R-1h	R/W1C-0h	R/W1C-0h

Table 30-23. SCIFLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	BE	R/W1C	0h	Bit Error Flag. This bit is effective in LIN mode only. This bit is set when there has been a bit error. This is detected by the bit monitor in the internal bit monitor. This bit is cleared by: - Reading the corresponding interrupt offset in the SCIINTVECT0/1 register - Setting the SWnRESET bit (SCIGCR1.7) - RESET bit (SCIGCR0.0) - System reset - Writing a 1 to this bit - Reception of a new sync break This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = No bit error detected. 1h (R/W) = Bit error detected.
30	PBE	R/W1C	0h	Physical Bus Error Flag. This bit is effective in LIN mode only. This bit is set when there has been a physical bus error. This is detected by the bit monitor in TED. This bit is cleared by: - Reading the corresponding interrupt offset in the SCIINTVECT0/1 register - Setting the SWnRESET bit (SCIGCR1.7) - RESET bit (SCIGCR0.0) - System reset - Writing a 1 to this bit - Reception of a new sync break Note: thie PBE will ony be flagged if no sync break can be generated. (because of a bus shortage to VBAT) or if no sync break delimeter can be generated (because of a bus shortage to GND). This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = No physiscal bus error detected. 1h (R/W) = Physical bus error detected.

Table 30-23. SCIFLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
29	CE	R/W1C	0h	Checksum Error Flag. This bit is effective in LIN mode only. This bit is set when there is checksum error detected by a receiving node. The type of checksum to be used depends on the SCIGCR1.CTYPE bit. This bit is cleared by: - Reading the corresponding interrupt offset in the SCIINTVECT0/1 register - Setting the SWnRESET bit (SCIGCR1.7) - RESET bit (SCIGCR0.0) - System reset - Writing a 1 to this bit - Reception of a new sync break This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = No Checksum error detected. 1h (R/W) = Checksum error detected.
28	ISFE	R/W1C	0h	Inconsistent Sync Field Error Flag. This bit is effective in LIN mode only. This bit is set when there has been an inconsistent Sync Field error detected by the synchronizer during header reception. See the "Header Reception and Adaptive Baudrate" section for more information. This bit is cleared by: - Reading the corresponding interrupt offset in the SCIINTVECT0/1 register - Setting the SWnRESET bit (SCIGCR1.7) - RESET bit (SCIGCR0.0) - System reset - Writing a 1 to this bit - Reception of a new sync break This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = No Inconsistent Sync Field error detected. 1h (R/W) = Inconsistent Sync Field error detected.
27	NRE	R/W1C	0h	No-Response Error Flag. This bit is effective in LIN mode only. This bit is set when there is no response to a Master's header completed within TFRAME_MAX. This timeout period is applied for message frames of unknown length (identifiers 0 to 61). This error is detected by the synchronizer of the module. This bit is cleared by: - Reading the corresponding interrupt offset in the SCIINTVECT0/1 register - Setting the SWnRESET bit (SCIGCR1.7) - RESET bit (SCIGCR0.0) - System reset - Writing a 1 to this bit - Reception of a new sync break This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = No No-Response error detected. 1h (R/W) = No-Response error detected.

Table 30-23. SCIFLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
26	FE	R/W1C	0h	Framing error flag. This bit is effective in LIN or SCI-compatible mode. This bit is set when an expected stop bit is not found. In SCI compatible mode, only the first stop bit is checked. The missing stop bit indicates that synchronization with the start bit has been lost and that the character is incorrectly framed. Detection of a framing error causes the SCI to generate an error interrupt if the RXERR INT ENA bit is set. This bit is cleared by: - Reading the corresponding interrupt offset in the SCIINTVECT0/1 register - Setting the SWnRESET bit (SCIGCR1.7) - RESET bit (SCIGCR0.0) - System reset - Writing a 1 to this bit - Reception of a new character (SCI-compatible mode), or frame (LIN mode) In multibuffer mode the frame is defined in the SCIFORMAT register. Reset type: SYSRSn 0h (R/W) = No framing error detected. 1h (R/W) = Framing error detected.
25	OE	R/W1C	0h	Overrun error flag. This bit is effective in LIN or SCI-compatible mode. This bit is set when the transfer of data from SCIRXSHF to SCIRD overwrites unread data already in SCIRD or the RDy buffers. Detection of an overrun error causes the LIN to generate an error interrupt if the SET OE INT bit is one. This bit is cleared by: - Reading the corresponding interrupt offset in the SCIINTVECT0/1 register - Setting the SWnRESET bit (SCIGCR1.7) - RESET bit (SCIGCR0.0) - System reset - Writing a 1 to this bit Reset type: SYSRSn 0h (R/W) = No overrun error detected. 1h (R/W) = Overrun error detected.
24	PE	R/W1C	0h	Parity error flag. This bit is effective in LIN or SCI-compatible mode. This bit is set when a parity error is detected in the received data. In SCI address-bit mode, the parity is calculated on the data and address bit fields of the received frame. In idle-line mode, only the data is used to calculate parity. An error is generated when a character is received with a mismatch between the number of 1s and its parity bit. For more information on parity checking, see the "SCI Global Control Register (SCIGCR1)" description. If the parity function is disabled (that is, SCIGCR1.2 = 0), the PE flag is disabled and read as 0. Detection of a parity error causes the LIN to generate an error interrupt if the SET PE INT bit = 1. This bit is cleared by: - Reading the corresponding interrupt offset in the SCIINTVECT0/1 register - Setting the SWnRESET bit (SCIGCR1.7) - RESET bit (SCIGCR0.0) - System reset - Reception of a new charcter (SCI-compatible mode) or frame (LIN mode) - Writing a 1 to this bit Reset type: SYSRSn 0h (R/W) = No parity error or parity disabled. 1h (R/W) = Parity error detected.
23-16	RESERVED	R	0h	Reserved
15	RESERVED	R	0h	Reserved

Table 30-23. SCIFLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
14	IDRXFLAG	R/W1C	0h	Identifier On Receive Flag. This bit is effective in LIN mode only. This flag is set once an identifier is received with an RX match and no ID-parity error. See the "Message Filtering and Validation" section for more details. When this flag is set it indicates that a new valid identifier has been received on an RX match. This bit is cleared by: - Reading the corresponding interrupt offset in the SCIINTVECT0/1 register - Setting the SWnRESET bit (SCIGCR1.7) - RESET bit (SCIGCR0.0) - System reset - Reading the LINID register - Writing a 1 to this bit This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = No valid ID received. 1h (R/W) = Valid ID RX received in LINID[23:16] on RX match.
13	IDTXFLAG	R/W1C	0h	Identifier On Transmit Flag. This bit is effective in LIN mode only. This flag is set once an identifier is received with a TX match and no ID-parity error. See the "Message Filtering and Validation" section for more details. When this flag is set it indicates that a new valid identifier has been received on a TX match. This bit is cleared by: - Reading the corresponding interrupt offset in the SCIINTVECT0/1 register - RESET bit (SCIGCR0.0) - Setting SWnRESET - System reset - Reading the LINID register - Writing a 1 to this bit This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = No valid ID received. 1h (R/W) = Valid ID received in LINID[23:16] on TX match.
12	RXWAKE	R	0h	Receiver wakeup detect flag. This bit is effective in SCI-compatible mode only. The SCI sets this bit to indicate that the data currently in SCIRD is an address. This bit is cleared by: - RESET bit - Setting the SWnRESET bit (SCIGCR1.7) - System reset - Receipt of a data frame This bit is writable in SCI mode only. Reset type: SYSRSn 0h (R/W) = The data in SCIRD is not an address. 1h (R/W) = The data in SCIRD is an address. See [1] Section 3.4.4, Sleep Mode for Multiprocessor Communication, on page 16 for more information on using the RXWAKE bit with sleep mode.

Table 30-23. SCIFLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11	TXEMPTY	R	1h	Transmitter Empty flag. The value of this flag indicates the contents of the transmitter's buffer register(s) (SCITD/TDy) and shift register (SCITXSHF). In multibuffer mode, this flag indicates the value of the TDx registers and shift register (SCITXSHF). In non multibuffer mode, this flag indicates the value of LINTD0 (byte) and shift register (SCITXSHF). This bit is set by: - RESET bit (SCIGCR0.0) - Setting the SWnRESET bit (SCIGCR1.7) - System reset. Note: This bit does not cause an interrupt request. Reset type: SYSRSn 0h (R/W) = Compatible mode or LIN with no multibuffer: Transmitter buffer or shift register (or both) are loaded with data. In LIN mode using multibuffer mode: Multibuffer or shift register (or all) are loaded with data. 1h (R/W) = Compatible mode or LIN with no multibuffer: Transmitter buffer and shift registers are both empty. In LIN mode using multibuffer mode: Multibuffer and shift registers are all empty.
10	TXWAKE	R/W	0h	SCI transmitter wakeup method select. This bit is effective in SCI-compatible mode only. The TXWAKE bit controls whether the data in SCITD should be sent as an address or data frame using multiprocessor communication format. This bit is set to 1 or 0 by software before a byte is written to SCITD and is cleared by the SCI when data is transferred from SCITD to SCITXSHF or by a system reset. TXWAKE is not cleared by the SWnRESET bit (SCIGCR1.7). Reset type: SYSRSn 0h (R/W) = Address-bit mode: Frame to be transmitted will be data (address bit = 0). Idle-line mode: Frame to be transmitted will be data. 1h (R/W) = Address-bit mode: Frame to be transmitted will be an address (address bit=1). Idle-line mode: Following frame to be transmitted will be an address (writing a 1 to this bit followed by writing dummy data to the SCITD will result in a idle period of 11 bit periods before the next frame is transmitted).
9	RXRDY	R/W1C	0h	Receiver ready flag. In SCI compatibility mode, the receiver sets this bit to indicate that the SCIRD contains new data and is ready to be read by the CPU. In LIN mode, RXRDY is set once a valid frame is received in multibuffer mode, a valid frame being a message frame received with no errors. In non multibuffer mode RXRDY is set for each received byte and will be set for the last byte of the frame if there are no errors. The SCI/LIN generates a receive interrupt when RXRDY flag bit is set if the interrupt-enable bit is set (SCISETINT.9). RXRDY is cleared by: - RESET bit (SCIGCR0.0) - Setting the SWnRESET - System reset - Writing a 1 to this bit - Reading SCIRD in while in SCI compatibility mode - Reading last data byte RDy of the response in LIN mode Note: The RXRDY flag cannot be cleared by reading the corresponding interrupt offset in the SCIINTVECT0/1 register. Reset type: SYSRSn 0h (R/W) = No new data in SCIRD/RDy. 1h (R/W) = New data ready to be read from SCIRD.

Table 30-23. SCIFLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	TXRDY	R	1h	Transmitter buffer register ready flag. When set, this bit indicates that the transmit buffer(s) register (SCITD in compatibility mode and LINTD0, LINTD1 in MBUF mode) is/are ready to get another character from a CPU write. In SCI compatibility mode, writing data to SCITD automatically clears this bit. In LIN mode, this bit is cleared once byte 0 (TD0) is written to LINTD0. This bit is set after the data of the TX buffer are shifted into the SCITXSHF register. This event can trigger a transmit DMA event if the DMA enable bit is set. This bit is set to 1 by: - RESET bit (SCIGCR0.0) - Setting the SWnRESET (SCIGCR1.7) - System reset Note: The TXRDY flag cannot be cleared by reading the corresponding interrupt offset in the SCIINTVECT0/1 register. Note: The transmit interrupt request can be eliminated until the next series of data is written into the transmit buffers LINTD0 and LINTD1, by disaLINg the corresponding interrupt via the SCICLEARINT register or by disaLINg the transmitter via the TXENA bit (SCIGCR1.25=0). Reset type: SYSRSn 0h (R/W) = Compatible mode: SCITD is full. LIN mode: The multibuffers are full. 1h (R/W) = Compatible mode: SCITD is ready to receive the next character. LIN mode: The multibuffers are ready to receive the next character(s).
7	TOA3WUS	R/W1C	0h	Timeout After 3 Wakeup Signals flag. This bit is effective in LIN mode only. This flag is set if there is no Sync Break received after 3 wakeup signals and a period of 1.5 seconds have passed. Such expiration time is used before issuing another round of wakeup signals. This bit is cleared by: - Reading the corresponding interrupt offset in the SCIINTVECT0/1 register - Setting the SWnRESET bit (SCIGCR1.7) - RESET bit (SCIGCR0.0) - System reset - Writing a 1 to this bit This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = No timeout after 3 wakeup signals. 1h (R/W) = Timeout after 3 wakeup signals and 1.5s time.
6	TOAWUS	R/W1C	0h	Timeout After Wakeup Signal flag. This bit is effective in LIN mode only. This bit is set if there is no Sync Break received after a wakeup signal has been sent. A minimum of 150 ms expiration time is used before issuing another wakeup signal. This bit is cleared by: - Reading the corresponding interrupt offset in the SCIINTVECT0/1 register - Setting the SWnRESET bit (SCIGCR1.7) - RESET bit (SCIGCR0.0) - System reset - Writing a 1 to this bit This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = No timeout after one wakeup signal (150 ms). 1h (R/W) = Timeout after one wakeup signal.
5	RESERVED	R	0h	Reserved

Table 30-23. SCIFLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	TIMEOUT	R/W1C	0h	LIN Bus IDLE timeout flag. This bit is effective in LIN mode only. This bit is set if there is no LIN bus activity for at least 4 seconds. LIN bus activity being a transition from recessive to dominant. This bit is cleared by: - Reading the corresponding interrupt offset in the SCIINTVECT0/1 register - Setting the SWnRESET bit (SCIGCR1.7) - RESET bit (SCIGCR0.0) - System reset - Writing a 1 to this bit This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = No bus idle detected. 1h (R/W) = LIN bus idle detected.
3	BUSY	R	0h	Bus BUSY flag. This bit is effective in LIN mode and SCI-compatible mode. This bit indicates whether the receiver is in the process of receiving a frame. As soon as the receiver detects the beginning of a start bit, the BUSY bit is set to 1. When the reception of a frame is complete, the BUSY bit is cleared. If SET WAKEUP INT is set and power down is requested while this bit is set, the SCI/LIN automatically prevents low-power mode from being entered and generates wakeup interrupt. The BUSY bit is controlled directly by the SCI receiver but can be cleared by: - Setting the SWnRESET bit (SCIGCR1.7) - RESET bit (SCIGCR0.0) - System reset. Reset type: SYSRSn 0h (R/W) = Receiver is not currently receiving a frame. 1h (R/W) = Receiver is currently receiving a frame.
2	IDLE	R	1h	SCI receiver in idle state. This bit is effective in SCI-compatible mode only. While this bit is set, the SCI looks for an idle period to resynchronize itself with the bit stream. The receiver does not receive any data while the bit is set. The bus must be idle for 11 bit periods to clear this bit. The SCI enters this state: - After a system reset - Setting the SWnRESET bit (SCIGCR1.7) - After coming out of power down This bit is writable in SCI mode only. Reset type: SYSRSn 0h (R/W) = Idle period detected, the SCI is ready to receive. 1h (R/W) = Idle period not detected, the SCI will not receive any data.
1	WAKEUP	R/W1C	0h	Wake-up flag. This bit is effective in LIN mode only. This bit is set by the SCI/LIN when receiver or transmitter activity has taken the module out of power-down mode. An interrupt is generated if the SET WAKEUP INT bit (SCISETINT.1) is set. This bit is cleared by: - Reading the corresponding interrupt offset in the SCIINTVECT0/1 register. - Setting the SWnRESET bit (SCIGCR1.7) - RESET bit (SCIGCR0.0) - System reset - Writing a 1 to this bit. This field is writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = Do not wake up from power-down mode. 1h (R/W) = Wake up from power-down mode.

Table 30-23. SCIFLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	BRKDT	R/W1C	0h	SCI break-detect flag. This bit is effective in SCI-compatible mode only. This bit is set when the SCI detects a break condition on the LINRX pin. A break condition occurs when the LINRX pin remains continuously low for at least 10 bits after a missing first stop bit, that is, after a framing error. Detection of a break condition causes the SCI to generate an error interrupt if the BRKDT INT ENA bit is set. The BRKDT bit is cleared by the following: - Reading the corresponding interrupt offset in the SCIINTVECT0/1 register. - Setting the SWnRESET bit (SCIGCR1.7) - RESET bit (SCIGCR0.0) - System reset - By writing a 1 to this bit. This bit is writable in SCI mode only. Reset type: SYSRSn 0h (R/W) = No break condition detected. 1h (R/W) = Break condition detected.

30.7.2.9 SCIINTVECT0 Register (Offset = 20h) [Reset = 0h]

SCIINTVECT0 is shown in Figure 30-34 and described in Table 30-24.

Return to the Summary Table.

The SCIINTVECT0 register indicates the offset for the INT0 interrupt line.

Figure 30-34. SCIINTVECT0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											INTVECT0				
R-0h											R-0h				

Table 30-24. SCIINTVECT0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-5	RESERVED	R	0h	Reserved
4-0	INTVECT0	R	0h	Interrupt vector offset for INT0. This register indicates the offset for interrupt line INT0. A read to this register updates its value to the next highest priority pending interrupt in SCIFLR and clears the flag corresponding to the offset that was read. Note: The flags for the receive (SCIFLR.9) and the transmit (SCIFLR.8) interrupts cannot be cleared by reading the corresponding offset vector in this register (see detailed description in SCIFLR register). Reset type: SYSRSn

30.7.2.10 SCIINTVECT1 Register (Offset = 24h) [Reset = 0h]

SCIINTVECT1 is shown in Figure 30-35 and described in Table 30-25.

Return to the Summary Table.

The SCIINTVECT1 register indicates the offset for the INT1 interrupt line.

Figure 30-35. SCIINTVECT1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED											INTVECT1				
R-0h											R-0h				

Table 30-25. SCIINTVECT1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-5	RESERVED	R	0h	Reserved
4-0	INTVECT1	R	0h	Interrupt vector offset for INT1. This register indicates the offset for interrupt line INT1. A read to this register updates its value to the next highest priority pending interrupt in SCIFLR and clears the flag corresponding to the offset that was read. Note: The flags for the receive (SCIFLR.9) and the transmit (SCIFLR.8) interrupts cannot be cleared by reading the corresponding offset vector in this register (see detailed description in SCIFLR register). Reset type: SYSRSn

30.7.2.11 SCIFORMAT Register (Offset = 28h) [Reset = 0h]

SCIFORMAT is shown in Figure 30-36 and described in Table 30-26.

Return to the Summary Table.

The SCIFORMAT register is used to set up the character and frame lengths.

Figure 30-36. SCIFORMAT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED													LENGTH		
R-0h													R/W-0h		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED													CHAR		
R-0h													R/W-0h		

Table 30-26. SCIFORMAT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Reserved
18-16	LENGTH	R/W	0h	Frame length control bits. In LIN mode, these bits indicate the number of bytes in the response field from 1 to 8 bytes. In buffered SCI mode, these bits indicate the number of characters. When these bits are used to indicate LIN response length (SCIGCR1[0] = 1), then when there is an ID RX match, this value should be updated with the expected length of the response. In buffered SCI mode, these bits indicate the number of characters with SCIFORMAT[2:0] bits per character. i.e. these bits indicate the transmitter/receiver format for the number of characters: 1 to 8. There can be up to eight characters with eight bits each. Reset type: SYSRSn 0h (R/W) = The response field has 1 bytes/characters. 1h (R/W) = The response field has 2 bytes/characters. 2h (R/W) = The response field has 3 bytes/characters. 3h (R/W) = The response field has 4 bytes/characters. 4h (R/W) = The response field has 5 bytes/characters. 5h (R/W) = The response field has 6 bytes/characters. 6h (R/W) = The response field has 7 bytes/characters. 7h (R/W) = The response field has 8 bytes/characters.
15-3	RESERVED	R	0h	Reserved
2-0	CHAR	R/W	0h	Character length control bits. These bits are effective in SCI compatible mode only. These bits set the SCI character length from 1 to 8 bits. Note: In compatibility mode or buffered SCI mode, when data of fewer than eight bits in length is received, it is left justified in SCIRD/RDy and padded with trailing zeros. Data read from the SCIRD should be shifted by software to make the received data right justified. Note: Data written to the SCITD should be right justified but does not need to be padded with leading zeros. These bits are witable in SCI mode only. Reset type: SYSRSn 0h (R/W) = The character is 1 bits long. 1h (R/W) = The character is 2 bits long. 2h (R/W) = The character is 3 bits long. 3h (R/W) = The character is 4 bits long. 4h (R/W) = The character is 5 bits long. 5h (R/W) = The character is 6 bits long. 6h (R/W) = The character is 7 bits long. 7h (R/W) = The character is 8 bits long.

30.7.2.12 BRSR Register (Offset = 2Ch) [Reset = 0h]

BRSR is shown in Figure 30-37 and described in Table 30-27.

Return to the Summary Table.

The BRSR register is used to configure the baud rate of the LIN module.

Figure 30-37. BRSR Register

31	30	29	28	27	26	25	24
RESERVED	U			M			
R-0h	R/W-0h			R/W-0h			
23	22	21	20	19	18	17	16
SCI_LIN_PSH							
R/W-0h							
15	14	13	12	11	10	9	8
SCI_LIN_PSL							
R/W-0h							
7	6	5	4	3	2	1	0
SCI_LIN_PSL							
R/W-0h							

Table 30-27. BRSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved
30-28	U	R/W	0h	Superfractional Divider Selection. (U) These bits are an additional fractional part for the baudrate specification. These bits allow a super fine tuning of the fractional baudrate with 7 more intermediate values for each of the M fractional divider values. See the Superfractional Divider section for more details. Reset type: SYSRSn
27-24	M	R/W	0h	SCI/LIN 4-bit Fractional Divider Selection. (M) These bits are effective in LIN or SCI asynchronous mode. These bits are used to select a baud rate for the SCI/LIN module, and they are a fractional part for the baud rate specification. The M divider allows fine-tuning of the baud rate over the P prescaler with 15 additional intermediate values for each of the P integer values. Reset type: SYSRSn
23-16	SCI_LIN_PSH	R/W	0h	PRESCALER P (High Bits). SCI/LIN 24-bit Integer Prescaler Selection. These bits are used to select a baudrate for the SCI/LIN module. These bits are effective in LIN mode and SCI compatible mode. The SCI/LIN has an internally generated serial clock determined by the LIN module input clock and the prescalers P and M in this register. The SCI/LIN uses the 24-bit integer prescaler P value to select 1 of over 16,700,000 available baud rates. The additional 4-bit fractional prescaler M refines the baudate selection. Reset type: SYSRSn

Table 30-27. BRSR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
15-0	SCI_LIN_PSL	R/W	0h	PRESCALER P (Low Bits). SCI/LIN 24-bit Integer Prescaler Selection. These bits are used to select a baudrate for the SCI/LIN module. These bits are effective in LIN mode and SCI compatible mode. The SCI/LIN has an internally generated serial clock determined by the LIN module input clock and the prescalers P and M in this register. The SCI/LIN uses the 24-bit integer prescaler P value to select 1 of over 16,700,000 available baud rates. The additional 4-bit fractional prescaler M refines the baudate selection. Reset type: SYSRSn

30.7.2.13 SCIED Register (Offset = 30h) [Reset = 0h]

SCIED is shown in Figure 30-38 and described in Table 30-28.

Return to the Summary Table.

The SCIED register is a duplicate copy of SCIRD register that has no affect on the RXRDY flag for use with an emulator.

Figure 30-38. SCIED Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								ED							
R-0h																								R-0h							

Table 30-28. SCIED Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	ED	R	0h	Receiver Emulation Data. This bit is effective in SCI-compatible mode only. Reading SCIED(7-0) does not clear the RXRDY flag. This register should be used only by an emulator that must continually read the data buffer without affecting the RXRDY flag. Reset type: SYSRSn

30.7.2.14 SCIRD Register (Offset = 34h) [Reset = 0h]

SCIRD is shown in Figure 30-39 and described in Table 30-29.

Return to the Summary Table.

The SCIRD register is where received data is stored and can be read from.

Figure 30-39. SCIRD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								RD							
R-0h																								R-0h							

Table 30-29. SCIRD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	RD	R	0h	Received Data. This bit is effective in SCI-compatible mode only. When a frame has been completely received, the data in the frame is transferred from the receiver shift register SCIRXSHF to this register. As this transfer occurs, the RXRDY flag is set and a receive interrupt is generated if RX INT ENA (SCISETINT0.9) is set. When the data is read from SCIRD, the RXRDY flag is automatically cleared. When the SCI receives data that is fewer than eight bits in length, it loads the data into this register in a left justified format padded with trailing zeros. Therefore, your software should perform a logical shift on the data by the correct number of positions to make it right justified. Reset type: SYSRSn

30.7.2.15 SCITD Register (Offset = 38h) [Reset = 0h]

SCITD is shown in Figure 30-40 and described in Table 30-30.

Return to the Summary Table.

The SCITD register is where data to be transmitted is written to by application software.

Figure 30-40. SCITD Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																								TD							
R-0h																								R/W-0h							

Table 30-30. SCITD Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7-0	TD	R/W	0h	Transmit data This bit is effective in SCI-compatible mode only. Data to be transmitted is written to this register. The transfer of data from this register to the transmit shift register SCITXSHF sets the TXRDY flag (SCIFLR.23), which indicates that SCITD is ready to be loaded with another byte of data. Note: If TX INT ENA (SCISETINT.8) is set, this data transfer also causes an interrupt. Note: Data written to the SCIRD register that is fewer than eight bits long must be right justified, but it does not need to be padded with leading zeros. Reset type: SYSRSn

30.7.2.16 SCIPIO0 Register (Offset = 3Ch) [Reset = 0h]

SCIPIO0 is shown in Figure 30-41 and described in Table 30-31.

Return to the Summary Table.

The SCIPIO0 register is used to enable the LINTX and LINRX pins.

Figure 30-41. SCIPIO0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					TXFUNC	RXFUNC	RESERVED
R-0h					R/W-0h	R/W-0h	R-0h

Table 30-31. SCIPIO0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-3	RESERVED	R	0h	Reserved
2	TXFUNC	R/W	0h	Transmit pin function. This bit is effective in LIN or SCI mode. This bit defines the function of LINTX pin. Reset type: SYSRSn 0h (R/W) = LINTX pin is disabled. 1h (R/W) = LINTX pin is enabled.
1	RXFUNC	R/W	0h	Receive pin function. This bit is effective in LIN or SCI mode. This bit defines the function of the LINRX pin. Reset type: SYSRSn 0h (R/W) = LINRX pin is disabled. 1h (R/W) = LINRX pin is enabled.
0	RESERVED	R	0h	Reserved

30.7.2.17 SCIPIO2 Register (Offset = 44h) [Reset = 0h]

SCIPIO2 is shown in Figure 30-42 and described in Table 30-32.

Return to the Summary Table.

The SCIPIO2 register indicates the current status of the LINTX and LINRX pins.

Figure 30-42. SCIPIO2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED					TXIN	RXIN	RESERVED
R-0h					R-0h	R-0h	R-0h

Table 30-32. SCIPIO2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-3	RESERVED	R	0h	Reserved
2	TXIN	R	0h	Transmit data in. This bit is effective in LIN or SCI-compatible mode. This bit contains the current value on the LINTX pin. Reset type: SYSRSn
1	RXIN	R	0h	Receive data in. This bit is effective in LIN or SCI-compatible mode. This bit contains the current value on the LINRX pin. Reset type: SYSRSn
0	RESERVED	R	0h	Reserved

30.7.2.18 LINCOMP Register (Offset = 60h) [Reset = 0h]

LINCOMP is shown in Figure 30-43 and described in Table 30-33.

Return to the Summary Table.

The LINCOMPARE register is used to configure the sync delimeter and sync break extension.

Figure 30-43. LINCOMP Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED															
R-0h															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED						SDEL		RESERVED					SBREAK		
R-0h						R/W-0h		R-0h					R/W-0h		

Table 30-33. LINCOMP Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-10	RESERVED	R	0h	Reserved
9-8	SDEL	R/W	0h	2-bit Sync Delimiter compare. These bits are effective in LIN mode only. These bits are used to configure the number of Tbit for the sync delimiter in the sync field. The time delay calculation for the synchronization delimiter is: TSDEL = (SDEL + 1)Tbit These bits are writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = The sync delimiter has 1 Tbit. 1h (R/W) = The sync delimiter has 2 Tbit. 2h (R/W) = The sync delimiter has 3 Tbit. 3h (R/W) = The sync delimiter has 4 Tbit.
7-3	RESERVED	R	0h	Reserved
2-0	SBREAK	R/W	0h	3-bit Sync Break extend. LIN mode only. These bits are used to configure the number of Tbits for the sync break to extend the minimum 13 Tbit in the Sync Field to a maximum of 20 Tbit. The time delay calculation for the sync break is: TSYNBRK = 13Tbit + SBREAK x Tbit These bits are writable in LIN mode only. Reset type: SYSRSn 0h (R/W) = The sync break has no additional Tbit. 1h (R/W) = The sync break has 1 additional Tbit. 2h (R/W) = The sync break has 2 additional Tbit. 3h (R/W) = The sync break has 3 additional Tbit. 4h (R/W) = The sync break has 4 additional Tbit. 5h (R/W) = The sync break has 5 additional Tbit. 6h (R/W) = The sync break has 6 additional Tbit. 7h (R/W) = The sync break has 7 additional Tbit.

30.7.2.19 LINRD0 Register (Offset = 64h) [Reset = 0h]

LINRD0 is shown in Figure 30-44 and described in Table 30-34.

Return to the Summary Table.

The LINRD0 register contains the lower 4 bytes of the received LIN frame data.

Figure 30-44. LINRD0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD0								RD1								RD2								RD3							
R-0h								R-0h								R-0h								R-0h							

Table 30-34. LINRD0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RD0	R	0h	8-bit Receive Buffer 0 Each response data-byte that is received in the SCIRXSHFT register is transferred to the corresponding RDy register according to the number of bytes received. A read of this byte clears the RXDY byte. Note: RD<x-1> is equivalent to Data byte <x> of the LIN frame. Reset type: SYSRSn
23-16	RD1	R	0h	8-bit Receive Buffer 1. Each response data-byte that is received in the SCIRXSHFT register is transferred to the corresponding RDy register according to the number of bytes received. Reset type: SYSRSn
15-8	RD2	R	0h	8-bit Receive Buffer 2. Each response data-byte that is received in the SCIRXSHFT register is transferred to the corresponding RDy register according to the number of bytes received. Reset type: SYSRSn
7-0	RD3	R	0h	8-bit Receive Buffer 3. Each response data-byte that is received in the SCIRXSHFT register is transferred to the corresponding RDy register according to the number of bytes received. Reset type: SYSRSn

30.7.2.20 LINRD1 Register (Offset = 68h) [Reset = 0h]

LINRD1 is shown in Figure 30-45 and described in Table 30-35.

Return to the Summary Table.

The LINRD1 regsiter contains the upper 4 bytes of the received LIN frame data.

Figure 30-45. LINRD1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RD4								RD5								RD6								RD7							
R-0h								R-0h								R-0h								R-0h							

Table 30-35. LINRD1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RD4	R	0h	8-bit Receive Buffer 4 Each response data-byte that is received in the SCIRXSHFT register is transferred to the corresponding RDy register according to the number of bytes received. Reset type: SYSRSn
23-16	RD5	R	0h	8-bit Receive Buffer 5. Each response data-byte that is received in the SCIRXSHFT register is transferred to the corresponding RDy register according to the number of bytes received. Reset type: SYSRSn
15-8	RD6	R	0h	8-bit Receive Buffer 6. Each response data-byte that is received in the SCIRXSHFT register is transferred to the corresponding RDy register according to the number of bytes received. Reset type: SYSRSn
7-0	RD7	R	0h	8-bit Receive Buffer 7. Each response data-byte that is received in the SCIRXSHFT register is transferred to the corresponding RDy register according to the number of bytes received. Reset type: SYSRSn

30.7.2.21 LINMASK Register (Offset = 6Ch) [Reset = 0h]

LINMASK is shown in Figure 30-46 and described in Table 30-36.

Return to the Summary Table.

The LINMASK register is used to configure the masks used for filtering incoming ID messages for receive and transmit frames.

Figure 30-46. LINMASK Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								RXIDMASK								RESERVED								TXIDMASK							
R-0h								R/W-0h								R-0h								R/W-0h							

Table 30-36. LINMASK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	RXIDMASK	R/W	0h	Receive ID mask. This field is effective in LIN mode only.This 8-bit mask is used for filtering an incoming ID message and compare it to the ID-byte. A compare match of the received ID witht the RX ID mask will set the ID RX flag and trigger and ID interrupt if enabled. A 0 bit in the mask indicates that bit is compared to the ID-byte. A 1 bit in the mask indicates that that bit is filtered and therefore not used in the compare. When HGENCTRL is set to 1, this field must be set to 0xFF. Reset type: SYSRSn
15-8	RESERVED	R	0h	Reserved
7-0	TXIDMASK	R/W	0h	Transmit ID mask. This field is effective in LIN mode only. This 8-bit mask is used for filtering an incoming ID message and compare it to the ID-byte. A compare match of the received ID with the TX ID Mask will set the ID TX flag and trigger an ID interrupt if enabled. A 0 bit in the mask indicates that bit is compared to the ID-byte. A 1 bit in the mask indicates that bit is filtered and therefore not used for the compare. When HGENCTRL is set to 1, this field must be set to 0xFF. Reset type: SYSRSn

30.7.2.22 LINID Register (Offset = 70h) [Reset = 0h]

LINID is shown in Figure 30-47 and described in Table 30-37.

Return to the Summary Table.

The LINID register contains the identification fields for LIN communication.
NOTE: For software compatibility with future LIN modules, the HGEN CTRL bit must be set to 1, the RX ID MASK field must be set to FFh, and the TX ID MASK field must be set to FFh.

Figure 30-47. LINID Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED								RECEIVEDID							
R-0h								R-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IDSLAVETASKBYTE								IDBYTE							
R/W-0h								R/W-0h							

Table 30-37. LINID Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	RECEIVEDID	R	0h	Received ID. This bit is effective in LIN mode only. This byte contains the current message identifier. During header reception the received ID is copied from the SCIRXSHF register to this byte if there is no ID-parity error and there has been an RX/TX match. Note: If a framing error (FE) is detected during ID reception, the received ID will also not be copied to the LINID register. Reset type: SYSRSn
15-8	IDSLAVETASKBYTE	R/W	0h	ID Slave Task byte. This field is effective in LIN mode only. This byte contains the identifier to which the received ID of an incoming header will be compared in order to decide whether a RX response, a TX response, or no action needs to be done by the LIN node. These bits are writable in LIN mode only. Reset type: SYSRSn
7-0	IDBYTE	R/W	0h	ID byte. This field is effective in LIN mode only. This byte is the LIN mode message ID. On a Master node, a write to this register by the CPU initiates a header transmission. For a Slave task, this byte is used for message filtering when HGENCTRL (SCIGCR1.12) is '0'. These bits are writable in LIN mode only. Reset type: SYSRSn

30.7.2.23 LINTD0 Register (Offset = 74h) [Reset = 0h]

LINTD0 is shown in Figure 30-48 and described in Table 30-38.

Return to the Summary Table.

The LINTD0 register contains the lower 4 bytes of the data to be transmitted.
NOTE: TD<x-1> is equivalent to Data byte <x> of the LIN frame.

Figure 30-48. LINTD0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TD0								TD1								TD2								TD3							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 30-38. LINTD0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	TD0	R/W	0h	8-bit Transmit Buffer 0. Byte 0 to be transmitted is written into this register and then copied to SCITXSHF for transmission. Once byte 0 is written in TDO buffer, transmission will be initiated. Reset type: SYSRSn
23-16	TD1	R/W	0h	8-bit Transmit Buffer 3. Byte 1 to be transmitted is written into this register and then copied to SCITXSHF for transmission. Reset type: SYSRSn
15-8	TD2	R/W	0h	8-bit Transmit Buffer 2. Byte 2 to be transmitted is written into this register and then copied to SCITXSHF for transmission. Reset type: SYSRSn
7-0	TD3	R/W	0h	8-bit Transmit Buffer 3. Byte 3 to be transmitted is written into this register and then copied to SCITXSHF for transmission. Reset type: SYSRSn

30.7.2.24 LINTD1 Register (Offset = 78h) [Reset = 0h]

LINTD1 is shown in Figure 30-49 and described in Table 30-39.

Return to the Summary Table.

The LINTD1 register contains the upper 4 bytes of the data to be transmitted.
NOTE: TD<x-1> is equivalent to Data byte <x> of the LIN frame.

Figure 30-49. LINTD1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TD4								TD5								TD6								TD7							
R/W-0h								R/W-0h								R/W-0h								R/W-0h							

Table 30-39. LINTD1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	TD4	R/W	0h	8-bit Transmit Buffer 4. Byte4 to be transmitted is written into this register and then copied to SCITXSHF for transmission. Reset type: SYSRSn
23-16	TD5	R/W	0h	8-bit Transmit Buffer 5. Byte 5 to be transmitted is written into this register and then copied to SCITXSHF for transmission. Reset type: SYSRSn
15-8	TD6	R/W	0h	8-bit Transmit Buffer 6. Byte 6 to be transmitted is written into this register and then copied to SCITXSHF for transmission. Reset type: SYSRSn
7-0	TD7	R/W	0h	8-bit Transmit Buffer 7. Byte 7 to be transmitted is written into this register and then copied to SCITXSHF for transmission. Reset type: SYSRSn

30.7.2.25 MBRSR Register (Offset = 7Ch) [Reset = DACh]

MBRSR is shown in Figure 30-50 and described in Table 30-40.

Return to the Summary Table.

The MBRSR register is used to configure the expected maximum baud rate of the LIN network.

Figure 30-50. MBRSR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																			MBR												
R-0h																			R/W-DACh											

Table 30-40. MBRSR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12-0	MBR	R/W	DACh	Maximum Baud Rate Prescaler. This field is effective in LIN mode only. This 13-bit prescaler is used during the synchronization phase (see the "Header Reception and Adaptive Baudrate" section) of a Slave module if the ADAPT bit is set. In this way, a SCI/LIN Slave using an automatic or select bit rate modes detects any LIN bus legal rate automatically. The MBR value should be programmed to allow a maximum baud rate that is not more than 10% above the expected operating baud rate in the LIN network. Otherwise a s 0x00 data byte could mistakenly be detected as sync break. The default value is for a 70MHz LINCLK (0xDAC). This MBR prescaler is used by the wake-up and idle time counters for a constant expiration time relative to a 20kHz rate. Reset type: SYSRSn

30.7.2.26 IODFTCTRL Register (Offset = 90h) [Reset = 500h]

IODFTCTRL is shown in Figure 30-51 and described in Table 30-41.

Return to the Summary Table.

The IODFTCTRL register is used to emulate various error and test conditions.

Figure 30-51. IODFTCTRL Register

31	30	29	28	27	26	25	24
BERRENA	PBERRENA	CERRENA	ISFERRENA	RESERVED	FERRENA	PERRENA	BRKDTERRENA
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
RESERVED			PINSAMPLEMASK		TXSHIFT		
R-0h			R/W-0h		R/W-0h		
15	14	13	12	11	10	9	8
RESERVED				IODFTENA			
R-0h				R/W-5h			
7	6	5	4	3	2	1	0
RESERVED						LPBENA	RXPENA
R-0h						R/W-0h	R/W-0h

Table 30-41. IODFTCTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	BERRENA	R/W	0h	Bit Errror Enable bit. This bit is effective in LIN mode only. This bit is used to create a Bit error. When this bit is set, the bit received is ORed with 1 and passed to the Bit monitor circuitry. Reset type: SYSRSn
30	PBERRENA	R/W	0h	Physical Bus Error Enable bit. This bit is effective in LIN mode only. This bit is used to create a Physical Bus Error. When this bit is set, the bit received during Sync Break field transmission is ORed with 1 and passed to the Bit monitor circuitry Reset type: SYSRSn
29	CERRENA	R/W	0h	Checksum Error Enable bit. This bit is effective in LIN mode only. This bit is used to create a checksum error. When this bit is set, the polarity of the CTYPE (checksum type) in the receive checksum calculator is changed so that a checksum error is generated. Reset type: SYSRSn
28	ISFERRENA	R/W	0h	Inconsistent Sync Field Error Enable bit. This bit is effective in LIN mode only. This bit is used to create an ISF error. When this bit is set, the bit widths in the sync field are varied so that the ISF check fails and the error flag is set. Reset type: SYSRSn
27	RESERVED	R	0h	Reserved
26	FERRENA	R/W	0h	This bit is used to create a Frame Error. This bit is effective in SCI-compatible mode only. When this bit is set, the stop bit received is ANDed with '0' and passed to the stop bit check circuitry. Reset type: SYSRSn
25	PERRENA	R/W	0h	Compatible Mode only This bit is effective in SCI-compatible mode only. This bit is used to create a Parity Error. When this bit is set, in compatible mode, the parity bit received is toggled so that a parity error occurs. Reset type: SYSRSn

Table 30-41. IODFTCTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
24	BRKDTERRENA	R/W	0h	Compatible Mode only This bit is effective in SCI-compatible mode only. This bit is used to create BRKDT error (SCI mode only). When this bit is set, the stop bit of the frame is ANDed with '0' and passed to the RSM so that a frame error occurs. Then the RX Pin is forced to continuous low for 10 Tbits so that a BRKDT error occurs. Reset type: SYSRSn
23-21	RESERVED	R	0h	Reserved
20-19	PINSAMPLEMASK	R/W	0h	Pin sample mask. These bits define the sample number at which the TX Pin value that is being transmitted will be inverted to verify the receive pin samples correctly with the majority detection circuitry. Note: During IODFT mode testing for the pin sample mask, the prescalar P must be programmed to be greater than 2. Reset type: SYSRSn 0h (R/W) = No Mask 1h (R/W) = Invert the TX Pin value at TBIT_CENTER 2h (R/W) = Invert the TX Pin value at TBIT_CENTER + SCLK 3h (R/W) = Invert the TX Pin value at TBIT_CENTER + 2 SCLK
18-16	TXSHIFT	R/W	0h	Transmit shift. These bits define the delay by which the value on LINTX is delayed so that the value on LINRX is asynchronous. (Not applicable to Start Bit) Reset type: SYSRSn 0h (R/W) = No Delay 1h (R/W) = Delay by 1 SCLK 2h (R/W) = Delay by 2 SCLK 3h (R/W) = Delay by 3 SCLK 4h (R/W) = Delay by 4 SCLK 5h (R/W) = Delay by 5 SCLK 6h (R/W) = Delay by 6 SCLK 7h (R/W) = Delay by 7 SCLK
15-12	RESERVED	R	0h	Reserved
11-8	IODFTENA	R/W	5h	IO DFT Enable Key This field is used to enable the IODFT mode of the SCI/LIN module for testing. Reset type: SYSRSn 0h (R/W) = IODFT is disabled 1h (R/W) = IODFT is disabled 2h (R/W) = IODFT is disabled 3h (R/W) = IODFT is disabled 4h (R/W) = IODFT is disabled 5h (R/W) = IODFT is disabled 6h (R/W) = IODFT is disabled 7h (R/W) = IODFT is disabled 8h (R/W) = IODFT is disabled 9h (R/W) = IODFT is disabled Ah (R/W) = IODFT is enabled Bh (R/W) = IODFT is disabled Ch (R/W) = IODFT is disabled Dh (R/W) = IODFT is disabled Eh (R/W) = IODFT is disabled Fh (R/W) = IODFT is disabled
7-2	RESERVED	R	0h	Reserved
1	LPBENA	R/W	0h	Module loopback enable. In analog loopback mode the complete communication path through the I/Os can be tested, whereas in digital loopback mode the I/O buffers are excluded from this path. Reset type: SYSRSn 0h (R/W) = Digital loopback is enabled. 1h (R/W) = Analog loopback is enabled in module I/O DFT mode (when IODFTENA = 1010)

Table 30-41. IODFTCTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	RXPENA	R/W	0h	Module Analog loopback through receive pin enable. This bit defines whether the I/O buffers for the transmit or the receive pin are included in the communication path in analog loopback mode only. Reset type: SYSRSn 0h (R/W) = Analog loopback through the transmit pin is enabled. 1h (R/W) = Analog loopback through the receive pin is enabled.

30.7.2.27 LIN_GLB_INT_EN Register (Offset = E0h) [Reset = 0h]

LIN_GLB_INT_EN is shown in Figure 30-52 and described in Table 30-42.

Return to the Summary Table.

The LIN_GLB_INT_EN register is used to enable the INT0 and INT1 interrupt lines to propagate to the PIE block.

Figure 30-52. LIN_GLB_INT_EN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						GLBINT1_EN	GLBINT0_EN
R-0h						R/W-0h	R/W-0h

Table 30-42. LIN_GLB_INT_EN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	GLBINT1_EN	R/W	0h	Global Interrupt Enable for LIN INT1. This bit determines whether the INT1 interrupt line generates an interrupt to the PIE or not Reset type: SYSRSn 0h (R/W) = LIN INT1 line does not generate an interrupt to the PIE. 1h (R/W) = LIN INT1 line generates an interrupt to the PIE if an enabled interrupt condition occurs.
0	GLBINT0_EN	R/W	0h	Global Interrupt Enable for LIN INT0. This bit determines whether the INT0 interrupt line generates an interrupt to the PIE or not. Reset type: SYSRSn 0h (R/W) = LIN INT0 line does not generate an interrupt to the PIE. 1h (R/W) = LIN INT0 line generates an interrupt to the PIE if an enabled interrupt condition occurs.

30.7.2.28 LIN_GLB_INT_FLG Register (Offset = E4h) [Reset = 0h]

LIN_GLB_INT_FLG is shown in Figure 30-53 and described in Table 30-43.

Return to the Summary Table.

The LIN_GLB_INT_FLG register contains the current status of the INT0 and INT1 flags.

Figure 30-53. LIN_GLB_INT_FLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						INT1_FLG	INT0_FLG
R-0h						R-0h	R-0h

Table 30-43. LIN_GLB_INT_FLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	INT1_FLG	R	0h	Global Interrupt Flag for LIN INT1. This bit indicates if an interrupt was generated to the PIE due to an enabled interrupt on the INT1 interrupt line. Refer to the LIN Interrupt Status Register for the condition that generated the interrupt. This bit can be cleared by writing a 1 to the corresponding bit in the LIN_GLB_INT_CLR register. Reset type: SYSRSn 0h (R/W) = No interrupt is active on the INT1 line. 1h (R/W) = An interrupt was generated due to an enabled interrupt on the INT1 interrupt line.
0	INT0_FLG	R	0h	Global Interrupt Flag for LIN INT0. This bit indicates if an interrupt was generated to the PIE due to an enabled interrupt on the INT0 interrupt line. Refer to the LIN Interrupt Status Register for the condition that generated the interrupt. This bit can be cleared by writing a 1 to the corresponding bit in the LIN_GLB_INT_CLR register. Reset type: SYSRSn 0h (R/W) = No interrupt is active on the INT0 line. 1h (R/W) = An interrupt was generated due to an enabled interrupt on the INT0 interrupt line.

30.7.2.29 LIN_GLB_INT_CLR Register (Offset = E8h) [Reset = 0h]

LIN_GLB_INT_CLR is shown in Figure 30-54 and described in Table 30-44.

Return to the Summary Table.

The LIN_GLB_INT_CLR register is used to clear the interrupt flags in LIN_GLB_INT_FLG register.

Figure 30-54. LIN_GLB_INT_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						INT1_FLG_CLR	INT0_FLG_CLR
R-0h						R/W1C-0h	R/W1C-0h

Table 30-44. LIN_GLB_INT_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	INT1_FLG_CLR	R/W1C	0h	Global Interrupt flag clear for LIN INT1. This bit is used to clear the corresponding bit in the LIN_GLB_INT_FLG register. Write 1 to clear the INT1_FLG bit. Writing 0 has no effect. Reset type: SYSRSn
0	INT0_FLG_CLR	R/W1C	0h	Global Interrupt flag clear for LIN INT0. This bit is used to clear the corresponding bit in the LIN_GLB_INT_FLG register. Write 1 to clear the INT0_FLG bit. Writing 0 has no effect. Reset type: SYSRSn

30.7.3 LIN Registers to Driverlib Functions

Table 30-45. LIN Registers to Driverlib Functions

File	Driverlib Function
SCIGCR0	
lin.h	LIN_enableModule
lin.h	LIN_disableModule
SCIGCR1	
lin.h	LIN_setLINMode
lin.h	LIN_setMessageFiltering
lin.h	LIN_enableParity
lin.h	LIN_disableParity
lin.h	LIN_setCommMode
lin.h	LIN_enableAutomaticBaudrate
lin.h	LIN_disableAutomaticBaudrate
lin.h	LIN_stopExtendedFrame
lin.h	LIN_setChecksumType

Table 30-45. LIN Registers to Driverlib Functions (continued)

File	Driverlib Function
lin.h	LIN_enableSCIMode
lin.h	LIN_disableSCIMode
lin.h	LIN_setSCICommMode
lin.h	LIN_enableSCIParity
lin.h	LIN_disableSCIParity
lin.h	LIN_setSCIStopBits
lin.h	LIN_enableSCISleepMode
lin.h	LIN_disableSCISleepMode
lin.h	LIN_enterSCILowPower
lin.h	LIN_exitSCILowPower
lin.h	LIN_setSCICharLength
lin.h	LIN_setSCIFrameLength
lin.h	LIN_isSCIDataAvailable
lin.h	LIN_isSCISpaceAvailable
lin.h	LIN_readSCICharNonBlocking
lin.h	LIN_readSCICharBlocking
lin.h	LIN_writeSCICharNonBlocking
lin.h	LIN_writeSCICharBlocking
lin.h	LIN_enableSCIModuleErrors
lin.h	LIN_disableSCIModuleErrors
lin.h	LIN_enableSCIInterrupt
lin.h	LIN_disableSCIInterrupt
lin.h	LIN_clearSCIInterruptStatus
lin.h	LIN_setSCIInterruptLevel0
lin.h	LIN_setSCIInterruptLevel1
lin.h	LIN_isSCIReceiverIdle
lin.h	LIN_getSCITxFrameType
lin.h	LIN_getSCIRxFrameType
lin.h	LIN_isSCIBreakDetected
lin.h	LIN_enableDataTransmitter
lin.h	LIN_disableDataTransmitter
lin.h	LIN_enableDataReceiver
lin.h	LIN_disableDataReceiver
lin.h	LIN_performSoftwareReset
lin.h	LIN_enterSoftwareReset
lin.h	LIN_exitSoftwareReset
lin.h	LIN_enableIntLoopback
lin.h	LIN_disableIntLoopback
lin.h	LIN_enableMultibufferMode
lin.h	LIN_disableMultibufferMode
lin.h	LIN_setDebugSuspendMode
SCIGCR2	
lin.h	LIN_sendWakeupSignal
lin.h	LIN_enterSleep
lin.h	LIN_sendChecksum

Table 30-45. LIN Registers to Driverlib Functions (continued)

File	Driverlib Function
lin.h	LIN_triggerChecksumCompare
lin.h	LIN_enterSCILowPower
lin.h	LIN_exitSCILowPower
SCISETINT	
lin.h	LIN_enableInterrupt
lin.h	LIN_setInterruptLevel1
lin.h	LIN_enableSCIInterrupt
lin.h	LIN_setSCIInterruptLevel1
lin.h	LIN_getInterruptLevel
SCICLEARINT	
lin.h	LIN_disableInterrupt
lin.h	LIN_setInterruptLevel0
lin.h	LIN_disableSCIInterrupt
lin.h	LIN_setSCIInterruptLevel0
SCISETINTLVL	
lin.h	LIN_setInterruptLevel1
lin.h	LIN_setSCIInterruptLevel1
lin.h	LIN_getInterruptLevel
SCICLEARINTLVL	
lin.h	LIN_setInterruptLevel0
lin.h	LIN_setSCIInterruptLevel0
SCIFLR	
lin.h	LIN_isTxReady
lin.h	LIN_isRxReady
lin.h	LIN_isTxMatch
lin.h	LIN_isRxMatch
lin.h	LIN_clearInterruptStatus
lin.h	LIN_isSCIDataAvailable
lin.h	LIN_isSCISpaceAvailable
lin.h	LIN_clearSCIInterruptStatus
lin.h	LIN_isSCIReceiverIdle
lin.h	LIN_getSCITxFrameType
lin.h	LIN_getSCIRxFrameType
lin.h	LIN_isSCIBreakDetected
lin.h	LIN_isBusBusy
lin.h	LIN_isTxBufferEmpty
lin.h	LIN_getInterruptStatus
SCIINTVECT0	
lin.h	LIN_getInterruptLine0Offset
SCIINTVECT1	
lin.h	LIN_getInterruptLine1Offset
SCIFORMAT	
lin.c	LIN_sendData
lin.c	LIN_getData
lin.h	LIN_setFrameLength

Table 30-45. LIN Registers to Driverlib Functions (continued)

File	Driverlib Function
lin.h	LIN_setSCICharLength
lin.h	LIN_setSCIFrameLength
BRSR	
lin.h	LIN_setBaudRatePrescaler
SCIED	
lin.h	LIN_readSCICharNonBlocking
lin.h	LIN_readSCICharBlocking
SCIRD	
lin.h	LIN_readSCICharNonBlocking
lin.h	LIN_readSCICharBlocking
SCITD	
lin.h	LIN_writeSCICharNonBlocking
lin.h	LIN_writeSCICharBlocking
SCIPIO0	
lin.h	LIN_enableModule
lin.h	LIN_disableModule
SCIPIO2	
lin.h	LIN_getPinStatus
COMP	
lin.h	LIN_setSyncFields
RD0	
lin.c	LIN_getData
RD1	
-	
MASK	
lin.h	LIN_setTxMask
lin.h	LIN_setRxMask
lin.h	LIN_getTxMask
lin.h	LIN_getRxMask
ID	
lin.h	LIN_setIDByte
lin.h	LIN_setIDSlaveTask
lin.h	LIN_getRxIdentifier
TD0	
lin.c	LIN_sendData
lin.h	LIN_sendWakeupSignal
TD1	
-	
MBRSR	
lin.h	LIN_setMaximumBaudRate
IODFTCTRL	
lin.h	LIN_enableModuleErrors
lin.h	LIN_disableModuleErrors
lin.h	LIN_enableSCIModuleErrors
lin.h	LIN_disableSCIModuleErrors

Table 30-45. LIN Registers to Driverlib Functions (continued)

File	Driverlib Function
lin.h	LIN_enableExtLoopback
lin.h	LIN_disableExtLoopback
lin.h	LIN_setTransmitDelay
lin.h	LIN_setPinSampleMask
GLB_INT_EN	
lin.h	LIN_enableGlobalInterrupt
lin.h	LIN_disableGlobalInterrupt
GLB_INT_FLG	
lin.h	LIN_getGlobalInterruptStatus
GLB_INT_CLR	
lin.h	LIN_clearGlobalInterruptStatus

This page intentionally left blank.

Chapter 31

Fast Serial Interface (FSI)

This chapter contains a general description of the Fast Serial Interface (FSI) module. The FSI is a serial peripheral capable of reliable high-speed communication across isolation barriers.

31.1 Introduction

The Fast Serial Interface (FSI) module is a serial communication peripheral capable of reliable high-speed communication across isolation devices. Galvanic isolation devices are used in situations where two different electronic circuits, which do not have common power and ground connections, must exchange information. Though isolation devices facilitate these signal communications, they can also introduce a large delay on the signal lines and add skew between the signals. The FSI is designed specifically to ensure reliable high-speed communication for system scenarios that involve communication across isolation barriers without adding components.

The FSI consists of independent transmitter (FSITX) and receiver (FSIRX) cores. The FSITX and FSIRX cores are configured and operated independently.

For additional information on the FSI module, refer to the *Fast Serial Interface (FSI) Skew Compensation Application Report*.

31.1.1 FSI Related Collateral

Foundational Materials

- C2000 Academy - Communications

Getting Started Materials

- *Fast Serial Interface (FSI) Skew Compensation Application Report*
- Fast serial interface (FSI) adapter board evaluation module
- *Using the Fast Serial Interface (FSI) With Multiple Devices in an Application Application Report*

Expert Materials

- *Design Guide: TIDM-02006Distributed Multi-axis Servo Drive Over Fast Serial Interface (FSI) Reference Design*
- *The Essential Guide for Developing With C2000 Real-Time Microcontrollers Application Report*
  - Refer to the See sections 'Distributed Real-Time Control Across an Isolation Boundary' and 'Solving Event Synchronization Across Multiple Controllers in Decentralized Control Systems'. section

31.1.2 FSI Features

The FSI module includes the following features:

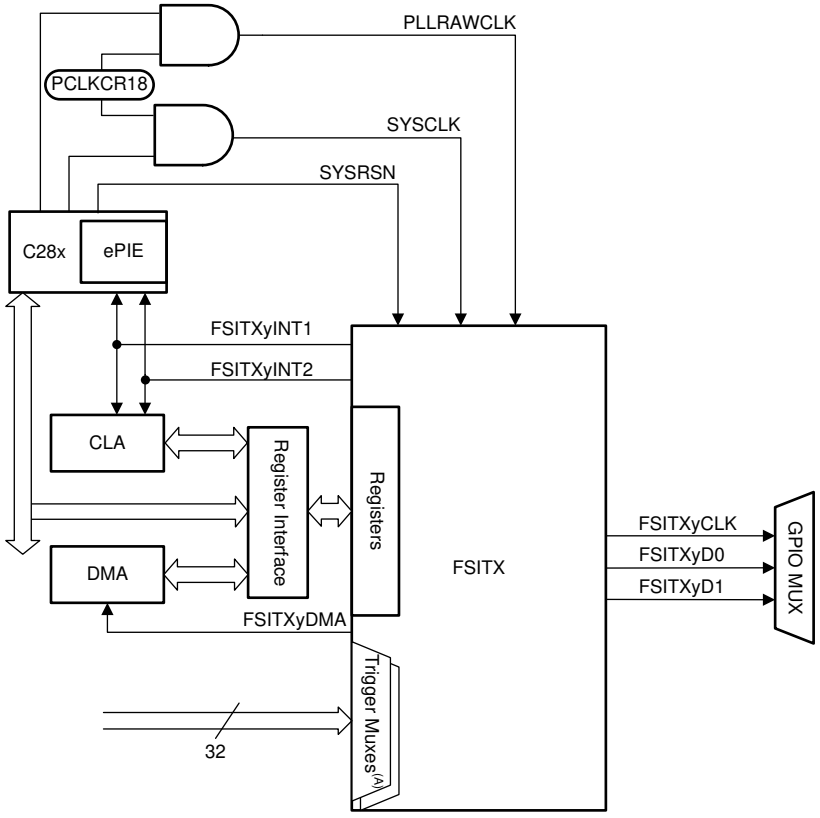
- Independent transmitter and receiver cores
- Source-synchronous transmission
- Double Data Rate (DDR)
- One or two data lines
- Programmable data length
- Skew adjustment block to compensate for board and system delay mismatches
- Frame error detection
- Programmable frame tagging for message filtering
- Hardware ping to detect line breaks during communication (ping watchdog)
- Two interrupts per FSI core
- Externally triggered frame generation
- Hardware- or software-calculated CRC
- Embedded ECC computation module
- Register write protection
- FSI-SPI compatibility mode (limited features available)
- Tag match notifications

31.2 System-level Integration

This section describes the device-level integration of the FSI module. Some of the features may require additional configuration of modules that are not within the scope of this chapter, the details can be found elsewhere in this TRM.

31.2.1 CPU Interface

The following diagrams show the CPU interface of each FSI module.



A. The signals connected to the trigger muxes are described in Section 31.2.5.

Figure 31-1. FSI Transmitter (FSITX) CPU Interface

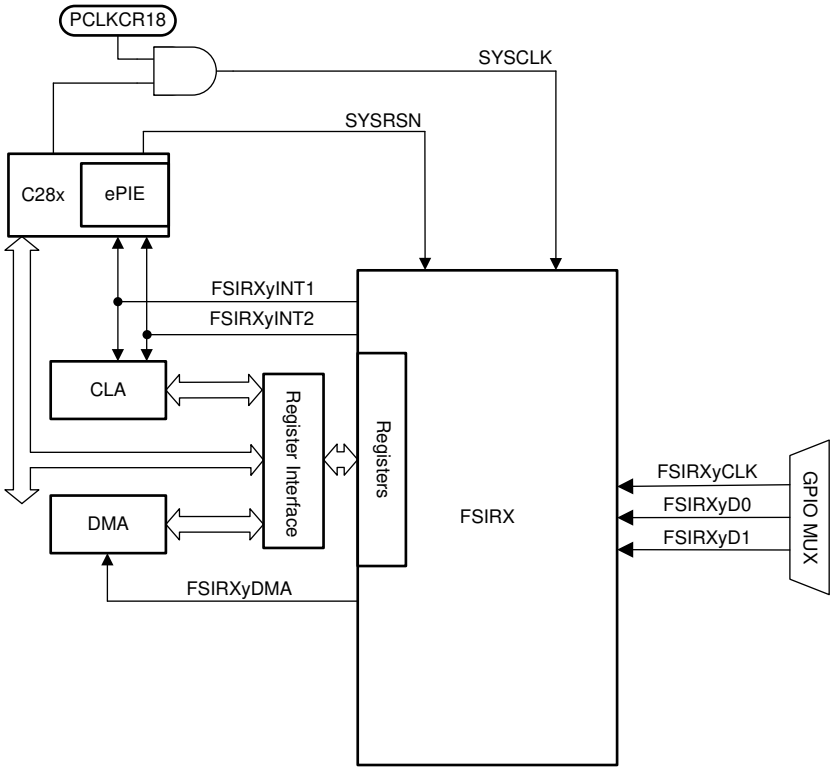


Figure 31-2. FSI Receiver (FSIRX) CPU Interface

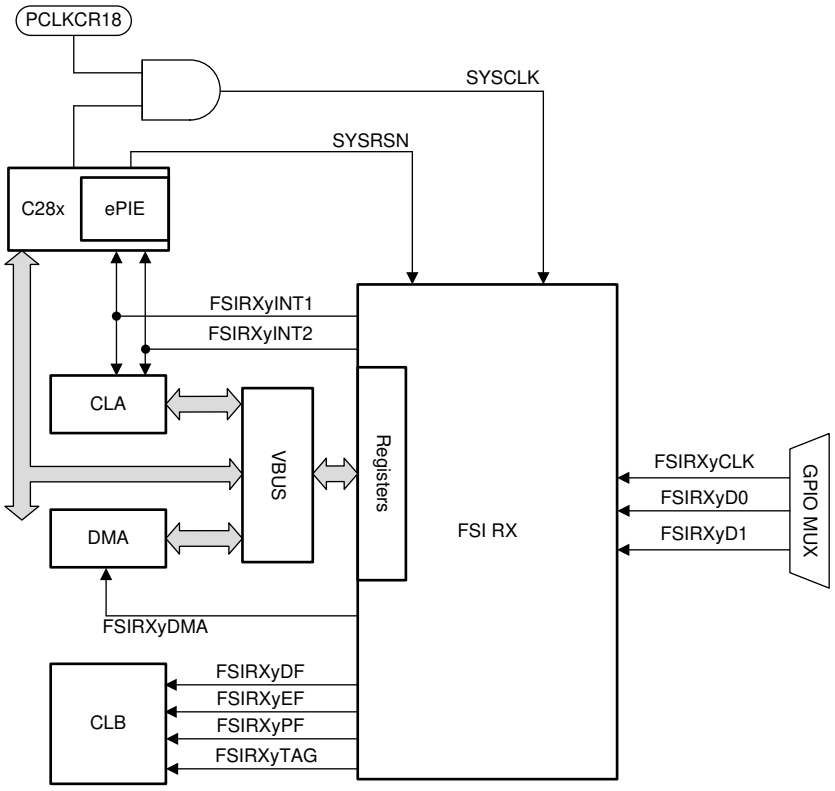


Figure 31-3. FSI Receiver (FSIRX) CPU Interface with CLB

31.2.2 Signal Description

FSI is a point-to-point communication protocol. Hence, an FSI transmitter core will communicate directly to a single FSI receiver core. Similarly, an FSI receiver core will receive data from a single FSI transmitter core.

Each FSI core has three signals associated with it: one clock and two data signals. Data is always transmitted or received with the most significant bit of each frame field being first. If multi-lane transmissions are not used, the TXD1 and RXD1 signals can be left unconnected and their GPIOs repurposed for other application needs. Table 31-1 and Table 31-2 describe the various signals that can be selected by the PADCONFIG register to be brought out to device pins.

CAUTION
The maximum RXCLK rate is SYSCLK/2 and should not exceed this limit.

Table 31-1. FSI Receiver Core Signals

Signal Name	Direction	Description	Inactive Level(1)
RXCLK	Input	This is the receive clock input signal for the FSI receive module. This must should be connected to TXCLK of the transmitting FSI module.	Logic High
RXD0	Input	This is the primary data input line for reception. This should be connected to the TXD0 of the transmitting FSI module.	Logic High
RXD1	Input	This is an additional data input line for reception. This signal should be connected to the TXD1 of the transmitting FSI module if multi-lane transmission is used.	Logic High

(1) Inactive level refers to the state of the pin while the module is not actively receiving data.

Table 31-2. FSI Transmitter Core Signals

Signal Name	Direction	Description	Inactive Level(1)
TXCLK	Output	This is the transmit clock. It is driven by the FSI transmit module. During a transmission, four clock edges will be transmitted before the start of frame phase (preamble) and four clock edges will follow the last bit of the frame (postamble). Data is transmitted on both edges of the clock. In FSI-SPI compatibility mode, the preamble and the post frame clock edges will not be transmitted. Data is transmitted only on one edge of the clock. Data will transmit on rising edge and received on falling edge of the clock.	Logic High
TXD0	Output	This is the primary data output line for transmission. This signal is driven by the FSI transmit module. When the FSI is configured for multi-lane transmission, TXD0 will contain all the even numbered bits of the data and CRC bytes. Other frame fields such as frame type, start-of-frame, tag, and end-of-frame will be transmitted in full.	Logic High
TXD1	Output	This is an additional data output line for transmission if the FSI is configured for multi-lane transmission. This signal is driven by the FSI transmit module. During transmission, the data bits are split between TXD0 and TXD1. TXD1 will contain all the odd numbered bits of the data and CRC bytes. This applies only to the data words and the CRC bytes. Other data frame related information like Frame Type, Start-of-Frame, Tag and End-of-frame, the state of this line will be identical to TXD0.	Logic High

(1) Inactive level refers to the state of the pin while the module is not actively transmitting, or held in reset.

31.2.2.1 Configuring Device Pins

The GPIO mux registers must be configured to connect this peripheral to the device pins. To avoid glitches on the pins, the GPyGMUX bits must be configured first (while keeping the corresponding GPyMUX bits at the default of zero), followed by writing the GPyMUX register to the desired value.

Some IO functionality is defined by GPIO register settings independent of this peripheral. For input signals, the GPIO input qualification should be set to asynchronous mode by setting the appropriate GPxQSELn register bits to 0x3. The internal pullups can be configured in the GPyPUD register. See the *General Purpose Input-Output (GPIO)* chapter for more details on the GPIO mux and settings.

31.2.3 FSI Interrupts

Each FSI module contains multiple interrupt sources which can be assigned to two different interrupt vectors: INT1 and INT2. Each interrupt source has an associated status flag, force, and clear bits in the EVT_STS, EVT_FRC, and the EVT_CLR registers, respectively.

Each interrupt can be assigned to either interrupt vector, INT1 and INT2, to allow for two priority levels. Alternately, the interrupt source can be prevented from generating any interrupt, though the status flag can still be set and monitored by software. The transmitter events are assigned to either interrupt vector in the TX_INT_CTRL register. The receiver events are assigned an interrupt vector using RX_INT1_CTRL and RX_INT2_CTRL registers. If an interrupt is not required, ensure the bit is not set in the respective INT_CTRL register.

31.2.3.1 Transmitter Interrupts

The transmitter can generate the following interrupts:

- **Frame Done (FRAME_DONE):** This event indicates that FSI has completed transmitting a frame.
- **Buffer Underrun (BUF_UNDERRUN):** This event indicates that the transmit buffer has experienced underrun. Buffer underrun occurs when the transmitter tries to read data from a location which has not yet be written to by the CPU, or DMA.
- **Buffer Overrun (BUF_OVERRUN):** The buffer overrun interrupt is generated when the buffer has experienced overrun. Buffer overrun may occur if a piece of data is overwritten before it has been transmitted.
- **Ping Frame Triggered (PING_TRIGGERED):** The ping frame triggered interrupt is generated when the ping frame has been triggered. This bit will be set when the ping counter has timed out or an external ping trigger event has occurred.

31.2.3.2 Receiver Interrupts

The receiver core is capable of generating interrupts from many different events:

- **Ping Watchdog Timeout (PING_WD_TO):** This event indicates that the ping watchdog timer has timed out. The receiver has not received a valid frame within the time period specified in the RX_PING_WD_REF register.
- **Frame Watchdog Timeout (FRAME_WD_TO):** This event indicates that the frame watchdog timer has timed out. The conditions of this timeout are set using the RX_FRAME_WD_CTRL register. As soon as the start of frame phase is detected, the frame watchdog counter will start counting from 0. The end of frame phase must complete by the time the watchdog counter reaches the reference value. If this does not happen, the watchdog will time out and this event will be generated. If this event occurs, the receiver must undergo a soft reset and subsequent resynchronization in order to guarantee proper operation.
- **CRC Error (CRC_ERR):** This error indicates that a CRC error has occurred. A CRC error will be generated when the received CRC and the computed CRC do not match.
- **Frame Type Error (TYPE_ERR):** This error indicates that an invalid frame type has been received. If this error occurs, the receiver must undergo a soft reset and subsequent resynchronization in order to guarantee proper operation.
- **End-of-Frame Error (EOF_ERR):** This error indicates that an invalid end-of-frame bit pattern has been received. If this error occurs, the receiver must undergo a soft reset and subsequent resynchronization in order to guarantee proper operation.
- **Receive Buffer Overrun (BUF_OVERRUN):** This event indicates that an overrun condition has occurred in the receive buffer.
- **Receive Buffer Underrun (BUF_UNDERRUN):** This event indicates that an underrun condition has occurred in the receive buffer. This condition occurs when software reads the buffer while it is empty.
- **Frame Done (FRAME_DONE):** This event indicates that a valid frame has been received without error.
- **Error Frame Received (ERR_FRAME):** This event indicates that an error frame has been received.
- **Ping Frame Received (PING_FRAME):** This event indicates that a ping frame has been received.
- **Frame Overrun (FRAME_OVERRUN):** This event indicates that a new frame has been received while the FRAME_DONE flag was still set.
- **Data Frame Received (DATA_FRAME):** This event indicates that a data frame has been received.
- **Ping Tag Matched (PING_TAG_MATCH):** This event indicates that a ping frame with a matching tag has been received.
- **Data Tag Matched (DATA_TAG_MATCH):** This event indicates that a data frame with a matching tag has been received.
- **Error Tag Matched (ERROR_TAG_MATCH):** This event indicates that an error frame with a matching tag has been received.

31.2.3.3 Configuring Interrupts

To configure interrupts on the FSI, the application should select the interrupt vector for each desired event using the TX_INT_CTRL register for the transmitter, and RX_INT1_CTRL and RX_INT2_CTRL registers for the receiver. There is no module-level interrupt enable bit to configure.

Note

If an event is registered for both interrupt vectors, both interrupts will fire. There are no hardware checks for overlapping interrupt vector assignments.

31.2.3.4 Handling Interrupts

Inside the interrupt service routine (ISR), the user should clear the event flag using the EVT_CLR register and then acknowledge the CPU interrupt.

If the one event occurs multiple times before the corresponding bit is cleared by software, no new interrupt will be generated.

If multiple events occur simultaneously, or very close in time, it is possible to handle multiple conditions within a single interrupt. Each flag is independently set by hardware and must be cleared by application software. If multiple different events occur, the ISR can handle each in whatever order is deemed necessary by the application. It is not advisable to clear the full interrupt status register in every ISR. This may cause the application to miss events that may be detrimental to the application. A sample sequence for handling interrupts on the receiver follows; the transmitter routine will be similar.

- On receiving an interrupt, copy the current state of the receive event and error status flag register (RX_EVT_STS) into a local snapshot variable.
- Read all of the bits from the snapshot to determine the events that require action.
- Perform the necessary actions for each fo the events seen in the snapshot.
- Write to the receive event and error clear register (RX_EVT_CLR) with the snapshot to clear only those interrupts that were set at the beginning of the ISR.
- Repeat this sequence for every generated ISR.

There is a chance that another event occurred during the just-handled ISR since only the snapshot of events was handled and then cleared; an event flag may still be set at the end of the ISR. As soon as the ISR completes, a new interrupt will be generated and this flag will still be set and can be handled accordingly.

Software accesses tied to multiple events and handled within the same ISR may cause race conditions which will cause the software to not function as desired. For example, it is recommended to use different interrupt lines if the user wants to enable events for both ping and data frames. If both are handled within the same interrupt line, the software may only respond to one of the events if they both occur close in time.

31.2.4 DMA Interface

Both the transmitter and receiver are capable of using the DMA for automatic data transfers. The DMA trigger is independent from the interrupt signals. DMA events are only triggered on the completion of a data frame.

The transmitter DMA trigger is enabled by setting TX_DMA_CTRL.DMA_EVT_EN to 1. The transmitter must also set TX_OPER_CTRL_LO.START_MODE to 0x2 to allow either a write to the TX_FRAME_CTRL.START bit or to the TX_FRAME_TAG_UDATA register to start the transmission.

The receiver DMA trigger is enabled by setting RX_DMA_CTRL.DMA_EVT_EN to 1.

Refer to Section 31.3.2 and Section 31.3.3 for more DMA information specific to each FSI Module.

31.2.5 External Frame Trigger Mux

The FSI has two muxes connected to the transmitter module. These muxes are used to select triggers to start ping frames, and/or generic frames. These muxes are independently configured for each type of frame. The application may select one trigger source per frame type. Use of these triggers are optional.

The external ping frame trigger is configured by setting TX_PING_CTRL.EXT_TRIG_SEL to the index of the desired trigger. TX_PING_CTRL.EXT_TRIG_EN must also be set to allow the trigger to generate a ping frame.

The generic frame trigger is configured by setting TX_OPER_CTRL_HI.EXT_TRIG_SEL to the index of the desired trigger. TX_OPER_CTRL_LO.START_MODE must be set to 0x1 in order for a frame to be transmitted by an external trigger.

Note

Triggers generated by the CLB and EPWM XBAR are asynchronous and must be at least 3 SYSCLKs wide.

Table 31-3. External Trigger Sources and Their Index

Index	External Trigger Source
0	EPWM XBAR0-TRIP4
1	EPWM XBAR1-TRIP5
2	EPWM XBAR2-TRIP7
3	EPWM XBAR3-TRIP8
4	EPWM XBAR4-TRIP9
5	EPWM XBAR5-TRIP10
6	EPWM XBAR6-TRIP11
7	EPWM XBAR7-TRIP12
8	EPWM1-SOCA
9	EPWM1-SOCB
10	EPWM2-SOCA
11	EPWM2-SOCB
12	EPWM3-SOCA
13	EPWM3-SOCB
14	EPWM4-SOCA
15	EPWM4-SOCB
16	EPWM5-SOCA
17	EPWM5-SOCB
18	EPWM6-SOCA
19	EPWM6-SOCB
20	EPWM7-SOCA
21	EPWM7-SOCB
22	EPWM8-SOCA
23	EPWM-SOCB
24:39	Reserved
40	CLB1.CLBOUT30
41	CLB1.CLBOUT31
42	CLB2.CLBOUT30
43	CLB2.CLBOUT31
44	CLB3.CLBOUT30
45	CLB3.CLBOUT31
46	CLB4.CLBOUT30

Table 31-3. External Trigger Sources and Their Index (continued)

Index	External Trigger Source
47	CLB4.CLBOUT31
48:51	Reserved
52	ADCSOCA
53	ADCSOCB
54	TIMER0INT
55	TIMER1INT
56	TIMER2INT
57	FSI RX_TRIG0
58	FSI RX_TRIG1
59	FSI RX_TRIG2
60	FSI RX_TRIG3
61:63	Reserved

31.3 FSI Functional Description

31.3.1 FSI Functional Description

The Fast Serial Interface Transmitter and Receiver modules (FSI_TX/FSI_RX) are two completely independent modules on the device. Each module has an independent set of control registers, clocking, and interrupts. The following sections describe the frame format and the various initialization and configuration procedures for both the transmitter and receiver.

31.3.2 FSI Transmitter Module

The FSI transmitter module handles the framing of data, CRC generation, and signal generation of TXCLK, TXD0, and TXD1, as well as interrupt generation. The operation of the transmitter core is controlled and configured through programmable control registers. The transmitter control registers allow the CPU to program, control, and monitor the operation of the FSI receiver. The transmit data buffer is accessible by the CPU and the DMA.

The transmitter has the following features:

- Automated ping frame generation
- Externally triggered ping frames
- Externally triggered data frames
- Software-configurable frame lengths
- 16-word data buffer
- Data buffer underrun and overrun detection
- Hardware-generated CRC on data bits
- Software ECC calculation on select data
- DMA support

Figure 31-4 shows the high-level block diagram of the FSI transmitter. Figure 31-5 shows the block diagram of the transmitter core submodule.

The following sections describe the various aspects of the FSI transmitter in detail.

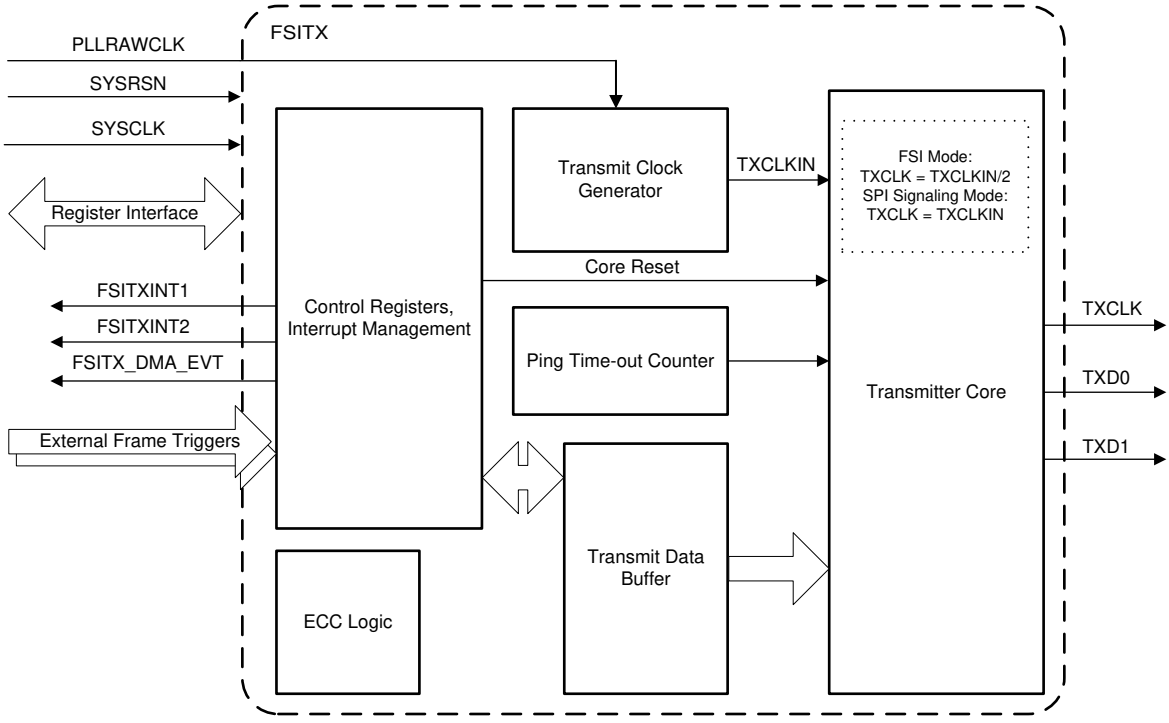


Figure 31-4. FSI Transmitter Block Diagram

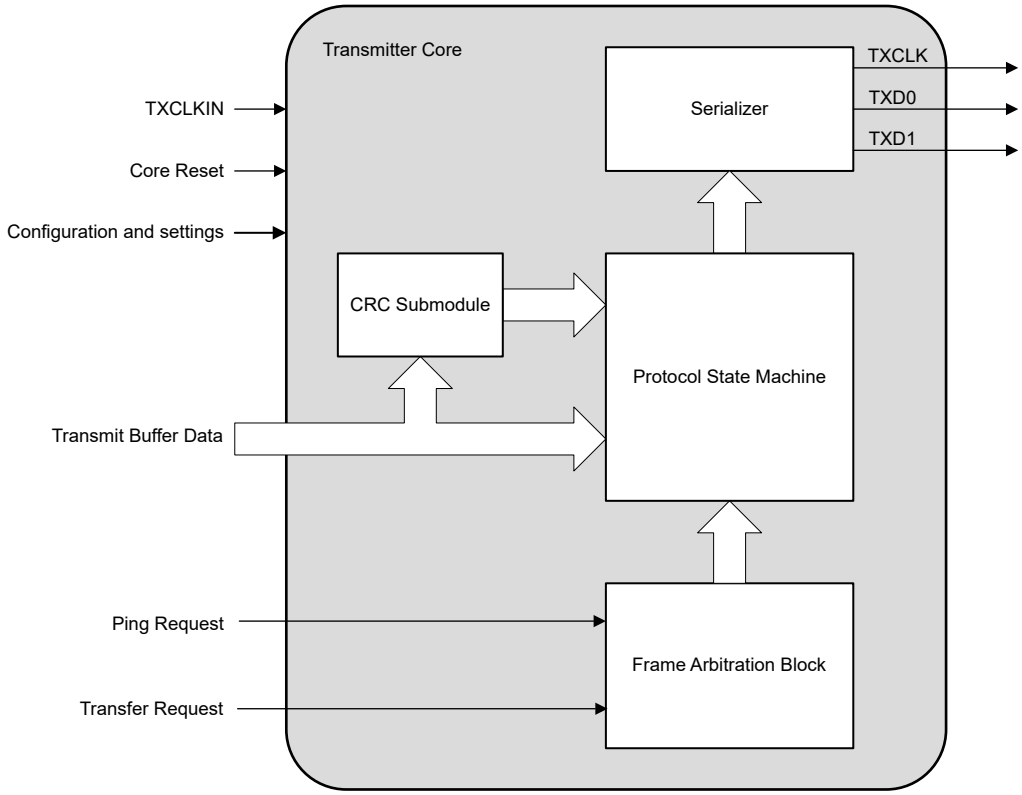


Figure 31-5. FSI Transmitter Core Block Diagram

31.3.2.1 Initialization

On the first initialization or after a module reset due to an underrun condition, the transmitter module should execute the following initialization sequence in order to start or resume transmit operations.

1. Initialize the transmitter clock by setting TX_CLK_CTRL.CLK_RST to 1 and subsequently clearing it.
2. Set the clock to the transmitter core to PLLRAWCLK by setting TX_OPER_CTRL_LO.SEL_PLLCLK to 1.
3. Set the clock prescaler value to the desired rate by writing to TX_CLK_CTRL.PRESCALE_VAL.
4. Enable the transmitter clock divider by setting TX_CLK_CTRL.CLK_EN to 1.
5. Assert the transmitter module soft reset by writing 0xA501 to TX_MASTER_CTRL.
6. Wait four TXCLK cycles.
7. Release the transmitter core from reset by writing 0xA500 to TX_MASTER_CTRL

After initialization and configuration, the transmitter module should synchronize with receiver module before transmitting. The synchronization sequence is described in Section 31.4.1.

CAUTION

Do not change TX_CLK_CTRL.PRESCALE_VAL while the clock is enabled (TX_CLK_CTRL.CLK_EN = 1). Doing so may cause undefined behavior.

31.3.2.2 FSI_TX Clocking

The transmitter core registers and control logic run off of the device system clock (SYSCLK).

The FSI Transmit Clock (TXCLK) is derived from PLLRAWCLK. PLLRAWCLK is divided down by configuring the clock prescaler value (TX_CLK_CTRL.PRESCALE_VAL) then setting the clock divider enable bit (TX_CLK_CTRL.CLK_EN). The clock prescaler value can be set to divide PLLRAWCLK by 1 (TX_CLK_CTRL.PRESCALE_VAL = 0x0 or 0x1) through 255(TX_CLK_CTRL.PRESCAL_VAL = 0xFF). Though TXCLK and SYSCLK are both derived from PLLRAWCLK, TXCLK is asynchronous with respect to SYSCLK.

CAUTION

TXCLK should never be configured to be faster than SYSCLK/2.

31.3.2.3 Transmitting Frames

On the transmitter, the ping frame is the only frame which can be set up and transmitted without any further software or DMA intervention. Ping frames can be transmitted by any (or all) of the three sources: automatic ping timer, software, or external triggers.

Each available frame type can be sent multiple ways. Generically, the following steps must be executed before the frame is sent. These steps may be executed in any order before the start condition is set.

1. Configure the frame type
2. Set the frame tag
3. If the frame to be sent is a data frame:
   - Set the user data
   - Write to the data buffer
   - Set the word length if the frame is a software defined frame length
4. Set the start condition

Note

Transmit Frame Start Restriction:

A new frame transmission can be initiated by one of the methods selected in the TX_OPER_CTRL_LO.START_MODE bits. If there is already a PING frame transmission taking place, due to a hardware initiated PING timer, the new frame transmission will begin as soon as the on-going PING transmission is completed.

Once a START of frame has been initiated, the next START of frame will be recognized when the first frame has started transmitting the End-of-Frame (EOF) field. If a new START trigger arrives before the current transmission has reached the EOF field, the trigger will be lost without a notification.

Note

There is no hardware check implemented to check whether the type field written by software is valid or not. If an invalid type is used and a frame transmission is initiated, the behavior will be as follows:

- The transmitted frame structure will be exactly like an NWORD data frame. The size of the data frame will be determined by the value in the TX_FRAME_CTRL.N_WORDS register.
- The frame type field of the transmitted data frame will be transmitted as programmed. If this is received by an FSI receiver, it will generate a Type error.

This mechanism can be used for force a Type error in a received frame for testing purposes.

The following sections describe the specific configuration for each frame type and start condition.

31.3.2.3.1 Software Triggered Frames

The most basic way to transmit a data frame is through software. Each step must be handled by the application. To send a data frame using software, the following steps should be executed. Steps 1-6 may be executed in any order before setting TX_FRAME_CTRL.START. Some fields may not need to be reconfigured for every transmission. The frame tag, user data, and frame type are sticky and will be re-transmitted in the subsequent frame unless they are modified by software.

1. Write the data to be transmitted to the next location of the transmit data buffer.
2. Set TX_FRAME_CTRL.FRAME_TYPE to the appropriate value for the type of frame to be transmitted.
3. Set TX_FRAME_CTRL.N_WORDS to 1 less than the number of words to be transmitted if TX_FRAME_CTRL.FRAME_TYPE is set to 0011, the frame type of the software-defined length data frame. That is, if 16 words will be transmitted, N = 16, set TX_FRAME_CTRL.N_WORDS to 15.
4. When the frame is assembled before transmitting, the FSITX hardware will calculate the CRC to be transmitted. If TX_OPER_CTRL_LO.SW_CRC is 1, the application may calculate a custom CRC value and then set TX_USER_CRC to the result.
5. Set TX_FRAME_TAG_UDATA.FRAME_TAG to the desired tag.
6. Set TX_FRAME_TAG_UDATA.USER_DATA to the desired user data.
7. Set TX_FRAME_CTRL.START to 1 to initiate the transmission of the data frame.

Once the frame transmission has started, the TX_FRAME_CTRL.START will be cleared by hardware. To monitor if the frame has completed, the software can poll TX_EVT_STS.FRAME_DONE.

31.3.2.3.2 Externally Triggered Frames

The transmitter can transmit frames when triggered by an external source. See Section 31.2.5 for more information on the available external triggers.

To transmit frames using an external trigger, the application must follow the same procedure as described in Section 31.3.2.3.1. The only difference is that in Step 7, the start condition will be automatically set when the external trigger condition is met rather than by software.

It is important to note that by externally triggering frames, the frame information to be sent will be pulled from the same registers described in the previous section. Because of this, it is possible to send any type of frame from an external trigger including ping, error, and data frames. Also, there is no hardware mechanism by which the FSI can determine if multiple triggers occur. The FSITX will take the data as is, and the application software should ensure that this data has been updated as necessary.

Using TX_EVT_STS fields either by polling or by interrupts, the application can populate or update the frame information to be sent in the next frame

31.3.2.3.3 Ping Frame Generation

Assuming the FSI transmitter has already been properly initialized, the following sequences can be used to configure and send ping frames.

31.3.2.3.3.1 Automatic Ping Frames

To generate periodic ping frames, the following steps must be followed:

1. Initialize the ping counter by writing 1 to TX_PING_CTRL.CNT_RST.
2. Set the desired ping tag to TX_PING_TAG.TAG.
3. Set the ping timer reference value to TX_PING_TO_REF.TO_REF.
4. Enable the ping timer by writing 1 to TX_PING_CTRL.TIMER_EN.

The ping timer is a free-running counter which will count up from 0. The current value of the ping timer counter is found in TX_PING_TO_CNT. When the current value of TX_PING_TO_CNT matches the reference value TX_PING_TO_REF.TO_REF, TX_EVT_STS.PING_TRIGGERED will be set. TX_PING_TO_CNT will reset to 0 and resume counting until the next match has occurred or the ping timer is halted by software (TX_PING_CTRL.TIMER_EN is set to 0).

31.3.2.3.3.2 Software Triggered Ping Frame

Software can also manually generate a ping frame. The process for sending a ping frame with software is very similar to sending the other types of frames. The following steps must be followed:

1. Set TX_FRAME_CTRL.FRAME_TYPE to 0000'b to denote that the frame being sent will be a Ping Frame.
2. Set TX_FRAME_TAG_UDATA.FRAME_TAG to the desired value.
3. Write 1 to TX_FRAME_CTRL.START. This will start the transmission.

Once the frame transmission has started, the TX_FRAME_CTRL.START will be cleared by hardware. To monitor if the frame has completed, the software can poll TX_EVT_STS.FRAME_DONE.

31.3.2.3.3.3 Externally Triggered Ping Frame

The last source for generating ping frames is an external trigger. One of up to 32 different triggers may be selected. See Section 31.2.5 for the list of input sources.

CAUTION

Ping frames can be triggered by both an external trigger source and the internal ping timer. If TX_PING_CTRL.EXT_TRIG_EN is set to 1, it will take precedence and the ping timer will be ignored.

31.3.2.3.4 Transmitting Frames with DMA

The FSI transmitter can send data which is continuously fed with the DMA. A DMA trigger will be generated every time a data frame transmission is completed. This is concurrent with the FRAME_DONE signal that sets the TX_EVT_STS.FRAME_DONE flag.

In order to transmit continuous data with the DMA, some configurations need to be made on the transmitter:

First, set TX_DMA_CTRL.DMA_EVT_EN to 1. This will allow the DMA trigger to propagate to the DMA module. Next, TX_OPER_CTRL_LO.START_MODE must be set to 0x2. The transmitter is now able to start a transmission using a software write to TX_FRAME_CTRL.START or TX_FRAME_TAG_UDATA..

The DMA must also be configured properly for the FSI to send the data. One possible solution of using the DMA to continuously feed the transmit buffer is shown below:

- Set up two DMA channels to be triggered by the same FSI transmitter and DMA trigger.
- Configure one channel to fill the transmit buffer.
- Configure the other channel to set the frame tag and user data fields
- Since the FSI transmit buffer is a 16-word circular buffer, ensure the DMA channel servicing the data buffer wraps the after 16 words are copied.

Note

Because the frame tag and user data must written in to in order to initiate the transmission of the frame, use two consecutive DMA channels. This ensures that the DMA channels are always executed in sequence. The DMA channel servicing the data buffer should be the lower numbered channel and the tag/user data channel should be the next. For example, configure DMA channel 3 to service the data buffer, and configure DMA channel 4 to service the tag and user data.

31.3.2.4 Transmit Buffer Management

The FSI transmitter has a 16-word buffer from which it pulls data to transmit. This buffer is implemented as a circular buffer, not a FIFO, so some care must be taken to properly interpret buffer overrun and underrun, as well as the TX_BUF_PTR_STS register. These flags and pointers work under the assumption that the software or DMA is using the buffer as a circular buffer. This mode of operation is the only way in which the overrun, underrun, and pointer status are meaningful. If data is being sourced by the DMA and there is some other periodic trigger mechanism trying to initiate transfers, underrun becomes a critical error. If an underrun happens, it means that the buffer went out of sync. This would not only affect the current transfer, but all future transfers also could not be guaranteed due to the ring buffer. Under such conditions, the underrun would need a soft reset to cleanly recover. Alternately, the software could manually stop the transmitting, reset the buffer pointers, clear the remaining error conditions, and then restart transmission. The software method involves a few steps, while the soft reset is a single action and will guarantee a full reset of the control registers.

Due to the flexibility of the transmit buffer, it is possible for software to implement a simple ping-pong buffer, or to randomly load and send from any location of the buffer. If the buffer is used in this manner, error flags and status fields may be ignored without adversely affecting the transmitter capability. Additionally, the CURR_WORD_CNT will also be invalid if used in this way. The application can set the buffer pointer manually by writing the 4-bit index to TX_BUF_PTR_LOAD. This will force the transmitter to start picking the data from the indicated location in the buffer.

31.3.2.5 CRC Submodule

The FSI transmitter can supply the CRC to the frame being transmitted through the embedded hardware CRC submodule or by supplying a user-defined value. This is controlled by setting TX_OPER_CTRL_LO.SW_CRC appropriately.

If hardware CRC generation is selected (TX_OPER_CTRL_LO.SW_CRC = 0, the default), the CRC is computed by hardware on the data and user data fields using the CRC polynomial 0x7 ($x^8 + x^2 + x + 1$). The transmitter module will automatically compute the CRC on the data fields without user intervention when the frame is transmitted. For more information on how the CRC is generated by the CRC Submodule, refer to Section 31.3.7.

If software CRC generation is selected (TX_OPER_CTRL_LO.SW_CRC = 1), the CRC must be computed by software and placed in the TX_USER_CRC register. The next frame to be transmitted will use the value placed in the TX_USER_CRC register in place of the CRC value generated by the hardware.

As the TX_USER_CRC register is software-programmable, the application may use this field as an extra data field for application specific purposes. If TX_USER_CRC is used in this manner, the CRC detection on the receiver will not be valid and should be ignored.

31.3.2.6 Conditions in Which the Transmitter Must Undergo a Soft Reset

Unlike the receiver, there are no detectable errors that require a soft reset. A buffer overrun or underrun interrupt may or may not require a soft reset in order to resume proper operation. This determination is up to the application software. Refer to Section 31.3.2.4 for more information on the transmit buffer.

31.3.2.7 Reset

The entire transmitter module and all transmitter registers are reset by SYSRSn. The transmitter core is reset by SYSRSn or by writing a 1 to TX_MASTER_CTRL.CORE_RST.

A module reset will cause the registers to be reset to their default state.

31.3.3 FSI Receiver Module

The receiver module interfaces to the FSI clock (RXCLK), and data lines (RXD0 and RXD1) after they pass through an optional programmable delay line. The receiver core handles the data framing, CRC computation, and frame-related error checking. The receiver bit clock and state machine are run by the RXCLK input, which is asynchronous to the device system clock.

The receiver control registers allow the CPU to program, control, and monitor the operation of the FSI receiver. The receive data buffer is accessible by the CPU and the DMA.

The receiver core has the following features:

- 16-word data buffer
- Multiple supported frame types
- Ping frame watchdog
- Frame watchdog
- CRC calculation and comparison in hardware
- ECC detection
- Programmable delay line control on incoming signals
- DMA support
- FSI-SPI compatibility mode

Figure 31-6 provides a high-level overview of the internal modules present in the FSI receiver. Figure 31-7 shows a view of the FSI receiver core submodule. Not all data paths and internal connections are shown.

The following sections describe the various aspects of the FSI receiver module.

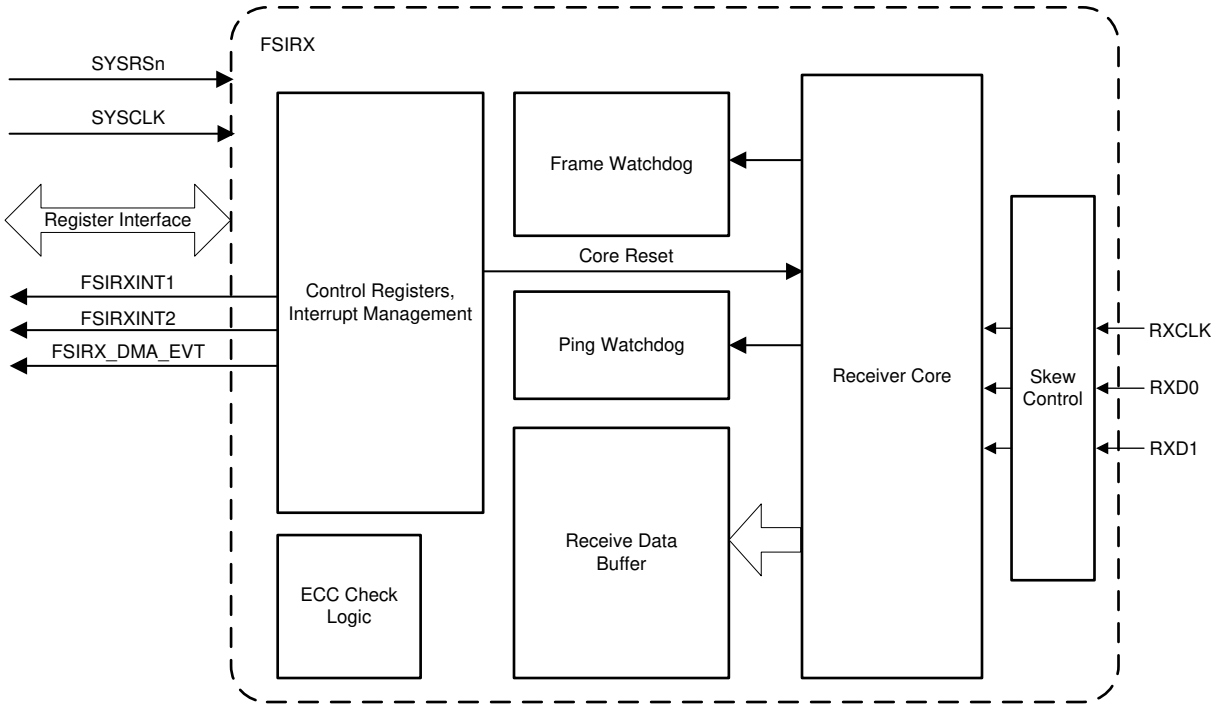


Figure 31-6. FSI Receiver Block Diagram

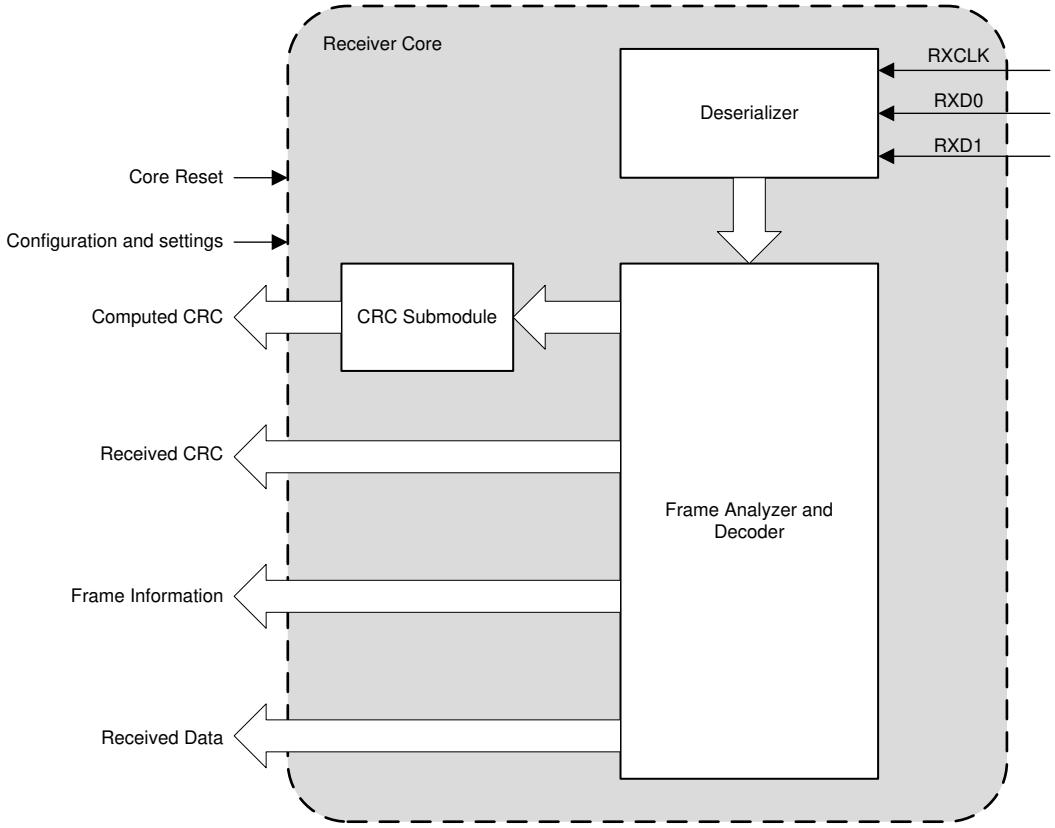


Figure 31-7. FSI Receiver Core Block Diagram

31.3.3.1 Initialization

On the first initialization or after a module reset following any frame error, the receiver module should assert and release the receiver core reset bit (RX_MASTER_CTRL.CORE_RST) prior to any other initialization. Once the receiver module is initialized, the following steps may be executed:

1. If required, assign interrupt sources to the necessary interrupt line.
2. If required, configure the ping watchdog to periodically check for an active link to the transmitter. See Section 31.3.3.4 for configuration details.
3. If required, configure the frame watchdog to ensure that each frame is received within a predetermined window. See Section 31.3.3.5 for configuration details.
4. Initialize the receive buffer pointer by writing to the RX_BUF_PTR_LOAD register. Received data will be placed into the buffer starting with the address loaded in this register.
5. Ensure all errors and flags have been cleared from the RX_EVT_STS register.

At this point the receiver is ready to receive any incoming frames. Software can now either poll on the RX_EVT_STS register for various conditions. For example, when the RX_EVT_STS.FRAME_DONE and no other flags are set, the receiver has successfully received a frame without error.

Next, the application should configure the various features such as the ping and frame watchdogs, DMA, external triggering, and so on. These features are described in subsequent sections. The receiver module is now ready to synchronize with the transmitter then begin reception. The synchronization sequence is described in Section 31.4.1.

31.3.3.2 FSI_RX Clocking

The receiver module registers and control logic are clocked by the device system clock (SYSCLK). The receiver state machine is clocked by the receiver input clock pin (RXCLK).

CAUTION

RXCLK should never be faster than SYSCLK.

31.3.3.3 Receiving Frames

Once the receiver has been properly configured and synchronized, incoming messages are handled as described below. It is important to note that there is no equivalent to a chip select signal to gate incoming data. Every valid clock edge will latch data into the receiver.

The header information of the received frame will be placed in their respective register fields.

- RX_FRAME_INFO.FRAME_TYPE will contain the received frame type.
- RX_FRAME_TAG_UDATA.FRAME_TAG will contain the received frame tag.
- RX_FRAME_TAG_UDATA.USER_DATA will contain the received user data.

If any error conditions occur during reception such as a CRC mismatch, frame error, frame timeout, buffer overrun, or ping watchdog timeout, the corresponding flag will be set in the RX_EVT_STS register.

Note

If at any point during operation a frame error occurs, the receiver module must be reset and re-synchronized with the transmitter before the next frame can be successfully received. The follow errors are classified as frame errors:

- Type error
- CRC error
- End of frame error

31.3.3.3.1 Receiving Frames with DMA

The FSI receiver can continuously receive data and move it from the receiver buffer with the DMA. A DMA trigger will be generated every time a data frame has been received. This is concurrent with the FRAME_DONE signal that sets the RX_EVT_STS.FRAME_DONE flag. In order to receive continuous data with the DMA, some configurations need to be made on the receiver.

First, set RX_DMA_CTRL.DMA_EVT_EN to 1. This will allow the DMA trigger to propagate to the DMA module. The receiver is now able to trigger a DMA event upon the reception of a data frame.

The DMA must also be configured properly for the FSI to receive the data. One possible solution for using the receiver to continuously feed the DMA is show below:

- Set up two DMA channels to be triggered by the FSI Receiver DMA Trigger.
- Configure one DMA channel to copy data from the receive buffer to a larger data buffer.
- Configure the next DMA channel to copy the received frame tag and user data to another data buffer.
- Since the FSI receive buffer is a 16-word circular buffer, ensure the DMA channel servicing the data buffer wraps after 16 words are copied.

Unlike the transmitter, there is no requirement to have the DMA channel which is handling the data buffer, execute before the DMA channel handling the received tag and user data.

31.3.3.4 Ping Frame Watchdog

The ping frame watchdog is a hardware-enabled automatic error detection of the connection status to the transmitter. This watchdog monitors the time elapsed between ping frames. If the transmitter has been set up to periodically send out a ping frame, the receiver can be set up to monitor whether this frame has been received within a specified amount of time. If the time between ping frames has exceeded the programmed number of clock cycles, an event will be triggered which can generate an interrupt or be monitored by software.

This watchdog has a dedicated counter which is reset and restarted upon the successful reception of a ping frame. The watchdog counter will be incremented at the rate of SYSCLK. Optionally, the watchdog can be configured to be reset upon the successful reception of any frame. This option allows the receiver to monitor for any successful frame to indicate that the connection is still alive and the transmitter is still functioning as expected.

To configure the ping frame watchdog for operation:

1. Reset the ping watchdog counter by setting RX_PING_WD_CTRL.PING_WD_RST to 1 and then subsequently clearing it to 0.
2. Set RX_OPER_CTRL.PING_WD_RST_MODE to the desired watchdog reset event, either 0 for ping frames only, or 1 for any frame.
3. Set RX_PING_WD_REF to the maximum time between frames. Add 10 additional SYSCLK cycles to account for clock synchronization.
4. Enable the ping watchdog by setting RX_PING_WD_CTRL.PING_EN to 1.

The ping watchdog is now enabled and can now monitor for ping frames.

If the RX_PING_WD_CNT value reaches the value programmed in RX_PING_WD_REF, the RX_EVT_STS.PING_WD_TO flag will be set. If configured, and interrupt can be generated on this event.

31.3.3.5 Frame Watchdog

The frame watchdog is an additional feature the receiver can use to monitor for any error conditions. This dedicated watchdog monitors the duration that it takes for a single frame to be received. The watchdog starts incrementing at the time the receiver detects a proper start of frame condition. If the end of frame condition is not detected within the expected number of SYSCLK cycles, the frame watchdog will be triggered which can generate an interrupt or be monitored by software.

This watchdog is automatically started and stopped at the start-of-frame and end-of-frame conditions respectively. The frame watchdog is connected to SYSCLK.

To configure the frame watchdog for operation:

1. Reset the frame watchdog counter by setting RX_FRAME_WD_CTRL. FRAME_WD_CNT_RST to 1 and then subsequently clearing it to 0.
2. Set RX_FRAME_WD_REF.FRAME_WD_REF to the maximum number of SYSCLK cycles expected to be in the longest frame that may be received. Add an additional 10 SYSCLK cycles to account for clock synchronization.
3. Enable the frame watchdog by setting RX_FRAME_WD_CTRL.FRAME_WD_CNT_EN to 1.

The frame watchdog is now enabled and can detect a failed frame.

If the RX_FRAME_WD_CNT reaches the value programmed in RX_FRAME_WD_REF, the RX_EVT_STS.FRAME_WD_TO flag will be set. If enabled, an interrupt can be generated on this event.

If the frame watchdog interrupt ever occurs, the receiver core is in an invalid state to receive a new transmission. The only way to recover from a frame watchdog time out is to undergo a soft reset, and subsequently resynchronizing with the transmitter.

31.3.3.6 Delay Line Control

The receiver module has a programmable delay line on each of the external signal inputs: RXCLK, RXD0, and RXD1. The delay elements introduce delays on the respective lines. This is to facilitate adjustment for signal delays introduced by system level components such as signal buffers, ferrite beads, isolators, and so on, or board delays such as uneven trace lengths, long cable length, and so on. The length of the delay is controlled by setting the RX_DLY_LINE_CTRL register values for each line. By default, no delay is introduced by the delay line elements. The delay values should only be adjusted while the FSIRX is held in soft reset, ensuring that there are no active transmissions during this process. Figure 31-8 shows a representation of the delay line circuitry for the input signals. The implementation for RXCLK, RXD0, and RXD1 are replicas of this diagram. All circuits will behave similarly.

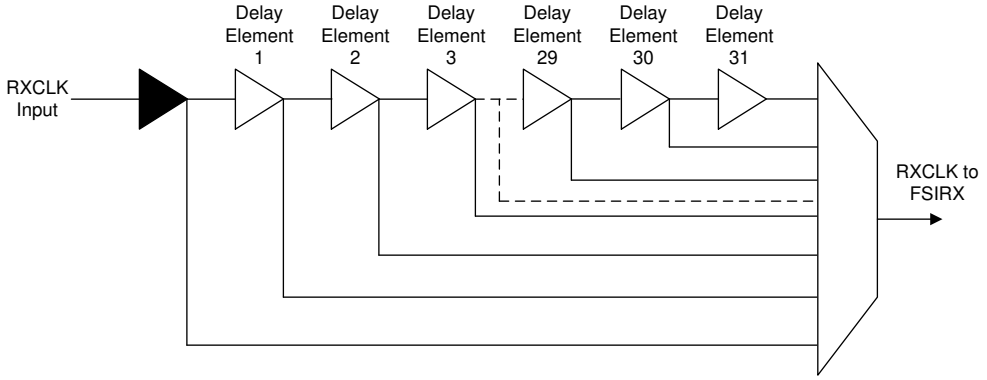


Figure 31-8. Delay Line Control Circuit

For more information on skew compensation, refer to the *Fast Serial Interface (FSI) Skew Compensation Application Report*.

The FSITX module also has delay line control circuitry that is placed before the FSITX signals (TXCLK, TXD0, and TXD1) are sent to the TDM signal selection mux (controlled by the SEL_TDM_PATH signal). The TX_DLY_LINE_CTRL register determines the length of the delay for each output line.

31.3.3.7 Buffer Management

The FSI receiver has a 16-word buffer into which the data is copied to when it has been received. This buffer is implemented as a circular buffer, not a FIFO, so some care must be taken to properly interpret buffer overrun and underrun as well as the RX_BUF_PTR_STS register. These flags and pointers work under the assumption that the software or DMA is using the buffer as a circular buffer. If the receiver state machine enters into an erroneous state, there is no way for software to cleanly handle this because there is no guaranteed receive clock. For the receiver to detect a clean resynchronization, the state machine needs to be operational and not in the error state. The only way to recover from the error state is to reset the entire receiver module. For overrun and underrun, the receiver can no longer guarantee which values in the buffer are valid. As such, the best way to recover is to reset the FSI and resynchronize with the transmitter.

Due to the flexibility of the receive buffer, it is possible for software to implement a simple ping-pong buffer, or to randomly receive and read from any location of the buffer. If the buffer is used in this manner, these flags and status fields may be ignored without adversely affecting the receiver capability. Additionally, the CURR_WORD_CNT will also be invalid if used in this way. The application can set the buffer pointer manually by writing the 4-bit index to RX_BUF_PTR_LOAD. This will force the receiver to start storing the received data starting at the indicated location in the buffer.

31.3.3.8 CRC Submodule

The receive module will automatically calculate the CRC on the incoming data. The received CRC value is placed into RX_CRC_INFO.RX_CRC. The CRC value calculated by hardware on the received data is placed into RX_CRC_INFO.CALC_CRC. These values are compared by hardware and will set RX_EVT_STS.CRC_ERROR if there is a mismatch. The receiver can generate an interrupt based on RX_EVT_STS.CRC_ERROR if enabled.

Since the CRC is only used in data frames, the values found in RX_CRC_INFO.RX_CRC and RX_CRC_INFO.CALC_CRC are undefined during ping and error frames.

For more information on how the CRC is calculated, refer to Section 31.3.7.

If the transmitting module is sending a software-defined CRC value (FSITX.TX_OPER_CTRL_LO.SW_CRC = 1) the receiver module will trigger a CRC Error event if the received value does not match the hardware-calculated value. As this is an application level decision, the FSIRX can safely disregard the CRC error event. Application software will need to calculate and verify the incoming CRC using the same custom algorithm used on the transmitter and act appropriately.

The CRC field may have also been used as an application-specific value, not a CRC, the application can use the RX_CRC_INFO.RX_CRC as required. All CRC errors and flags can be ignored in this situation.

31.3.3.9 Using the Zero Bits of the Receiver Tag Registers

The receiver tag registers (receiver frame tag and user data (RX_FRAME_TAG_UDATA) register and receiver ping tag (RX_PING_TAG) register) have the Least Significant Bit is set to 0. The actual received tag is in the bit positions 4:1. The reason for this is to facilitate user software to create a table of functions which can be called depending on the tag value. A function pointer needs a 32-bit storage space and hence each successive pointer will be offset by 2. If the first pointer is at address x, then the second pointer will be at address x+2, the third at address x+4 and so on. By keeping the LSB 0, the five bits of the tag register (bits 4:0) can now be directly used as an index into a table of function pointers.

31.3.3.10 Conditions in Which the Receiver Must Undergo a Soft Reset

The receiver will receive data on every clock edge. While there are specific patterns which determine the a start of a frame, and denote the end of a frame, these patterns are able to occur at any point during normal operation inside of the frame. If there ever is a point at which the receiver fails to detect a successful frame, the module must be reset in order to ensure that subsequent frames are received properly.

When any of the following errors occur in a received frame, the receiver may be required to be reset and resynchronized with the transmitter:

- Frame type error
- End of frame error
- Ping frame watchdog timeout
- Frame watchdog timeout
- Receiver in an invalid state due to noisy clock

The receiver core status (RX_VIS_1.RX_CORE_STS) can be monitored to determine if it has entered into an error state requiring a soft reset in order to resume communication. Incorrect frame type and end of frame errors will always cause this bit to become set. A soft reset is required in these cases. A frame watchdog timeout will always require a reset due to the fact that the receiver state machine is still expecting more information when the watchdog timed out. RX_CORE_STS can be used to determine if a noise event was the cause of the failed frame. The ping frame watchdog also will not cause RX_CORE_STS to be set. Similar to the frame watchdog, a corrupt receiver may not be the reason for the ping frame to have timed out. The transmitter may have gone offline and never sent a ping frame. Alternately, during idle time, a noise event may have occurred, thereby putting the receiver into a corrupt state. As the receiver will be able to detect this during the ping frame watchdog timeout interrupt handler, this type of event will not be lost and the application can act appropriately.

As the receiver is clocked by RXCLK, not SYSCLK, a noisy clock or data line may cause some internal design constraints to be violated, putting the receiver core logic into undefined states. Ensure that the clock and data

lines satisfy the Electrical Characteristics and timing requirements of the FSI module found in the data manual for this device. Failure to do so may cause the receiver state machine to go into an unrecoverable error state. The receiver can only be recovered by undergoing a soft reset. To determine the state of the receiver core after an unexpected frame error, the application should check the receiver core status bit.

In addition to the above errors, buffer overrun or underrun may warrant a soft reset in order to resynchronize with the local application software. Refer to Section 31.3.3.8 for more information on the receive buffers. The requirement of resetting the receiver due to overrun or underrun is up to the application.

After the receiver has been placed into soft reset, the application must notify the other device's transmitter to begin a new synchronization phase. The simplest way to achieve this is through a ping or error frame sent with a designated tag. If the application is not using the FSITX on the device with the detected error, some other method must be established. The other device should stop transmitting and begin a new synchronization phase.

31.3.3.11 FSI_RX Reset

The receiver module and its registers are reset by SYSRSn. The receiver core is reset by SYSRSn or by writing a 1 to RX_MASTER_CTRL.CORE_RST.

A module reset will cause the registers to be reset to their default state. After a module reset, the receiver module must be re-initialized, and the data link re-established.

31.3.4 Frame Format

The FSI module transmits and receives information in frames. Each frame will contain multiple phases where different information can be found. The number of phases as well as the total length of the frame will vary depending on the frame type being transmitted. Frames may be as short as 16 bits long for a ping or error frame or 288 bits long for a 16-word data frame.

In normal transmission mode, there are four preamble clock edges before the start of the frame and four post-frame clock edges (postamble). Data is transmitted on both edges of the clock (double data rate). The basic frame structure is shown below. Each phase of the frame (such as start-of-frame, frame type, and so on) will be transmitted with the most significant bit first. Table 31-4 describes the basic frame structure used by the FSI and adapted according to which frame type is transmitted.

Table 31-4. Basic Frame Structure

Idle State	Preamble	Start of Frame	Frame Type	User Data	Data Words	CRC Byte	Frame Tag	End of Frame	Postamble	Idle State
	1111	1001	4 bits	8 bits	1-16 words	8 bits	4 bits	0110	1111	

The FSI also supports a FSI-SPI compatibility mode. The SPI compatible frame structure is similar to a standard FSI frame, but there are differences. Refer to Section 31.3.13 for more information on how to configure and use the FSI-SPI compatibility mode.

Note

One word of the FSI refers to 16 bits.

The terms “frame” and “packet” can be used interchangeably to describe the signaling format of the FSI.

31.3.4.1 FSI Frame Phases

The different phases of the frame structure are described in detail.

- **Idle State:** During the idle state, the clock and data lines will be driven high, the inactive state.
- **Preamble:** The preamble phase contains four clock edges (or two complete clock pulses) with the data signals held in the high state. These clock edges serve to flush the receiver logic and prepare it for receiving a new frame. This phase is not present in SPI compatibility mode.
- **Start of Frame:** The start of frame phase contains two clock pulses with four bits, 1001, transmitted on the data lines.
- **Frame Type:** The frame type phase contains two clock pulses with the 4-bit frame type code being transmitted on the data lines. The different frame types are described in detail in Section 31.3.4.2. The transmitter must set the TX_FRAME_CTRL.FRAME_TYPE field before transmitting a frame. The received frame type is stored in the RX_FRAME_INFO.FRAME_TYPE.
- **User Data:** The user data phase contains a fully user-configurable data field. There are no restrictions on how this field is used. This phase is only available in data frames. The user data to be transmitted is set by writing to TX_FRAME_TAG_UDATA.USER_DATA. The received user data is stored in RX_FRAME_TAG_UDATA.USER_DATA.
- **Data:** The data phase contains the data that is being transmitted. The data is pulled from the transmit buffer of the transmitter and will be placed in the receive buffer of the receiver. Word 0 is transmitted first. This phase is only present in data frames. Depending on the type of frame transmitted, this can contain anywhere between 1 and 16 words depending on the frame type selected. More information on data frames is found in Section 31.3.4.2.3.
- **CRC Byte:** The CRC byte contains the CRC of the transmitted data. The value present in this phase can be sourced from either hardware or software based on the TX_OPER_CTRL_LO.SW_CRC bit. Refer to the module-specific section of the CRC Submodule for more information on the CRC is generated or used, for the transmitter and receiver modules respectively. The CRC byte is only present in data frames.
- **Frame Tag:** The frame tag contains the 4-bit user-defined frame tag. There are no restrictions on how this field is used in an application. The transmitter supplies this tag into the TX_FRAME_TAG_UDATA.FRAME_TAG bits for data frames. Ping frames use the tag defined in TX_PING_TAG.TAG. The receiver can access the received frame tag in RX_FRAME_TAG_UDATA.FRAME_TAG.
- **End of Frame:** The end of frame contains four clock edges with four bits, 0110, transmitted on the data lines.
- **Postamble:** The postamble contains four additional clock edges with the data lines held in the high state. After the postamble, the clock and data lines will be driven high, their inactive state. This phase is not present in FSI-SPI compatibility mode.

31.3.4.2 Frame Types

The FSI hardware can generate and handle many predefined frame types. The different frame types can be used by the application to signal different types of events or convey different information to the receiver. The different frame types influence which phases and data fields to include in the transmitted frames.

Table 31-5 provides a short overview of the different frame types used by the FSI. Each frame type is described in more detail in the following subsections.

Table 31-5. Frame Types and Their 4-bit Codes

Frame Type	4-bit Frame Code	Description
PING	0000	This is the ping frame that can be sent either by software or automatically by hardware.
ERROR	1111	This should be used typically during error conditions or any condition where one side wants to signal the other side for attention. However, the user software is at liberty to use this for any purpose.
DATA_1_WORD	0100	1 word data packet (16 bits of data)
DATA_2_WORD	0101	2 word data packet (32 bits of data)
DATA_4_WORD	0110	4 word data packet (64 bits of data)
DATA_6_WORD	0111	6 word data packet (96 bits of data)
DATA_N_WORD	0011	N(1-16) word data packet where software has programmed the number of the data words in a designated register. Both transmitter and receiver modules should have the same value programmed.
Reserved	0001, 0010, and 1000-1110	Reserved

31.3.4.2.1 Ping Frames

Ping frames are one of the most basic frames that can be generated by the FSI. Table 31-6 shows the structure of the ping frames.

Table 31-6. Ping Frame

Idle State	Preamble	SOF	Frame Type	Frame Tag	EOF	Postamble	Idle State
	1111	1001	0000	xxxx	0110	1111	

The ping frame type is always 0000. The frame tag is defined by the application. Separate frame tags exist for timer and software initiated ping frames. No data or CRC is transmitted in a ping frame.

The main purpose of the ping frame is to periodically send a notification to the receiver to ensure an active connection between the transmitter and receiver. The transmitter and receiver cores implement different features to allow the ping frame to operate as a line break detect feature.

On the transmitter, the ping frame is the only frame which can be set up and transmitted without any further software or DMA intervention. Ping frames can be transmitted by any (or all) of the three sources: automatic ping timer, software, or external triggers. See Section 31.3.2.3.3 for information on how the transmitter configures and sends the ping frames.

The receiver has a ping watchdog that can detect if a ping frame has not been received in a predetermined window. This allows the receiver to know if the connection between it and the transmitter has been broken. See Section 31.3.3.4 for information on how the receiver handles ping frames.

31.3.4.2.2 Error Frames

Error frames are similar to ping frames in that there are no data fields transmitted. Despite the naming of this frame as an “error frame,” the usage of it is up to the application, as no restrictions are placed on how and when this type of frame is transmitted. Table 31-7 shows the structure of an error frame.

Table 31-7. Error Frame

Idle State	Preamble	SOF	Frame Type	Frame Tag	EOF	Postamble	Idle State
	1111	1001	1111	xxxx	0110	1111	

The structure of the error frame is the same as a ping frame. No data or CRC values are transmitted. The frame type is 1111 for all error frames, and the frame tag is defined by software in the TX_FRAME_TAG_UDATA register.

The receiver can detect if an error frame has been received based on the frame type field. Because of this, the receiver can read the incoming frame tag from the RX_FRAME_TAG_UDATA register and act on up to 16 different conditions.

31.3.4.2.3 Data Frames

Data frames are the most complex frames. As the name indicates, these frames are used to transfer data. Table 31-8 shows the general structure of data frames.

Table 31-8. Data Frame

Idle State	Preamble	SOF	Frame Type	User Data	Data Words	CRC Byte	Frame Tag	EOF	Postamble	Idle State
	1111	1001	0xxx	xxxx xxxx	1-16 words	xxxx xxxx	xxxx	0110	1111	

The frame type field will reflect the 4-bit code of the frame type. A list of frame types can be seen in Table 31-5. The number of the data words transmitted will be determined by the frame type chosen.

There are four fixed-length data frames supported by the frame type: 1 word, 2 words, 4 words, and 6 words.

Additionally, there is a user-defined data length frame type where the number of data words is fixed by software. Anywhere from 1 to 16 words can be transmitted in this frame type. This length must be configured in the N_WORDS field of the transmitter’s TX_FRAME_CTRL register and receiver’s RX_OPER_CTRL register.

31.3.4.3 Multi-Lane Transmission

The FSI is capable of transmitting and receiving data on two parallel data lines. When enabled, data bits will be split between the data lines while the start of frame, frame type, frame tag, and end of frame fields will be identical and complete on each line. The user data, data, and CRC fields will be split between the data lines. Starting with the most significant bit, the odd-numbered bits appear on D0 and even-numbered bits appear on D1.

In the following example, assume the following:

8-bit user data: u7u6u5u4u3u2u1u0

16-bit data: d15d14d13d12…d1d0

8-bit CRC: c7c6c5c4c3c2c1c0

Table 31-9. Multi-Lane Frame Format

Idle State	Preamble	SOF	Frame Type	User Data	Data Words	CRC Byte	Frame Tag	EOF	Postamble	Idle State
TXD0	1111	1001	0011	$u^7u^5u^3u^1$	$d^{15}d^{13}…d^1$	$c^7c^5c^3c^1$	xxxx	0110	1111	
TXD1	1111	1001	0011	$u^6u^4u^2u^0$	$d^{14}d^{12}…d^0$	$c^6c^4c^2c^0$	xxxx	0110	1111	

31.3.5 Flush Sequence

Every time there is a soft reset of the receiver, the receiver requires a flush sequence from the transmitter before it can receive and decode frames. The receiver core has an asynchronous reset mechanism which will allow the receive module to be reset even in the absence of the receive clocks. However, due to the design, this reset will be released synchronous to the receive clock (RXCLK). Thus, the receiver will require five full clock pulses to be able to come out of reset. Sending the flush pattern will ensure that these clock edges are received and any subsequent frames sent to the receiver will be correctly interpreted.

The flush sequence consists of a single toggle on both of the data lines as well as five consecutive pulses on the clock line.

If the FSI receiver is receiving data from a standard SPI, a data word of 0xFFFF from the SPI will have the same effect as a flush sequence.

Figure 31-9 shows a sample plot of the flush sequence.

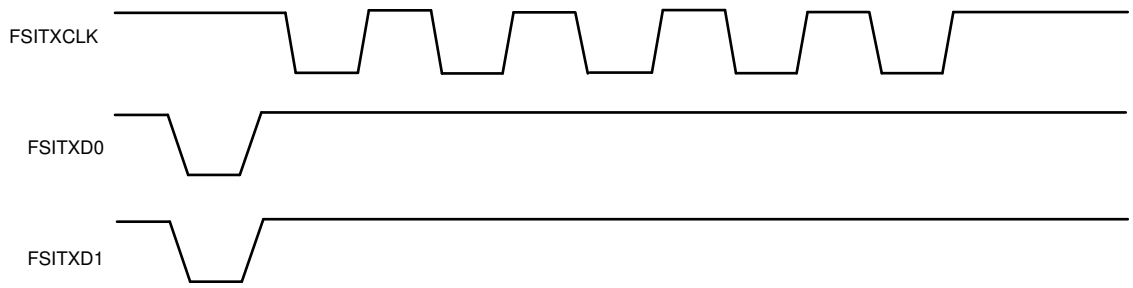


Figure 31-9. Flush Sequence Signals

31.3.6 Internal Loopback

The transmitter and receiver cores can be connected together internally to allow for development and debug. This is achieved by setting RX_MASTER_CTRL.INT_LOOPBACK to 1. Internal loopback routes the signals from the corresponding transmitter to the appropriate receiver pin. No configuration needs to be done in the transmitter.

Figure 31-10 shows the signal connections with internal loopback.

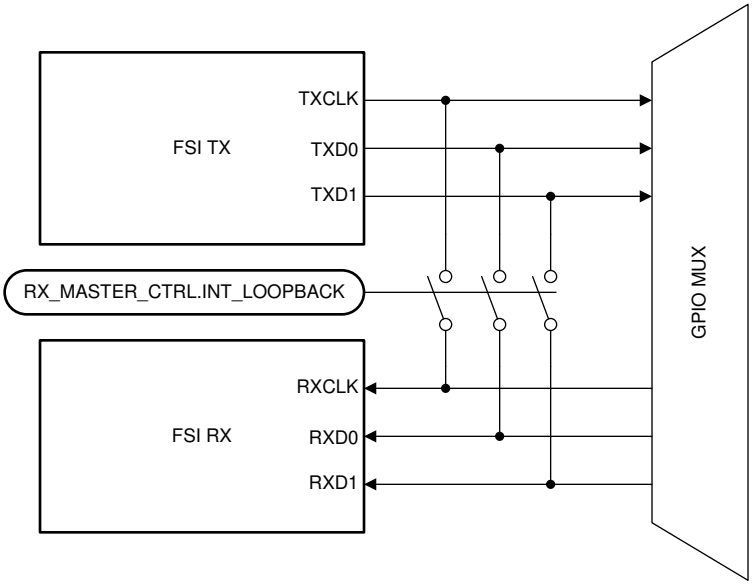


Figure 31-10. FSI with Internal Loopback

31.3.7 CRC Generation

The FSI uses CRC-8 with the polynomial 0x07 for the internal hardware CRC generation. This polynomial is also represented as x^8+x^2+x+1.

For example, for a 2-word data packet the following calculation would occur:

Data-1 = 0x4433

Data-0 = 0x2211

User Data = 0xAA

The CRC would be computed with the bytes being taken in the following order (first to last):

0xAA – Byte 0, User Data

0x11 – Byte 1, Data-0, Least-significant byte

0x22 – Byte 2, Data-0, Most-significant byte

0x33 – Byte 3, Data-1, Least-significant byte

0x44 – Byte 4, Data-1, Most-significant byte

31.3.8 ECC Module

The FSI module comes with a 16-bit or 32-bit ECC computation module in both the transmitter and receiver. Use of this module is optional.

Note that the ECC is independent and unrelated to the hardware CRC computation module present in both the transmitter and receiver cores.

The following example shows a scenario in which the application requires ECC be calculated and transmitted on a 2-word data frame.

In the FSITX module:

1. Configure the ECC module for 32-bit data by setting TX_OPER_CTRL_HI.ECC_SEL to 1.
2. Write the data to the TX_ECC_DATA register as well as the transmit buffer.
3. Read TX_ECC_VAL Register. This register contains the 8-bit ECC value calculated on the data.
4. Copy the 8-bit data from TX_ECC_VAL to TX_FRAME_TAG_UDATA.USER_DATA.
5. Set the Start Condition to begin the transmission.

The reverse process is followed on the FSIRX module. Once the data frame is received, user software can do the following:

1. Copy the data from the receive buffer to the RX_ECC_DATA register.
2. Copy the received user data that contains the transmitted ECC value from RX_FRAME_TAG_UDATA.USER_DATA to the RX_ECC_VAL register.
3. Read the RX_ECC_LOG register. This contains the result of the ECC computation using the RX_ECC_DATA and RX_ECC_VAL registers.
   a. If no ECC errors were detected, RX_ECC_LOG will be 0. The correct data will be available in RX_ECC_SEC_DATA.
   b. If a single bit error was detected, RX_ECC_LOG.SBE will be 1. The autocorrected data will be available in RX_ECC_SEC_DATA.
   c. If multiple bit errors occurred, RX_ECC_LOG.MBE will be set to 1. The data in RX_ECC_SEC_DATA is invalid and should not be used.

Using a 2-word data frame plus using the user data for the ECC is one possible implementation for ECC detection. Another option is to use a larger data frame a allocate one of the data words to be the ECC value.

31.3.9 Tag Matching

The FSI receiver core has the capability of generating an interrupt when the received data or ping frame's tag matches the reference tag in RX_FRAME_TAG_CMP or RX_PING_TAG_CMP, respectively.

Each tag compare register allows the user to not only select a reference tag (TAG_REF) but also set up a mask (TAG_MASK) to ignore specified bits in the reference tag. In order for tag matching to be enabled, the CMP_EN bit must be set. When the tag compare is enabled and a successful match occurs, the PING_TAG_MATCH, the DATA_TAG_MATCH or the ERROR_TAG_MATCH in RX_EVT_ERR_STATUS will be set. The corresponding match register status can be cleared by writing to RX_EVT_ERR_CLEAR. Also, if it is required to set the match register status in software, the user must do so by writing to the RX_EVT_ERR_SET register.

The tag matching scheme is NOT a filtering scheme. Tag matching is only a notification scheme to alert the user when a specific tag is detected in a data or ping frame. Both RX_INTR_EVT_CTRL_1 and RX_INTR_EVT_CTRL_2 interrupts can be set up to generate an interrupt when a tag match event occurs.

Another feature used when tag matching is enabled, is the broadcast feature. The broadcast feature can be enabled by setting the BRDCST_EN in RX_FRAME_TAG_CMP or RX_PING_TAG_CMP. When broadcast mode is enabled, the third bit of the tag will be treated as a broadcast bit. If the received tag has it's third bit set, then it will be treated as a match regardless of the other tag bit comparisons. It is important to note that, a match caused by TAG_REF and TAG_MASK will also be considered a match.

It is important to reiterate that the tag matching scheme is not a filtering scheme. For example, if tag matching is enabled and a frame is received with a non matching tag, the frame is still saved in the buffer and the corresponding event status bits (FRAME_DONE, DATA_FRAME_RCVD etc.) will be set.

Tag matching should be used in TDM configuration described in Section 31.3.11. However it can be used in single slave configurations as needed.

31.3.10 User Data Filtering (UDATA Matching)

The FSI receiver core has the capability of filtering data frames and only receive packets when received data's UDATA (user data) matches the reference UDATA in RX_UDATA_FILTER.

UDATA filter compare register allows the user to not only select a reference UDATA (UDATA_REF) but also set up a mask (UDATA_MASK) to ignore specified bits in the reference user data. In order for user data filtering to be enabled, the RX_MASTER_CTRL.DATA_FILTER_EN bit must be set. When the user data filtering is enabled, only a successful match would allow the packet to be received in the FSIRX circular buffer, all non-matching packets are ignored.

The user data matching scheme **is a filtering scheme**.

User data filtering should be used in multi-slave configuration as described in Section 31.3.11; however, it can be used in single-slave configurations as needed.

31.3.11 TDM Configurations

The FSI module in this device supports multi-slave configurations, whereas a single master can control multiple slave devices.

In order to use the FSI module in the multi-slave configuration described, the slave device must utilize tag matching and user data filtering.

This multi-slave configuration is supported in the FSI module through time-division multiplexing. When TDM is enabled, each slave must also have tag matching enabled. Figure 31-11 shows a scheme where a single master is communicating with multiple slaves. All FSI receive modules in the slave devices are directly connected to the master device's transmit module. The transmit modules of the slave devices are chained serially such that each transmit module is connected to the next slave device and the last slave device's output connects to the master device's receive module. Each slave device will decide, based on the received frame's tag, whether to transmit their own data, or to enter bypass mode where they directly connect the previous slave device's transmit module to the next slave device. This is done by using the FSI transmit module's TDM_IN.

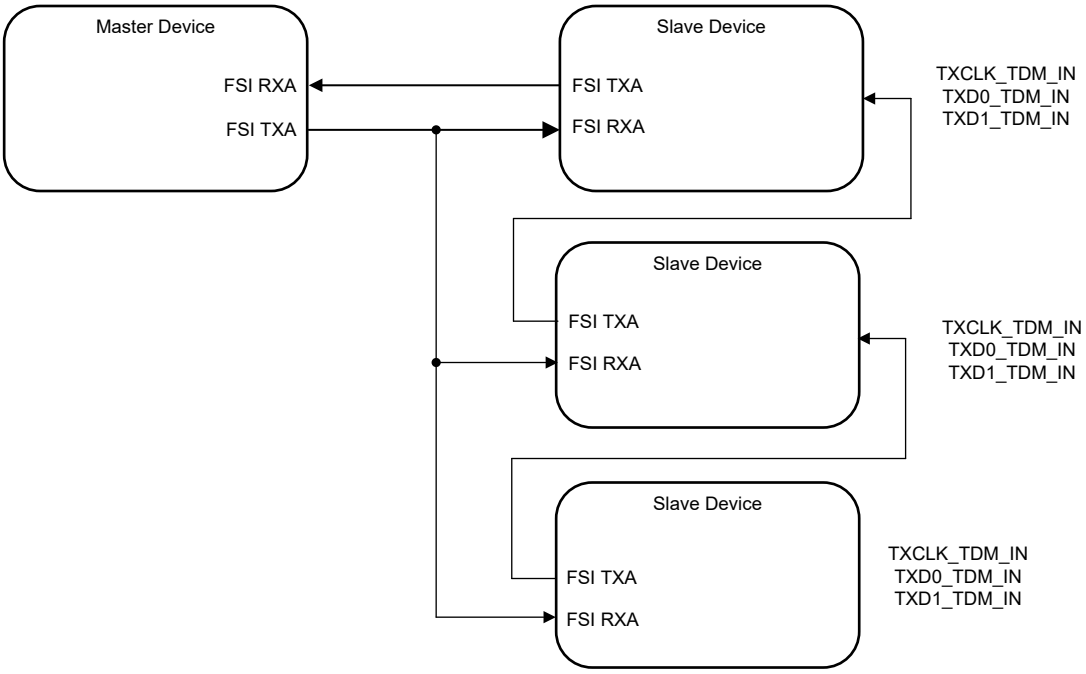


Figure 31-11. FSI Multi-Slave Configuration

When an FSI transmitter module is used in TDM mode, TXCLK_TDM_IN, TXD0_TDM_IN and TXD1_TDM_IN pins are used if the transmitter is required to enter bypass mode. Figure 31-12 shows how the FSI module operates when in multi-slave TDM mode.

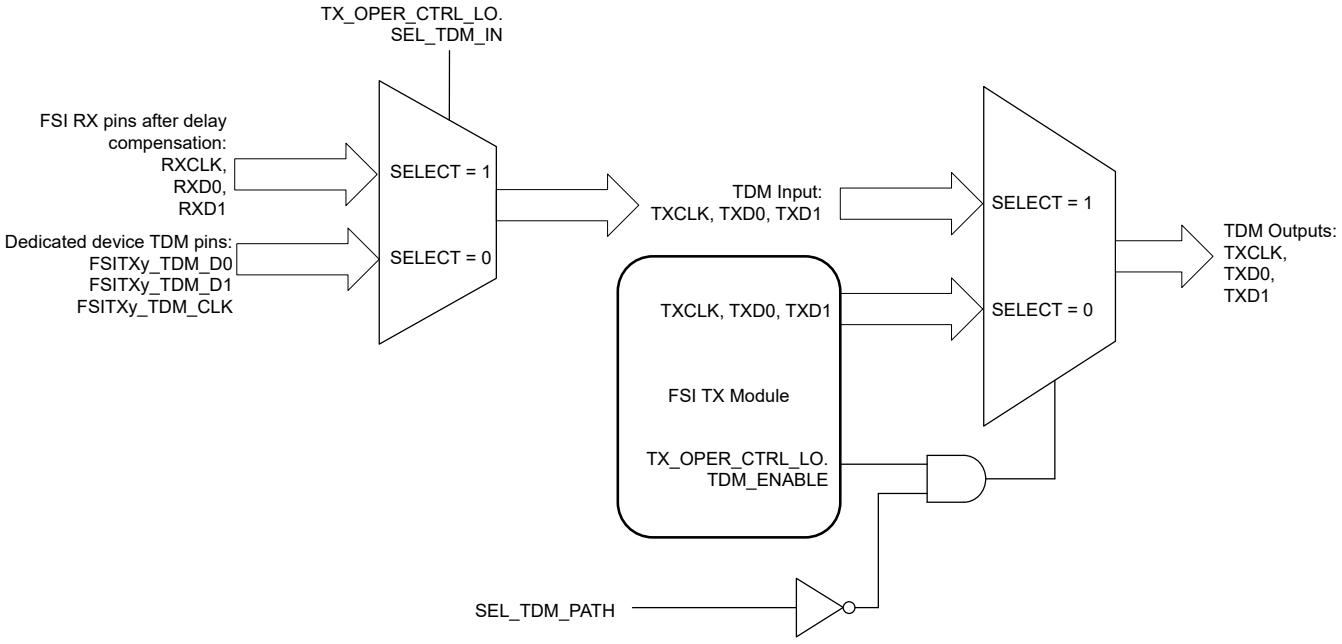


Figure 31-12. FSI Transmitter Multi-Slave Multiplexing

The SEL_TDM_PATH signal is sourced from the RX_TRIG0 signal (after being stretched by the programmable width module). The RX_TRIGx signal will also generate the transmit trigger for the FSI transmitter. The RX_TRIGx signal must be configured to decide when to generate the FSI transmit trigger based on the status of the data, ping, and frame tag match generated by the FSI receiver module. The FSITX module must be configured to transmit on an external trigger and the corresponding RX_TRIGx trigger input must be selected. In a broadcast scenario (FSI tag match will notify all slave devices that a match has occurred), the RX_TRIG0 signal inside each slave will generate a trigger and SEL_TDM_PATH signal. The main key here is that the trigger and the SEL_TDM_PATH signal must be generated at a different time interval in a non-overlapping manner. Figure 31-13 shows an example of FSI transmit triggers and the multi-slave SEL_TDM_PATH signals generated by the RX_TRIGx signal or the CLB module of the slave devices in a broadcast scenario.

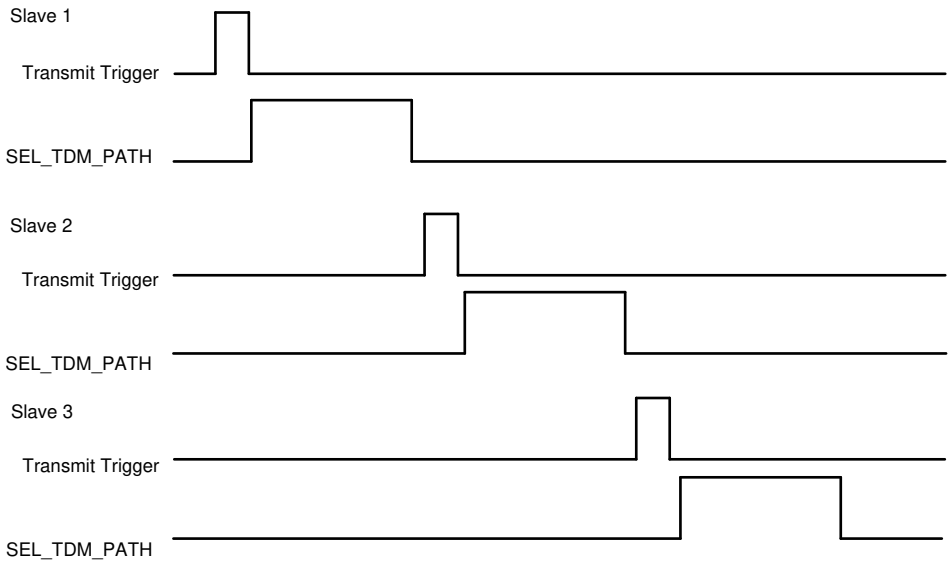


Figure 31-13. Generated Signals for FSI Multi-Slave Configuration

When using TDM mode, the user has the option of using the device dedicated TDM_IN pins or FSIRX pins. When the FSITX module is put in bypass mode, the TX_OPER_CTRL.SEL_TDM_IN bit determines which set of signals, FSIRX input signals or TDM_IN signals, is internally connected to the FSITX output signals.

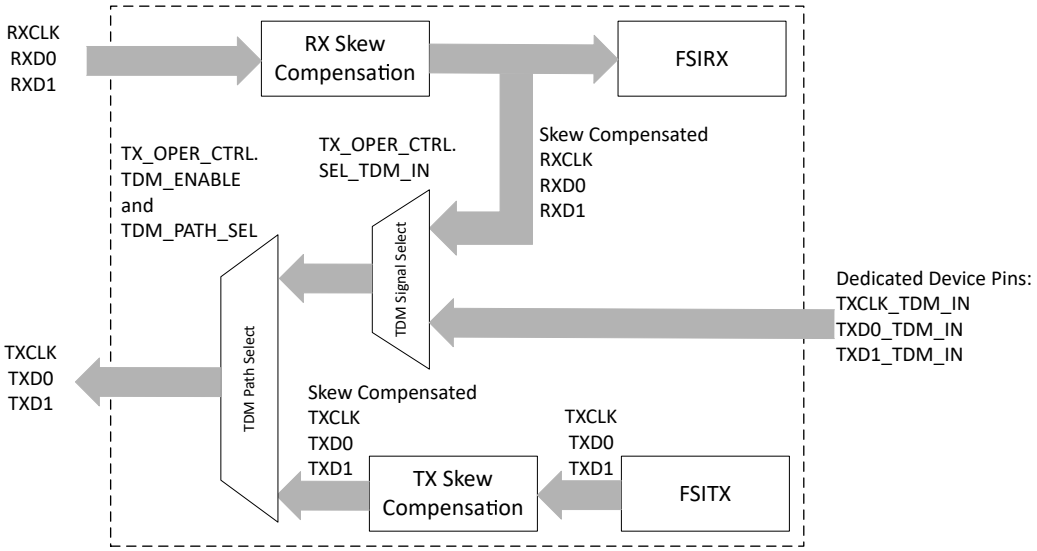


Figure 31-14. TDM Signal Selection

31.3.12 FSI Trigger Generation

The RX_TRIGx external trigger can be used to initiate FSITX transmission. RX_TRIG0 must be used if TDM mode (multi-slave configuration) is required. RX_TRIG0 must be used as the trigger source for start of transmission while the programmable stretch width RX_TRIG0 signal is used as the SEL_TDM_PATH signal (which decides whether the local FSITX is active or put in bypass mode).

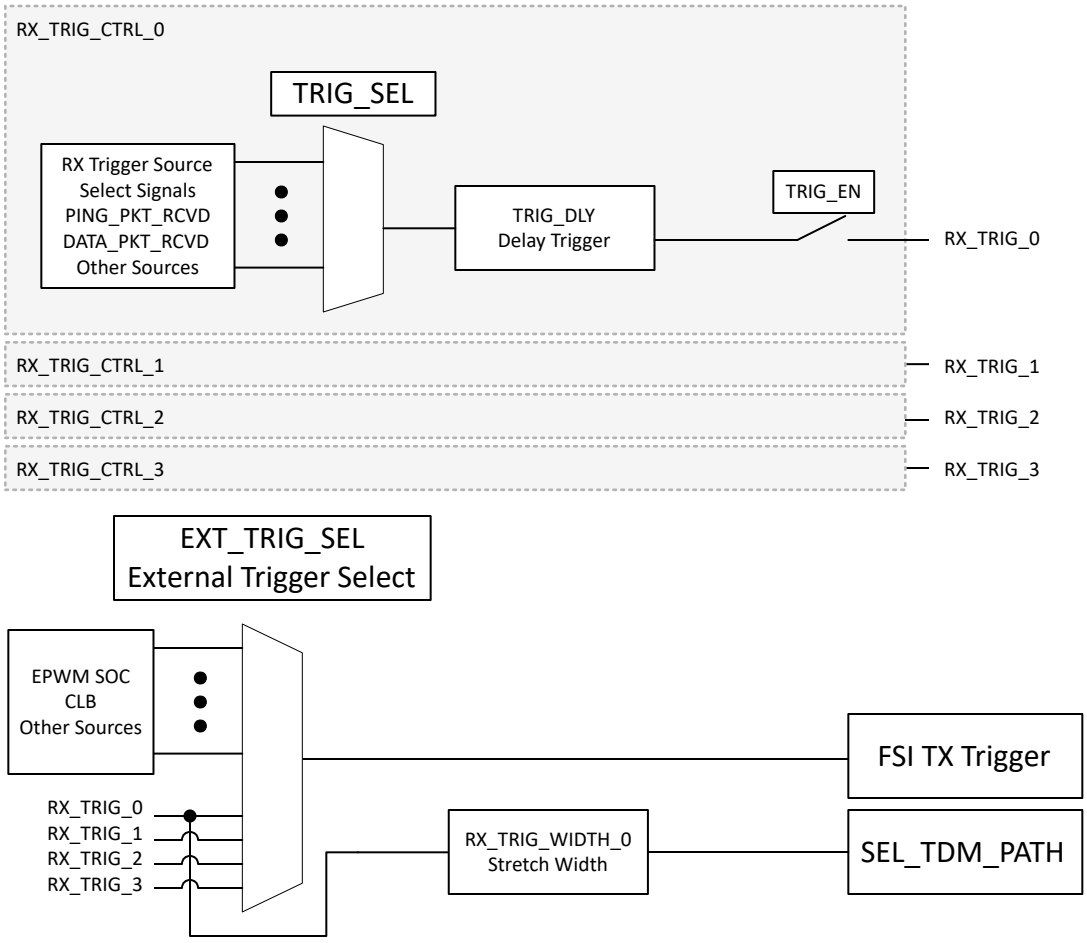


Figure 31-15. RX_TRIGx FSI Trigger

The signal source for the RX_TRIGx signal is selected through the RX_TRIG_CTRL_x.TRIG_SEL bits, as listed in Table 31-10.

Table 31-10. RX_TRIGx Trigger Select Signals

RX_TRIG_CTRLx.TRIG_SEL	Selected Signal
0	Ping Packet Received
1	Data Packet Received
2	Error Packet Received
3	Ping Frame Tag Match Occured
4	Data Frame Tag Match Occured
5	Error Frame Tag Match Occured
6	Frame Done
7	Reserved
8 to 15	Reserved

The RX_TRIGx signals can optionally be delayed (this can be used in TDM scenarios) through the RX_TRIG_CTRL_x.TRIG_DLY.

31.3.13 FSI-SPI Compatibility Mode

The FSI supports a SPI compatibility mode. While the FSI can communicate with a standard SPI module, the FSI supports a limited configuration. The features of this compatibility mode follow:

- Data will transmit on rising edge and receive on falling edge of the clock.
- Only 16-bit word size is supported.
- TXD1 will be driven like an active-low, chip-select signal. The signal will be low for the duration for the full frame transmission.
- No receiver chip-select input is required. RXD1 is not used. Data is shifted into the receiver on every active clock edge.
- No preamble or postamble clocks will be transmitted. All signals return to the IDLE state after the frame phase is finished.
- It is not possible to transmit in the SPI slave configuration because the FSI TXCLK cannot take an external clock source.

Table 31-11 lists the frame structure of the FSI-SPI compatibility mode. Each frame phase is present in this mode. If the FSI is transmitting to a standard SPI module, the SPI must decode the frame structure. Similarly, if the FSI is configured as a SPI slave, the standard SPI must encode the transmission to be sent.

Table 31-11. FSI-SPI Compatibility Frame Structure

Idle State	Start of Frame	Frame Type	User Data	Data Words	CRC byte(1)	Frame Tag	End of Frame	Idle State
	1001	4 bits	8 bits	1-16 words	8 bits	4 bits	0110	

(1) The CRC byte is present only in data frames.

Because of the requirement that the standard SPI module encodes the various frame data, this limits the type of modules that can be connected to the FSI in SPI mode. The paired SPI module must have enough functionality to encode and decode the frames.

If the FSI is transmitted to a standard 16-bit SPI, the data would be arranged in the following manner. The example provided in Table 31-12 assumes a DATA_2_WORD frame has been sent.

Table 31-12. Contents of Data Received by a Standard SPI

SPI Data	Data Contents
SPI word 0	1001, 0100, 8-bit User Data
SPI word 1	Data word 1
SPI word 2	Data word 2
SPI word 3	8-bit CRC, 4-bit Frame Tag, 0110

31.3.13.1 Available SPI Modes

There are a few wiring schemes available for the FSI to use when communicating with an SPI module.

31.3.13.1.1 FSITX as SPI Master, Transmit Only

The FSITX can operate as an independent SPI Master module. In this condition, TXCLK will be connected to SPICLK, TXD0 will be connected to SPISIMO, and TXD1 will be connected to $\overline{\text{SPISTE}}$, the chip select.

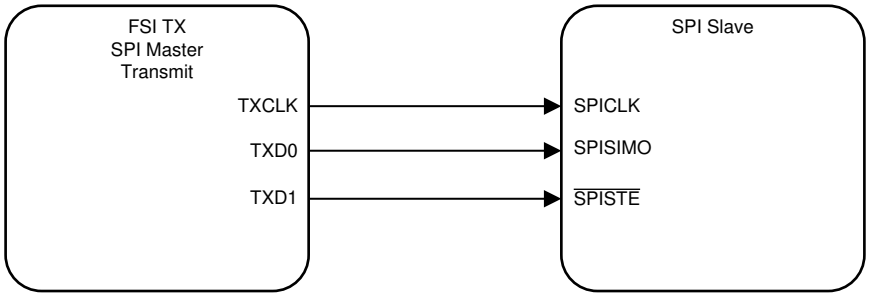


Figure 31-16. FSITX as SPI Master, Transmit Only

When the FSI is a SPI transmitter, the application has the ability to check for frame errors, line breaks, CRC errors, and ECC checks on data. These are all encoded by hardware in every FSI frame. The SPI receiver will require some software to act upon this information.

Table 31-13. FSI as Master Transmitter, SPI as Slave Receiver

Capability	Availability	Comment
Framing checks on the data frames	Yes	Can be implemented in software on the SPI receiver.
Ability to detect line breaks	Yes	Can be implemented in software on the SPI receiver but will require additional software overhead such as a timer, or watchdog.
CRC check	Yes	Can be implemented in software on the SPI receiver. For devices that have VCRC , this will be more efficient.
ECC on data	Yes	Can be implemented in software in the SPI receiver
Detection of abruptly terminated frames	No	
Double edge data rate	No	
Recovery from glitches on signal lines between frames	No	
Skew adjustment on signal lines	No	

31.3.13.1.1.1 Initialization

To configure the FSITX module to be a SPI master for transmit only, proceed through the standard FSITX initialization procedure. Before releasing the FSITX from reset, set TX_OPER_CTRL_LO.SPI_MODE to 1. This will enable the SPI clocking scheme and signaling structure.

31.3.13.1.1.2 Operation

The operation of the FSITX module in FSI-SPI Compatibility mode is the same as if the module is in standard FSI mode. The application can utilize the frame timer, ping frames, external frame triggers, and so on. Refer to Section 31.3.2 for more information on each of these features.

31.3.13.1.2 FSIRX as SPI Slave, Receive Only

The FSIRX can operate as an independent SPI slave module. In the usage, RXCLK will be connected to SPICLK, and RXD0 will be connected to SPISIMO. RXD1 is unused. There is no requirement for a chip select signal to be used when connected to the FSIRX. This is because the FSIRX will respond to any incoming clock edge. If there is any noise or unwanted clock transitions, a flush sequence will be required to resynchronize the FSIRX module with the Master.

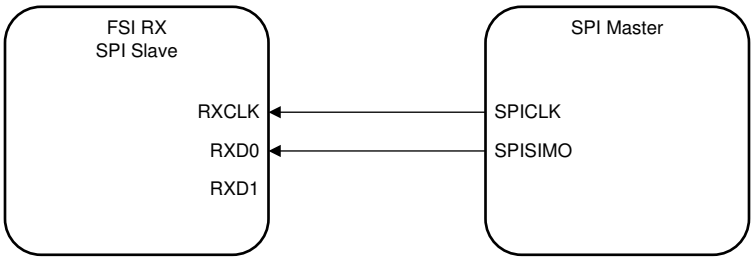


Figure 31-17. FSIRX as SPI Slave, Receive Only

When the FSI is a SPI receiver communicating with a SPI transmitter, the application has the ability to detect frame errors, line breaks, CRC errors, ECC checks on data, as well as abruptly terminated frames. Note that the FSI can handle all of this in hardware, but the SPI transmitter must encode the information into the data to be transmitted.

Table 31-14. SPI as Master Transmitter, FSI as Slave Receiver

Capability	Availability	Comment
Framing checks on the data frames	Yes	Standard on FSI
Ability to detect line breaks	Yes	Can be implemented in software in the SPI transmitter. But will have to use some timer or watchdog in the transmitting SPI device
CRC check	Yes	Can be implemented in software in the SPI transmitter.
ECC on data	Yes	Can be implemented in software in the SPI transmitter.
Detection of abruptly terminated frames	Yes	This is accomplished with the FSI setting up the frame watchdog counter.
Double edge data rate	No	
Recovery from glitches on signal lines between frames	Yes	Whenever glitches occur on either the clock or data lines in between transmissions, the initial flush pattern of a frame will discard the effects of these glitches and will cause the receiver to resynchronize when the real “start-of-frame” pattern is seen. So, the ability to reject glitches in between frames is hence very high.
Skew adjustment on signal lines	Yes	The FSI receiver has the ability to add delays to the incoming signal lines.

31.3.13.1.2.1 Initialization

To configure the FSIRX module to be a SPI slave for receiving only, proceed through the standard FSIRX initialization procedure. Before releasing the FSIRX from reset, set RX_OPER_CTRL.SPI_MODE to 1. This will enable the SPI clocking scheme and signaling structure.

31.3.13.1.2.2 Operation

The operation of the FSIRX module in FSI-SPI compatibility mode is the same as if the module is in standard FSI mode. The application can utilize the Frame and Ping Watchdogs, CRC and ECC checks, and so on. Refer to Section 31.3.3 for more information on each of these features.

31.3.13.1.3 FSITX and FSIRX Emulating a Full Duplex SPI Master

In this configuration, the FSITX is the clock master. The FSITX module will drive TXCLK (SPICLK), TXD0 (SPISIMO), and TXD1 (SPISTE/chip select), to the SPI slave. The SPISOMI signal will be connected back to the RXD0 signal. RXCLK can be fed either internally using the internal SPI pairing feature or wired externally, depending on the application requirements. Since the FSITX and RX modules are independent, the FSIRX could also be thought of as an additional SPI slave. Some software logic will be required in order for the FSI to emulate a SPI master fully.

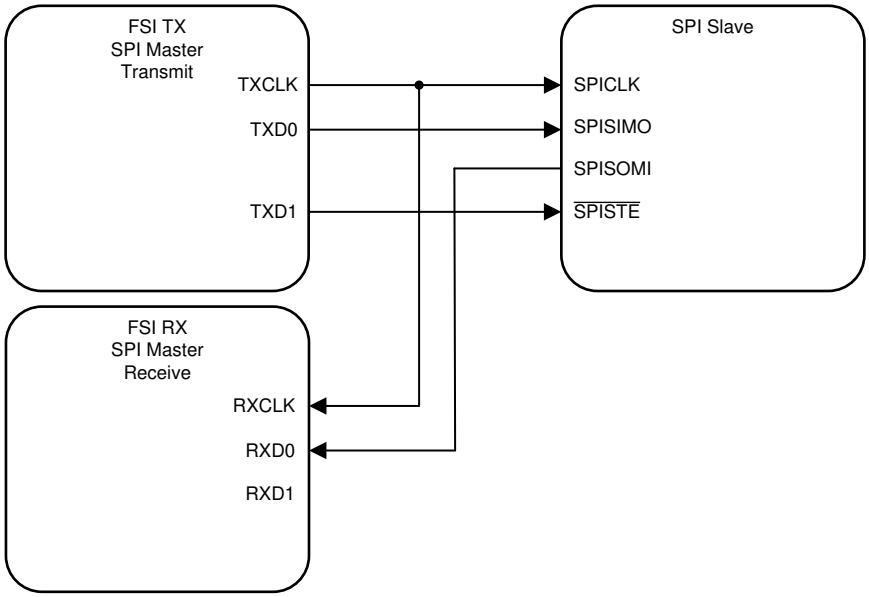


Figure 31-18. FSITX and FSIRX as SPI Master, Full Duplex

31.3.13.1.3.1 Initialization

To configure both FSITX and RX modules for full duplex SPI master operation, follow the initialization instructions for each module described in the preceding sections. Both FSITX and RX modules must set their respective SPI_MODE bits. This will enabled the SPI clocking scheme and signaling structures.

If internal clock loopback is desired, the FSIRX module must also set RX_MASTER_CTRL.SPI_PAIRING to 1. This will internally connect TXCLK to RXCLK. If using internal clock loopback, the GPIO used for RXCLK may be reallocated to other application requirements.

If the application requires an external clock loopback, ensure that TXCLK is connected to RXCLK. This may be required if the SPI Slave is across an isolation barrier and there is latency between TXCLK being launched and SPISOMI data being received on RXD0.

31.3.13.1.3.2 Operation

In this mode of operation, some higher level software must be written to emulate a full SPI master module. There is no path for the transmit module to determine what the receive module received. Both the TX and RX modules are still able to utilize the various other features available such as the ping frame timer, ping frame and frame watchdogs, CRC and ECC error checkers, and so on. The procedure for configuring these features is described elsewhere in this document.

31.4 FSI Programing Guide

This section describes various operational sequences and features for the FSI.

31.4.1 Establishing the Communication Link

Once the transmitter and receiver modules have been configured, some synchronization must occur before the modules may exchange data. Since the receiver will accept data on any clock transition, the receiver core logic must be flushed to properly interpret the start of a new, valid frame. This is especially true when the FSI modules reside on separate devices and are possibly isolated.

The following example provides a suggested approach for establishing a clean communication link on two separate devices that may power up in an arbitrary order. Note that this is only a sample synchronization. Depending on application requirements, a different approach can be followed. The single, most important aspect of synchronization is to ensure that the receiver is properly flushed and ready to receive a complete frame without error. How to achieve this is up to the application.

Figure 31-19 shows the connection of the devices in this example. While there is no true concept of a master or a slave node in the FSI protocol, the example uses this nomenclature as a simple way to describe the data flow.

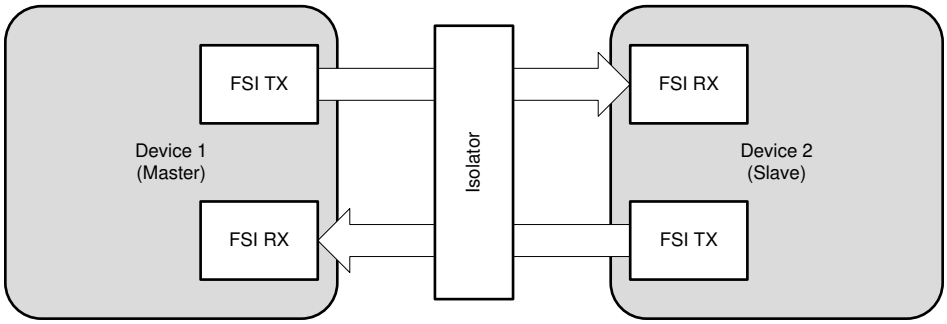


Figure 31-19. Point to Point Connection

Device 1 is the master node; it will be the driver of the initialization sequence. Device 2 is the slave node; it will respond to the master's commands. In this example, as well as in a real world use-case, neither the master nor the slave may know precisely when the other is ready to receive communication.

Sample sequences for both the master device and slave device are provided in the following subsections.

31.4.1.1 Establishing the Communication Link from the Master

The following sequence is an example of how the master node can establish the communication link with the slave without external signals outside of the standard communication link.

1. Assert the core reset to both the FSITX and FSIRX modules, and then deassert the resets.
2. Configure the transmitter and receiver for desired operation.
3. Set up the receiver interrupts to detect an incoming transmission.
4. Begin the ping loop:
   - Send the flush sequence.
   - Send a ping frame with the frame tag 0000.
   - Wait for some time. (determined by application)
   - If the FSIRX has received a valid ping frame, continue; else iterate the loop again.
   - If the received ping frame tag was 0001, continue; else iterate the loop again.
5. Send a ping frame with the frame tag 0001.

At this point both the master transmit and receive channels have successfully received a frame from their slave counterparts. The link has been established and standard application communication may begin.

31.4.1.2 Establishing the Communication Link from the Slave

The following sequence is an example of how the slave node can establish the communication link with the master without external signals outside of the standard communication link.

1. Apply the core reset to both the FSITX and FSIRX modules, and then release the reset.
2. Configure the transmitter and receiver for desired operation.
3. Set up the receiver interrupts to detect an incoming transmission.
4. Wait for a receiver interrupt.
5. If the FSIRX has received a valid ping frame, continue; else return to step 4.
6. If the received frame tag was 0000, continue; else discard the transmission and return to step 4.
7. Send the flush sequence.
8. Send a ping frame with the frame tag 0001.
9. Wait for a receiver interrupt.
10. If the FSIRX has received a valid ping frame, continue; else return to step 4.
11. If the received ping frame tag was 0001, continue; else if the received frame tag was 0000, return to step 9. This can happen if a second ping frame was already in transit before receiving the slave's response in step 8.

At this point, both the transmit and receive modules have successfully received ping frames from their master counterparts. The link has been established and regular communication may now proceed. The application may configure periodic ping frames from the transmitter, initialize the receiver's ping and frame watchdogs and begin the communication required by the application.

31.4.2 Register Protection

Both the FSITX and FSIRX modules contain control registers that have embedded write protection. This is accomplished through EALLOW, register keys, and a master register lock. These protections ensure that no spurious writes or unintentional modifications to these registers are accepted. For the list of registers with write protections available and the register and bit descriptions, refer to Section 31.6.

EALLOW Protection

EALLOW is a device-level register protection, refer to Section 3.1.3 for more information on EALLOW. For those registers with EALLOW protection, the EALLOW bit should be set before modifying the register. The application should then clear the EALLOW bit to re-enable the write protection when access to EALLOW-protected registers are complete.

Register Key Protection

In addition to EALLOW, some bits in the FSI registers are protected by a key. In order to write to these bits, the key must be written at the same time. For example, to put the transmitter core into reset, TX_MASTER_CTRL.CORE_RST must be set. To do this, write 0xA501 to TX_MASTER_CTRL, where 0xA500 is the KEY value, and 0x0001 is the CORE_RESET bit. Refer to the register descriptions for more information on which registers have write keys added.

Control Register Lock Protection

There also exists a master lock to prevent any modifications to the control registers. There is an independent lock for each FSI module. For the list of registers that are protected by this control register lock, refer to the register descriptions section. The control register lock will prevent any writes to the control registers until the lock is released. To set the control register lock, write 0xA501 to RX_LOCK_CTRL and TX_LOCK_CTRL for the receiver and transmitter respectively.

The control register lock cannot be disabled by the application until a SYSRSn has been asserted. This can occur at the device level, or by writing to the appropriate peripheral soft reset register (DEV_CFG_REGS.SOFTPRESx) for the FSI module. Refer to Section 31.3.2.7 for more information on SYSRSn.

31.4.3 Emulation Mode

There is no specific emulation mode or configuration supported. The FSI cores will always be in free running mode. CPU halts will not have any effect on the operation of the FSI. However, reads of registers and data buffers by the debugger will not affect any flags, or status of the data buffers.

If you want to stop the operation of either FSI module when the debugger halts, the following steps are required:

1. Set the debugger to real-time emulation mode.
2. Mark the FSI interrupt group as a time-critical interrupt. That is, enable the corresponding bit in the DBGIER register.
3. The ISR can check the DSTAT register and to determine if the ISR was called when the debugger was halted.
4. FSI operations can be disabled and the ISR can branch to a debug-specific halt location.

31.5 Software

31.5.1 FSI Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/fsi

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

31.5.1.1 FSI Loopback:CPU Control

FILE: fsi_ex1_loopback_cpucontrol.c

Example sets up infinite data frame transfers where trigger happens through *CPU*. Automatic (hardware triggered) Ping frame transmission is also setup along with data.

User can edit some of configuration parameters as per use case. These are as below. Default values can be referred in code where these globals are defined

- *nWords* - Number of words per transfer may be from 1 -16
- *nLanes* - Choice to select single or double lane for frame transfers
- *fsiClock* - FSI Clock used for transfers
- *txUserData* - User data to be sent with Data frame
- *txDataFrameTag* - Frame tag used for Data transfers
- *txPingFrameTag* - Frame tag used for Ping transfers
- *txPingTimeRefCntr* - Tx Ping timer reference counter
- *rxWdTimeoutRefCntr* - Rx Watchdog timeout reference counter

For any errors during transfers, that is, *error* events such as Frame Overrun, Underrun, Watchdog timeout and CRC/EOF/TYPE errors, execution will stop immediately and status variables can be looked into for more details. Execution will also stop for any mismatch between received data and sent ones and also if transfers takes unusually long time(detected through software counters - txTimeOutCntr and rxTimeOutCntr)

External Connections

For FSI internal loopback (EXTERNAL_FSI_ENABLE == 0), no external connections needed

For FSI external loopback (EXTERNAL_FSI_ENABLE == 1), external connections are required. The FSI TX pins should be connected to the FSI RX pins of the same device. See below for external connections to include and GPIOs used:

External Connections Required between FSI TX and RX of the same device:
- FSIRX_CLK to FSITX_CLK
- FSIRX_RX0 to FSITX_TX0
- FSIRX_RX1 to FSITX_TX1

ControlCard FSI Header GPIOs:
- GPIO_27 -> FSITX_CLK
- GPIO_26 -> FSITX_TX0
- GPIO_25 -> FSITX_TX1
- GPIO_13 -> FSIRX_CLK
- GPIO_12 -> FSIRX_RX0
- GPIO_11 -> FSIRX_RX1

Watch Variables
- *dataFrameCntr* Number of Data frame transferred
- *error* Non zero for transmit/receive data mismatch

31.5.1.2 FSI Loopback CLA Control

FILE: fsi_ex2_loopback_clacontrol.c

Example sets up infinite data frame transfers where trigger happens through *CLA*. Automatic (hardware triggered) Ping frame transmission is also setup along with data. This example is similar to fsi_ex1_loopback_cpucontrol and only different in the sense that data frame transfer are triggered from a CLA task. Using CLA will release some of load from CPU and help it in providing time for other tasks.

User can edit some of configuration parameters as per use case. These are as below. Default values can be referred in code where these globals are defined.

- *nWords* - Number of words per transfer may be from 1 -16
- *nLanes* - Choice to select single or double lane for frame transfers
- *fsiClock* - FSI Clock used for transfers
- *txUserData* - User data to be sent with Data frame
- *txDataFrameTag* - Frame tag used for Data transfers
- *txPingFrameTag* - Frame tag used for Ping transfers
- *txPingTimeRefCntr* - Tx Ping timer reference counter
- *rxWdTimeoutRefCntr* - Rx Watchdog timeout reference counter

For any errors during transfers, that is, *error* events such as Frame Overrun, Underrun, Watchdog timeout and CRC/EOF/TYPE errors, execution will stop immediately and status variables can be looked into for more details. Execution will also stop for any mismatch between received data and sent ones and also if transfers takes unusually long time(detected through software counters - txTimeOutCntr and rxTimeOutCntr)

External Connections

For FSI internal loopback (EXTERNAL_FSI_ENABLE == 0), no external connections needed

For FSI external loopback (EXTERNAL_FSI_ENABLE == 1), external connections are required. The FSI TX pins should be connected to the respective FSI RX pins of the same device. See below for external connections to include and GPIOs used:

External Connections Required between FSI TX and RX of the same device:
- FSIRX_CLK to FSITX_CLK
- FSIRX_RX0 to FSITX_TX0
- FSIRX_RX1 to FSITX_TX1

ControlCard FSI Header GPIOs:
- GPIO_27 -> FSITX_CLK
- GPIO_26 -> FSITX_TX0
- GPIO_25 -> FSITX_TX1
- GPIO_13 -> FSIRX_CLK
- GPIO_12 -> FSIRX_RX0
- GPIO_11 -> FSIRX_RX1

Watch Variables
- *dataFrameCntr* Number of Data frame transferred
- *error* Non zero for transmit/receive data mismatch

31.5.1.3 FSI DMA Frame Transfers:DMA Control

FILE: fsi_ex3_loopback_dmacontrol.c

Example sets up infinite data frame transfers where DMA trigger happens once through CPU and then DMA takes control to transfer data iteratively. This example demonstrates the FSI feature about triggering DMA events which in turn can copy data and trigger next transfer.

Two DMA channels are setup for FSI Tx operation and two for Rx. Four areas in GSx memories are also setup as source and sink for data and tag values of frame under transmission.

Automatic (hardware triggered) Ping frame transmission is also setup along with data.

If there are any comparison failures during transfers or any of error event occurs, execution will stop.

External Connections

For FSI internal loopback (EXTERNAL_FSI_ENABLE == 0), no external connections needed

For FSI external loopback (EXTERNAL_FSI_ENABLE == 1), external connections are required. The FSI TX pins should be connected to the respective FSI RX pins of the same device. See below for external connections to include and GPIOs used:

External Connections Required between FSI TX and RX of the same device:
- FSIRX_CLK to FSITX_CLK
- FSIRX_RX0 to FSITX_TX0
- FSIRX_RX1 to FSITX_TX1

ControlCard FSI Header GPIOs:
- GPIO_27 -> FSITX_CLK
- GPIO_26 -> FSITX_TX0
- GPIO_25 -> FSITX_TX1
- GPIO_13 -> FSIRX_CLK
- GPIO_12 -> FSIRX_RX0
- GPIO_11 -> FSIRX_RX1

Watch Variables
- *countDMAtransfers* Number of Data frame transferred
- *error* Non zero for transmit/receive data mismatch

31.5.1.4 FSI Data Transfer by External Trigger

FILE: fsi_ex4_loopback_epwmtrigger.c

FSI frame transfer can be triggered by external sources. It can connect up to 32 trigger sources but as of now, only 16 ePWMx-SOCy(x-1:8, y-A:B) are supported. FSI supports external trigger for both PING and DATA frame transfers and in this example we demonstrate how to setup infinite DATA transfers using selectable ePWM-SOC as a trigger source. The TB counter for ePWM operation is in up/down count mode for this example.

Automatic (hardware triggered) Ping frame transmission is also setup along with data.

If there are any comparison failures during transfers or any of error event occurs, execution will stop.

External Connections

For FSI internal loopback (EXTERNAL_FSI_ENABLE == 0), no external connections needed

For FSI external loopback (EXTERNAL_FSI_ENABLE == 1), external connections are required. The FSI TX pins should be connected to the respective FSI RX pins of the same device. See below for external connections to include and GPIOs used:

External Connections Required between FSI TX and RX of the same device:
- FSIRX_CLK to FSITX_CLK
- FSIRX_RX0 to FSITX_TX0
- FSIRX_RX1 to FSITX_TX1

ControlCard FSI Header GPIOs:
- GPIO_27 -> FSITX_CLK
- GPIO_26 -> FSITX_TX0
- GPIO_25 -> FSITX_TX1
- GPIO_13 -> FSIRX_CLK
- GPIO_12 -> FSIRX_RX0
- GPIO_11 -> FSIRX_RX1

Watch Variables
- *dataFrameCntr* Number of Data frame transferred
- *error* Non zero for transmit/receive data mismatch

31.5.1.5 FSI Data Transfers upon CPU Timer Event

FILE: fsi_ex5_periodic_frame.c

Example sets up infinite data frame transfers where trigger comes from ISR handling the periodic CPU Timer event. Automatic (hardware triggered) Ping frame transmission is also setup along with data.

CPU Timer0 is chosen for setting up periodic timer events. User can choose any other Timer-1/Timer-2 as well.

Automatic (hardware triggered) Ping frame transmission is also setup along with data.

If there are any comparison failures during transfers or any of error event occurs, execution will stop.

External Connections

For FSI internal loopback (EXTERNAL_FSI_ENABLE == 0), no external connections needed

For FSI external loopback (EXTERNAL_FSI_ENABLE == 1), external connections are required. The FSI TX pins should be connected to the respective FSI RX pins of the same device. See below for external connections to include and GPIOs used:

External Connections Required between FSI TX and RX of the same device:
- FSIRX_CLK to FSITX_CLK
- FSIRX_RX0 to FSITX_TX0
- FSIRX_RX1 to FSITX_TX1

ControlCard FSI Header GPIOs:
- GPIO_27 -> FSITX_CLK
- GPIO_26 -> FSITX_TX0
- GPIO_25 -> FSITX_TX1
- GPIO_13 -> FSIRX_CLK
- GPIO_12 -> FSIRX_RX0
- GPIO_11 -> FSIRX_RX1

Watch Variables
- *dataFrameCntr* Number of Data frame transferred
- *error* Non zero for transmit/receive data mismatch

31.5.1.6 FSI and SPI Communication(fsi_ex6_spi_master_tx)

FILE: fsi_ex6_spi_master_tx.c

FSI supports SPI compatibility mode to talk to the devices not having FSI but SPI module. Example sets up infinite data frame transfers where FSI acts like master Tx and SPI as slave Rx. API to decode FSI frame received at SPI end is implemented and checks are made to ensure received details(frame tag/type, userdata, data) match with transferred frame.

If there are any comparison failures during transfers or any of error event occurs, execution will stop.

External Connections

For FSI <-> SPI communication, make below connections in GPIO settings:

- GPIO_27 -> GPIO_3 :: To connect FSITX_CLK with SPICLKA
- GPIO_26 -> GPIO_2 :: To connect FSITX_TX0 with SPISIMOA
- GPIO_25 -> GPIO_0 :: To connect FSITX_TX1 with SPISTEA

Watch Variables
- *dataFrameCntr* Number of Data frame transferred
- *error* Non zero for transmit/receive data mismatch

31.5.1.7 FSI and SPI Communication(fsi_ex7_spi_slave_rx)

FILE: fsi_ex7_spi_slave_rx.c

FSI supports SPI compatibility mode to talk to the devices not having FSI but SPI module. Example sets up infinite data frame transfers where FSI acts like slave Rx and SPI as master Rx. API to build the FSI frame at SPI end before transfer is implemented in SW and checks are made to ensure received details(frame tag/type, userdata, data) on FSI Rx match with transferred data.

If there are any comparison failures during transfers or any of error event occurs, execution will stop.

External Connections

For FSI(Rx) <-> SPI(Tx) communication, make connections in GPIO settings:

There is no requirement for a chip select signal to be used when connected to the FSIRX. This is because the FSIRX will respond to any incoming clock edge.

- GPIO_13 -> GPIO_3 :: To connect FSIRXCLK with SPICLKA
- GPIO_12 -> GPIO_2 :: To connect FSIRX0 with SPISIMOA

Watch Variables

- *dataFrameCntr* Number of Data frame transferred
- *error* Non zero for transmit/receive data mismatch

31.5.1.8 FSI P2Point Connection:Rx Side

FILE: fsi_ex8_ext_p2pconnection_rx.c

Example sets up FSI receiving device in a point to point connection to the FSI transmitting device. Example code to set up FSI transmit device is implemented in a separate file.

In a real scenario two separate devices may power up in arbitrary order and there is a need to establish a clean communication link which ensures that receiver side is flushed to properly interpret the start of a new valid frame.

There is no true concept of a master or a slave node in the FSI protocol, but to simplify the data flow and connection we can consider transmitting device as master and receiving side as slave. Transmitting side will be driver of initialization sequence.

Handshake mechanism that must take place before actual data transmission can be use case specific; points described below can be taken as an example on how to implement the handshake from receiving side.

- Setup the receiver interrupts to detect PING type frame reception
- Begin the first PING loop
  - Wait for receiver interrupt
  - If the FSI Rx has received a PING frame with *FSI_FRAME_TAG0*, come out of loop. Otherwise iterate the loop again.
- Begin the second PING loop
  - Send the Flush sequence
  - Send the PING frame with tag *FSI_FRAME_TAG1*
  - Wait for receiver interrupt
  - If the FSI Rx has received a PING frame with *FSI_FRAME_TAG1*, come out of loop. Otherwise iterate the loop again.

Now, the receiver side has received the acknowledged PING frame (tag1), so it is ready for normal operation further.

After above synchronization steps, FSI Rx can be configured as per use case, that is, nWords, lane width, enabling events etc and start the infinite transfers. More details on establishing the communication link can be found in device TRM.

User can edit some of configuration parameters as per use case, similar to other examples.

- *nWords* - Number of words per transfer may be from 1 -16
- *nLanes* - Choice to select single or double lane for frame transfers
- *fsiClock* - FSI Clock used for transfers
- *txUserData* - User data to be sent with Data frame
- *txDataFrameTag* - Frame tag used for Data transfers
- *txPingFrameTag* - Frame tag used for Ping transfers
- *txPingTimeRefCntr* - Tx Ping timer reference counter
- *rxWdTimeoutRefCntr* - Rx Watchdog timeout reference counter

External Connections

For FSI external P2P connection, external connections are required to be made between two devices. Device 1's FSI TX and RX pins need to be connected to device 2's FSI RX and TX pins, respectively. See below for external connections to make and GPIOs used:

External connections required between independent RX and TX devices:

- FSIRX_CLK to FSITX_CLK
- FSIRX_RX0 to FSITX_TX0
- FSIRX_RX1 to FSITX_TX1

ControlCard FSI Header GPIOs:

- GPIO_27 -> FSITX_CLK
- GPIO_26 -> FSITX_TX0
- GPIO_25 -> FSITX_TX1
- GPIO_13 -> FSIRX_CLK
- GPIO_12 -> FSIRX_RX0
- GPIO_11 -> FSIRX_RX1

Watch Variables

- *dataFrameCntr* Number of Data frame received
- *error* Non zero for transmit/receive data mismatch

31.5.1.9 FSI P2Point Connection:Tx Side

FILE: fsi_ex8_ext_p2pconnection_tx.c

Example sets up FSI transmitting device in a point to point connection to the FSI receiving device. Example code to set up FSI receiving device is implemented in a separate file.

In a real scenario two separate devices may power up in arbitrary order and there is a need to establish a clean communication link which ensures that receiver side is flushed to properly interpret the start of a new valid frame.

There is no true concept of a master or a slave node in the FSI protocol, but to simplify the data flow and connection we can consider transmitting device as master and receiving side as slave. Transmitting side will be driver of initialization sequence.

Handshake mechanism which must take place before actual data transmission can be use case specific; points described below can be taken as an example on how to implement the handshake from transmitting side.

- Setup the receiver interrupts to detect PING type frame reception
- Begin the PING loop
  - Send the Flush sequence
  - Send a PING frame with the frame tag *FSI_FRAME_TAG0*
  - Wait for some time (determined by application)
  - If the FSI Rx has received a PING frame with *FSI_FRAME_TAG1*, come out of loop. Otherwise iterate the loop again

Send a PING frame with the frame tag *FSI_FRAME_TAG1*

After above synchronization steps, FSI Tx can be configured as per use case, that is, nWords, lane width, enabling events etc and start the infinite transfers. More details on establishing the communication link can be found in device TRM.

User can edit some of configuration parameters as per use case, similar to other examples.

- *nWords* - Number of words per transfer may be from 1 -16
- *nLanes* - Choice to select single or double lane for frame transfers
- *fsiClock* - FSI Clock used for transfers
- *txUserData* - User data to be sent with Data frame
- *txDataFrameTag* - Frame tag used for Data transfers
- *txPingFrameTag* - Frame tag used for Ping transfers
- *txPingTimeRefCntr* - Tx Ping timer reference counter
- *rxWdTimeoutRefCntr* - Rx Watchdog timeout reference counter

External Connections

For FSI external P2P connection, external connections are required to be made between two devices. Device 1's FSI TX and RX pins need to be connected to device 2's FSI RX and TX pins, respectively. See below for external connections to make and GPIOs used:

External connections required between independent RX and TX devices:

- FSIRX_CLK to FSITX_CLK
- FSIRX_RX0 to FSITX_TX0
- FSIRX_RX1 to FSITX_TX1

ControlCard FSI Header GPIOs:

- GPIO_27 -> FSITX_CLK
- GPIO_26 -> FSITX_TX0
- GPIO_25 -> FSITX_TX1
- GPIO_13 -> FSIRX_CLK
- GPIO_12 -> FSIRX_RX0
- GPIO_11 -> FSIRX_RX1

Watch Variables

- *dataFrameCntr* Number of Data frame transmitted
- *error* Non zero for transmit/receive data mismatch

31.6 FSI Registers

This section describes the Fast Serial Interface Registers. The FSI module contains two distinct sets of registers. One for the FSI receiver and one for the FSI transmitter.

31.6.1 FSI Base Address Table

Table 31-15. FSI Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
FsiTxaRegs	FSI_TX_REGS	FSITXA_BASE	0x0000_6600	YES	YES	YES	YES	YES
FsiRxaRegs	FSI_RX_REGS	FSIRXA_BASE	0x0000_6680	YES	YES	YES	YES	YES

31.6.2 FSI_TX_REGS Registers

Table 31-16 lists the memory-mapped registers for the FSI_TX_REGS registers. All register offset addresses not listed in Table 31-16 should be considered as reserved locations and the register contents should not be modified.

Table 31-16. FSI_TX_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	TX_MASTER_CTRL	Transmit main control register	EALLOW	Go
2h	TX_CLK_CTRL	Transmit clock control register	EALLOW and LOCK	Go
4h	TX_OPER_CTRL_LO	Transmit operation control register low	EALLOW and LOCK	Go
5h	TX_OPER_CTRL_HI	Transmit operation control register high	EALLOW and LOCK	Go
6h	TX_FRAME_CTRL	Transmit frame control register		Go
7h	TX_FRAME_TAG_UDATA	Transmit frame tag and user data register		Go
8h	TX_BUF_PTR_LOAD	Transmit buffer pointer control load register	EALLOW	Go
9h	TX_BUF_PTR_STS	Transmit buffer pointer control status register		Go
Ah	TX_PING_CTRL	Transmit ping control register	EALLOW and LOCK	Go
Bh	TX_PING_TAG	Transmit ping tag register		Go
Ch	TX_PING_TO_REF	Transmit ping timeout counter reference	EALLOW and LOCK	Go
Eh	TX_PING_TO_CNT	Transmit ping timeout current count		Go
10h	TX_INT_CTRL	Transmit interrupt event control register	EALLOW and LOCK	Go
11h	TX_DMA_CTRL	Transmit DMA event control register	EALLOW and LOCK	Go
12h	TX_LOCK_CTRL	Transmit lock control register	EALLOW and LOCK	Go
14h	TX_EVT_STS	Transmit event and error status flag register		Go
16h	TX_EVT_CLR	Transmit event and error clear register	EALLOW	Go
17h	TX_EVT_FRC	Transmit event and error flag force register	EALLOW	Go
18h	TX_USER_CRC	Transmit user-defined CRC register		Go
20h	TX_ECC_DATA	Transmit ECC data register		Go
22h	TX_ECC_VAL	Transmit ECC value register		Go
24h	TX_DLYLINE_CTRL	Transmit delay Line control register	EALLOW and LOCK	Go
40h + formula	TX_BUF_BASE_y	Base address for transmit buffer		Go

Complex bit access types are encoded to fit into small table cells. Table 31-17 shows the codes that are used for access types in this section.

Table 31-17. FSI_TX_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
Reset or Default Value		

Table 31-17. FSI_TX_REGS Access Type Codes (continued)

Access Type	Code	Description
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

31.6.2.1 TX_MASTER_CTRL Register (Offset = 0h) [Reset = 0h]

TX_MASTER_CTRL is shown in Figure 31-20 and described in Table 31-18.

Return to the Summary Table.

Transmit main control register

Figure 31-20. TX_MASTER_CTRL Register

15	14	13	12	11	10	9	8
KEY							
W-0h							
7	6	5	4	3	2	1	0
RESERVED						FLUSH	CORE_RST
R-0h						R/W-0h	R/W-0h

Table 31-18. TX_MASTER_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	KEY	W	0h	Write Key In order to write to any bit in this register, 0xA5 must be written to this field at the same time. Otherwise, writes are ignored. The key is cleared immediately after writing, so it must be written again for every change to this register. Reset type: SYSRSn
7-2	RESERVED	R	0h	Reserved
1	FLUSH	R/W	0h	Flush Operation Start bit This bit will cause the transmitter to initiate a flush pattern of a single toggle on the TXD0 and TXD1 followed by five full cycles of TXCLK. This bit should be written only when the CORE_RST bit is 0 and the clock to the Transmitter core is turned on. 0h (R/W) = Clear this bit. 1h (R/W) = Setting this bit will Initiate flush sequence. To properly execute a flush sequence, Set FLUSH to 1, wait for five TXCLK cycles then clear FLUSH to 0. Note: The KEY field must contain 0xA5 for any write to this bit to take effect. The software must keep this bit set to 1 for at least five TXCLK cycles before setting it back to 0. Reset type: SYSRSn
0	CORE_RST	R/W	0h	Transmitter Main Core Reset bit This bit controls the transmitter main core reset. In order to send any frame, this bit must be cleared. 0h (R/W) = Transmitter core is not in reset and can transmit frames. 1h (R/W) = Transmitter core is held in reset. Note: The KEY field must contatin 0xA5 for any write to this bit to take effect. Reset type: SYSRSn

31.6.2.2 TX_CLK_CTRL Register (Offset = 2h) [Reset = 0h]

TX_CLK_CTRL is shown in Figure 31-21 and described in Table 31-19.

Return to the Summary Table.

Transmit clock control register

Figure 31-21. TX_CLK_CTRL Register

15	14	13	12	11	10	9	8
RESERVED						PRESCALE_VAL	
R-0h						R/W-0h	
7	6	5	4	3	2	1	0
PRESCALE_VAL						CLK_EN	CLK_RST
R/W-0h						R/W-0h	R/W-0h

Table 31-19. TX_CLK_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9-2	PRESCALE_VAL	R/W	0h	Clock Divider Prescale Value The input clock is divided by this 8-bit value and fed into the transmitter core. This divided clock is the rate at which TXCLK will operate. 0h (R/W) = Reserved 1h (R/W) = Input clock /1 2h (R/W) = Input clock /2 3h (R/W) = Input clock /3 4h (R/W) = Input clock /4 ... FFh (R/W) = Input clock /255 TXCLKIN = Input clock / PRESCALE_VAL In FSI mode: TXCLK = TXCLKIN / 2 In SPI mode: TXCLK = TXCLKIN Reset type: SYSRSn
1	CLK_EN	R/W	0h	Clock Divider Enable bit This bit will enable and disable the input clock divider and start the clock to the transmitter core. 0h (R/W) = The input clock divider is not enabled and the clock is not connected to the transmitter core. 1h (R/W) = The input clock to the transmitter core is being divided by the PRESCALE_VAL and enabled. Reset type: SYSRSn
0	CLK_RST	R/W	0h	Clock Divider Reset bit This bit will reset the clock counter in the clock divider. 0h (R/W) = The clock divider is set based on the value in PRESCALE_VAL. The input clock will be divided by PRESCALE_VAL if CLK_EN is set. 1h (R/W) = The clock divider will be reset to 0 and will stay reset until software writes a 0 to this bit. Reset type: SYSRSn

31.6.2.3 TX_OPER_CTRL_LO Register (Offset = 4h) [Reset = 0h]

TX_OPER_CTRL_LO is shown in Figure 31-22 and described in Table 31-20.

Return to the Summary Table.

Transmit operation control register low

Figure 31-22. TX_OPER_CTRL_LO Register

15	14	13	12	11	10	9	8
RESERVED					SEL_TDM_IN	TDM_ENABLE	SEL_PLLCLK
R-0h					R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
PING_TO_MODE	SW_CRC	START_MODE			SPI_MODE	DATA_WIDTH	
R/W-0h	R/W-0h	R/W-0h			R/W-0h	R/W-0h	

Table 31-20. TX_OPER_CTRL_LO Register Field Descriptions

Bit	Field	Type	Reset	Description
15-11	RESERVED	R	0h	Reserved
10	SEL_TDM_IN	R/W	0h	Input TDM port Select bit This bit selects the input port for the transmitter core between the TDM input pins or the RX module. When this bit is '0', the inputs selected for TDM are from the TDM input pins. When this bit is '1', then inputs selected for TDM are from the RX module. Reset type: SYSRSn
9	TDM_ENABLE	R/W	0h	Transmit TDM Mode Enable bit. This bit enables the TDM Mode for multi-remote TDM operation. 0h (R/W) Transmit TDM Mode is not enabled. 1h (R/W) Transmit TDM Mode is enabled. Reset type: SYSRSn
8	SEL_PLLCLK	R/W	0h	Input Clock Select bit This bit selects the input clock source for the transmitter core. 0h (R/W) = SYSCLK is the source of the transmitter clock into the clock prescaler. 1h (R/W) = PLLRAWCLK is the source of the transmitter core clock into the clock prescaler. Reset type: SYSRSn
7	PING_TO_MODE	R/W	0h	Ping Counter Reset Mode Select bit This bit selects when the ping counter will reset. 0h (R/W) = The ping counter will reset and restart only on hardware initiated ping frames, when ping counter has timed out. 1h (R/W) = The ping counter will reset and restart on any software initiated frame as well as a ping counter timeout Reset type: SYSRSn
6	SW_CRC	R/W	0h	CRC Source Select bit This bit selects the source of the CRC value that is transmitted. 0h (R/W) = The transmitted CRC value is computed by hardware. 1h (R/W) = The transmitted CRC value is sourced from the value programmed in the TX_USER_CRC register. Reset type: SYSRSn

Table 31-20. TX_OPER_CTRL_LO Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
5-3	START_MODE	R/W	0h	Transmission Start Mode Select bit These bits select the method by which a new frame transmission is started. 0h (R/W) = Only a software write to TX_FRAME_CTRL.START initiate a new transmission. 1h (R/W) = The configured external trigger will initiate a new transmission. 2h (R/W) = Either writing to TX_FRAME_CTRL.START or the TX_FRAME_TAG_UDATA register will initiate a new transmission. All other combinations of bits are illegal and reserved for future use. Reset type: SYSRSn
2	SPI_MODE	R/W	0h	SPI Mode Select bit This bit enables and disables SPI compatibility mode. 0h (R/W) = FSI is in normal mode of operation. 1h (R/W) = FSI is operating in SPI compatibility mode. Reset type: SYSRSn
1-0	DATA_WIDTH	R/W	0h	Transmit Data Width Select bits These bits define the number of data lines used by the transmitter. 0h (R/W) = Data will be transmitted on one data line (TXD0) 1h (R/W) = Data will be transmitted on two data lines (TXD0 and TXD1). The format of the data is described in the preceeding chapter. 2h, 3h (R/W) = Reserved Reset type: SYSRSn

31.6.2.4 TX_OPER_CTRL_HI Register (Offset = 5h) [Reset = 0h]

TX_OPER_CTRL_HI is shown in Figure 31-23 and described in Table 31-21.

Return to the Summary Table.

Transmit operation control register high

Figure 31-23. TX_OPER_CTRL_HI Register

15	14	13	12	11	10	9	8
RESERVED			EXT_TRIG_SEL				
R-0h			R/W-0h				
7	6	5	4	3	2	1	0
EXT_TRIG_SEL	ECC_SEL	FORCE_ERR	RESERVED				
R/W-0h	R/W-0h	R/W-0h	R-0h				

Table 31-21. TX_OPER_CTRL_HI Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12-7	EXT_TRIG_SEL	R/W	0h	External Trigger Select bit These bits define which of the 64 external inputs will be used as the source for the external input trigger. 00h (R/W) = Trigger 1 is the source. 01h (R/W) = Trigger 2 is the source. 02h (R/W) = Trigger 3 is the source. ... 3Fh (R/W) = Trigger 64 is the source. Reset type: SYSRSn
6	ECC_SEL	R/W	0h	ECC Data Width Select bit This bit selects between 16-bit and 32-bit ECC computation. 0h (R/W) = 32-bit ECC is used. 1h (R/W) = 16-bit ECC is used. Reset type: SYSRSn
5	FORCE_ERR	R/W	0h	Error Frame Force bit This bit will force the the CRC value of the transmitted data frame to 0 whenever there is a buffer overrun or underrun condition. This can be used to force a corrupted CRC as the data is not guaranteed to be reliable. The receiver will treat the data as invalid and can handle this as needed. Note: DO NOT use FORCE_ERR if using the SW CRC mode (FSI Transmit). 0h (R/W) = The CRC will not be forced to 0. 1h (R/W) = The CRC will be forced to 0 in a buffer overrun or underrun condition. Reset type: SYSRSn
4-0	RESERVED	R	0h	Reserved

31.6.2.5 TX_FRAME_CTRL Register (Offset = 6h) [Reset = 0h]

TX_FRAME_CTRL is shown in Figure 31-24 and described in Table 31-22.

Return to the Summary Table.

Transmit frame control register

Figure 31-24. TX_FRAME_CTRL Register

15	14	13	12	11	10	9	8
START	RESERVED						
R/W-0h	R-0h						
7	6	5	4	3	2	1	0
N_WORDS				FRAME_TYPE			
R/W-0h				R/W-0h			

Table 31-22. TX_FRAME_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	START	R/W	0h	Start Transmission bit This bit will cause the FSI to start transmitting the next frame. 0h (R/W) = Writing a 0 to this bit will have no effect. 1h (R/W) = Start the next transmission. This bit will be cleared by hardware. Reset type: SYSRSn
14-8	RESERVED	R	0h	Reserved
7-4	N_WORDS	R/W	0h	Number of Words to be Transmitted This field defines the number of words which will be transmitted in a DATA_N_WORD frame. This is a user-defined field that must match the corresponding field in the receiver. Set this bitfield to be one less than the number of words to be transmitted. 0h (R/W) = 1 data word frame (16-bit data). 1h (R/W) = 2 data word frame (32-bit data). .. Fh (R/W) = 16 data word frame (256-bit data). Reset type: SYSRSn
3-0	FRAME_TYPE	R/W	0h	Transmit Frame Type This field determines the type of frame that will be transmitted next. 0000b (R/W) = Ping Frame. This frame can be sent either by software or automatically by hardware. 0100b (R/W) = DATA_1_WORD Frame. One word data frame (16-bit data). 0101b (R/W) = DATA_2_WORD Frame. Two word data frame (32-bit data). 0110b (R/W) = DATA_4_WORD Frame. Four word data frame (64-bit data). 0111b (R/W) = DATA_6_WORD Frame. Six word data frame (96-bit data). 0011b (R/W) = DATA_N_WORD Frame. The N_WORDS field will determine the number of words (1 to 16) to be sent. Both the transmitter and receiver must have the same value programmed. 1111b (R/W) = Error Frame. This frame can be used during error conditions or any condition where the transmitter wants to notify the receiver of a high priorty status. However, the user software is at liberty to use this for any purpose. 0001b, 0010b, and 1000b through 1110b are Reserved and should not be used. Reset type: SYSRSn

31.6.2.6 TX_FRAME_TAG_UDATA Register (Offset = 7h) [Reset = 0h]

TX_FRAME_TAG_UDATA is shown in Figure 31-25 and described in Table 31-23.

Return to the Summary Table.

Transmit frame tag and user data register

Figure 31-25. TX_FRAME_TAG_UDATA Register

15	14	13	12	11	10	9	8
USER_DATA							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED				FRAME_TAG			
R-0h				R/W-0h			

Table 31-23. TX_FRAME_TAG_UDATA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	USER_DATA	R/W	0h	User Data bits This is a user-defined value that will be loaded into the the user data phase of the frame. This 8-bit value can be used by the receiver for any application need. This value will not impact any hardware behavior. Reset type: SYSRSn
7-4	RESERVED	R	0h	Reserved
3-0	FRAME_TAG	R/W	0h	This will be used only for software initiated transmissions. Frame tag bits This is a user-defined value that will be loaded into the frame tag phase of the next transmission. The receiver may use the frame tag for any application need. This value will not impact any hardware behavior For external triggers do not use this register. Use the TX_PING_TAG register instead. Reset type: SYSRSn

31.6.2.7 TX_BUF_PTR_LOAD Register (Offset = 8h) [Reset = 0h]

TX_BUF_PTR_LOAD is shown in Figure 31-26 and described in Table 31-24.

Return to the Summary Table.

Transmit buffer pointer control load register

Figure 31-26. TX_BUF_PTR_LOAD Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				BUF_PTR_LOAD			
R-0h				R/W-0h			

Table 31-24. TX_BUF_PTR_LOAD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3-0	BUF_PTR_LOAD	R/W	0h	Buffer Pointer Load bits These bits are used to force the transmit buffer pointer to a desired index within the transmit buffer. The next transmission will begin picking data from this index and increment appropriately. This value will be reflected in TX_BUF_PTR_STS only after a minimum 3 SYSCLK cycles + 3 TXCLK cycles. This value should not be written while there is an active transmission as it may corrupt the ongoing frame or other undefined behavior. Reset type: SYSRSn

31.6.2.8 TX_BUF_PTR_STS Register (Offset = 9h) [Reset = 0h]

TX_BUF_PTR_STS is shown in Figure 31-27 and described in Table 31-25.

Return to the Summary Table.

Transmit buffer pointer control status register

Figure 31-27. TX_BUF_PTR_STS Register

15	14	13	12	11	10	9	8
RESERVED			CURR_WORD_CNT				
R-0h			R-0h				
7	6	5	4	3	2	1	0
RESERVED				CURR_BUF_PTR			
R-0h				R-0h			

Table 31-25. TX_BUF_PTR_STS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12-8	CURR_WORD_CNT	R	0h	Words Remaining in the transmit buffer This value indicates the number of words present in the data buffer which have not yet been transmitted. This value is only valid when there is no active transmission. Note: This value will not be valid if there is a buffer overrun or underrun condition. Reset type: SYSRSn
7-4	RESERVED	R	0h	Reserved
3-0	CURR_BUF_PTR	R	0h	Current Buffer Pointer Index This bitfield will show the current index of the buffer pointer. This value is only valid when there is no active transmission. Reset type: SYSRSn

31.6.2.9 TX_PING_CTRL Register (Offset = Ah) [Reset = 0h]

TX_PING_CTRL is shown in Figure 31-28 and described in Table 31-26.

Return to the Summary Table.

Transmit ping control register

Figure 31-28. TX_PING_CTRL Register

15	14	13	12	11	10	9	8
RESERVED							EXT_TRIG_SEL
R-0h							R/W-0h
7	6	5	4	3	2	1	0
EXT_TRIG_SEL					EXT_TRIG_EN	TIMER_EN	CNT_RST
R/W-0h					R/W-0h	R/W-0h	R/W-0h

Table 31-26. TX_PING_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-9	RESERVED	R	0h	Reserved
8-3	EXT_TRIG_SEL	R/W	0h	External Trigger Select bits This bitfield will select one of the 64 external trigger inputs to as the source to generate a ping frame. A ping frame will only be generated if the EXT_TRIG_EN bit is set. 0h (R/W) = Trigger 1 will be used to generate a ping frame. 1h (R/W) = Trigger 2 will be used to generate a ping frame. .. 3Fh (R/W) = Trigger 64 will be used to generate a ping frame. Reset type: SYSRSn
2	EXT_TRIG_EN	R/W	0h	External Trigger Enable bit This bit will allow the external trigger logic to generate a ping frame. 0h (R/W) = External triggers will not be used to generate ping frames. 1h (R/W) = The selected external trigger (selected by EXT_TRIG_SEL bits) will be able to generate a ping frame. The ping timer will be ignored if this bit is set. Reset type: SYSRSn
1	TIMER_EN	R/W	0h	Ping Timer Enable bit This bit will enable the ping timer for generating periodic ping frames. 0h (R/W) = The ping timer is disabled and will not generate ping frames. 1h (R/W) = The ping timer is enabled and can be used to generate ping frames.Once the timer count reaches the value set by the TX_PING_TO_REF register, it will initiate a ping frame transmission. Note: If the ping timer is used, EXT_TRIG_EN should not be set as it will override this function. Reset type: SYSRSn
0	CNT_RST	R/W	0h	Ping Counter Reset bit Writing a 1 to this bit will reset the ping counter to 0. The counter will stay in reset as long as this bit is set to 1. This bit needs to be cleared to 0 to use the counter. 0h (R/W) = Clear the CNT_RST. 1h (R/W) = The ping counter will be reset to 0. Reset type: SYSRSn

31.6.2.10 TX_PING_TAG Register (Offset = Bh) [Reset = 0h]

TX_PING_TAG is shown in Figure 31-29 and described in Table 31-27.

Return to the Summary Table.

Transmit ping tag register

Figure 31-29. TX_PING_TAG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				TAG			
R-0h				R/W-0h			

Table 31-27. TX_PING_TAG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3-0	TAG	R/W	0h	Ping Frame Tag This field contains a 4-bit tag which will be sent in any ping frame that is initiated by an external trigger or the ping timer. This field is user-defined and can be set based on the application requirement. If a ping frame is generated manually, the transmitted tag will be from TX_FRAME_TAG_UDATA.FRAME_TAG, not this value. Reset type: SYSRSn

31.6.2.11 TX_PING_TO_REF Register (Offset = Ch) [Reset = 0h]

TX_PING_TO_REF is shown in Figure 31-30 and described in Table 31-28.

Return to the Summary Table.

Transmit ping timeout counter reference

Figure 31-30. TX_PING_TO_REF Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TO_REF																															
R/W-0h																															

Table 31-28. TX_PING_TO_REF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TO_REF	R/W	0h	Ping Timer Reference Value. This is the 32-bit reference value for the ping timer. The timer will increment the counter starting from 0. When the reference value is reached, it will generate a timeout event, triggering a ping frame transmission. The counter will then reset to 0 and continue counting. Reset type: SYSRSn

31.6.2.12 TX_PING_TO_CNT Register (Offset = Eh) [Reset = 0h]

TX_PING_TO_CNT is shown in Figure 31-31 and described in Table 31-29.

Return to the Summary Table.

Transmit ping timeout current count

Figure 31-31. TX_PING_TO_CNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TO_CNT																															
R-0h																															

Table 31-29. TX_PING_TO_CNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	TO_CNT	R	0h	Ping Timer Counter Value This register contains the current value of the ping timer counter. After reset, this counter will increment until it reaches the reference value (TX_PING_TO_REF), at which point it generates a ping frame transmission. After this point, the counter will reset to 0 and continue counting. This is a free-running counter Reset type: SYSRSn

31.6.2.13 TX_INT_CTRL Register (Offset = 10h) [Reset = 0h]

TX_INT_CTRL is shown in Figure 31-32 and described in Table 31-30.

Return to the Summary Table.

Transmit interrupt event control register

Figure 31-32. TX_INT_CTRL Register

15	14	13	12	11	10	9	8
RESERVED				INT2_EN_PING_TO	INT2_EN_BUF_OVERRUN	INT2_EN_BUF_UNDERRUN	INT2_EN_FRAME_DONE
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED				INT1_EN_PING_TO	INT1_EN_BUF_OVERRUN	INT1_EN_BUF_UNDERRUN	INT1_EN_FRAME_DONE
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 31-30. TX_INT_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11	INT2_EN_PING_TO	R/W	0h	Enable PING Timer Interrupt to INT2 This bit allows the event to generate an interrupt on the INT2 line. 0h (R/W) = This event will not trigger an interrupt on TX_INT2. 1h (R/W) = The ping timer event will trigger an interrupt on TX_INT2. Reset type: SYSRSn
10	INT2_EN_BUF_OVERRUN	R/W	0h	Enable Buffer Overrun Interrupt to INT2 This bit allows the event to generate an interrupt on the INT2 line. 0h (R/W) = This event will not trigger an interrupt on TX_INT2. 1h (R/W) = A Buffer Overrun condition will trigger an interrupt on TX_INT2. Reset type: SYSRSn
9	INT2_EN_BUF_UNDERRUN	R/W	0h	Enable Buffer Underrun Interrupt to INT2 This bit allows the event to generate an interrupt on the INT2 line. 0h (R/W) = This event will not trigger an interrupt on TX_INT2. 1h (R/W) = A Buffer Underrun condition will trigger an interrupt on TX_INT2. Reset type: SYSRSn
8	INT2_EN_FRAME_DONE	R/W	0h	Enable Frame Done interrupt to INT2 This bit allows the event to generate an interrupt on the INT2 line. 0h (R/W) = This event will not trigger an interrupt on TX_INT2. 1h (R/W) = A Frame Done event will trigger an interrupt on TX_INT2. Reset type: SYSRSn
7-4	RESERVED	R	0h	Reserved
3	INT1_EN_PING_TO	R/W	0h	Enable Ping Timer Interrupt to INT1 This bit allows the event to generate an interrupt on the INT1 line. 0h (R/W) = This event will not trigger an interrupt on TX_INT1. 1h (R/W) = The ping timer event will trigger an interrupt on TX_INT1. Reset type: SYSRSn
2	INT1_EN_BUF_OVERRUN	R/W	0h	Enable Buffer Overrun Interrupt to INT1 This bit allows the event to generate an interrupt on the INT1 line. 0h (R/W) = This event will not trigger an interrupt on TX_INT1. 1h (R/W) = A Buffer Overrun condition will trigger an interrupt on TX_INT1. Reset type: SYSRSn
1	INT1_EN_BUF_UNDERRUN	R/W	0h	Enable Buffer Underrun Interrupt to INT1 This bit allows the event to generate an interrupt on the INT1 line. 0h (R/W) = This event will not trigger an interrupt on TX_INT1. 1h (R/W) = A Buffer Underrun condition will trigger an interrupt on TX_INT1. Reset type: SYSRSn

Table 31-30. TX_INT_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
0	INT1_EN_FRAME_DONE	R/W	0h	Enable Frame Done interrupt to INT1 This bit allows the event to generate an interrupt on the INT1 line. 0h (R/W) = This event will not trigger an interrupt on TX_INT1. 1h (R/W) = A Frame Done event will trigger an interrupt on TX_INT1. Reset type: SYSRSn

31.6.2.14 TX_DMA_CTRL Register (Offset = 11h) [Reset = 0h]

TX_DMA_CTRL is shown in Figure 31-33 and described in Table 31-31.

Return to the Summary Table.

Transmit DMA event control register

Figure 31-33. TX_DMA_CTRL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							DMA_EVT_EN
R-0h							R/W-0h

Table 31-31. TX_DMA_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R	0h	Reserved
0	DMA_EVT_EN	R/W	0h	DMA Event Enable bit This bit will enable the DMA event to be generated upon the completion of a transmit frame. 0h (R/W) = A DMA event will not be generated. 1h (R/W) = A DMA event will be generated upon the completion of a transmitted frame. Note: The DMA event will only be generated for data frames. Reset type: SYSRSn

31.6.2.15 TX_LOCK_CTRL Register (Offset = 12h) [Reset = 0h]

TX_LOCK_CTRL is shown in Figure 31-34 and described in Table 31-32.

Return to the Summary Table.

Transmit lock control register

Figure 31-34. TX_LOCK_CTRL Register

15	14	13	12	11	10	9	8
KEY							
W-0h							
7	6	5	4	3	2	1	0
RESERVED							LOCK
R-0h							R/W-0h

Table 31-32. TX_LOCK_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	KEY	W	0h	Write Key In order to write to this register, 0xA5 must be written to this field at the same time. Otherwise, writes are ignored. The key is cleared immediately after writing, so it must be written again for every change to this register. Reset type: SYSRSn
7-1	RESERVED	R	0h	Reserved
0	LOCK	R/W	0h	Control Register Lock Enable bit This bit locks the contents of all the transmit control registers that support a lock protection. Once locked, further writes will not take effect until a SYSRS has reset this register. Once set, further writes to this bit will be ignored. 0h (R/W) = Transmit control registers can be modified and are not locked. 1h (R/W) = Transmit control registers are locked and cannot be modified until this bit is cleared by SYSRS. Any further writes to this bit are ignored. Note: The KEY field must contatin 0xA5 for any write to this bit to take effect. Reset type: SYSRSn

31.6.2.16 TX_EVT_STS Register (Offset = 14h) [Reset = 0h]

TX_EVT_STS is shown in Figure 31-35 and described in Table 31-33.

Return to the Summary Table.

Transmit event and error status flag register

Figure 31-35. TX_EVT_STS Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				PING_TRIGGERED	BUF_OVERRUN	BUF_UNDERRUN	FRAME_DONE
R-0h				R-0h	R-0h	R-0h	R-0h

Table 31-33. TX_EVT_STS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	PING_TRIGGERED	R	0h	Ping Frame Triggered Flag Bit This bit indicates that a ping frame has been triggered. This bit is set by hardware when either the ping timer or an external trigger event have occured. Software can also force this bit to get set by writing to the TX_EVT_FRC register. 0h (R) = A ping frame has not been triggered. 1h (R) = A ping frame has been triggered by either the ping timer or external trigger. To clear this bit, write to the corresponding bit in the TX_EVT_CLR register. Reset type: SYSRSn
2	BUF_OVERRUN	R	0h	Buffer Overrun Flag Bit This bit inditcates that buffer overrun has occured.Software can also force this bit to get set by writing to the TX_EVT_FRC register. 0h (R) = Buffer Overrun has not occured. 1h (R) = Buffer Overrun has occured. To clear this bit, write to the corresponding bit in the TX_EVT_CLR register. Reset type: SYSRSn
1	BUF_UNDERRUN	R	0h	Buffer Underrun Flag Bit This bit inditcates that buffer underrun has occured.Software can also force this bit to get set by writing to the TX_EVT_FRC register. 0h (R) = Buffer Underrun has not occured. 1h (R) = Buffer Underrun has occured. To clear this bit, write to the corresponding bit in the TX_EVT_CLR register. Reset type: SYSRSn
0	FRAME_DONE	R	0h	Frame Done Flag Bit This bit inditcates a Frame Done condition. This bit is set by hardware when a frame transmission has been completed. Software can also force this bit to get set by writing to the TX_EVT_FRC register. 0h (R) = Frame Done condition has not occured. 1h (R) = Frame Done condition has occured. To clear this bit, write to the corresponding bit in the TX_EVT_CLR register. Reset type: SYSRSn

31.6.2.17 TX_EVT_CLR Register (Offset = 16h) [Reset = 0h]

TX_EVT_CLR is shown in Figure 31-36 and described in Table 31-34.

Return to the Summary Table.

Transmit event and error clear register

Figure 31-36. TX_EVT_CLR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				PING_TRIGGERED	BUF_OVERRUN	BUF_UNDERRUN	FRAME_DONE
R-0h				W-0h	W-0h	W-0h	W-0h

Table 31-34. TX_EVT_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	PING_TRIGGERED	W	0h	Ping Frame Triggered Flag Clear bit This bit clears the corresponding bit in the TX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the TX_EVT_STS register to 0. Note: This bit may not always be cleared when writing to the corresponding TX_EVT_CLR bit. If PING_TIMEOUT MODE is configured to be 0, a hardware ping timeout may occur when another frame is actively being transmitted. In this case, if this bit still shows as 1 after the clear bit is written then the ping frame has been triggered but not serviced. This bit does not indicate that the ping frame has been completely sent, only that it has been triggered by the timeout event. Reset type: SYSRSn
2	BUF_OVERRUN	W	0h	Buffer Overrun Flag Clear bit This bit clears the corresponding bit in the TX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the TX_EVT_STS register to 0. Reset type: SYSRSn
1	BUF_UNDERRUN	W	0h	Buffer Underrun Flag Clear bit This bit clears the corresponding bit in the TX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the TX_EVT_STS register to 0. Reset type: SYSRSn
0	FRAME_DONE	W	0h	Frame Done Flag Clear bit This bit clears the corresponding bit in the TX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the TX_EVT_STS register to 0. Reset type: SYSRSn

31.6.2.18 TX_EVT_FRC Register (Offset = 17h) [Reset = 0h]

TX_EVT_FRC is shown in Figure 31-37 and described in Table 31-35.

Return to the Summary Table.

Transmit event and error flag force register

Figure 31-37. TX_EVT_FRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				PING_TRIGGERED	BUF_OVERRUN	BUF_UNDERRUN	FRAME_DONE
R-0h				W-0h	W-0h	W-0h	W-0h

Table 31-35. TX_EVT_FRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3	PING_TRIGGERED	W	0h	Ping Frame Triggered Flag Force bit This bit will cause the corresponding bit in the TX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding flag bit in the TX_EVT_STS Register. Reset type: SYSRSn
2	BUF_OVERRUN	W	0h	Buffer Overrun Flag Force bit This bit will cause the corresponding bit in the TX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (R/W) = Writing a 0 to this bit will have no effect. 1h (R/W) = Force the corresponding flag bit in the TX_EVT_STS Register. Reset type: SYSRSn
1	BUF_UNDERRUN	W	0h	Buffer Underrun Flag Force bit This bit will cause the corresponding bit in the TX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding flag bit in the TX_EVT_STS Register. Reset type: SYSRSn
0	FRAME_DONE	W	0h	Frame Done Flag Force bit This bit will cause the corresponding bit in the TX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding flag bit in the TX_EVT_STS Register. Reset type: SYSRSn

31.6.2.19 TX_USER_CRC Register (Offset = 18h) [Reset = 0h]

TX_USER_CRC is shown in Figure 31-38 and described in Table 31-36.

Return to the Summary Table.

Transmit user-defined CRC register

Figure 31-38. TX_USER_CRC Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
USER_CRC							
R/W-0h							

Table 31-36. TX_USER_CRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	RESERVED	R	0h	Reserved
7-0	USER_CRC	R/W	0h	User-defined CRC This register contains the 8-bit CRC value to be transmitted in the next frame if the transmission is set for user-defined CRC option (TX_OPER_CTRL_LO.SW_CRC = 1). This register is ignored if the hardware CRC generation is enabled. Reset type: SYSRSn

31.6.2.20 TX_ECC_DATA Register (Offset = 20h) [Reset = 0h]

TX_ECC_DATA is shown in Figure 31-39 and described in Table 31-37.

Return to the Summary Table.

Transmit ECC data register

Figure 31-39. TX_ECC_DATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_HIGH																DATA_LOW															
R/W-0h																R/W-0h															

Table 31-37. TX_ECC_DATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DATA_HIGH	R/W	0h	Upper 16 bits of ECC Data Writing to this bitfield will cause the ECC logic to compute the ECC(SEC-DED) the entire 32-bit register and update TX_ECC_VAL register with the results. Software should write to these 16 bits of the register in a 32-bit write when needing to compute ECC for 32-bits for the full TX_ECC_DATA register. Reset type: SYSRSn
15-0	DATA_LOW	R/W	0h	Lower 16 bits of ECC Data Writing to this bitfield will cause the ECC logic to compute the ECC(SEC-DED) for these 16 bits and update the TX_ECC_VAL register with the results. Software should write to these register bits as a 16-bit write when needing to compute ECC for 16-bits. Reset type: SYSRSn

31.6.2.21 TX_ECC_VAL Register (Offset = 22h) [Reset = Ch]

TX_ECC_VAL is shown in Figure 31-40 and described in Table 31-38.

Return to the Summary Table.

Transmit ECC value register

Figure 31-40. TX_ECC_VAL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	ECC_VAL						
R-0h	R-Ch						

Table 31-38. TX_ECC_VAL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R	0h	Reserved
6-0	ECC_VAL	R	Ch	Computed ECC Value This field contains the ECC value computed using SEC-DED either for 16-bit or 32-bit data in the TX_ECC_DATA register. Reset type: SYSRSn

31.6.2.22 TX_DLYLINE_CTRL Register (Offset = 24h) [Reset = 0h]

TX_DLYLINE_CTRL is shown in Figure 31-41 and described in Table 31-39.

Return to the Summary Table.

Transmit delay Line control register

Figure 31-41. TX_DLYLINE_CTRL Register

15	14	13	12	11	10	9	8
RESERVED	TXD1_DLY					TXD0_DLY	
R-0h	R/W-0h					R/W-0h	
7	6	5	4	3	2	1	0
TXD0_DLY			TXCLK_DLY				
R/W-0h			R/W-0h				

Table 31-39. TX_DLYLINE_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14-10	TXD1_DLY	R/W	0h	Delay Line Tap Select for TXD1 This bitfield selects the number of delay elements inserted into the TXD1 path from the pin boundary to the receiver core. 0h (R/W) Zero delay elements are included in the TXD1 path. TXD1 is taken directly from the pin. 1h (R/W) One delay element is included in the TXD1 path. 2h (R/W) Two delay elements are included in the TXD1 path. ... 1Fh (R/W) 31 delay elements are included in the TXD1 path, the maximum. Reset type: SYSRSn
9-5	TXD0_DLY	R/W	0h	Delay Line Tap Select for TXD0 This bitfield selects the number of delay elements inserted into the TXD0 path from the pin boundary to the receiver core. 0h (R/W) Zero delay elements are included in the TXD0 path. TXD0 is taken directly from the pin. 1h (R/W) One delay element is included in the TXD0 path. 2h (R/W) Two delay elements are included in the TXD0 path. ... 1Fh (R/W) 31 delay elements are included in the TXD0 path, the maximum. Reset type: SYSRSn
4-0	TXCLK_DLY	R/W	0h	Delay Line Tap Select for TXCLK This bitfield selects the number of delay elements inserted into the TXCLK path from the pin boundary to the receiver core. 0h (R/W) Zero delay elements are included in the TXCLK path. TXCLK is taken directly from the pin. 1h (R/W) One delay element is included in the TXCLK path. 2h (R/W) Two delay elements are included in the TXCLK path. ... 1Fh (R/W) 31 delay elements are included in the TXCLK path, the maximum. Reset type: SYSRSn

31.6.2.23 TX_BUF_BASE_y Register (Offset = 40h + formula) [Reset = 0h]

TX_BUF_BASE_y is shown in Figure 31-42 and described in Table 31-40.

Return to the Summary Table.

Base address for transmit buffer

Offset = 40h + (y * 1h); where y = 0h to Fh

Figure 31-42. TX_BUF_BASE_y Register

15	14	13	12	11	10	9	8
BASE_ADDRESS							
R/W-0h							
7	6	5	4	3	2	1	0
BASE_ADDRESS							
R/W-0h							

Table 31-40. TX_BUF_BASE_y Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	BASE_ADDRESS	R/W	0h	Transmit Data Buffer Base Address This is the base address of the 16-word data buffer used by the transmitter. Reset type: SYSRSn

31.6.3 FSI_RX_REGS Registers

Table 31-41 lists the memory-mapped registers for the FSI_RX_REGS registers. All register offset addresses not listed in Table 31-41 should be considered as reserved locations and the register contents should not be modified.

Table 31-41. FSI_RX_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	RX_MASTER_CTRL	Receive main control register	EALLOW	Go
4h	RX_OPER_CTRL	Receive operation control register	EALLOW and LOCK	Go
6h	RX_FRAME_INFO	Receive frame control register		Go
7h	RX_FRAME_TAG_UDATA	Receive frame tag and user data register		Go
8h	RX_DMA_CTRL	Receive DMA event control register	EALLOW and LOCK	Go
Ah	RX_EVT_STS	Receive event and error status flag register		Go
Bh	RX_CRC_INFO	Receive CRC info of received and computed CRC		Go
Ch	RX_EVT_CLR	Receive event and error clear register	EALLOW	Go
Dh	RX_EVT_FRC	Receive event and error flag force register	EALLOW	Go
Eh	RX_BUF_PTR_LOAD	Receive buffer pointer load register	EALLOW	Go
Fh	RX_BUF_PTR_STS	Receive buffer pointer status register		Go
10h	RX_FRAME_WD_CTRL	Receive frame watchdog control register	EALLOW and LOCK	Go
12h	RX_FRAME_WD_REF	Receive frame watchdog counter reference	EALLOW and LOCK	Go
14h	RX_FRAME_WD_CNT	Receive frame watchdog current count		Go
16h	RX_PING_WD_CTRL	Receive ping watchdog control register	EALLOW and LOCK	Go
17h	RX_PING_TAG	Receive ping tag register		Go
18h	RX_PING_WD_REF	Receive ping watchdog counter reference	EALLOW and LOCK	Go
1Ah	RX_PING_WD_CNT	Receive pingwatchdog current count		Go
1Ch	RX_INT1_CTRL	Receive interrupt control register for RX_INT1	EALLOW and LOCK	Go
1Dh	RX_INT2_CTRL	Receive interrupt control register for RX_INT2	EALLOW and LOCK	Go
1Eh	RX_LOCK_CTRL	Receive lock control register	EALLOW and LOCK	Go
20h	RX_ECC_DATA	Receive ECC data register		Go
22h	RX_ECC_VAL	Receive ECC value register		Go
24h	RX_ECC_SEC_DATA	Receive ECC corrected data register		Go
26h	RX_ECC_LOG	Receive ECC log and status register		Go
28h	RX_FRAME_TAG_CMP	Receive frame tag compare register	EALLOW and LOCK	Go
29h	RX_PING_TAG_CMP	Receive ping tag compare register	EALLOW and LOCK	Go
2Ch	RX_TRIG_CTRL_0	Receive Trigger Control register 0	EALLOW and LOCK	Go
2Eh	RX_TRIG_WIDTH_0	Receive Trigger Wdith register 0	EALLOW and LOCK	Go
30h	RX_DLYLINE_CTRL	Receive delay line control register	EALLOW and LOCK	Go

Table 31-41. FSI_RX_REGS Registers (continued)

Offset	Acronym	Register Name	Write Protection	Section
32h	RX_TRIG_CTRL_1	Receive Trigger Control register 1	EALLOW and LOCK	Go
34h	RX_TRIG_CTRL_2	Receive Trigger Control register 2	EALLOW and LOCK	Go
36h	RX_TRIG_CTRL_3	Receive Trigger Control register 3	EALLOW and LOCK	Go
38h	RX_VIS_1	Receive debug visibility register 1		Go
3Ah	RX_UDATA_FILTER	Receive User Data Filter Control register	EALLOW and LOCK	Go
40h + formula	RX_BUF_BASE_y	Base address for receive data buffer		Go

Complex bit access types are encoded to fit into small table cells. Table 31-42 shows the codes that are used for access types in this section.

Table 31-42. FSI_RX_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

31.6.3.1 RX_MASTER_CTRL Register (Offset = 0h) [Reset = 0h]

RX_MASTER_CTRL is shown in Figure 31-43 and described in Table 31-43.

Return to the Summary Table.

Receive main control register

Figure 31-43. RX_MASTER_CTRL Register

15	14	13	12	11	10	9	8
KEY							
W-0h							
7	6	5	4	3	2	1	0
RESERVED			DATA_FILTER_EN	INPUT_ISOLATE	SPI_PAIRING	INT_LOOPBACK	CORE_RST
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 31-43. RX_MASTER_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	KEY	W	0h	Write Key. In order to write to this register, 0xA5 must be written to this field at the same time. Otherwise, writes are ignored. The key is cleared immediately after writing, so it must be written again for every change to this register. Reset type: SYSRSn
7-5	RESERVED	R	0h	Reserved
4	DATA_FILTER_EN	R/W	0h	Data Filter Enable Bit. 0h (R/W) = Data filtering is disabled. 1h (R/W) = Data filtering is enabled. Reset type: SYSRSn
3	INPUT_ISOLATE	R/W	0h	When set to 1, the FSI RX inputs (RXCLK, RXD0 and RXD1) will be isolated from what is driven from the device pins and will be held at inactive level of '1'.This isolation facilitates the user to switch the RX inputs to a different set of device pins and hence any potential glitch that could occur during the process of switching will not affect the RX module itself. Reset type: SYSRSn
2	SPI_PAIRING	R/W	0h	Clock Pairing for SPI-like Behavior Enable bit This bit enables the internal clock pairing with the FSI TX module. This feature internally connects the TXCLK to RXCLK allowing the FSI TX module, acting as a SPI controller, to clock data into the receiver and out of the transmitter like a standard SPI module. This configuration is valid when the Module is in SPI mode only (RX_OPER_CTRL.SPI_MODE = 1) 0h (R/W) = SPI clock pairing is not enabled. 1h (R/W) = SPI clock pairing is enabled. The RXCLK will be internally connected to the TXCLK of the corresponding FSI module. Note: The KEY field must contatin 0xA5 for any write to this bit to take effect. Reset type: SYSRSn

Table 31-43. RX_MASTER_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	INT_LOOPBACK	R/W	0h	Internal Loopback Enable bit This bit enables the internal loopback functionality of the FSI receiver. By enabling this bit, a mux will select the signals coming directly from the corresponding FSI transmitter module rather than from the pins. 0h (R/W) = Internal loopback is disabled. The FSI RX module will receive signals coming from the pins. 1h (R/W) = Internal loopback is enabled. The FSI RX module will receive signals from the directly from FSI TX module rather than the pins. Note: The KEY field must contatin 0xA5 for any write to this bit to take effect. Reset type: SYSRSn
0	CORE_RST	R/W	0h	Receiver Main Core Reset bit This bit controls the receiver main core reset. In order to receive any frame, this bit must be cleared. Note: For reset to take affect, the FSI RX module must be held in reset for at least 4 SYSCLK cycles. 0h (R/W) = Receiver core is not in reset and can receive frames. 1h (R/W) = Receiver core is held in reset. Note: The KEY field must contatin 0xA5 for any write to this bit to take effect. Reset type: SYSRSn

31.6.3.2 RX_OPER_CTRL Register (Offset = 4h) [Reset = 0h]

RX_OPER_CTRL is shown in Figure 31-44 and described in Table 31-44.

Return to the Summary Table.

Receive operation control register

Figure 31-44. RX_OPER_CTRL Register

15	14	13	12	11	10	9	8
RESERVED							PING_WD_RST_MODE
R-0h							R/W-0h
7	6	5	4	3	2	1	0
ECC_SEL	N_WORDS				SPI_MODE	DATA_WIDTH	
R/W-0h	R/W-0h				R/W-0h	R/W-0h	

Table 31-44. RX_OPER_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-9	RESERVED	R	0h	Reserved
8	PING_WD_RST_MODE	R/W	0h	Ping Watchdog Timeout Mode Select bit This bit selects the mode by which the ping watchdog counter is reset. The watchdog counter can be reset and restarted only by ping frames or by any received frame. 0h (R/W) = The ping watchdog counter will reset and restart only by ping frames. 1h (R/W) = The ping watchdog counter will reset and restart by any received frame. Reset type: SYSRSn
7	ECC_SEL	R/W	0h	ECC Data Width Select bit This bit selects between whether the ECC computation is done on 16-bit or 32-bit words. 0h (R/W) = 32-bit ECC is used. 1h (R/W) = 16-bit ECC is used. Reset type: SYSRSn
6-3	N_WORDS	R/W	0h	Number of Words to Receive This field defines the number of words which will be received in a DATA_N_WORD frame. This is a user-defined field that must match the corresponding field in the transmitter. Set this bitfield to be one less than the number of words to be received. This value is only applicable when the frame type received is DATA_N_WORD. 0h (R/W) = 1 data word frame (16-bit data). 1h (R/W) = 2 data word frame (32-bit data). .. Fh (R/W) = 16 data word frame (256-bit data). Reset type: SYSRSn
2	SPI_MODE	R/W	0h	SPI Mode Enable bit This bit enables and disables the SPI compatibility mode of the FSI RX. The received data must be formatted as an FSI frame in order for the data to properly be received. SPI compatibility mode will allow FSI RX to receive data that is sent using SPI signal format. Refer to the applicable section in the FSI TRM chapter for more information. 0h (R/W) = FSI is in normal mode of operation. 1h (R/W) = FSI is operating in SPI compatibility mode. Reset type: SYSRSn
1-0	DATA_WIDTH	R/W	0h	Receive Data Width Select bit These bits decide the number of data lines used for receiving data. 0h (R/W) = Data will be received on one data line, RXD0. 1h (R/W) = Data will be received on two data lines, RXD0 and RXD1. 2h, 3h (R/W) = Reserved Reset type: SYSRSn

31.6.3.3 RX_FRAME_INFO Register (Offset = 6h) [Reset = 0h]

RX_FRAME_INFO is shown in Figure 31-45 and described in Table 31-45.

Return to the Summary Table.

Receive frame control register

Figure 31-45. RX_FRAME_INFO Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				FRAME_TYPE			
R-0h				R-0h			

Table 31-45. RX_FRAME_INFO Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3-0	FRAME_TYPE	R	0h	Received Frame Type This field indicates the type of non-ping frame that was successfully received last. Note: Ping frame reception does not update this field, we want to retain the last successful non-ping frame FRAME_TYPE and PING_FRAME_RCVD flag already conveys PING info to the user. 0100b (R/W) = A DATA_1_WORD frame was received (16-bit data). 0101b (R/W) = A DATA_2_WORD frame was received (32-bit data). 0110b (R/W) = A DATA_4_WORD frame was received (64-bit data). 0111b (R/W) = A DATA_6_WORD frame was received (96-bit data). 0011b (R/W) = A DATA_N_WORD frame was received. The N_WORD field will determine the number of words (1 to 16) to be sent. The number of words received must equal the value programmed in RX_OPER_CTRL.N_WORDS. 1111b (R/W) = An error frame was received. This frame can be used during error conditions or any condition where the transmitter wants to signal the receiver for attention. However, the user software is at liberty to use this for any purpose. 0001b, 0010b, and 1000b through 1110b are Reserved and should not be used. Reset type: SYSRSn

31.6.3.4 RX_FRAME_TAG_UDATA Register (Offset = 7h) [Reset = 0h]

RX_FRAME_TAG_UDATA is shown in Figure 31-46 and described in Table 31-46.

Return to the Summary Table.

Receive frame tag and user data register

Figure 31-46. RX_FRAME_TAG_UDATA Register

15	14	13	12	11	10	9	8
USER_DATA							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			FRAME_TAG				RESERVED
R-0h			R-0h				R-0h

Table 31-46. RX_FRAME_TAG_UDATA Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	USER_DATA	R	0h	Received User Data This field contains the 8-bit user data field of the last successfully received frame. Reset type: SYSRSn
7-5	RESERVED	R	0h	Reserved
4-1	FRAME_TAG	R	0h	Received Frame Tag This field contains the 4-bit frame tag from the last successfully received frame. This is intentionally shifted into bits 4:1 so that the register can be used as a 32-bit address index based on the received tag. Reset type: SYSRSn
0	RESERVED	R	0h	Reserved

31.6.3.5 RX_DMA_CTRL Register (Offset = 8h) [Reset = 0h]

RX_DMA_CTRL is shown in Figure 31-47 and described in Table 31-47.

Return to the Summary Table.

Receive DMA event control register

Figure 31-47. RX_DMA_CTRL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							DMA_EVT_EN
R-0h							R/W-0h

Table 31-47. RX_DMA_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-1	RESERVED	R	0h	Reserved
0	DMA_EVT_EN	R/W	0h	DMA Event Enable bit This bit will enable a DMA Event to be generated upon the completion of a frame reception. 0h (R/W) = A DMA event will not be generated. 1h (R/W) = A DMA event will be generated upon the reception of a frame. Note: The DMA event will only be generated for data frames. Reset type: SYSRSn

31.6.3.6 RX_EVT_STS Register (Offset = Ah) [Reset = 0h]

RX_EVT_STS is shown in Figure 31-48 and described in Table 31-48.

Return to the Summary Table.

Receive event and error status flag register

Figure 31-48. RX_EVT_STS Register

15	14	13	12	11	10	9	8
RESERVED	ERROR_TAG_MATCH	DATA_TAG_MATCH	PING_TAG_MATCH	DATA_FRAME	FRAME_OVERRUN	PING_FRAME	ERR_FRAME
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
BUF_UNDERRUN	FRAME_DONE	BUF_OVERRUN	EOF_ERR	TYPE_ERR	CRC_ERR	FRAME_WD_TO	PING_WD_TO
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 31-48. RX_EVT_STS Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	ERROR_TAG_MATCH	R	0h	Error Tag Match Flag This bit indicates that an error frame was received with a tag comparison matching the masked tag reference. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = No tag-matched error frame received. 1h (R) = A tag-matched error frame has been received. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn
13	DATA_TAG_MATCH	R	0h	Data Tag Match Flag This bit indicates that a dataframe was received with a tag comparison matching the masked tag reference. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = No tag-matched data frame received. 1h (R) = A tag-matched data frame has been received. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn
12	PING_TAG_MATCH	R	0h	Ping Tag Match Flag This bit indicates that a ping frame was received with a tag comparison matching the masked tag reference. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = No tag-matched ping frame received. 1h (R) = A tag-matched ping frame has been received. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn
11	DATA_FRAME	R	0h	Data Frame Received Flag This bit indicates that an data frame has been received. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = No data frame has been received. 1h (R) = A data frame has been received. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn

Table 31-48. RX_EVT_STS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
10	FRAME_OVERRUN	R	0h	Frame Overrun Flag This bit indicates that a frame overrun condition has occured. This bit gets set to 1 when a new DATA/ERROR frame is received and the corresponding DATA_FRAME_RCVD/ERROR_FRAME_RCVD flag is still set to 1. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = Frame overrun has not ocurred. 1h (R) = Frame overrun has ocurred. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn
9	PING_FRAME	R	0h	Ping Frame Received Flag This bit indicates that an ping frame has been received. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = No ping frame has been received. 1h (R) = A ping frame has been received. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn
8	ERR_FRAME	R	0h	Error Frame Received Flag This bit indicates that an error frame has been received. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = No error frame has been received. 1h (R) = An error frame has been received. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn
7	BUF_UNDERRUN	R	0h	Receive Buffer Underrun Flag This bit indicates that a buffer underrun condition has occured in the receive buffer. This will happen when software reads the buffer which is empty and has no valid data. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = Receive Buffer Underrun has not ocurred. 1h (R) = Receive Buffer Underrun has ocurred. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn
6	FRAME_DONE	R	0h	Frame Done Flag This bit indicates that a frame has been successfully received without error. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = No frame has been successfully received. 1h (R) = A frame has been successfully received. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn
5	BUF_OVERRUN	R	0h	Receive Buffer Overrun Flag This bit indicates that a buffer overrun condition has occured in the receive buffer. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = Receive buffer overrun has not ocurred. 1h (R) = Receive buffer overrun has ocurred. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn

Table 31-48. RX_EVT_STS Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4	EOF_ERR	R	0h	End-of-Frame Error Flag This bit indicates that an invalid end-of-frame bit pattern has been received. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = Invalid end-of-frame has not been received. 1h (R) = Invalid end-of-frame has been received To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn
3	TYPE_ERR	R	0h	Frame Type Error Flag This bit inditcates that an invalid frame type has been received. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = Invalid frame type has not been received. 1h (R) = Invalid frame type has been received To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn
2	CRC_ERR	R	0h	CRC Error Flag This bit indicates that a CRC error has occured. A CRC error will be generated on a data frame where the received CRC and the computed CRC do not match. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = CRC error has not occured. 1h (R) = CRC error has occured. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn
1	FRAME_WD_TO	R	0h	Frame Watchdog Timeout Flag This bit indicates that the frame watchdog timer has timed out. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = Frame watchdog timeout has not occured. 1h (R) = Frame watchdog timeout has occured. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn
0	PING_WD_TO	R	0h	Ping Watchdog Timeout Flag This bit indicates that the ping watchdog timer has timed out. Software can also force this bit to get set by writing to the RX_EVT_FRC register. 0h (R) = Ping watchdog timeout has not occured. 1h (R) = Ping watchdog timeout has occured. To clear this bit, write to the corresponding bit in the RX_EVT_CLR register. Reset type: SYSRSn

31.6.3.7 RX_CRC_INFO Register (Offset = Bh) [Reset = 0h]

RX_CRC_INFO is shown in Figure 31-49 and described in Table 31-49.

Return to the Summary Table.

Receive CRC info of received and computed CRC

Figure 31-49. RX_CRC_INFO Register

15	14	13	12	11	10	9	8
CALC_CRC							
R-0h							
7	6	5	4	3	2	1	0
RX_CRC							
R-0h							

Table 31-49. RX_CRC_INFO Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	CALC_CRC	R	0h	Harware Calculated CRC Value This bitfield contains the CRC value that was calculated on the last received data. The contents of this bitfield are valid only when data frames are received. Note: The contents of this bitfield are invalid for ping and error frames. Reset type: SYSRSn
7-0	RX_CRC	R	0h	Received CRC Value This bitfield contains the CRC value that was last received a frame. The contents of this bitfield are valid only when data frames are received. Note: The contents of this bitfield are invalid for ping and error frames. Reset type: SYSRSn

31.6.3.8 RX_EVT_CLR Register (Offset = Ch) [Reset = 0h]

RX_EVT_CLR is shown in Figure 31-50 and described in Table 31-50.

Return to the Summary Table.

Receive event and error clear register

Figure 31-50. RX_EVT_CLR Register

15	14	13	12	11	10	9	8
RESERVED	ERROR_TAG_MATCH	DATA_TAG_MATCH	PING_TAG_MATCH	DATA_FRAME	FRAME_OVERRUN	PING_FRAME	ERR_FRAME
R-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
BUF_UNDERRUN	FRAME_DONE	BUF_OVERRUN	EOF_ERR	TYPE_ERR	CRC_ERR	FRAME_WD_TO	PING_WD_TO
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

Table 31-50. RX_EVT_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	ERROR_TAG_MATCH	W	0h	Error Tag Match Glag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
13	DATA_TAG_MATCH	W	0h	Data Tag Match Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
12	PING_TAG_MATCH	W	0h	Ping Tag Match Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
11	DATA_FRAME	W	0h	Data Frame Received Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
10	FRAME_OVERRUN	W	0h	Frame Overrun Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
9	PING_FRAME	W	0h	Ping Frame Received Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn

Table 31-50. RX_EVT_CLR Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
8	ERR_FRAME	W	0h	Error Frame Received Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
7	BUF_UNDERRUN	W	0h	Receive Buffer Underrun Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (R/W) = Writing a 0 to this bit will have no effect. 1h (R/W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
6	FRAME_DONE	W	0h	Frame Done Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
5	BUF_OVERRUN	W	0h	Receive Buffer Overrun Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
4	EOF_ERR	W	0h	End-of-Frame Error Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
3	TYPE_ERR	W	0h	Frame Type Error Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
2	CRC_ERR	W	0h	CRC Error Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
1	FRAME_WD_TO	W	0h	Frame Watchdog Timeout Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn
0	PING_WD_TO	W	0h	Ping Watchdog Timeout Flag Clear bit This bit clears the corresponding bit in the RX_EVT_STS register. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Writing a 1 to this bit will clear the corresponding bit in the RX_EVT_STS register to 0. Reset type: SYSRSn

31.6.3.9 RX_EVT_FRC Register (Offset = Dh) [Reset = 0h]

RX_EVT_FRC is shown in Figure 31-51 and described in Table 31-51.

Return to the Summary Table.

Receive event and error flag force register

Figure 31-51. RX_EVT_FRC Register

15	14	13	12	11	10	9	8
RESERVED	ERROR_TAG_MATCH	DATA_TAG_MATCH	PING_TAG_MATCH	DATA_FRAME	FRAME_OVERRUN	PING_FRAME	ERR_FRAME
R-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h
7	6	5	4	3	2	1	0
BUF_UNDERRUN	FRAME_DONE	BUF_OVERRUN	EOF_ERR	TYPE_ERR	CRC_ERR	FRAME_WD_TO	PING_WD_TO
W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h	W-0h

Table 31-51. RX_EVT_FRC Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	ERROR_TAG_MATCH	W	0h	Error Tag Match Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn
13	DATA_TAG_MATCH	W	0h	Data Tag Match Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn
12	PING_TAG_MATCH	W	0h	Ping Tag Match Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn
11	DATA_FRAME	W	0h	Data Frame Received Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn
10	FRAME_OVERRUN	W	0h	Frame Overrun Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn

Table 31-51. RX_EVT_FRC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	PING_FRAME	W	0h	Ping Frame Received Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn
8	ERR_FRAME	W	0h	Error Frame Received Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn
7	BUF_UNDERRUN	W	0h	Receive Buffer Underrun Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn
6	FRAME_DONE	W	0h	Frame Done Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn
5	BUF_OVERRUN	W	0h	Receive Buffer Overrun Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn
4	EOF_ERR	W	0h	End-of-Frame Error Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn
3	TYPE_ERR	W	0h	Frame Type Error Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn
2	CRC_ERR	W	0h	CRC Error Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn

Table 31-51. RX_EVT_FRC Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	FRAME_WD_TO	W	0h	Frame Watchdog Timeout Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn
0	PING_WD_TO	W	0h	Ping Watchdog Timeout Flag Force bit This bit will cause the corresponding bit in the RX_EVT_STS register to get set. The purpose of this register is to allow software to simulate the effect of the event and test the associated software/ISR. 0h (W) = Writing a 0 to this bit will have no effect. 1h (W) = Force the corresponding bit in the RX_EVT_STS Register. Reset type: SYSRSn

31.6.3.10 RX_BUF_PTR_LOAD Register (Offset = Eh) [Reset = 0h]

RX_BUF_PTR_LOAD is shown in Figure 31-52 and described in Table 31-52.

Return to the Summary Table.

Receive buffer pointer load register

Figure 31-52. RX_BUF_PTR_LOAD Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				BUF_PTR_LOAD			
R-0h				R/W-0h			

Table 31-52. RX_BUF_PTR_LOAD Register Field Descriptions

Bit	Field	Type	Reset	Description
15-4	RESERVED	R	0h	Reserved
3-0	BUF_PTR_LOAD	R/W	0h	Buffer Pointer Load. This is the value to be loaded into the receive word pointer when written. This is to allow software to force the receiver to start storing the received data starting at a specific location in the buffer. NOTE: The value of the CURR_BUF_PTR in the RX_BUF_PTR_STS will not get reflected immediately. This will take effect only when there is a valid receive operation with incoming clocks after (3 RXCLK + 3 SYCLK) cycles. Reset type: SYSRSn

31.6.3.11 RX_BUF_PTR_STS Register (Offset = Fh) [Reset = 0h]

RX_BUF_PTR_STS is shown in Figure 31-53 and described in Table 31-53.

Return to the Summary Table.

Receive buffer pointer status register

Figure 31-53. RX_BUF_PTR_STS Register

15	14	13	12	11	10	9	8
RESERVED			CURR_WORD_CNT				
R-0h			R-0h				
7	6	5	4	3	2	1	0
RESERVED				CURR_BUF_PTR			
R-0h				R-0h			

Table 31-53. RX_BUF_PTR_STS Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12-8	CURR_WORD_CNT	R	0h	Words Available in the Receive Buffer This bitfield indicates the number of valid data words present in the receive buffer that have not been read by the application software. This bitfield is only valid when there is no active transfer. Note: This value will not be valid if there has been a buffer overrun or underrun condition. Reset type: SYSRSn
7-4	RESERVED	R	0h	Reserved
3-0	CURR_BUF_PTR	R	0h	Current Buffer Pointer Index This bitfield will show the current index of the buffer pointer. This value is only valid when there is no active transmission. Reset type: SYSRSn

31.6.3.12 RX_FRAME_WD_CTRL Register (Offset = 10h) [Reset = 0h]

RX_FRAME_WD_CTRL is shown in Figure 31-54 and described in Table 31-54.

Return to the Summary Table.

Receive frame watchdog control register

Figure 31-54. RX_FRAME_WD_CTRL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						FRAME_WD_EN	FRAME_WD_CNT_RST
R-0h						R/W-0h	R/W-0h

Table 31-54. RX_FRAME_WD_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-2	RESERVED	R	0h	Reserved
1	FRAME_WD_EN	R/W	0h	Frame Watchdog Counter Enable bit This bit will enable or disable the frame watchdog counter. The counter (RX_FRAME_WD_CNT) will begin counting from 0 when a valid start-of-frame pattern is received. When the reference value (RX_FRAME_WD_REF) is reached, it will generate a frame watchdog timeout event (RX_EVT_STS.FRAME_WD_TO) and the counter value will reset to 0 and continue counting on the next valid start-of-frame. 0h (R/W) = The frame watchdog counter is disabled and not running. 1h (R/W) = The frame watchdog counter logic is enabled and running. Reset type: SYSRSn
0	FRAME_WD_CNT_RST	R/W	0h	Frame Watchdog Counter Reset bit This bit will reset the frame watchdog counter to 0. Writing a 1 to this bit will reset the frame watchdog counter to 0. The counter will stay in reset as long as this bit is set to 1. This bit needs to be cleared to 0 to use the counter 0h (R/W) = Clear the FRAME_WD_CNT_RST. 1h (W) = The frame watchdog counter will be reset to 0. Reset type: SYSRSn

31.6.3.13 RX_FRAME_WD_REF Register (Offset = 12h) [Reset = 0h]

RX_FRAME_WD_REF is shown in Figure 31-55 and described in Table 31-55.

Return to the Summary Table.

Receive frame watchdog counter reference

Figure 31-55. RX_FRAME_WD_REF Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRAME_WD_REF																															
R/W-0h																															

Table 31-55. RX_FRAME_WD_REF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FRAME_WD_REF	R/W	0h	Frame Watchdog Counter Reference Value This is the 32-bit reference value for the frame watchdog timeout counter. The counter will count up starting from 0 at a valid start-of-frame pattern and continue counting until this value is reached. Reset type: SYSRSn

31.6.3.14 RX_FRAME_WD_CNT Register (Offset = 14h) [Reset = 0h]

RX_FRAME_WD_CNT is shown in Figure 31-56 and described in Table 31-56.

Return to the Summary Table.

Receive frame watchdog current count

Figure 31-56. RX_FRAME_WD_CNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FRAME_WD_CNT																															
R-0h																															

Table 31-56. RX_FRAME_WD_CNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	FRAME_WD_CNT	R	0h	Frame Watchdog Counter Value This is the 32-bit read-only register which shows the current value of the frame watchdog counter. This counter is reset to 0 in a variety of ways: A write to FRME_WD_CNT_RST, a match with FRAME_WD_REF, or the reception of a successful data frame. Reset type: SYSRSn

31.6.3.15 RX_PING_WD_CTRL Register (Offset = 16h) [Reset = 0h]

RX_PING_WD_CTRL is shown in Figure 31-57 and described in Table 31-57.

Return to the Summary Table.

Receive ping watchdog control register

Figure 31-57. RX_PING_WD_CTRL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						PING_WD_EN	PING_WD_RST
R-0h						R/W-0h	R/W-0h

Table 31-57. RX_PING_WD_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-2	RESERVED	R	0h	Reserved
1	PING_WD_EN	R/W	0h	Ping Watchdog Counter Enable bit This bit will enable or disable the ping watchdog counter. The counter (RX_PING_WD_CNT) will begin counting from 0 when it is enabled. When the reference value (RX_PING_WD_REF) is reached, it will generate a ping watchdog timeout event (RX_EVT_STS.PING_WD_TO) and the counter value will reset to 0, and resume counting 0h (R/W) = The ping watchdog counter is disabled and not running. 1h (R/W) = The ping watchdog counter logic is enabled and running. Reset type: SYSRSn
0	PING_WD_RST	R/W	0h	Ping Watchdog Counter Reset bit This bit will reset the ping watchdog counter to 0. Writing a 1 to this bit will reset the ping watchdog counter to 0. The counter will stay in reset as long as this bit is set to 1. This bit needs to be cleared to 0 to use the counter 0h (R/W) = Clear the PING_WD_RST. 1h (W) = The ping watchdog counter will be reset to 0. Reset type: SYSRSn

31.6.3.16 RX_PING_TAG Register (Offset = 17h) [Reset = 0h]

RX_PING_TAG is shown in Figure 31-58 and described in Table 31-58.

Return to the Summary Table.

Receive ping tag register

Figure 31-58. RX_PING_TAG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			PING_TAG				RESERVED
R-0h			R-0h				R-0h

Table 31-58. RX_PING_TAG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R	0h	Reserved
4-1	PING_TAG	R	0h	Received Ping Frame Tag This field contains the 4-bit frame tag from the last successfully received ping frame. This is intentionally shifted into bits 4:1 so that the register can be used as a 32-bit address index based on the received tag. Reset type: SYSRSn
0	RESERVED	R	0h	Reserved

31.6.3.17 RX_PING_WD_REF Register (Offset = 18h) [Reset = 0h]

RX_PING_WD_REF is shown in Figure 31-59 and described in Table 31-59.

Return to the Summary Table.

Receive ping watchdog counter reference

Figure 31-59. RX_PING_WD_REF Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PING_WD_REF																															
R/W-0h																															

Table 31-59. RX_PING_WD_REF Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PING_WD_REF	R/W	0h	Ping Watchdog Counter Reference Value This is the 32-bit reference value for the ping watchdog timeout counter. The counter will count up starting from 0 and continue counting until this value is reached. Reset type: SYSRSn

31.6.3.18 RX_PING_WD_CNT Register (Offset = 1Ah) [Reset = 0h]

RX_PING_WD_CNT is shown in Figure 31-60 and described in Table 31-60.

Return to the Summary Table.

Receive pingwatchdog current count

Figure 31-60. RX_PING_WD_CNT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PING_WD_CNT																															
R-0h																															

Table 31-60. RX_PING_WD_CNT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PING_WD_CNT	R	0h	Ping Watchdog Counter Value This is the 32-bit read-only register which shows the current value of the ping watchdog counter. This counter is reset to 0 in a variety of ways: A write to PING_WD_RST, a match with PING_WD_REF, or the reception of a ping frame. Reset type: SYSRSn

31.6.3.19 RX_INT1_CTRL Register (Offset = 1Ch) [Reset = 0h]

RX_INT1_CTRL is shown in Figure 31-61 and described in Table 31-61.

Return to the Summary Table.

Receive interrupt control register for RX_INT1

Figure 31-61. RX_INT1_CTRL Register

15	14	13	12	11	10	9	8
RESERVED	INT1_EN_ERROR_TAG_MATCH	INT1_EN_DATA_TAG_MATCH	INT1_EN_PING_TAG_MATCH	INT1_EN_DATA_FRAME	INT1_EN_FRAME_OVERRUN	INT1_EN_PING_FRAME	INT1_EN_ERR_FRAME
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INT1_EN_UNDERRUN	INT1_EN_FRAME_DONE	INT1_EN_OVERRUN	INT1_EN_EOF_ERR	INT1_EN_TYPE_ERR	INT1_EN_CRC_ERR	INT1_EN_FRAME_WD_TO	INT1_EN_PING_WD_TO
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 31-61. RX_INT1_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	INT1_EN_ERROR_TAG_MATCH	R/W	0h	Enable Error Frame Received with Tag Match Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = An error frame received with matching tag will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
13	INT1_EN_DATA_TAG_MATCH	R/W	0h	Enable Data Frame Received with Tag Match Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A data frame received with matching tag will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
12	INT1_EN_PING_TAG_MATCH	R/W	0h	Enable Ping Frame Received with Tag Match Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A ping frame received with matching tag will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
11	INT1_EN_DATA_FRAME	R/W	0h	Enable Data Frame Received Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A data frame received event will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
10	INT1_EN_FRAME_OVERRUN	R/W	0h	Enable Frame Overrun Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A frame overrun event will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn

Table 31-61. RX_INT1_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	INT1_EN_PING_FRAME	R/W	0h	Enable Ping Frame Received Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A ping frame received event will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
8	INT1_EN_ERR_FRAME	R/W	0h	Enable ERROR Frame Received Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A error frame received event will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
7	INT1_EN_UNDERRUN	R/W	0h	Enable Buffer Underrun Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A buffer underrun event will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
6	INT1_EN_FRAME_DONE	R/W	0h	Enable Frame Done Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A frame done event will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
5	INT1_EN_OVERRUN	R/W	0h	Enable Receive Buffer Overrun Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A receive buffer overrun event will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
4	INT1_EN_EOF_ERR	R/W	0h	Enable End-of-Frame Error Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = An end-of-frame error event will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
3	INT1_EN_TYPE_ERR	R/W	0h	Enable Frame Type Error Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A frame type error event will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn

Table 31-61. RX_INT1_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	INT1_EN_CRC_ERR	R/W	0h	Enable CRC Error Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A CRC error will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
1	INT1_EN_FRAME_WD_TO	R/W	0h	Enable Frame Watchdog Timeout Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A frame watchdog timeout event will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
0	INT1_EN_PING_WD_TO	R/W	0h	Enable Ping Watchdog Timeout Interrupt to INT1 bit This is an enable register which decides whether an interrupt (RX_INT1) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT1. 1h (R/W) = A ping watchdog timeout event will trigger an interrupt on RX_INT1. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn

31.6.3.20 RX_INT2_CTRL Register (Offset = 1Dh) [Reset = 0h]

RX_INT2_CTRL is shown in Figure 31-62 and described in Table 31-62.

Return to the Summary Table.

Receive interrupt control register for RX_INT2

Figure 31-62. RX_INT2_CTRL Register

15	14	13	12	11	10	9	8
RESERVED	INT2_EN_ERROR_TAG_MATCH	INT2_EN_DATA_TAG_MATCH	INT2_EN_PING_TAG_MATCH	INT2_EN_DATA_FRAME	INT2_EN_FRAME_OVERRUN	INT2_EN_PING_FRAME	INT2_EN_ERR_FRAME
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
INT2_EN_UNDERRUN	INT2_EN_FRAME_DONE	INT2_EN_OVERRUN	INT2_EN_EOF_ERR	INT2_EN_TYPE_ERR	INT2_EN_CRC_ERR	INT2_EN_FRAME_WD_TO	INT2_EN_PING_WD_TO
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 31-62. RX_INT2_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14	INT2_EN_ERROR_TAG_MATCH	R/W	0h	Enable Error Frame Received with Tag Match Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = An error frame received with matching tag will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
13	INT2_EN_DATA_TAG_MATCH	R/W	0h	Enable Data Frame Received with Tag Match Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A data frame received with matching tag will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
12	INT2_EN_PING_TAG_MATCH	R/W	0h	Enable Ping Frame Received with Tag Match Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A ping frame received with matching tag will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
11	INT2_EN_DATA_FRAME	R/W	0h	Enable Data Frame Received Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A data frame received event will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
10	INT2_EN_FRAME_OVERRUN	R/W	0h	Enable Frame Overrun Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A frame overrun event will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn

Table 31-62. RX_INT2_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
9	INT2_EN_PING_FRAME	R/W	0h	Enable Ping Frame Received Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A ping frame received event will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
8	INT2_EN_ERR_FRAME	R/W	0h	Enable Error Frame Received Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A error frame received event will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
7	INT2_EN_UNDERRUN	R/W	0h	Enable Buffer Underrun Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A buffer underrun event will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
6	INT2_EN_FRAME_DONE	R/W	0h	Enable Frame Done Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A frame done event will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
5	INT2_EN_OVERRUN	R/W	0h	Enable Buffer Overrun Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A buffer overrun event will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
4	INT2_EN_EOF_ERR	R/W	0h	Enable End-of-Frame Error Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = An end-of-frame error event will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
3	INT2_EN_TYPE_ERR	R/W	0h	Enable Frame Type Error Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A frame type error event will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn

Table 31-62. RX_INT2_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	INT2_EN_CRC_ERR	R/W	0h	Enable CRC Error Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A CRC error will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
1	INT2_EN_FRAME_WD_TO	R/W	0h	Enable Frame Watchdog Timeout Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A frame watchdog timeout event will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn
0	INT2_EN_PING_WD_TO	R/W	0h	Enable Ping Watchdog Timeout Interrupt to INT2 bit This is an enable register which decides whether an interrupt (RX_INT2) will be generated on the enabled event. 0h (R/W) = This event will not trigger an interrupt on RX_INT2. 1h (R/W) = A ping watchdog timeout event will trigger an interrupt on RX_INT2. The event itself will be latched in the corresponding bit in the RX_EVT_STS Register Reset type: SYSRSn

31.6.3.21 RX_LOCK_CTRL Register (Offset = 1Eh) [Reset = 0h]

RX_LOCK_CTRL is shown in Figure 31-63 and described in Table 31-63.

Return to the Summary Table.

Receive lock control register

Figure 31-63. RX_LOCK_CTRL Register

15	14	13	12	11	10	9	8
KEY							
W-0h							
7	6	5	4	3	2	1	0
RESERVED							LOCK
R-0h							R/W-0h

Table 31-63. RX_LOCK_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	KEY	W	0h	Write Key. In order to write to this register, 0xA5 must be written to this field at the same time. Otherwise, writes are ignored. The key is cleared immediately after writing, so it must be written again for every change to this register. Reset type: SYSRSn
7-1	RESERVED	R	0h	Reserved
0	LOCK	R/W	0h	Control Register Lock Enable bit This bit locks the contents of all the receive control registers that support a lock protection. Once locked, further writes will not take effect until SYSRS unlocks the register. Once set, further writes even to this bit will be ignored. 0h (R/W) = Receive control registers can be modified and are not locked. 1h (R/W) = Receive control registers are locked and cannot be modified until this bit is cleared by SYSRS. Any further writes to this bit are ignored. Note: The KEY field must contatin 0xA5 for any write to this bit to take effect. Reset type: SYSRSn

31.6.3.22 RX_ECC_DATA Register (Offset = 20h) [Reset = 0h]

RX_ECC_DATA is shown in Figure 31-64 and described in Table 31-64.

Return to the Summary Table.

Receive ECC data register

Figure 31-64. RX_ECC_DATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA_HIGH																DATA_LOW															
R/W-0h																R/W-0h															

Table 31-64. RX_ECC_DATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	DATA_HIGH	R/W	0h	Upper 16 bits of ECC Data Writing to this bitfield will cause the ECC logic to compute the ECC(SEC-DED) the entire 32-bit register and update TX_ECC_VAL register with the results. Software should write to these 16 bits of the register in a 32-bit write when needing to compute ECC for 32-bits for the full TX_ECC_DATA register. Reset type: SYSRSn
15-0	DATA_LOW	R/W	0h	Lower 16 bits of ECC Data Writing to this bitfield will cause the ECC logic to compute the ECC(SEC-DED) for these 16 bits and update the TX_ECC_VAL register with the results. Software should write to these register bits as a 16-bit write when needing to compute ECC for 16-bits. Reset type: SYSRSn

31.6.3.23 RX_ECC_VAL Register (Offset = 22h) [Reset = 0h]

RX_ECC_VAL is shown in Figure 31-65 and described in Table 31-65.

Return to the Summary Table.

Receive ECC value register

Figure 31-65. RX_ECC_VAL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	ECC_VAL						
R-0h	R/W-0h						

Table 31-65. RX_ECC_VAL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-7	RESERVED	R	0h	Reserved
6-0	ECC_VAL	R/W	0h	ECC Value for SEC-DED check This field contains the ECC value to be used for SEC-DED either for 16-bit or 32-bit data in the RX_ECC_DATA register. Reset type: SYSRSn

31.6.3.24 RX_ECC_SEC_DATA Register (Offset = 24h) [Reset = 0h]

RX_ECC_SEC_DATA is shown in Figure 31-66 and described in Table 31-66.

Return to the Summary Table.

Receive ECC corrected data register

Figure 31-66. RX_ECC_SEC_DATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEC_DATA																															
R-0h																															

Table 31-66. RX_ECC_SEC_DATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SEC_DATA	R	0h	ECC Single Error Corrected Data The ECC corrected data will be available in this register. This value is valid only when there are no bit errors, or a single bit error was detected. Otherwise, the contents of this register are invalid and should not be used. Reset type: SYSRSn

31.6.3.25 RX_ECC_LOG Register (Offset = 26h) [Reset = 3h]

RX_ECC_LOG is shown in Figure 31-67 and described in Table 31-67.

Return to the Summary Table.

Receive ECC log and status register

Figure 31-67. RX_ECC_LOG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						MBE	SBE
R-0h						R-1h	R-1h

Table 31-67. RX_ECC_LOG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-2	RESERVED	R	0h	Reserved
1	MBE	R	1h	Multiple Bit Errors Detected This bit indicates the occurrence of multiple bit errors.The data is corrupted and cannot be corrected. If this bit is set, the data present in RX_ECC_SEC_DATA is invalid and should not be used. 0h (R) Multiple Bit Errors were not detected. Check the SBE bit for single bit errors. 1h (R) Multiple Bit Errors were detected. The data is not able to be corrected. The value present in RX_ECC_SEC_DATA is invalid and should not be used. Reset type: SYSRSn
0	SBE	R	1h	Single Bit Error Detected This bit indicates the occurrence of a single bit error in the data. The data is autocorrected and placed into the RX_ECC_SEC_DATA register. This bit is valid only if MBE is 0. 0h (R) No bit errors were detected. The value in RX_ECC_SEC_DATA is correct. 1h (R) A single bit error was detected and corrected. The corrected data is present in RX_ECC_SEC_DATA. Reset type: SYSRSn

31.6.3.26 RX_FRAME_TAG_CMP Register (Offset = 28h) [Reset = 0h]

RX_FRAME_TAG_CMP is shown in Figure 31-68 and described in Table 31-68.

Return to the Summary Table.

Receive frame tag compare register

Figure 31-68. RX_FRAME_TAG_CMP Register

15	14	13	12	11	10	9	8
RESERVED						BROADCAST_EN	CMP_EN
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
TAG_MASK				TAG_REF			
R/W-0h				R/W-0h			

Table 31-68. RX_FRAME_TAG_CMP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9	BROADCAST_EN	R/W	0h	Broadcast Enable bit This will enable the reception of a ping frame broadcast. When this bit is set, bit 3 of the received tag will be treated as a broadcast notification. If bit 3 of the received tag is set to 1, a ping tag match event will be triggered regardless of the. A match caused by the comparison of TAG_MASK and TAG_REF will still be considered a match and the frame tag match event will be triggered as normal This bit only takes effect only if CMP_EN is set to 1. 0h (R/W) Broadcast frame match disabled. 1h (R/W) Broadcast frame match enabled. Reset type: SYSRSn
8	CMP_EN	R/W	0h	Frame Tag Compare Enable bit Set this bit to enable the comparison of an incoming frame tag and the value stored in the frame tag reference. A match caused by the comparison of TAG_MASK, TAG_REF, and the incoming frame tag will trigger the apprpriate frame tag match event. 0h (R/W) Frame tag comparison is disabled. 1h (R/W) Frame tag comparison is enabled. Reset type: SYSRSn
7-4	TAG_MASK	R/W	0h	Frame Tag Mask Any bit position in this register set to 0 will be used in the comparison of the incoming frame tag and the value stored in TAG_REF. A bit position set to 1 will be ignored in the tag comparison. This mask value is used only for non-ping frames. Reset type: SYSRSn
3-0	TAG_REF	R/W	0h	Frame Tag Reference The reference tag to check against when comparing the TAG_MASK and the incoming frame tag. This reference value is used only for non-ping frames. Reset type: SYSRSn

31.6.3.27 RX_PING_TAG_CMP Register (Offset = 29h) [Reset = 0h]

RX_PING_TAG_CMP is shown in Figure 31-69 and described in Table 31-69.

Return to the Summary Table.

Receive ping tag compare register

Figure 31-69. RX_PING_TAG_CMP Register

15	14	13	12	11	10	9	8
RESERVED						BROADCAST_EN	CMP_EN
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
TAG_MASK				TAG_REF			
R/W-0h				R/W-0h			

Table 31-69. RX_PING_TAG_CMP Register Field Descriptions

Bit	Field	Type	Reset	Description
15-10	RESERVED	R	0h	Reserved
9	BROADCAST_EN	R/W	0h	Broadcast Enable bit This will enable the reception of a ping frame broadcast. When this bit is set, bit 3 of the received tag will be treated as a broadcast notification. If bit 3 of the received tag is set to 1, a ping tag match event will be triggered regardless of the. A match caused by the comparison of TAG_MASK and TAG_REF will still be considered a match and the ping tag match event will be triggered as normal This bit only takes effect only if CMP_EN is set to 1. 0h (R/W) Broadcast frame match disabled. 1h (R/W) Broadcast frame match enabled. Reset type: SYSRSn
8	CMP_EN	R/W	0h	Ping Tag Compare Enable bit Set this bit to enable the comparison of an incoming ping tag and the value stored in the ping tag reference. A match caused by the comparison of TAG_MASK, TAG_REF, and the incoming ping tag will trigger a ping frame tag match event. 0h (R/W) Ping tag comparison is disabled. 1h (R/W) Ping tag comparison is enabled. Reset type: SYSRSn
7-4	TAG_MASK	R/W	0h	Ping Tag Mask Any bit position in this register set to 0 will be used in the comparison of the incoming ping frame tag and the value stored in TAG_REF. A bit position set to 1 will be ignored in the tag comparison. This mask value is used only for ping frames. Reset type: SYSRSn
3-0	TAG_REF	R/W	0h	Ping Tag Reference The reference tag to check against when comparing the TAG_MASK and the incoming ping tag. This reference value is used only for ping frames. Reset type: SYSRSn

31.6.3.28 RX_TRIG_CTRL_0 Register (Offset = 2Ch) [Reset = 0h]

RX_TRIG_CTRL_0 is shown in Figure 31-70 and described in Table 31-70.

Return to the Summary Table.

Receive Trigger Control register 0

Figure 31-70. RX_TRIG_CTRL_0 Register

31	30	29	28	27	26	25	24
RX_TRIG_DLY							
R/W-0h							
23	22	21	20	19	18	17	16
RX_TRIG_DLY							
R/W-0h							
15	14	13	12	11	10	9	8
RX_TRIG_DLY							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED			TRIG_SEL				TRIG_EN
R-0h			R/W-0h				R/W-0h

Table 31-70. RX_TRIG_CTRL_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RX_TRIG_DLY	R/W	0h	This is the 24 bit count of the trigger delay in SYSCLK cycles. If enabled, the Trigger-1 output of the trigger module will generate a 3 SYSCLK wide trigger pulse after the selected input trigger source sees a rising edge with a delay defined by this 24-bt value. Reset type: SYSRSn
7-5	RESERVED	R	0h	Reserved
4-1	TRIG_SEL	R/W	0h	This is the mux select value which selects which of the inputs will be used as the trigger source. Reset type: SYSRSn
0	TRIG_EN	R/W	0h	This is the enable for the RX output trigger generation. The output triggers will be generated only if this bit is set to 1. If this bit is 0, then no trigger will be generated by this module. Reset type: SYSRSn

31.6.3.29 RX_TRIG_WIDTH_0 Register (Offset = 2Eh) [Reset = 0h]

RX_TRIG_WIDTH_0 is shown in Figure 31-71 and described in Table 31-71.

Return to the Summary Table.

Receive Trigger Wdith register 0

Figure 31-71. RX_TRIG_WIDTH_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																RX_TRIG_WIDTH															
R-0h																R/W-0h															

Table 31-71. RX_TRIG_WIDTH_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	RX_TRIG_WIDTH	R/W	0h	This register decides the width(in SYSCLK cycles) of wide pulse output of the RX trigger module. Reset type: SYSRSn

31.6.3.30 RX_DLYLINE_CTRL Register (Offset = 30h) [Reset = 0h]

RX_DLYLINE_CTRL is shown in Figure 31-72 and described in Table 31-72.

Return to the Summary Table.

Receive delay line control register

Figure 31-72. RX_DLYLINE_CTRL Register

15	14	13	12	11	10	9	8
RESERVED	RXD1_DLY					RXD0_DLY	
R-0h	R/W-0h					R/W-0h	
7	6	5	4	3	2	1	0
RXD0_DLY			RXCLK_DLY				
R/W-0h			R/W-0h				

Table 31-72. RX_DLYLINE_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15	RESERVED	R	0h	Reserved
14-10	RXD1_DLY	R/W	0h	Delay Line Tap Select for RXD1 This bitfield selects the number of delay elements inserted into the RXD1 path from the pin boundary to the receiver core. 0h (R/W) Zero delay elements are included in the RXD1 path. RXD1 is taken directly from the pin. 1h (R/W) One delay element is included in the RXD1 path. 2h (R/W) Two delay elements are included in the RXD1 path. ... 1Fh (R/W) 31 delay elements are included in the RXD1 path, the maximum. Reset type: SYSRSn
9-5	RXD0_DLY	R/W	0h	Delay Line Tap Select for RXD0 This bitfield selects the number of delay elements inserted into the RXD0 path from the pin boundary to the receiver core. 0h (R/W) Zero delay elements are included in the RXD0 path. RXD0 is taken directly from the pin. 1h (R/W) One delay element is included in the RXD0 path. 2h (R/W) Two delay elements are included in the RXD0 path. ... 1Fh (R/W) 31 delay elements are included in the RXD0 path, the maximum. Reset type: SYSRSn
4-0	RXCLK_DLY	R/W	0h	Delay Line Tap Select for RXCLK This bitfield selects the number of delay elements inserted into the RXCLK path from the pin boundary to the receiver core. 0h (R/W) Zero delay elements are included in the RXCLK path. RXCLK is taken directly from the pin. 1h (R/W) One delay element is included in the RXCLK path. 2h (R/W) Two delay elements are included in the RXCLK path. ... 1Fh (R/W) 31 delay elements are included in the RXCLK path, the maximum. Reset type: SYSRSn

31.6.3.31 RX_TRIG_CTRL_1 Register (Offset = 32h) [Reset = 0h]

RX_TRIG_CTRL_1 is shown in Figure 31-73 and described in Table 31-73.

Return to the Summary Table.

Receive Trigger Control register 1

Figure 31-73. RX_TRIG_CTRL_1 Register

31	30	29	28	27	26	25	24
RX_TRIG_DLY							
R/W-0h							
23	22	21	20	19	18	17	16
RX_TRIG_DLY							
R/W-0h							
15	14	13	12	11	10	9	8
RX_TRIG_DLY							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED			TRIG_SEL				TRIG_EN
R-0h			R/W-0h				R/W-0h

Table 31-73. RX_TRIG_CTRL_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RX_TRIG_DLY	R/W	0h	This is the 24 bit count of the trigger delay in SYSCLK cycles. If enabled, the Trigger-1 output of the trigger module will generate a 3 SYSCLK wide trigger pulse after the selected input trigger source sees a rising edge with a delay defined by this 24-bt value. Reset type: SYSRSn
7-5	RESERVED	R	0h	Reserved
4-1	TRIG_SEL	R/W	0h	This is the mux select value which selects which of the inputs will be used as the trigger source. Reset type: SYSRSn
0	TRIG_EN	R/W	0h	This is the enable for the RX output trigger generation. The output triggers will be generated only if this bit is set to 1. If this bit is 0, then no trigger will be generated by this module. Reset type: SYSRSn

31.6.3.32 RX_TRIG_CTRL_2 Register (Offset = 34h) [Reset = 0h]

RX_TRIG_CTRL_2 is shown in Figure 31-74 and described in Table 31-74.

Return to the Summary Table.

Receive Trigger Control register 2

Figure 31-74. RX_TRIG_CTRL_2 Register

31	30	29	28	27	26	25	24
RX_TRIG_DLY							
R/W-0h							
23	22	21	20	19	18	17	16
RX_TRIG_DLY							
R/W-0h							
15	14	13	12	11	10	9	8
RX_TRIG_DLY							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED			TRIG_SEL				TRIG_EN
R-0h			R/W-0h				R/W-0h

Table 31-74. RX_TRIG_CTRL_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RX_TRIG_DLY	R/W	0h	This is the 24 bit count of the trigger delay in SYSCLK cycles. If enabled, the Trigger-1 output of the trigger module will generate a 3 SYSCLK wide trigger pulse after the selected input trigger source sees a rising edge with a delay defined by this 24-bt value. Reset type: SYSRSn
7-5	RESERVED	R	0h	Reserved
4-1	TRIG_SEL	R/W	0h	This is the mux select value which selects which of the inputs will be used as the trigger source. Reset type: SYSRSn
0	TRIG_EN	R/W	0h	This is the enable for the RX output trigger generation. The output triggers will be generated only if this bit is set to 1. If this bit is 0, then no trigger will be generated by this module. Reset type: SYSRSn

31.6.3.33 RX_TRIG_CTRL_3 Register (Offset = 36h) [Reset = 0h]

RX_TRIG_CTRL_3 is shown in Figure 31-75 and described in Table 31-75.

Return to the Summary Table.

Receive Trigger Control register 3

Figure 31-75. RX_TRIG_CTRL_3 Register

31	30	29	28	27	26	25	24
RX_TRIG_DLY							
R/W-0h							
23	22	21	20	19	18	17	16
RX_TRIG_DLY							
R/W-0h							
15	14	13	12	11	10	9	8
RX_TRIG_DLY							
R/W-0h							
7	6	5	4	3	2	1	0
RESERVED			TRIG_SEL				TRIG_EN
R-0h			R/W-0h				R/W-0h

Table 31-75. RX_TRIG_CTRL_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RX_TRIG_DLY	R/W	0h	This is the 24 bit count of the trigger delay in SYSCLK cycles. If enabled, the Trigger-1 output of the trigger module will generate a 3 SYSCLK wide trigger pulse after the selected input trigger source sees a rising edge with a delay defined by this 24-bt value. Reset type: SYSRSn
7-5	RESERVED	R	0h	Reserved
4-1	TRIG_SEL	R/W	0h	This is the mux select value which selects which of the inputs will be used as the trigger source. Reset type: SYSRSn
0	TRIG_EN	R/W	0h	This is the enable for the RX output trigger generation. The output triggers will be generated only if this bit is set to 1. If this bit is 0, then no trigger will be generated by this module. Reset type: SYSRSn

31.6.3.34 RX_VIS_1 Register (Offset = 38h) [Reset = 0h]

RX_VIS_1 is shown in Figure 31-76 and described in Table 31-76.

Return to the Summary Table.

Receive debug visibility register 1

Figure 31-76. RX_VIS_1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RX_CORE_STS	RESERVED		
R-0h				R-0h	R-0h		

Table 31-76. RX_VIS_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	RX_CORE_STS	R	0h	Receiver Core Status bit This bit indicates the status of the receiver core. If this bit is set, the receiver should undergo a reset and subsequent resynchronization with the transmitter. This bit will be always be set when the receiver has detected and end of frame error or a frame type error. This bit can also be set if the receiver becomes corrupted due to noise on the signal lines. If the receiver has experienced a ping watchdog or frame watchdog timeout, this bit should be read to determine if the cause was due to a corrupt transaction, thus putting the receiver core into an unrecoverable state. Only a soft reset will reset the recevier core and thus reset this bit. 0h (R) The receiver core is operating normally. 1h (R) The receiver core has entered into an error state and should be reset. Reset type: SYSRSn
2-0	RESERVED	R	0h	Reserved

31.6.3.35 RX_UDATA_FILTER Register (Offset = 3Ah) [Reset = 0h]

RX_UDATA_FILTER is shown in Figure 31-77 and described in Table 31-77.

Return to the Summary Table.

Receive User Data Filter Control register

Figure 31-77. RX_UDATA_FILTER Register

15	14	13	12	11	10	9	8
UDATA_MASK							
R/W-0h							
7	6	5	4	3	2	1	0
UDATA_REF							
R/W-0h							

Table 31-77. RX_UDATA_FILTER Register Field Descriptions

Bit	Field	Type	Reset	Description
15-8	UDATA_MASK	R/W	0h	Bit Mask to be used for comparing the USERDATA field when filtering is enabled. Every bit that is '1' in this register will be masked for comparison. If a bit position is '1', then it will be considered a successful match for that bit position. Reset type: SYSRSn
7-0	UDATA_REF	R/W	0h	Reference to be used for comparing the USERDATA field when filtering is enabled. Reset type: SYSRSn

31.6.3.36 RX_BUF_BASE_y Register (Offset = 40h + formula) [Reset = 0h]

RX_BUF_BASE_y is shown in Figure 31-78 and described in Table 31-78.

Return to the Summary Table.

Base address for receive data buffer

Offset = 40h + (y * 1h); where y = 0h to Fh

Figure 31-78. RX_BUF_BASE_y Register

15	14	13	12	11	10	9	8
BASE_ADDRESS							
R-0h							
7	6	5	4	3	2	1	0
BASE_ADDRESS							
R-0h							

Table 31-78. RX_BUF_BASE_y Register Field Descriptions

Bit	Field	Type	Reset	Description
15-0	BASE_ADDRESS	R	0h	Receive Data Buffer Base Address This is the base address of the 16-word data buffer used by the receiver. Reset type: SYSRSn

31.6.4 FSI Registers to Driverlib Functions

Table 31-79. FSI Registers to Driverlib Functions

File	Driverlib Function
TX_MASTER_CTRL	
fsi.c	FSI_resetTxModule
fsi.c	FSI_clearTxModuleReset
fsi.h	FSI_sendTxFlush
fsi.h	FSI_stopTxFlush
TX_CLK_CTRL	
fsi.c	FSI_resetTxModule
fsi.c	FSI_clearTxModuleReset
fsi.h	FSI_enableTxClock
fsi.h	FSI_disableTxClock
fsi.h	FSI_configPrescalar
TX_OPER_CTRL_LO	
fsi.h	FSI_selectTxPLLClock
fsi.h	FSI_setTxDataWidth
fsi.h	FSI_enableTxSPIMode
fsi.h	FSI_disableTxSPIMode
fsi.h	FSI_setTxStartMode
fsi.h	FSI_setTxPingTimeoutMode
fsi.h	FSI_enableTxTDMMode
fsi.h	FSI_disableTxTDMMode
fsi.h	FSI_enableRxTDMMode
fsi.h	FSI_disableRxTDMMode
fsi.h	FSI_enableTxUserCRC
fsi.h	FSI_disableTxUserCRC

Table 31-79. FSI Registers to Driverlib Functions (continued)

File	Driverlib Function
TX_OPER_CTRL_HI	
fsi.h	FSI_setTxExtFrameTrigger
fsi.h	FSI_enableTxCRCForceError
fsi.h	FSI_disableTxCRCForceError
fsi.h	FSI_setTxECCComputeWidth
TX_FRAME_CTRL	
fsi.h	FSI_setTxFrameType
fsi.h	FSI_setTxSoftwareFrameSize
fsi.h	FSI_startTxTransmit
TX_FRAME_TAG_UDATA	
fsi.h	FSI_setTxFrameTag
fsi.h	FSI_setTxUserDefinedData
TX_BUF_PTR_LOAD	
fsi.h	FSI_setTxBufferPtr
TX_BUF_PTR_STS	
fsi.h	FSI_getTxBufferPtr
fsi.h	FSI_getTxWordCount
TX_PING_CTRL	
fsi.c	FSI_resetTxModule
fsi.c	FSI_clearTxModuleReset
fsi.h	FSI_enableTxPingTimer
fsi.h	FSI_disableTxPingTimer
fsi.h	FSI_enableTxExtPingTrigger
fsi.h	FSI_disableTxExtPingTrigger
TX_PING_TAG	
fsi.h	FSI_enableTxPingTimer
fsi.h	FSI_setTxPingTag
TX_PING_TO_REF	
fsi.h	FSI_enableTxPingTimer
TX_PING_TO_CNT	
fsi.h	FSI_getTxCurrentPingTimeoutCounter
TX_INT_CTRL	
fsi.h	FSI_enableTxInterrupt
fsi.h	FSI_disableTxInterrupt
TX_DMA_CTRL	
fsi.h	FSI_enableTxDMAEvent
fsi.h	FSI_disableTxDMAEvent
TX_LOCK_CTRL	
fsi.h	FSI_lockTxCtrl
TX_EVT_STS	
fsi.h	FSI_getTxEventStatus
TX_EVT_CLR	
fsi.h	FSI_clearTxEvents
TX_EVT_FRC	
fsi.h	FSI_forceTxEvents

Table 31-79. FSI Registers to Driverlib Functions (continued)

File	Driverlib Function
TX_USER_CRC	
fsi.h	FSI_enableTxUserCRC
TX_ECC_DATA	
fsi.h	FSI_setTxECCdata
TX_ECC_VAL	
fsi.h	FSI_getTxECCValue
TX_DLYLINE_CTRL	
-	
TX_BUF_BASE	
fsi.c	FSI_writeTxBuffer
fsi.h	FSI_getTxBufferAddress
RX_MASTER_CTRL	
fsi.c	FSI_resetRxModule
fsi.c	FSI_clearRxModuleReset
fsi.h	FSI_enableRxInternalLoopback
fsi.h	FSI_disableRxInternalLoopback
fsi.h	FSI_enableRxSPIPairing
fsi.h	FSI_disableRxSPIPairing
RX_OPER_CTRL	
fsi.h	FSI_setRxDataWidth
fsi.h	FSI_enableRxSPIMode
fsi.h	FSI_disableRxSPIMode
fsi.h	FSI_setRxSoftwareFrameSize
fsi.h	FSI_setRxECCComputeWidth
fsi.h	FSI_setRxPingTimeoutMode
RX_FRAME_INFO	
fsi.h	FSI_getRxFrameType
RX_FRAME_TAG_UDATA	
fsi.h	FSI_getRxFrameTag
fsi.h	FSI_getRxUserDefinedData
RX_DMA_CTRL	
fsi.h	FSI_enableRxDMAEvent
fsi.h	FSI_disableRxDMAEvent
RX_EVT_STS	
fsi.h	FSI_getRxEventStatus
RX_CRC_INFO	
fsi.h	FSI_getRxReceivedCRC
fsi.h	FSI_getRxComputedCRC
RX_EVT_CLR	
fsi.h	FSI_clearRxEvents
RX_EVT_FRC	
fsi.h	FSI_forceRxEvents
RX_BUF_PTR_LOAD	
fsi.h	FSI_setRxBufferPtr
RX_BUF_PTR_STS	

Table 31-79. FSI Registers to Driverlib Functions (continued)

File	Driverlib Function
fsi.h	FSI_getRxBufferPtr
fsi.h	FSI_getRxWordCount
RX_FRAME_WD_CTRL	
fsi.c	FSI_resetRxModule
fsi.c	FSI_clearRxModuleReset
fsi.h	FSI_enableRxFrameWatchdog
fsi.h	FSI_disableRxFrameWatchdog
RX_FRAME_WD_REF	
fsi.h	FSI_enableRxFrameWatchdog
RX_FRAME_WD_CNT	
fsi.h	FSI_getRxFrameWatchdogCounter
RX_PING_WD_CTRL	
fsi.c	FSI_resetRxModule
fsi.c	FSI_clearRxModuleReset
fsi.h	FSI_enableRxPingWatchdog
fsi.h	FSI_disableRxPingWatchdog
RX_PING_TAG	
fsi.h	FSI_getRxPingTag
fsi.h	FSI_setRxPingTagRef
fsi.h	FSI_getRxPingTagRef
fsi.h	FSI_setRxPingTagMask
fsi.h	FSI_getRxPingTagMask
fsi.h	FSI_enableRxPingTagCompare
fsi.h	FSI_disableRxPingTagCompare
fsi.h	FSI_enableRxPingBroadcast
fsi.h	FSI_disableRxPingBroadcast
RX_PING_WD_REF	
fsi.h	FSI_enableRxPingWatchdog
RX_PING_WD_CNT	
fsi.h	FSI_getRxPingWatchdogCounter
RX_INT1_CTRL	
fsi.h	FSI_enableRxInterrupt
fsi.h	FSI_disableRxInterrupt
RX_INT2_CTRL	
fsi.h	FSI_enableRxInterrupt
fsi.h	FSI_disableRxInterrupt
RX_LOCK_CTRL	
fsi.h	FSI_lockRxCtrl
RX_ECC_DATA	
fsi.h	FSI_setRxECCData
RX_ECC_VAL	
fsi.h	FSI_setRxReceivedECCValue
RX_ECC_SEC_DATA	
fsi.h	FSI_getRxECCCorrectedData
RX_ECC_LOG	

Table 31-79. FSI Registers to Driverlib Functions (continued)

File	Driverlib Function
fsi.h	FSI_getRxECCLog
RX_FRAME_TAG_CMP	
fsi.h	FSI_setRxFrameTagRef
fsi.h	FSI_getRxFrameTagRef
fsi.h	FSI_setRxFrameTagMask
fsi.h	FSI_getRxFrameTagMask
fsi.h	FSI_enableRxFrameTagCompare
fsi.h	FSI_disableRxFrameTagCompare
fsi.h	FSI_enableRxFrameBroadcast
fsi.h	FSI_disableRxFrameBroadcast
RX_PING_TAG_CMP	
fsi.h	FSI_setRxPingTagRef
fsi.h	FSI_getRxPingTagRef
fsi.h	FSI_setRxPingTagMask
fsi.h	FSI_getRxPingTagMask
fsi.h	FSI_enableRxPingTagCompare
fsi.h	FSI_disableRxPingTagCompare
fsi.h	FSI_enableRxPingBroadcast
fsi.h	FSI_disableRxPingBroadcast
RX_TRIG_CTRL_0	
-	
RX_TRIG_WIDTH_0	
-	
RX_DLYLINE_CTRL	
fsi.c	FSI_configRxDelayLine
RX_TRIG_CTRL_1	
-	
RX_TRIG_CTRL_2	
-	
RX_TRIG_CTRL_3	
-	
RX_VIS_1	
-	
RX_UDATA_FILTER	
-	
RX_BUF_BASE	
fsi.c	FSI_readRxBuffer
fsi.h	FSI_getRxBufferAddress

This page intentionally left blank.

Chapter 32

Configurable Logic Block (CLB)

This chapter describes the features and operation of the configurable logic block (CLB) that is a collection of configurable blocks that may be inter-connected using software to implement custom digital logic functions.

32.1 Introduction

The configurable logic block (CLB) is a collection of configurable blocks that can be inter-connected using software to implement custom digital logic functions. The CLB is able to enhance existing peripherals through a set of crossbar interconnections, which provide a high level of connectivity to existing control peripherals such as enhanced pulse width modulators (ePWM), enhanced capture modules (eCAP), and enhanced quadrature encoder pulse modules (eQEP). The crossbars also allow the CLB to be connected to external GPIO pins. In this way, the CLB can be configured to interact with device peripherals to perform small logical functions such as simple PWM generators, or to implement custom serial data exchange protocols.

The CLB peripheral is configured through the CLB tool. For more information on the CLB tool, available examples, application reports, and user's guide, refer to the following location in your C2000WARE package (C2000Ware_2_00_00_03 and higher):

C2000WARE_INSTALL_LOCATION\utilities\clb_tool\clb_syscfg\doc

32.1.1 CLB Related Collateral

Foundational Materials

- C2000 Academy - CLB
- C2000 Configurable Logic Block (CLB) Series (Video)
- Customizing on-chip peripherals defies conventional logic
- *Enable Differentiation and win with CLB in various applications Application Report*
- Enable Differentiation with Configurable Logic in Various Automotive Applications (Video)

Getting Started Materials

- *C2000 Position Manager PTO API Reference Guide Application Report*
- *CLB Tool User Guide*
  - Basic examples are 7 - 15 (start with these). More involved examples are 1-6.
- *Designing With The C2000 Configurable Logic Block Application Report*
- How do I add SYSCONFIG support (Pinmux and Peripheral Initialization) to an existing driverlib project?
- *How to Migrate Custom Logic From an FPGA/CPLD to C2000 Microcontrollers Application Report*
  - Chpaters 1-3 are very useful for getting started and learning the CLB. Later chapters are very useful Expert materials for migrating from FPGA/CPLD to C2000's CLB.

Expert Materials

- *Achieve Delayed Protection for Three-Level Inverter With CLB Application Report*
- *Tamagawa T-Format Absolute-Encoder Master Interface Reference Design for C2000 MCUs*

32.2 Description

The CLB subsystem contains a number of identical tiles. There are four such tiles in the CLB subsystem; other devices may contain more or fewer tiles. Tiles are numbered 1 to ‘N’, where ‘N’ is the total tile count on the device. Each tile contains combinational and sequential logic blocks, as well as other dedicated hardware to be described later in this document. Figure 32-1 shows the structure of the CLB subsystem in the device.

The tile contains the core logic, providing the logic reconfiguration capability. Each CLB tile is associated with a separate CPU interface, which contains the registers needed to control and configure the logic in the tile. The CPU interface also contains data transfer buffers that can be used as part of the configurable logic to exchange data with the rest of the device. Figure 32-2 shows the connections between the tile, the CPU interface, and the device.

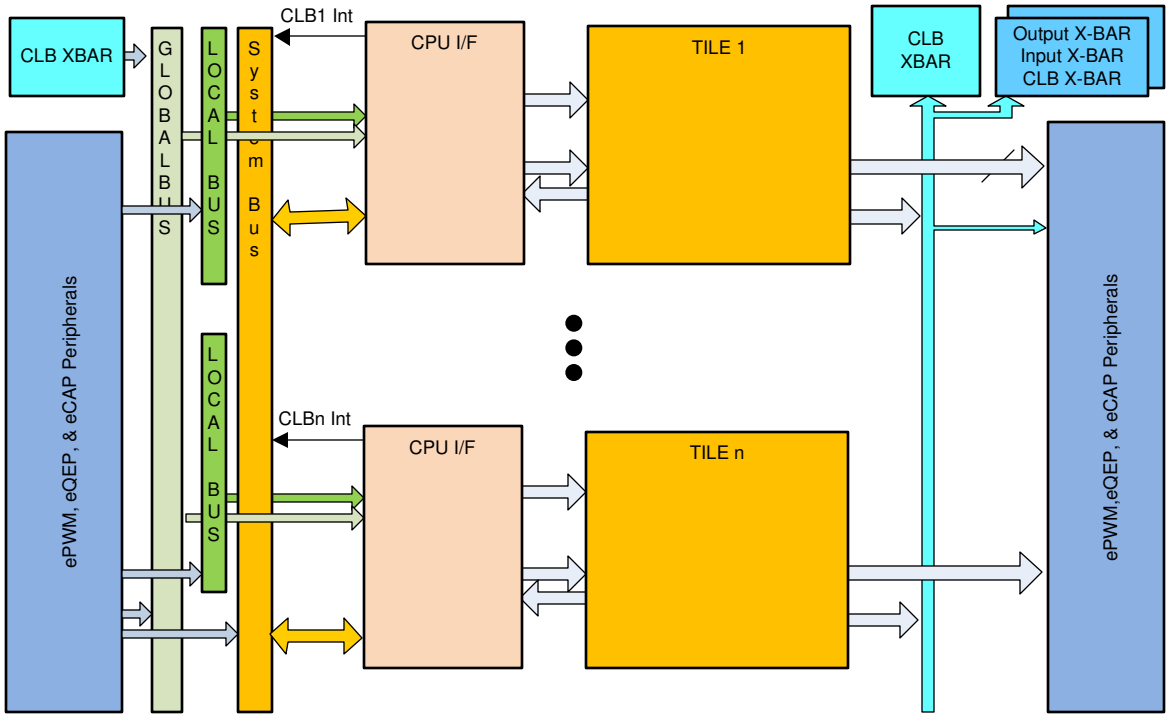


Figure 32-1. Block Diagram of the CLB Subsystem in the Device

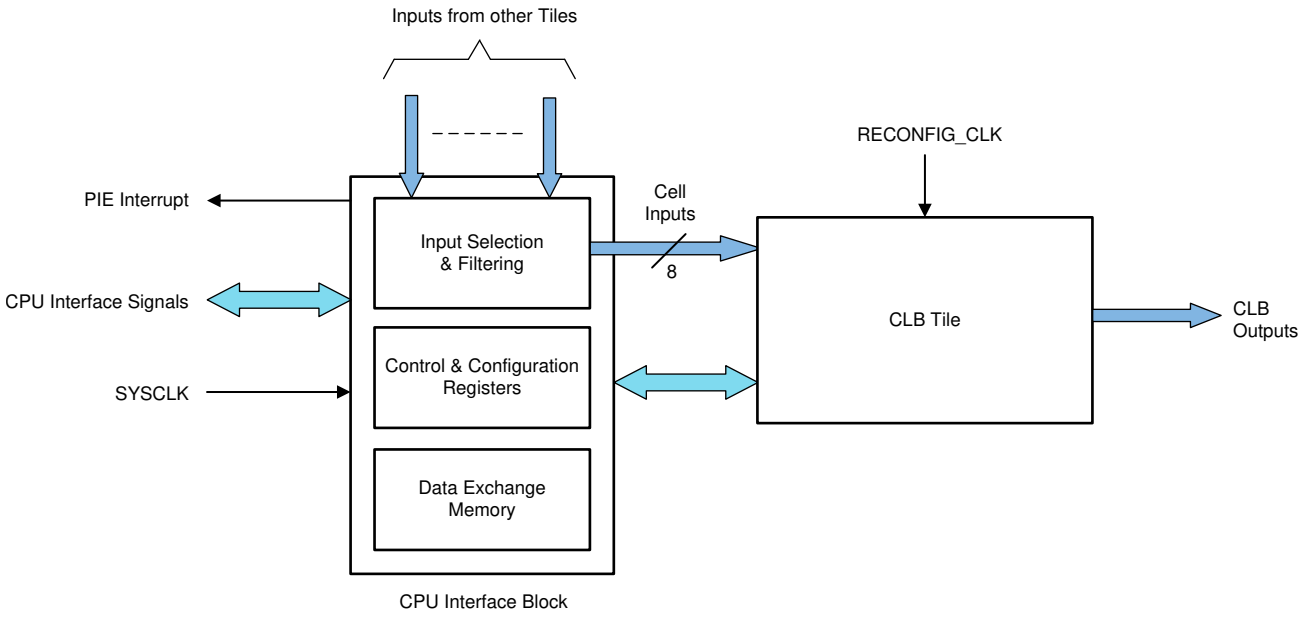


Figure 32-2. Block Diagram of a CLB Tile and CPU Interface

32.3 CLB Input/Output Connection

32.3.1 Overview

There are four instances of the CLB module in the device. Each CLB instance has a common set of input signals referred to as global input signals. Additionally, each CLB instance has a specific set of input signals that are unique to each instance, and are referred to as local input signals. Each of the eight inputs of a CLB can be chosen from any of the global input signals or the local input signals.

32.3.2 CLB Input Selection

Each CLB module has eight inputs that feed into the reconfigurable logic cell. Each of these inputs can be selectively driven by a predefined set of signals. A two-level mux structure allows each input of each CLB instance to select a signal.

A set of signals is common to all the CLB instances. These are referred to as global inputs in Figure 32-3. A separate set of signals is unique to each instance of the CLB. These are referred to as local inputs in Figure 32-3.

Registers CLB_LCL_MUX_SEL_1 and CLB_LCL_MUX_SEL_2 control the local mux selection for each of the eight inputs. The mux control registers CLB_GLBL_MUX_SEL_1 and CLB_GLBL_MUX_SEL_2 control the global mux selection for each of the eight inputs.

The local mux select value of 0 causes the selected global mux input signal to be connected to the corresponding CLB Input. For example, setting CLB_LCL_MUX_SEL_IN_0 = 0 and CLB_GLBL_MUX_SEL_IN_0 = 8 causes the global mux input number 8 to be connected to CLB Input 0. The input filter feature can be used to enable edge detection on the CLB inputs. The input filter feature can also synchronize the input with the CLB clock.

The global and local input mux settings are shown in Table 32-1 and Table 32-2.

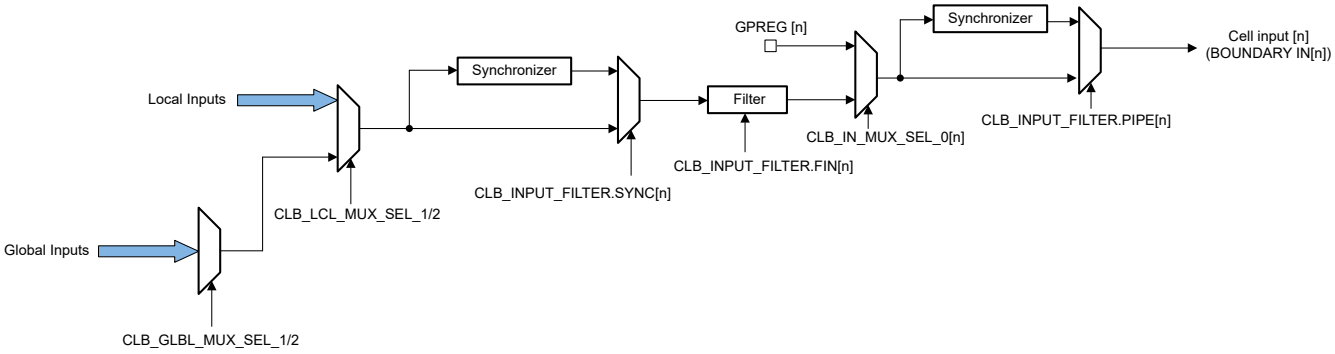


Figure 32-3. CLB Input Mux and Filter

Table 32-1. Global Signals and Mux Selection

Global Input Mux Bit Position	SYNC/ASYNC	Signal Name	Instance Name
CLB1 to CLB4			
0	ASYNC	ePWMxA	EPWM1
1	ASYNC	PWMA[OE]	EPWM1
2	ASYNC	ePWMxB	EPWM1
3	ASYNC	PWMB[OE]	EPWM1
4	SYNC	CTR=ZERO	EPWM1
5	SYNC	CTR=PRD	EPWM1
6	SYNC	CTR_Dir	EPWM1
7	SYNC	TBCLK	EPWM1
8	SYNC	CTR=CMPA	EPWM1
9	SYNC	CTR=CMPB	EPWM1
10	SYNC	CTR=CMPC	EPWM1
11	SYNC	CTR=CMPD	EPWM1
12	SYNC	PWMA[AQ]	EPWM1
13	SYNC	PWMB[AQ]	EPWM1
14	SYNC	PWMA[DB]	EPWM1
15	SYNC	PWMB[DB]	EPWM1
16	ASYNC	ePWMxA	EPWM2
17	ASYNC	PWMA[OE]	EPWM2
18	ASYNC	ePWMxB	EPWM2
19	ASYNC	PWMB[OE]	EPWM2
20	SYNC	CTR=ZERO	EPWM2
21	SYNC	CTR=PRD	EPWM2
22	SYNC	CTR_Dir	EPWM2
23	SYNC	TBCLK	EPWM2
24	SYNC	CTR=CMPA	EPWM2
25	SYNC	CTR=CMPB	EPWM2
26	SYNC	CTR=CMPC	EPWM2
27	SYNC	CTR=CMPD	EPWM2
28	SYNC	PWMA[AQ]	EPWM2
29	SYNC	PWMB[AQ]	EPWM2
30	SYNC	PWMA[DB]	EPWM2
31	SYNC	PWMB[DB]	EPWM2
32	ASYNC	ePWMxA	EPWM3
33	ASYNC	PWMA[OE]	EPWM3
34	ASYNC	ePWMxB	EPWM3
35	ASYNC	PWMB[OE]	EPWM3
36	SYNC	CTR=ZERO	EPWM3
37	SYNC	CTR=PRD	EPWM3
38	SYNC	CTR_Dir	EPWM3
39	SYNC	TBCLK	EPWM3
40	SYNC	CTR=CMPA	EPWM3
41	SYNC	CTR=CMPB	EPWM3
42	SYNC	CTR=CMPC	EPWM3
43	SYNC	CTR=CMPD	EPWM3

Table 32-1. Global Signals and Mux Selection (continued)

Global Input Mux Bit Position	SYNC/ASYNC	Signal Name	Instance Name
CLB1 to CLB4			
44	SYNC	PWMA[AQ]	EPWM3
45	SYNC	PWMB[AQ]	EPWM3
46	SYNC	PWMA[DB]	EPWM3
47	SYNC	PWMB[DB]	EPWM3
48	ASYNC	ePWMxA	EPWM4
49	ASYNC	PWMA[OE]	EPWM4
50	ASYNC	ePWMxB	EPWM4
51	ASYNC	PWMB[OE]	EPWM4
52	SYNC	CTR=ZERO	EPWM4
53	SYNC	CTR=PRD	EPWM4
54	SYNC	CTR_Dir	EPWM4
55	SYNC	TBCLK	EPWM4
56	SYNC	CTR=CMPA	EPWM4
57	SYNC	CTR=CMPB	EPWM4
58	SYNC	CTR=CMPC	EPWM4
59	SYNC	CTR=CMPD	EPWM4
60	SYNC	PWMA[AQ]	EPWM4
61	SYNC	PWMB[AQ]	EPWM4
62	SYNC	PWMA[DB]	EPWM4
63	SYNC	PWMB[DB]	EPWM4
64	ASYNC	AUXSIG0	CLB X BAR
65	ASYNC	AUXSIG1	CLB X BAR
66	ASYNC	AUXSIG2	CLB X BAR
67	ASYNC	AUXSIG3	CLB X BAR
68	ASYNC	AUXSIG4	CLB X BAR
69	ASYNC	AUXSIG5	CLB X BAR
70	ASYNC	AUXSIG6	CLB X BAR
71	ASYNC	AUXSIG7	CLB X BAR
72	SYNC	CLB1_OUT16	CLB1
73	SYNC	CLB1_OUT17	CLB1
74	SYNC	CLB1_OUT18	CLB1
75	SYNC	CLB1_OUT19	CLB1
76	SYNC	CLB1_OUT20	CLB1
77	SYNC	CLB1_OUT21	CLB1
78	SYNC	CLB1_OUT22	CLB1
79	SYNC	CLB1_OUT23	CLB1
80	SYNC	CLB2_OUT16	CLB2
81	SYNC	CLB2_OUT17	CLB2
82	SYNC	CLB2_OUT18	CLB2
83	SYNC	CLB2_OUT19	CLB2
84	SYNC	CLB2_OUT20	CLB2
85	SYNC	CLB2_OUT21	CLB2
86	SYNC	CLB2_OUT22	CLB2
87	SYNC	CLB2_OUT23	CLB2

Table 32-1. Global Signals and Mux Selection (continued)

Global Input Mux Bit Position	SYNC/ASYNC	Signal Name	Instance Name
CLB1 to CLB4			
88	SYNC	CLB3_OUT16	CLB3
89	SYNC	CLB3_OUT17	CLB3
90	SYNC	CLB3_OUT18	CLB3
91	SYNC	CLB3_OUT19	CLB3
92	SYNC	CLB3_OUT20	CLB3
93	SYNC	CLB3_OUT21	CLB3
94	SYNC	CLB3_OUT22	CLB3
95	SYNC	CLB3_OUT23	CLB3
96	SYNC	CLB4_OUT16	CLB4
97	SYNC	CLB4_OUT17	CLB4
98	SYNC	CLB4_OUT18	CLB4
99	SYNC	CLB4_OUT19	CLB4
100	SYNC	CLB4_OUT20	CLB4
101	SYNC	CLB4_OUT21	CLB4
102	SYNC	CLB4_OUT22	CLB4
103	SYNC	CLB4_OUT23	CLB4
104	SYNC	ERAD event 0	CPU1.ERAD
105	SYNC	ERAD event 1	CPU1.ERAD
106	SYNC	ERAD event 2	CPU1.ERAD
107	SYNC	ERAD event 3	CPU1.ERAD
108	SYNC	ERAD event 4	CPU1.ERAD
109	SYNC	ERAD event 5	CPU1.ERAD
110	SYNC	ERAD event 6	CPU1.ERAD
111	SYNC	ERAD event 7	CPU1.ERAD
112	SYNC	DATA_PKT_RCVD	FSIRXA
113	SYNC	ERROR_PKT_RCVD	FSIRXA
114	SYNC	PING_PKT_RCVD	FSIRXA
115	SYNC	frame_done	FSIRXA
116	SYNC	PING_TAG_MATCH	FSIRXA
117	SYNC	DATA_TAG_MATCH	FSIRXA
118	SYNC	ERR_TAG_MATCH	FSIRXA
119	SYNC	RX_TRIG2	FSIRXA
120	ASYNC	SPICLK(OUT)	SPIA
121	ASYNC	SPISOMI(IN)	SPIA
122	ASYNC	SPISTE(OUT)	SPIA
123	ASYNC	SPICLK(OUT)	SPIB
124	ASYNC	SPISOMI(IN)	SPIB
125	ASYNC	SPISTE(OUT)	SPIB
126	SYNC		
127	SYNC	RX_TRIG3	FSIRXA

Table 32-2. Local Signals and Mux Selection

Bit Position	SYNC/ ASYNC	Signal Name	Instance Name (for CLB1)	Instance Name (for CLB2)	Instance Name (for CLB3)	Instance Name (for CLB4)
0		Global Mux input	Global Mux input	Global Mux input	Global Mux input	Global Mux input
1	ASYNC	DCAEVT1	EPWM1	EPWM2	EPWM3	EPWM4
2	ASYNC	DCAEVT2	EPWM1	EPWM2	EPWM3	EPWM4
3	ASYNC	DCBEVT1	EPWM1	EPWM2	EPWM3	EPWM4
4	ASYNC	DCBEVT2	EPWM1	EPWM2	EPWM3	EPWM4
5	ASYNC	DCAH	EPWM1	EPWM2	EPWM3	EPWM4
6	ASYNC	DCAL	EPWM1	EPWM2	EPWM3	EPWM4
7	ASYNC	DCBH	EPWM1	EPWM2	EPWM3	EPWM4
8	ASYNC	DCBL	EPWM1	EPWM2	EPWM3	EPWM4
9	ASYNC	OST	EPWM1	EPWM2	EPWM3	EPWM4
10	ASYNC	CBC	EPWM1	EPWM2	EPWM3	EPWM4
11	ASYNC	ECAPIN	ECAP1	ECAP2	ECAP3	
12	SYNC	ECAP_OUT	ECAP1	ECAP2	ECAP3	
13	SYNC	ECAP_OUT_EN	ECAP1	ECAP2	ECAP3	
14	SYNC	CEVT1	ECAP1	ECAP2	ECAP3	
15	SYNC	CEVT2	ECAP1	ECAP2	ECAP3	
16	SYNC	CEVT3	ECAP1	ECAP2	ECAP3	
17	SYNC	CEVT4	ECAP1	ECAP2	ECAP3	
18	ASYNC	EQEPA	EQEP1	EQEP2		
19	ASYNC	EQEPB	EQEP1	EQEP2		
20	ASYNC	EQEPI	EQEP1	EQEP2		
21	ASYNC	EQEPS	EQEP1	EQEP2		
22	SYNC	CPU1.TBCLKSYNC	CPU1.TBCLKSYNC	CPU1.TBCLKSYNC	CPU1.TBCLKSYNC	CPU1.TBCLKSYNC
23	SYNC					
24	SYNC	CPU1.HALT	CPU1.HALT	CPU1.HALT	CPU1.HALT	CPU1.HALT
25	ASYNC	SPISIMO(OUT)	SPIA	SPIB		
26	ASYNC	SPI Clock (IN)	SPIA	SPIB		
27	ASYNC	SPISIMO(IN)	SPIA	SPIB		
28	ASYNC	SPI STE(IN)	SPIA	SPIB		
29	ASYNC	SCI TX	SCIA	SCIB	SCIA	SCIB
30	ASYNC	SPISOMI(OUT)	SPIA	SPIB		
31	SYNC	Local prescaled clock	CLB1	CLB2	CLB3	CLB4
32	ASYNC	PWMA	EPWM5	EPWM5	EPWM5	EPWM5
33	ASYNC	PWMA_OE	EPWM5	EPWM5	EPWM5	EPWM5
34	ASYNC	PWMB	EPWM5	EPWM5	EPWM5	EPWM5
35	ASYNC	PWMB_OE	EPWM5	EPWM5	EPWM5	EPWM5
36	ASYNC	PWMA	EPWM6	EPWM6	EPWM6	EPWM6
37	ASYNC	PWMA_OE	EPWM6	EPWM6	EPWM6	EPWM6
38	ASYNC	PWMB	EPWM6	EPWM6	EPWM6	EPWM6
39	ASYNC	PWMB_OE	EPWM6	EPWM6	EPWM6	EPWM6
40	ASYNC	PWMA	EPWM7	EPWM7	EPWM7	EPWM7
41	ASYNC	PWMA_OE	EPWM7	EPWM7	EPWM7	EPWM7
42	ASYNC	PWMB	EPWM7	EPWM7	EPWM7	EPWM7
43	ASYNC	PWMB_OE	EPWM7	EPWM7	EPWM7	EPWM7

Table 32-2. Local Signals and Mux Selection (continued)

Bit Position	SYNC/ ASYNC	Signal Name	Instance Name (for CLB1)	Instance Name (for CLB2)	Instance Name (for CLB3)	Instance Name (for CLB4)
44	ASYNC	PWMA	EPWM8	EPWM8	EPWM8	EPWM8
45	ASYNC	PWMA_OE	EPWM8	EPWM8	EPWM8	EPWM8
46	ASYNC	PWMB	EPWM8	EPWM8	EPWM8	EPWM8
47	ASYNC	PWMB_OE	EPWM8	EPWM8	EPWM8	EPWM8
48	ASYNC	INPUT1	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR
49	ASYNC	INPUT2	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR
50	ASYNC	INPUT3	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR
51	ASYNC	INPUT4	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR
52	ASYNC	INPUT5	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR
53	ASYNC	INPUT6	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR
54	ASYNC	INPUT7	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR
55	ASYNC	INPUT8	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR
56	ASYNC	INPUT9	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR
57	ASYNC	INPUT10	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR
58	ASYNC	INPUT11	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR
59	ASYNC	INPUT12	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR
60	ASYNC	INPUT13	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR
61	ASYNC	INPUT14	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR
62	ASYNC	INPUT15	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR
63	ASYNC	INPUT16	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR	CLB INPUT XBAR

Note

For the signals marked as ASYNC, the input filter synchronizer must be enabled.

The GPREG is accessible by the CPU and the bits of this register can be used as BOUNDARY INPUTs for the CLB Tiles. For example, CLB1's GPREG[0] can be used as BOUNDARY IN0 (Cell Input 0) for the corresponding CLB Tile.

In order to connect multiple tiles to each other, you can either use the CLBx OUT4/5 and connect it to CLBy BOUNDARY INz through the CLB X-BAR and the Global Signals Mux.

Another option is to connect the CLBx OUT0-7 to a GPIO and then use the INPUT X-BAR to bring the signal back in to the device and connect it to the CLBy BOUNDARY INz through the CLB X-BAR and the Global Signals Mux.

In order to use GPIOs as inputs to the CLB, you must utilize the Input X-BAR and the CLB X-BAR. Figure 32-4 shows how GPIOs can be used as inputs to the CLB tiles.

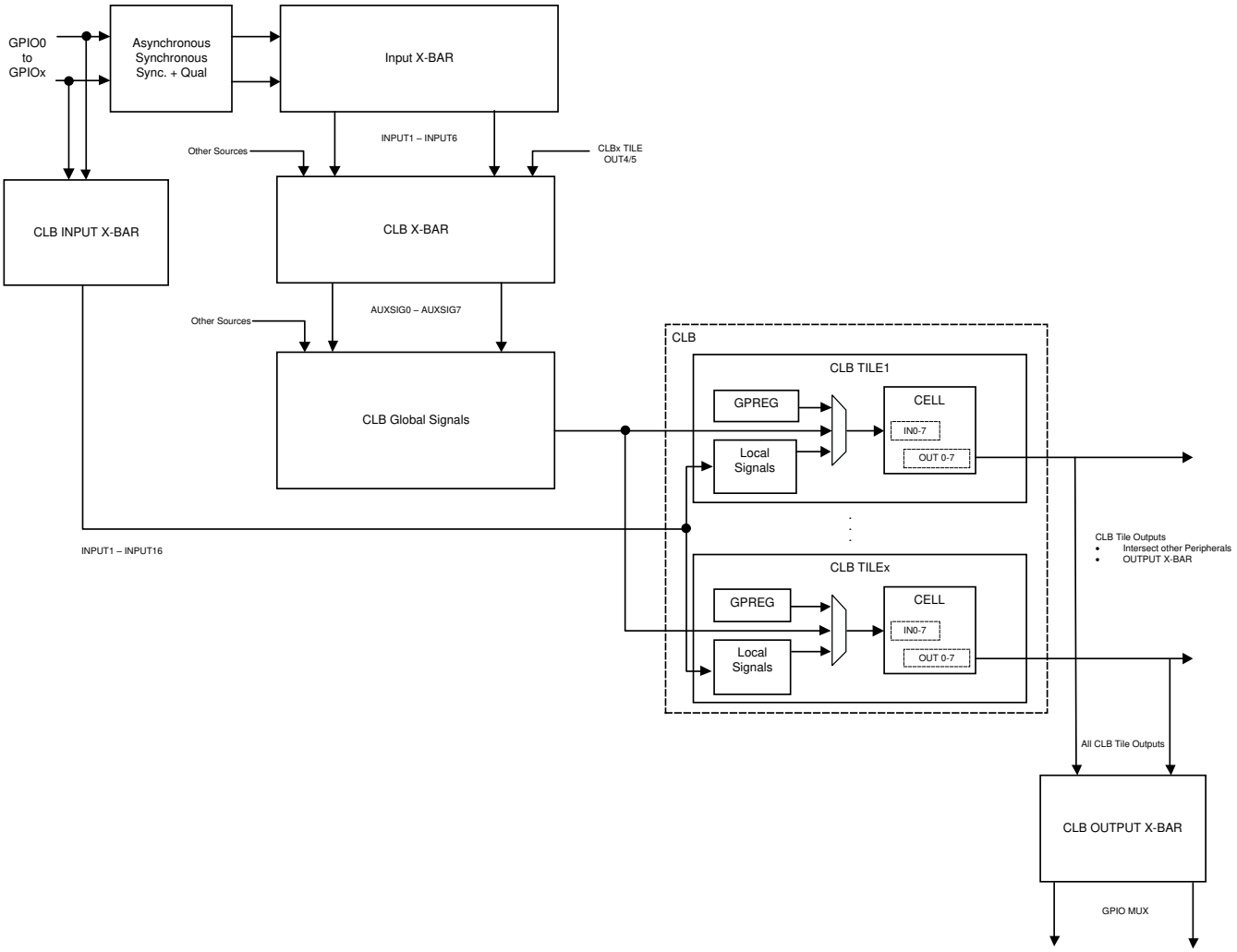


Figure 32-4. GPIO to CLB Tile Connections

32.3.3 CLB Output Selection

The eight outputs of the CLB are replicated to create 16 output signals. Each of these 16 outputs has a separate enable bit defined in the CLB output enable register, CLB_OUT_EN. The CLB outputs go to the ePWM, eCAP, eQEP and the crossbar module in the device. This allows the user to enhance the functionality of these modules with the CLB. Figure 32-5 shows the CLB outputs.

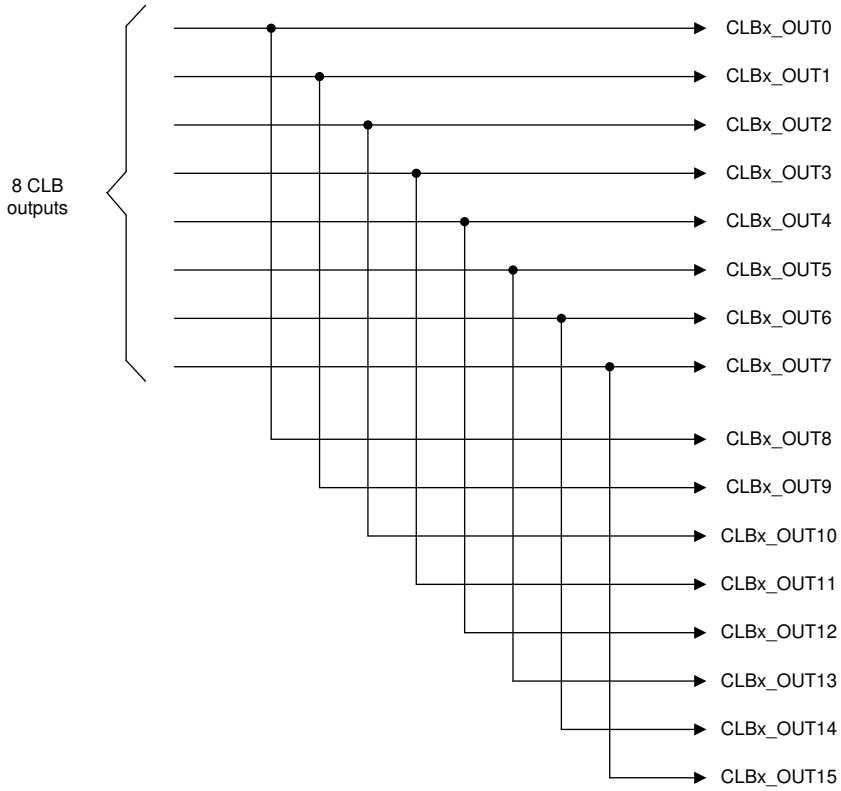


Figure 32-5. CLB Outputs

Note

CLBx_OUT0_1, CLBx_OUT1_1, CLBx_OUT2_1, … CLBx_OUT7_1 is the same as CLBx_OUT8, CLBx_OUT9, CLBx_OUT10, … CLBx_OUT15.

32.3.4 Peripheral Signal Multiplexer

Each CLB output signal passes through an external multiplexer that intersects a specific peripheral signal, see Figure 32-6. The output of the multiplexer is connected to the destination of the original peripheral signal and the default multiplexer setting is that the peripheral signal is passed through. The multiplexer is controlled by a bit in the CLB output enable register CLB_OUT_EN.

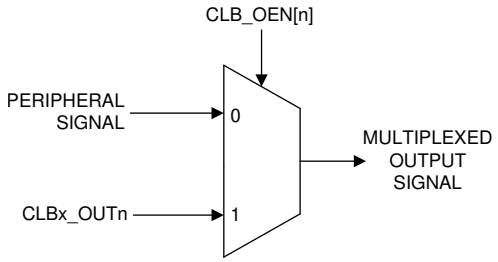


Figure 32-6. Peripheral Signal Multiplexer

For example, if the CLB1 OUT0 must override the EPWM1A signal, the OUPTUT ENABLE bit for OUT0 must be set to 1.

Table 32-3 shows the allocation of peripheral signals and the CLB outputs that they are multiplexed.

Table 32-3. Peripheral Signal Multiplexer Table

CLB Instance	CLB Output Signal	Peripheral Signal	Peripheral Name
CLB1			
CLB1	CLB1_OUT0_0	PWMA	EPWM1
CLB1	CLB1_OUT1_0	PWMA_OE	EPWM1
CLB1	CLB1_OUT2_0	PWMB	EPWM1
CLB1	CLB1_OUT3_0	PWMB_OE	EPWM1
CLB1	CLB1_OUT4_0	AQ_PWMA	EPWM1
CLB1	CLB1_OUT5_0	AQ_PWMB	EPWM1
CLB1	CLB1_OUT6_0	DB_PWMA	EPWM1
CLB1	CLB1_OUT7_0	DB_PWMB	EPWM1
CLB1	CLB1_OUT0_1	QCLK	EQEP1
CLB1	CLB1_OUT1_1	QDIR	EQEP1
CLB1	CLB1_OUT2_1	QB	EQEP1
CLB1	CLB1_OUT3_1	QA	EQEP1
CLB1	CLB1_OUT4_1	-	All XBARs (CLB, OUTPUT, EPWM)
CLB1	CLB1_OUT5_1	-	All XBARs (CLB, OUTPUT, EPWM)
CLB1	CLB1_OUT6_1	ECAP1.ECAPIN 16, ECAP2.ECAPIN 18, ECAP3.ECAPIN 16	ECAP1, ECAP2, ECAP3
CLB1	CLB1_OUT7_1	ECAP1.ECAPIN 17, ECAP2.ECAPIN 19, ECAP3.ECAPIN 17	ECAP1, ECAP2, ECAP3
CLB1	CLB1_OUT16	-	Global Mux
CLB1	CLB1_OUT17	-	Global Mux
CLB1	CLB1_OUT18	-	Global Mux
CLB1	CLB1_OUT19	-	Global Mux
CLB1	CLB1_OUT20	-	Global Mux

Table 32-3. Peripheral Signal Multiplexer Table (continued)

CLB Instance	CLB Output Signal	Peripheral Signal	Peripheral Name
CLB1	CLB1_OUT21	SPISTE(OUT)	SPIA,Global Mux
CLB1	CLB1_OUT22	SPISIMO(OUT)	SPIA,Global Mux
CLB1	CLB1_OUT23	SPISOMI(OUT)	SPIA,Global Mux
CLB1	CLB1_OUT24	SPICLK(IN)	SPIA
CLB1	CLB1_OUT25	SPISIMO(IN)	SPIA
CLB1	CLB1_OUT26	SPISTE(IN)	SPIA
CLB1	CLB1_OUT27	RX	SCIA
CLB1	CLB1_OUT28	ECAPOUT_EN	ECAP1
CLB1	CLB1_OUT29	ECAPOUT	ECAP1
CLB1	CLB1_OUT30	EXT_TRIG_40, PING_TRIG_40	FSITXA
CLB1	CLB1_OUT31	EXT_TRIG_41, PING_TRIG_41	FSITXA
CLB2			
CLB2	CLB2_OUT0_0	PWMA	EPWM2
CLB2	CLB2_OUT1_0	PWMA_OE	EPWM2
CLB2	CLB2_OUT2_0	PWMB	EPWM2
CLB2	CLB2_OUT3_0	PWMB_OE	EPWM2
CLB2	CLB2_OUT4_0	AQ_PWMA	EPWM2
CLB2	CLB2_OUT5_0	AQ_PWMB	EPWM2
CLB2	CLB2_OUT6_0	DB_PWMA	EPWM2
CLB2	CLB2_OUT7_0	DB_PWMB	EPWM2
CLB2	CLB2_OUT0_1	QCLK	EQEP2
CLB2	CLB2_OUT1_1	QDIR	EQEP2
CLB2	CLB2_OUT2_1	QB	EQEP2
CLB2	CLB2_OUT3_1	QA	EQEP2
CLB2	CLB2_OUT4_1	-	All XBARs (CLB, OUTPUT, EPWM)
CLB2	CLB2_OUT5_1	-	All XBARs (CLB, OUTPUT, EPWM)
CLB2	CLB2_OUT6_1	ECAP1.ECAPIN 18, ECAP2.ECAPIN 16 ECAP3.ECAPIN 18	ECAP1, ECAP2, ECAP3
CLB2	CLB2_OUT7_1	ECAP1.ECAPIN 19, ECAP2.ECAPIN 17 ECAP3.ECAPIN 19	ECAP1, ECAP2, ECAP3
CLB2	CLB2_OUT16	-	Global Mux
CLB2	CLB2_OUT17	-	Global Mux
CLB2	CLB2_OUT18	-	Global Mux
CLB2	CLB2_OUT19	-	Global Mux
CLB2	CLB2_OUT20	-	Global Mux
CLB2	CLB2_OUT21	SPISTE(OUT)	SPIB,Global Mux
CLB2	CLB2_OUT22	SPISIMO(OUT)	SPIB,Global Mux
CLB2	CLB2_OUT23	SPISOMI(OUT)	SPIB,Global Mux
CLB2	CLB2_OUT24	SPICLK(IN)	SPIB
CLB2	CLB2_OUT25	SPISIMO(IN)	SPIB
CLB2	CLB2_OUT26	SPISTE(IN)	SPIB
CLB2	CLB2_OUT27	RX	SCIB
CLB2	CLB2_OUT28	ECAPOUT_EN	ECAP2

Table 32-3. Peripheral Signal Multiplexer Table (continued)

CLB Instance	CLB Output Signal	Peripheral Signal	Peripheral Name
CLB2	CLB2_OUT29	ECAPOUT	ECAP2
CLB2	CLB2_OUT30	EXT_TRIG_42, PING_TRIG_42	FSITXA
CLB2	CLB2_OUT31	EXT_TRIG_43, PING_TRIG_43	FSITXA
CLB3			
CLB3	CLB3_OUT0_0	PWMA	EPWM3
CLB3	CLB3_OUT1_0	PWMA_OE	EPWM3
CLB3	CLB3_OUT2_0	PWMB	EPWM3
CLB3	CLB3_OUT3_0	PWMB_OE	EPWM3
CLB3	CLB3_OUT4_0	AQ_PWMA	EPWM3
CLB3	CLB3_OUT5_0	AQ_PWMB	EPWM3
CLB3	CLB3_OUT6_0	DB_PWMA	EPWM3
CLB3	CLB3_OUT7_0	DB_PWMB	EPWM3
CLB3	CLB3_OUT0_1	-	-
CLB3	CLB3_OUT1_1	-	-
CLB3	CLB3_OUT2_1	-	-
CLB3	CLB3_OUT3_1	-	-
CLB3	CLB3_OUT4_1	-	All XBARs (CLB, OUTPUT, EPWM)
CLB3	CLB3_OUT5_1	-	All XBARs (CLB, OUTPUT, EPWM)
CLB3	CLB3_OUT6_1	-	-
CLB3	CLB3_OUT7_1	-	-
CLB3	CLB3_OUT16	-	Global Mux
CLB3	CLB3_OUT17	-	Global Mux
CLB3	CLB3_OUT18	-	Global Mux
CLB3	CLB3_OUT19	-	Global Mux
CLB3	CLB3_OUT20	-	Global Mux
CLB3	CLB3_OUT21	-	Global Mux
CLB3	CLB3_OUT22	-	Global Mux
CLB3	CLB3_OUT23	-	Global Mux
CLB3	CLB3_OUT24	-	-
CLB3	CLB3_OUT25	-	-
CLB3	CLB3_OUT26	-	-
CLB3	CLB3_OUT27	RX	LINA
CLB3	CLB3_OUT28	ECAPOUT_EN	ECAP3
CLB3	CLB3_OUT29	ECAPOUT	ECAP3
CLB3	CLB3_OUT30	EXT_TRIG_44, PING_TRIG_44	FSITXA
CLB3	CLB3_OUT31	EXT_TRIG_45, PING_TRIG_45	FSITXA
CLB4			
CLB4	CLB4_OUT0_0	PWMA	EPWM4
CLB4	CLB4_OUT1_0	PWMA_OE	EPWM4
CLB4	CLB4_OUT2_0	PWMB	EPWM4
CLB4	CLB4_OUT3_0	PWMB_OE	EPWM4
CLB4	CLB4_OUT4_0	AQ_PWMA	EPWM4
CLB4	CLB4_OUT5_0	AQ_PWMB	EPWM4
CLB4	CLB4_OUT6_0	DB_PWMA	EPWM4

Table 32-3. Peripheral Signal Multiplexer Table (continued)

CLB Instance	CLB Output Signal	Peripheral Signal	Peripheral Name
CLB4	CLB4_OUT7_0	DB_PWMB	EPWM4
CLB4	CLB4_OUT0_1	-	-
CLB4	CLB4_OUT1_1	-	-
CLB4	CLB4_OUT2_1	-	-
CLB4	CLB4_OUT3_1	-	-
CLB4	CLB4_OUT4_1	-	All XBARs (CLB, OUTPUT, EPWM)
CLB4	CLB4_OUT5_1	-	All XBARs (CLB, OUTPUT, EPWM)
CLB4	CLB4_OUT6_1	-	-
CLB4	CLB4_OUT7_1	-	-
CLB4	CLB4_OUT16	-	Global Mux
CLB4	CLB4_OUT17	-	Global Mux
CLB4	CLB4_OUT18	-	Global Mux
CLB4	CLB4_OUT19	-	Global Mux
CLB4	CLB4_OUT20	-	Global Mux
CLB4	CLB4_OUT21	-	Global Mux
CLB4	CLB4_OUT22	-	Global Mux
CLB4	CLB4_OUT23	-	Global Mux
CLB4	CLB4_OUT24	-	Global Mux
CLB4	CLB4_OUT25	-	-
CLB4	CLB4_OUT26	-	-
CLB4	CLB4_OUT27	RX	LINB
CLB4	CLB4_OUT28	-	-
CLB4	CLB4_OUT29	-	-
CLB4	CLB4_OUT30	EXT_TRIG_46, PING_TRIG_46	FSITXA
CLB4	CLB4_OUT31	EXT_TRIG_47, PING_TRIG_47	FSITXA

32.4 CLB Tile

The purpose of the CLB tile is to provide the logic reconfiguration capability of the CLB. The CLB tile contains the following submodules:

- **Counter:** The counter submodule can be configured either as an adder, a counter, or a shifter. When functioning as an adder, it can either add or subtract. When functioning as a counter, it can count up or count down. When functioning as a shifter, it can shift left or shift right. The counter event inputs, as well as the reset input, may be freely connected to any of the other submodules in the same tile. There are three counters in each tile.
- **LUT4:** The LUT4 submodule has a 4-input look-up table functionality and is capable of realizing any combinatorial Boolean equation of up to four inputs. There are three LUT4 submodules in each CLB tile.
- **FSM:** The Finite State Machine (FSM) submodule can be configured either as a single four-state finite state machine, or as two independent two-state finite state machines. The FSM accepts two external inputs, and generates two state outputs and one combination output. When not used as a state machine, the FSM submodule can accept two external inputs and function as a 4-input LUT. There are three FSM submodules in each CLB tile.
- **Output LUT:** The output LUT is a 3-input lookup table submodule capable of realizing any combinatorial Boolean equation of up to three inputs. There are eight such blocks in a CLB tile, each associated with one of the tile outputs.
- **High Level Controller:** The High Level Controller (HLC) submodule is an event-driven block that can handle up to four concurrent events. The event can be an activity on any of the other block outputs. A predefined set of operations is executed when each event occurs. The HLC also provides a data exchange and interrupt mechanism to the CPU subsystem. There are four working registers (R0, R1, R2, and R3) which can be used for basic operations, and to modify or set up values for the three counter blocks. Unlike the other submodules, there is only one HLC in each CLB tile.
- **Configurable Switch Block:** The configurable switch block provides dynamic connectivity between all of the blocks listed above. Submodules can be connected by the user, with the sole restriction that they must not form a combinational loop within the tile.

A CLB tile consists of three sets each of the counter block, FSM, and LUT4, one high-level controller, and eight output LUT blocks. The submodule numbering is shown in Figure 32-7.

The functionality of the LUT submodules is configured using a register field containing the binary pattern of the output of the desired look-up table. For example, a 4-input LUT has 16 possible input permutations, each of which corresponds to a desired binary 0 or 1 at its output. The register field would therefore be 16-bits in length, with each bit representing the desired result of a binary pattern. Input pattern sequences start at 0000 and continue sequentially to 1111. A similar method is used to encode the 16-bit state equations in the FSM submodule.

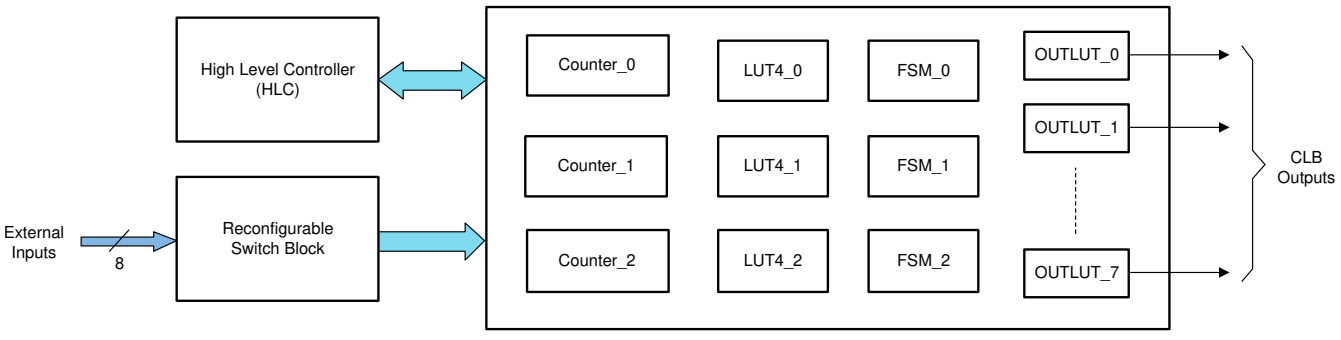


Figure 32-7. CLB Tile Submodules

32.4.1 Static Switch Block

The Static Switch Block provides the configurable connectivity between the submodules in the CLB tile. The outputs of all the submodules and the eight external inputs are connected to a common internal bus inside the tile. Every input port has a 32-to-1 multiplexer and an associated 5-bit selection value that allows the user to select one of the inputs on the bus. The only restrictions are certain signals (described below) that are tied off in the design to prevent creation of accidental combinatorial loops.

Table 32-4. Output Table

Bit Position	Signal Connection	Comment
0	Always '0'	This select value will be used to tie an input to '0'.
1	COUNTER_0 MATCH2	
2	COUNTER_0 ZERO	
3	COUNTER_0 MATCH1	
4	FSM_0 STATE_BIT_0	
5	FSM_0 STATE_BIT_1	
6	FSM_0 LUT output	
7	LUT4_0 output	
8	Always '1'	This select value will be used to tie an input to '1'.
9	COUNTER_1 MATCH2	
10	COUNTER_1 ZERO	
11	COUNTER_1 MATCH1	
12	FSM_1 STATE_BIT_0	
13	FSM_1 STATE_BIT_1	
14	FSM_1 LUT output	
15	LUT4_1 output	
16	Always '0'	
17	COUNTER_2 MATCH2	
18	COUNTER_2 ZERO	
19	COUNTER_2 MATCH1	
20	FSM_2 STATE_BIT_0	
21	FSM_2 STATE_BIT_1	
22	FSM_2 LUT output	
23	LUT4_2 output	
24	External Input 0	
25	External Input 1	
26	External Input 2	
27	External Input 3	
28	External Input 4	
29	External Input 5	
30	External Input 6	
31	External Input 7	

Table 32-5. Input Table

Module Name	Port Name	Description
Counter Block	RESET	Acts as an active high reset when used as a counter
	MODE_0	Acts as a enable when used as a counter. The counter will count only when this input is ‘1’.
	MODE_1	Acts as a direction control when used as a counter. If this input is ‘1’, then the counter counts up else, it counts down.
LUT	IN0	Input 0 of the 4-input LUT.
	IN1	Input 1 of the 4-input LUT.
	IN2	Input 2 of the 4-input LUT.
	IN3	Input 3 of the 4-input LUT.
FSM	EXT_IN0	Input 0 of the FSM block.
	EXT_IN1	Input 1 of the FSM block.
	EXTRA_EXT_IN0	Extra external Input 0 of the FSM block. This input matters only if it is configured in the LUT mode.
	EXTRA_EXT_IN1	Extra external Input 1 of the FSM block. This input matters only if it is configured in the LUT mode.

The static switch block allows the user to define the input connection of any submodule to come from any of the outputs in Table 32-4. It is therefore easy to create a combinatorial loop. In order to prevent this, certain paths are broken in the input path of each submodule. These port positions are tied to '0', as shown in Table 32-6.

Table 32-6. Ports Tied Off to Prevent Combinatorial Loops

Module Name	Ports of Input MUX Tied Off to ‘0’ to Prevent Combinational Loops
LUT_0	LUT_0 , LUT_1 and LUT_2 output, FSM_0, FSM_1 and FSM_2 output.
FSM_0	LUT_1 and LUT_2 output, FSM_0, FSM_1 and FSM_2 output.
LUT_1	LUT_1 and LUT_2 output, FSM_1 and FSM_2 output.
FSM_1	LUT_2 output, FSM_1 and FSM_2 output.
LUT_2	LUT_2 output, FSM_2 output.
FSM_2	FSM_2 output.

32.4.2 Counter Block

32.4.2.1 Counter Description

The counter block is a complex functional submodule that can be configured either as a counter, an adder, or a shifter. Apart from the normal operational control, this block has a dedicated EVENT input, which can trigger an addition, subtraction or shift operation, or load data into the counter register. The inputs to the counter submodule are shown in Figure 32-8.

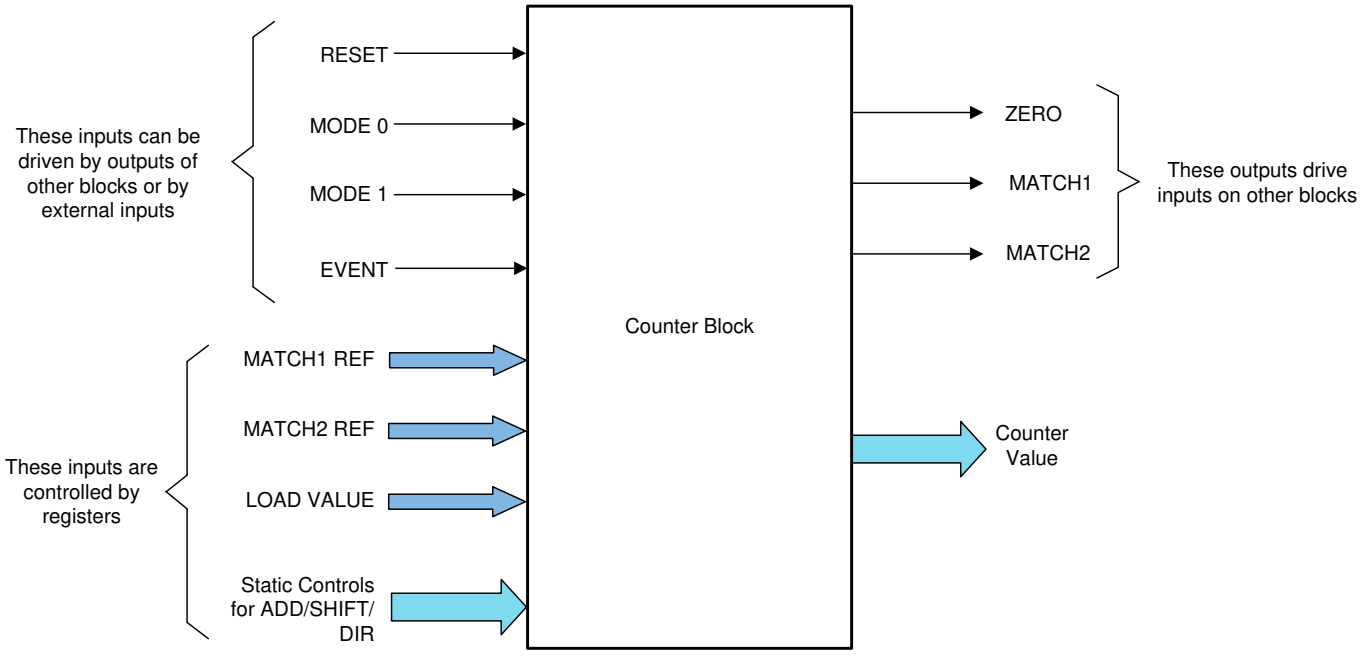


Figure 32-8. Counter Block

32.4.2.2 Counter Operation

At the heart of the counter block is a 32-bit count register. This register can either be loaded statically before counting commences, or dynamically at run time. The operation of the counter submodule is determined by the inputs described below. Note that each of the inputs can be connected to the outputs of any of the other blocks in the CLB tile. These connections are made by configuring the configurable switch block.

The counter inputs are:

- **RESET:** This is the highest priority input and takes precedence over all other inputs. The input is level sensitive and as long as it remains high, the counter will reset to 0 on the next clock cycle.
- **MODE_0:** This input is an enable for the counter. The counter begins counting (up or down depending on the MODE_1 setting) only when this input is high. If this input is low, then no counting takes place.
- **MODE_1:** This input is the direction control for the counter. If this input is high, then the counter will increment on every clock cycle where it sees MODE_0 to be high. If this input is low, then the counter decrements for every clock cycle it sees MODE_0 high. The counter wraps around to 0x00000000 after 0xFFFFFFFF when counting up. The counter wraps around to 0xFFFFFFFF after 0x00000000 when counting down. The only exception to this is when an EVENT occurs at exactly the same time, causing a different value to be loaded into the counter.

- **EVENT:** This input is defined for the purpose of triggering actions in the counter based on certain events. The event itself can be any of the outputs of the other blocks or an external input to the tile. The counter's static control inputs define the behavior of the counter on an active event. An active event is defined as a rising edge on the EVENT input. The counter can be configured to perform one of the following actions:
  - Load a predefined 32-bit value from the LOAD VALUE register into the count register
  - Shift the contents of the counter register left or right by a predefined amount between 0 and 31
  - Add or subtract a predefined 32-bit value. Addition and subtraction are treated as 32-bit unsigned operations and there is no saturation.

  Note that the effect of a rising edge on the EVENT input only lasts for one cycle. On the next cycle, the counter operation will continue based on the MODE_0, MODE_1, and RESET inputs.

MATCH1 REF and MATCH2 REF are 32-bit reference values that are used to generate the MATCH1 and MATCH2 outputs. The MATCH1 output becomes active high whenever the counter register value matches the 32-bit MATCH1 REF value. MATCH2 behaves in a similar manner in relation to the MATCH2 REF register. The reference values for MATCH1 and MATCH2 can either be setup once before the start of operation, or can be modified dynamically. The High Level Controller can load desired values into the MATCH1 REF and MATCH2 REF registers.

Note that the counter load and match registers are not memory-mapped. For more information, see Section 32.5.2.

The three logic outputs of the counter block are:

- **ZERO:** This output goes high whenever the counter register is 0.
- **MATCH1:** This output goes high whenever the counter register is equal to the MATCH1 REF input register.
- **MATCH2:** This output goes high whenever the counter register is equal to the MATCH2 REF input register.

The operation of the counter block is controlled by the CFG_MISC_CTRL register. The following three bits of this register are relevant for each counter. The "x" below refers to the counter instance; 0, 1, or 2. For more information, see the CLB_MISC_CONTROL register description located in Section 32.8.

- COUNT_EVENT_CTRL_x: This bit defines whether the counter performs an addition or a shift on an event. A value of 0 means that on an event, the counter will just load the static value; 1 means and add/shift operation will be performed.
- COUNT_ADD_SHIFT_x. 1 means add, 0 means shift.
- COUNT_DIR_x. 1 means left shift or add. 0 means right shift or subtract.

Table 32-7 shows the logical operation of the counter block in terms of the inputs and control register bits. Count up and down modes are the normal operation with EVENT = 0. The operations on the CNTVAL register are:

Load: CNTVAL = EVENT_LOAD_VAL

Shift right: CNTVAL = CNTVAL >> EVENT_LOAD_VAL

Shift left: CNTVAL = CNTVAL << EVENT_LOAD_VAL

Subtract: CNTVAL = CNTVAL - EVENT_LOAD_VAL

Add: CNTVAL = CNTVAL + EVENT_LOAD_VAL

Table 32-7. Counter Block Operating Modes

EVENT	MODE_0	MODE_1	COUNT_EVENT_CTRL_x	COUNT_ADD_SHIFT_x	COUNT_DIR_x	Action on CNTVAL
0	0	0	X	X	X	None
0	0	1	X	X	X	None
0	1	0	X	X	X	Count down
0	1	1	X	X	X	Count up
1	X	X	0	X	X	Load
1	X	X	1	0	0	Shift right
1	X	X	1	0	1	Shift left
1	X	X	1	1	0	Subtract
1	X	X	1	1	1	Add

32.4.2.3 Serializer Mode

Starting with CLB Type 2, the Counter module can operate as a serializer. In this mode of operation, this module acts as a shift register (also referred to as a serializer). In serializer mode of operation, the EVENT input is used to shift one bit of data into the serializer. Either of MATCH1 and MATCH2 can be configured to send out the shift register data.Using the MATCH1/2_TAP_SEL bit of CLB_COUNT_MATCH_TAP_SEL, any bit position of the counter can be brought out on the MATCH1/MATCH2 outputs. The shifting and direction of the counter in this mode is controlled by MODE_0(enable) and MODE_1(direction).

To enable the Serializer mode, CLB_MISC_CONTROL.COUNT_SERIALIZER_0 (for Counter 0) must be set.

32.4.2.4 Linear Feedback Shift Register (LFSR) Mode

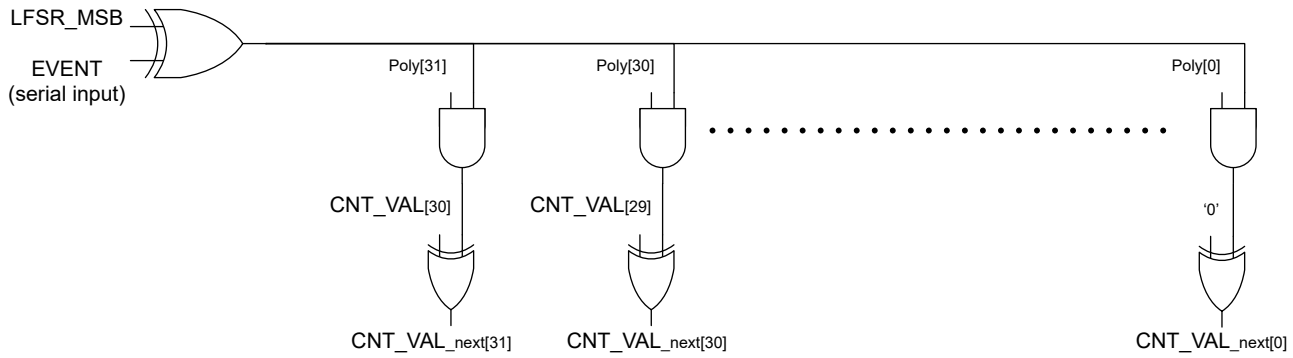
Starting with CLB Type 2, the Counter module operates as a linear feedback shift register. By configuring the characteristics of the LFSR, the counter module is used to compute the CRC on a serial bit stream. The polynomial for LFSR is in the MATCH2 reference register. The feedback bit position is in the MATCH1 reference register.

To enable the LFSR mode, CLB_MISC_CONTROL.COUNT_SERIALIZER_0 (for Counter 0) must be set along with COUNT0_LFSR_EN.

There are two types of LSFR that can be selected by changing the MODE1 value (0 or 1), as shown in Figure 32-9.

Structure for LFSR Type 1 (MODE_1 = 0)

CNT_VAL is the 32-bit counter's active register
CNT_VAL_next is the 32-bit value to be written into CNT_VAL on the next active cycle (clock edge when MODE_0 == 1)
LFSR_MSB = CNT_VAL[MATCH1_REF[4:0]]
Poly[31:0] is MATCH2_REF which acts as the CRC polynomial



Structure for LFSR Type 2 (MODE_1 = 1)

CNT_VAL is the 32-bit counter register
CNT_VAL_next is the 32-bit value to be written into CNT_VAL on the next active cycle (clock edge when MODE_0 == 1)
LFSR_MSB = CNT_VAL[MATCH1_REF[4:0]]
Poly[31:0] is MATCH2_REF which acts as the CRC polynomial

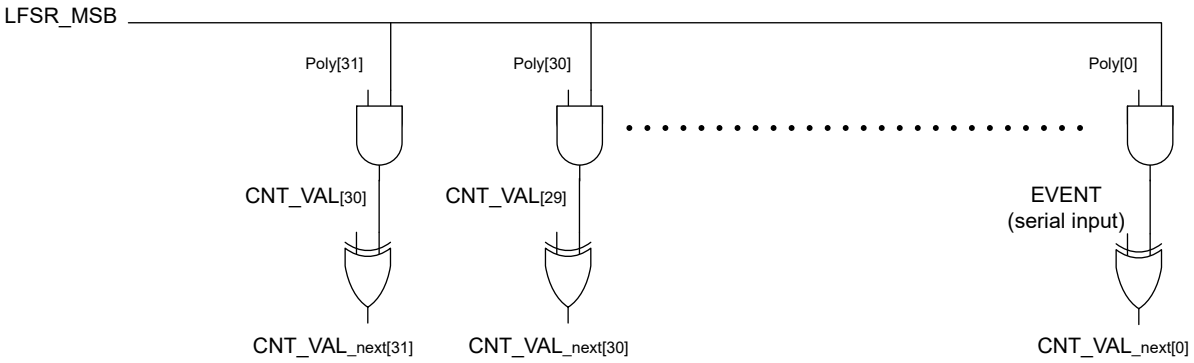


Figure 32-9. LFSR Modes

32.4.3 FSM Block

The Finite State Machine (FSM) block provides the ability to build programmable finite state machines with up to four states. It has two register bits and two external inputs, and can be programmed either as two 2-state machines or as a single 4-state machines. For additional flexibility, there are two auxiliary inputs (EXTRA_EXT_IN0 and EXTRA_EXT_IN1) that can be used to create larger combinational functions by giving up a state functionality. The structure of the FSM is shown in Figure 32-10.

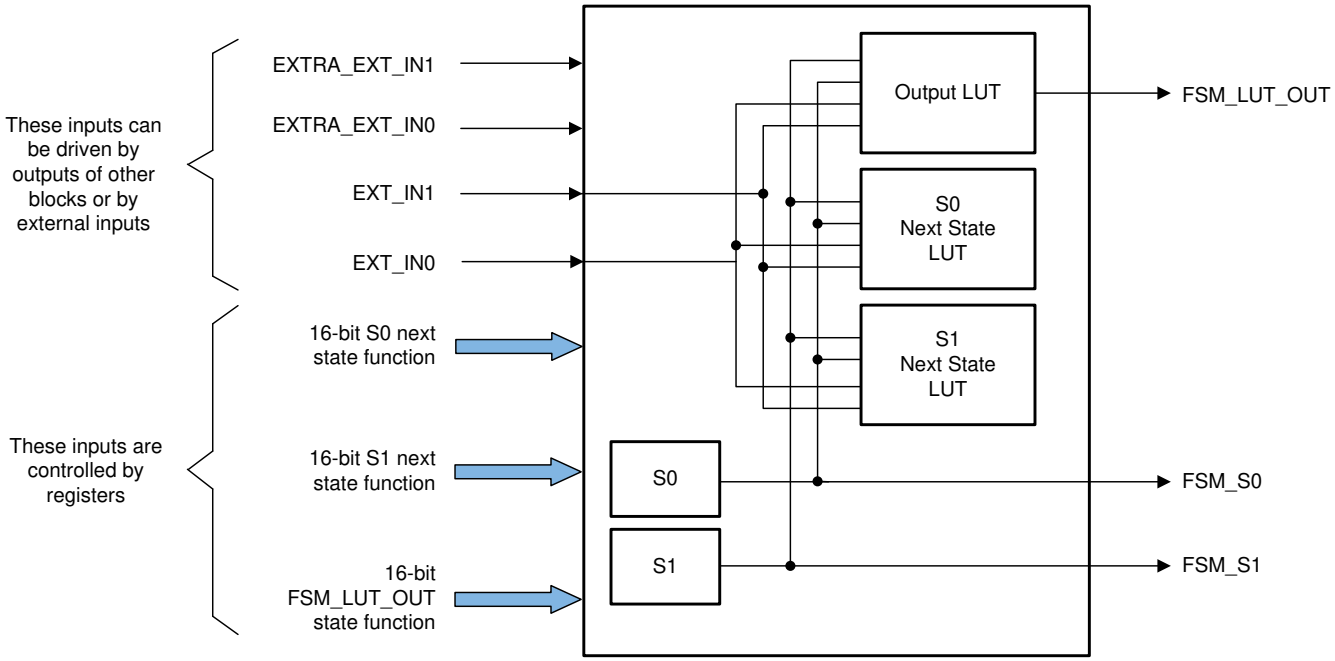


Figure 32-10. FSM Block

The signals and functionality of the FSM block are:

- **EXT_IN0 and EXT_IN1:** These are the two external inputs that can be used to control the output FSM_LUT_OUT or either of the states S0 and S1.
- **S0** and **S1** are two state bits that have independent state control equations.
- **16-bit S0 equation** defines a function (EXT_IN1, EXT_IN0, S1, S0). The four bits in the order defined are used as an index into the 16-bit register to decide the next state of S0.
- 1**6-bit S1 equation** defines a function (EXT_IN1, EXT_IN0, S1, S0). The four bits in the order defined are used as an index into the 16-bit register to decide the next state of S1.
- **16-bit output equation** defines a function (EXT_IN1, EXT_IN0, S1, S0). The four bits in the order defined are used as an index into the 16-bit register to decide the output value of FSM_LUT_OUT. An additional level of configurability is provided such that FSM_LUT_OUT can use extra inputs in case the states S0 and S1 are unused.

  One extra bit is used to select EXTRA_EXT_IN0 instead of S0. One extra bit is used to select EXTRA_EXT_IN1 instead of S1. Using these, one can effectively build 3-input or a 4-input LUT for the FSM_LUT_OUT by giving up one or two state bits, respectively.

The CFG_MISC_CTRL register controls the operation of the FSM block. Two bits in this register are used for each FSM Block to determine whether the FSM output LUT function uses the state variable S0/S1, or the corresponding extra external input signal FSM_EXTRA_EXT_INx. ‘0’ means use the state bit, and ‘1’ means use the FSM_EXTRA_EXT_IN0 / FSM_EXTRA_EXT_IN1 signal.

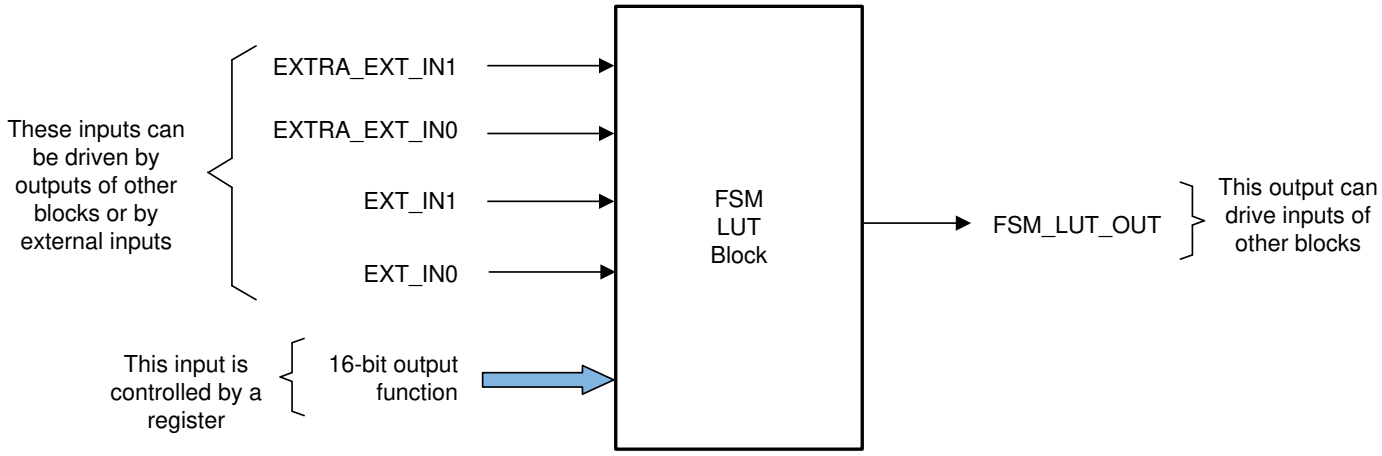


Figure 32-11. FSM LUT Block

32.4.4 LUT4 Block

This is a simple four input Look-Up table (LUT) block with inputs IN0, IN1, IN2, and IN3 (see Figure 32-12). Any combinatorial Boolean equation using the four inputs can be realized by programming the 16-bit control register associated with each LUT4 block. For more information, see the LUT4 register descriptions located in Section 32.8.

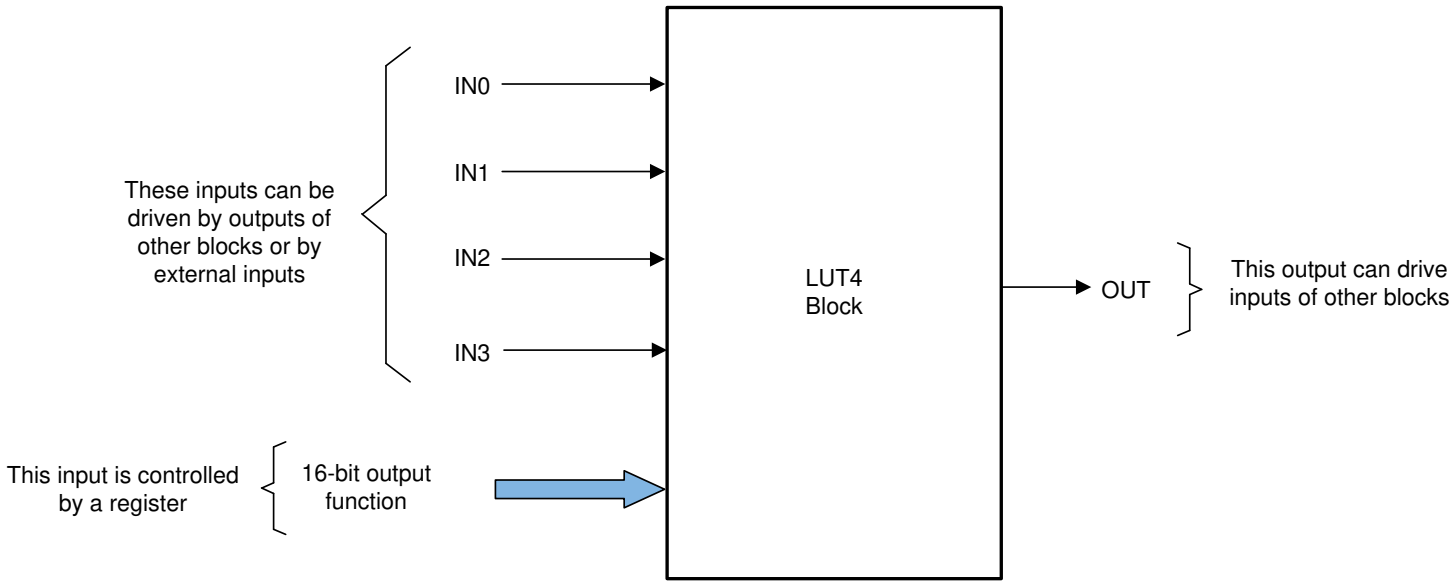


Figure 32-12. LUT4 Block

32.4.5 Output LUT Block

The output LUT block (Figure 32-13) is very similar in functionality to the LUT4 block, except that it has three inputs. Unlike the other sub blocks, the outputs of these blocks are meant to go out of the tile and hence they cannot be used by any other block within the tile. Any combinatorial function of the three inputs can be realized by the output LUT block. For more information, see the output LUT register descriptions located in Section 32.8.

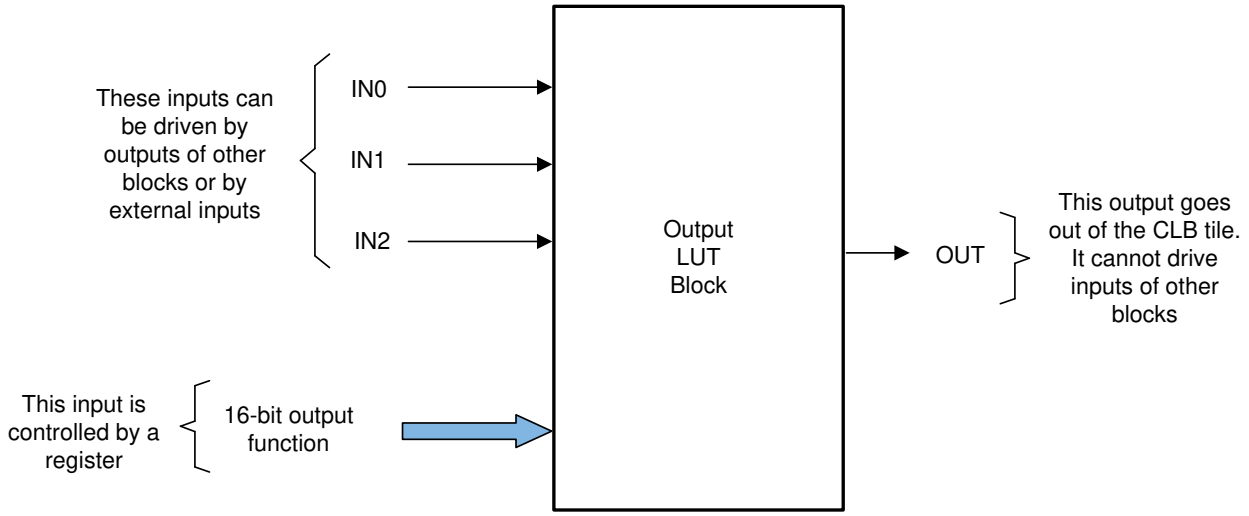


Figure 32-13. Output LUT Block

32.4.6 Asynchronous Output Conditioning (AOC) Block

The logic in the AOC block is organized into three stages with the inputs passing through different types of logic modification at each stage before proceeding to the next level. This is shown in Figure 32-14.

There are 8 inputs to this block. Each of these 8 inputs can pick the corresponding BOUNDARY input to the CLB or the CLB TILE output (for example the INPUT 0 of the AOC block can choose between CLB BOUNDARY INPUT0 and CLB TILE OUTPUT 0). If the CLB TILE OUTPUT 0 is selected, it is always registered before being sent to the subsequent asynchronous signal conditioning stages. In each of the three stages, there is always an option to do nothing and just send the signal as is to the next stage (bypass).

Stage 1 : The input signal can be inverted before sending it to the next stage.

Stage 2 : The signal coming from Stage 1 can be GATED with a gating control signal. The gating control signal can either be from a software register or can be any of the CLB TILE outputs. The GATING function can be a logical AND, OR or XOR.

Stage 3 : The input signal can be used to either set or clear the output on the rising edge of the signal. This is a purely asynchronous set/clear that occurs without needing any clocks. The release control signal, when high, will restore the output to it's default state (HIGH if asynchronous clear is selected and LOW if asynchronous set is selected). The release control signal can be either from a software register or can be any of the CLB TILE outputs. Optionally, instead of any of the asynchronous set/clear operation, the input signal can just be delayed by a clock cycle.

The interaction of the CLB TILE and the AOC block is shown in Figure 32-15.

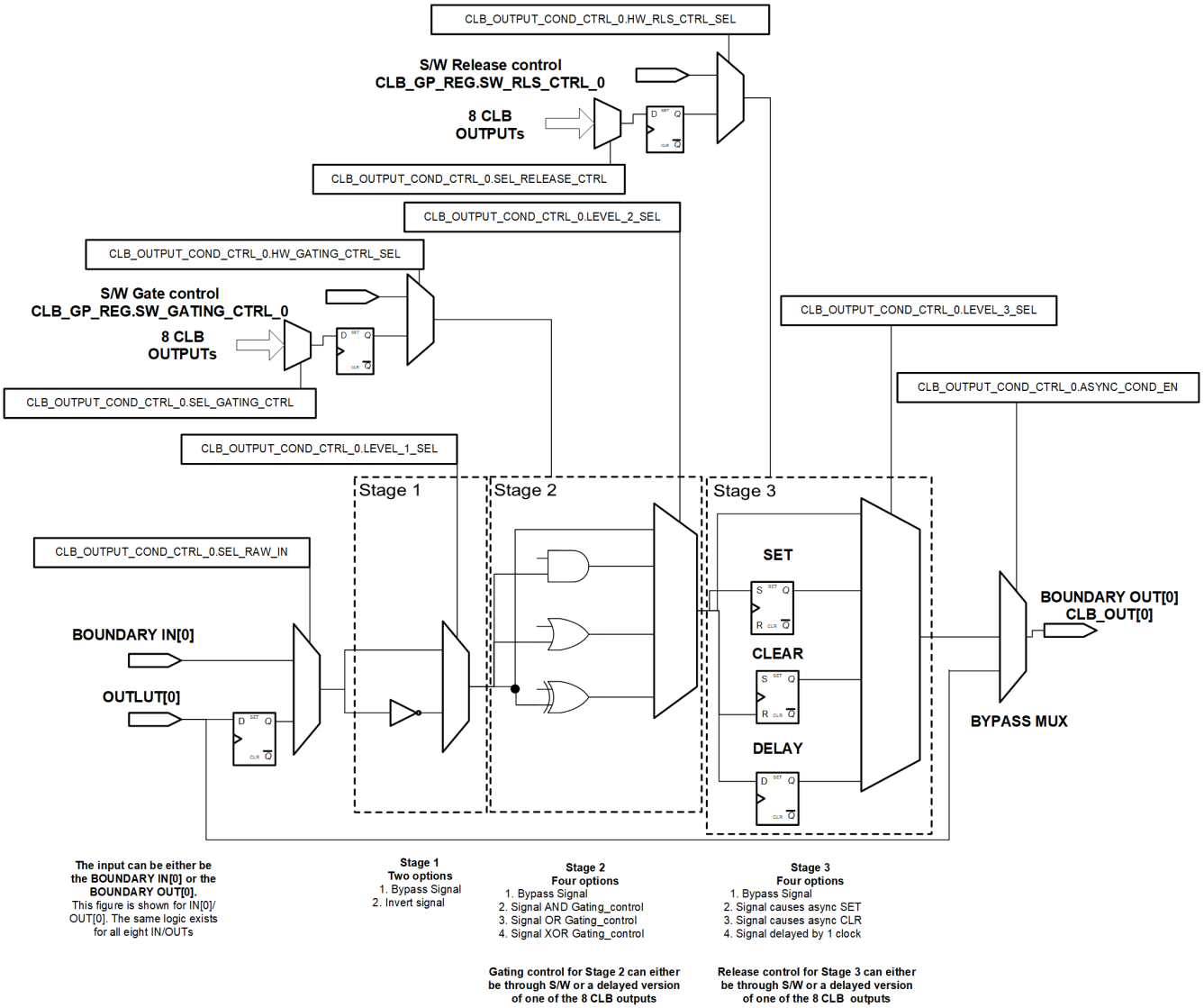


Figure 32-14. AOC Block

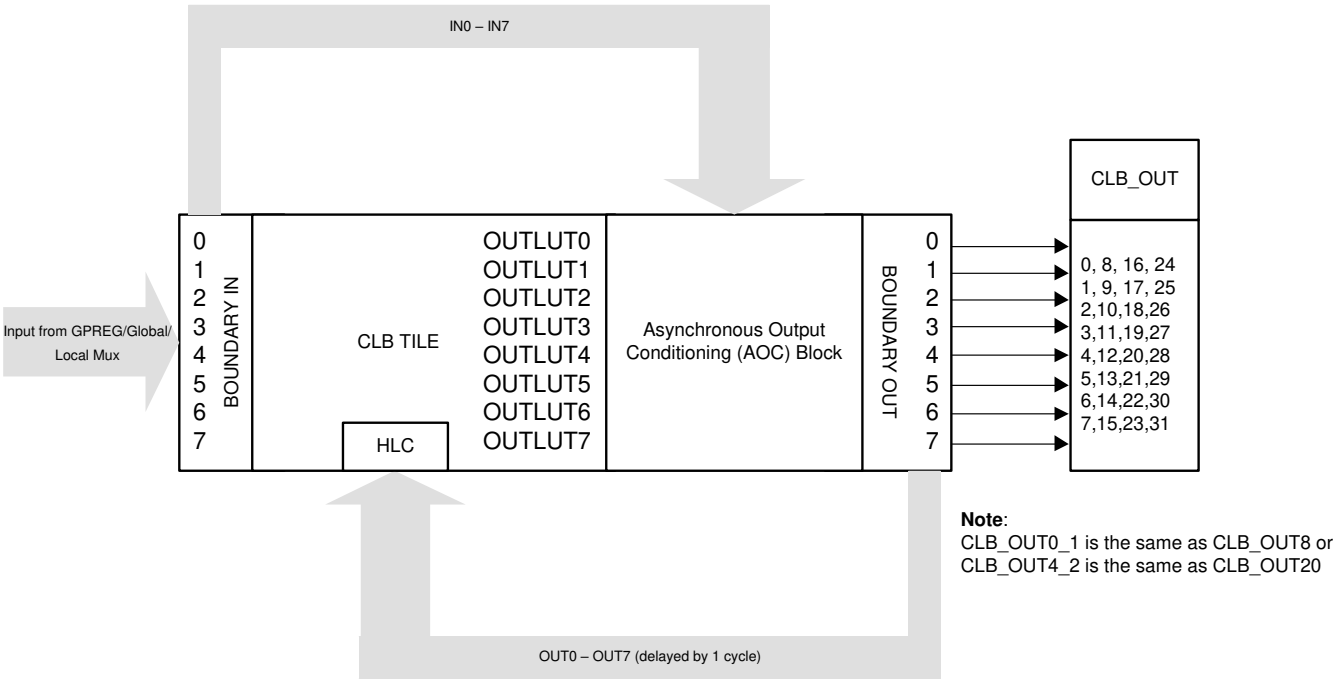


Figure 32-15. AOC Block and The CLB TILE

Note

Only CLB_OUT12 to CLB_OUT15 can be used as ASYNC outputs. This is the same as CLB_OUT4_1 to CLB_OUT7_1. GPIO Output XBAR can be used to OUT4_1 and OUT5_1 ASYNC outputs to GPIOs.

32.4.7 High Level Controller (HLC)

The High Level Controller (HLC) is significantly more complex than the other blocks in the CLB tile. It performs two main functions:

- Provides a means of communication and data exchange with the rest of the device.
- Provides a programmable, event-based action system, which performs computation, manipulation of logic functionality, and data movement. In other words, events can be configured to trigger a predefined set of actions in the CLB tile, or to initiate data exchange with the rest of the device.

The architecture of the HLC is shown in Figure 32-16. It is an event-based system capable of handling up to four simultaneous events that can be selected from any outputs of the other blocks within the tile or from an external input.

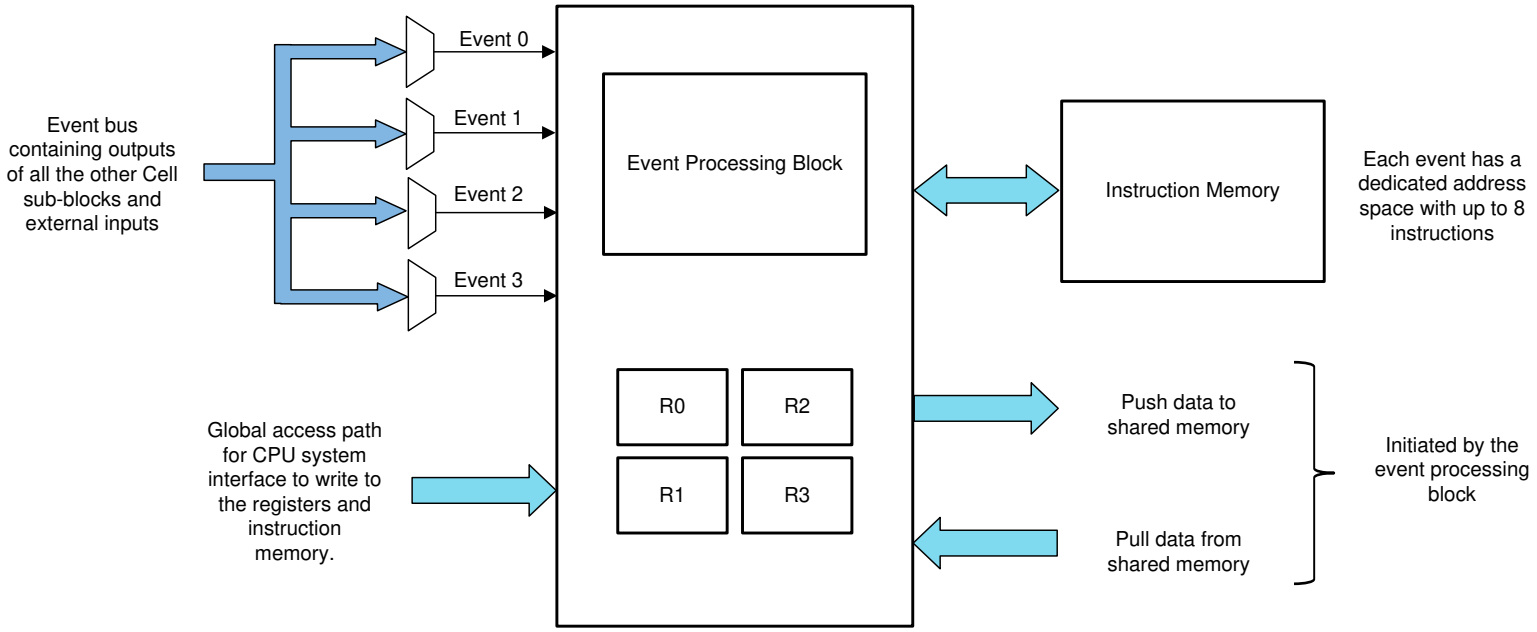


Figure 32-16. High Level Controller Block

32.4.7.1 High Level Controller Events

Each of the four HLC events has a dedicated address from which instructions are executed. Events are selected from the set of signals listed in Table 32-8. The lowest numbered event (Event 0) has the highest priority, and the highest numbered event (Event 3) has the lowest priority.

Table 32-8. HLC Event List

Index	HLC Event Mux
0	Always '0'
1	COUNTER_0 MATCH2
2	COUNTER_0 ZERO
3	COUNTER_0 MATCH1
4	FSM_0 STATE_BIT_0
5	FSM_0 STATE_BIT_1
6	FSM_0 LUT output
7	LUT4_0 output
8	Always '1'
9	COUNTER_1 MATCH2
10	COUNTER_1 ZERO
11	COUNTER_1 MATCH1
12	FSM_1 STATE_BIT_0
13	FSM_1 STATE_BIT_1
14	FSM_1 LUT output
15	LUT4_1 output
16	Always '0'
17	COUNTER_2 MATCH2
18	COUNTER_2 ZERO
19	COUNTER_2 MATCH1
20	FSM_2 STATE_BIT_0
21	FSM_2 STATE_BIT_1
22	FSM_2 LUT output
23	LUT4_2 output
24	External Input 0
25	External Input 1
26	External Input 2
27	External Input 3
28	External Input 4
29	External Input 5
30	External Input 6
31	External Input 7

32.4.7.2 High Level Controller Instructions

The instruction memory supports up to eight instructions per event. Each instruction sequence gets triggered on the rising edge of the corresponding event.

The HLC memory supports up to eight instructions per event, starting at the beginning of the fixed address range shown in Table 32-9. An instruction sequence is triggered on the rising edge of the corresponding event. If two or more events occur simultaneously, the associated instruction sequences will each be executed sequentially in priority order.

Table 32-9. HLC Instruction Address Ranges

Address	Instructions for
00000 to 00111	Event 0
01000 to 01111	Event 1
10000 to 10111	Event 2
11000 to 11111	Event 3

The HLC instruction format is shown in Table 32-10.

Table 32-10. HLC Instruction Format

Last Instruction Bit	5-Bit Opcode	3-Bit Source	3-Bit Destination
This bit when set to 1 will stop execution after the current instruction.	MOV 00000 MOV_T1 00001 MOV_T2 00010 PUSH 00011 PULL 00100 ADD 00101 SUB 00110 INTR 00111	Source can be R0, R1, R2, R3, C0, C1, C2.	Destination can be R0, R1, R2, R3, C0, C1, C2. Note that for ADD/SUB instructions, only R0 to R3 can be the destination.

R0, R1, R2, and R3 are four 32-bit general-purpose registers in the HLC. C0, C1, and C2 are three counter registers present in the CLB tile. <Src> is used to indicate the source and <Dest> is used to indicate the destination. Table 32-11 describes the HLC instructions.

Table 32-11. HLC Instruction Description

Instruction	Description
ADD <Src> <Dest>	This instruction performs an unsigned 32-bit addition. <Dest> = <Dest> + <Src>. The <Src> can be R0, R1, R2, R3, C0, C1, or C2. The <Dest> can only be R0 or R3.
INTR <6-bit constant>	This instruction will flag an interrupt through the CPU interface. The 6-bit constant is stored in the interrupt flag register CLB_INTR_TAG_REG.
MOV <Src> <Dest>	This instruction moves <Src> to <Dest>. Both <Src> and <Dest> can be any of R0, R1, R2, R3, C0, C1, or C2.
MOV_T1 <Src> <Dest>	This instruction moves <Src> to the Match1 register of the <Dest> counter. <Src> can be any of the registers R0, R1, R2, R3, or the counter values associated with C0, C1, or C2. <Dest> is the Match1 register of any of the counters C0, C1, or C2. Examples are: • This instruction moves the count value in C1 into register R0: MOV_T1 C1 R0 • This instruction moves the value in R2 into the Match1 register of counter C0: MOV_T1 R2 C0
MOV_T2 <Src> <Dest>	This instruction is similar to MOV_T1. It moves <Src> to the Match2 register of the <Dest> counter. <Src> can be any of the registers R0, R1, R2, R3, or the counter values associated with C0, C1, or C2. <Dest> is the Match2 register of any of the counters C0, C1, or C2.

Table 32-11. HLC Instruction Description (continued)

Instruction	Description
PULL <Dest>	This instruction transfers data from to the data exchange pull memory buffer in the CPU interface to the <Dest> register. <Dest> can be any of R0, R1, R2, R3. The PULL instruction is used as seen from the High Level Controller and a PULL operation reads (pulls) data from an internal 4-word FIFO.
PUSH <Src>	This instruction transfers data from <Src> to the data exchange push memory buffer in the CPU interface. <Src> can be any of R0, R1, R2, R3, C0, C1, or C2. The PUSH instruction is used as seen from the High Level Controller and pushes data into an internal 4 word FIFO.
SUB <Src> <Dest>	This instruction performs an unsigned 32-bit subtraction. <Dest> = <Dest> - <Src>. The <Src> can be R0, R1, R2, R3, C0, C1, or C2. The <Dest> can only be R0 or R3.

32.4.7.3 <Src> and <Dest>

Three bits are used to encode the <Src> and <Dest> registers as shown in Table 32-12.

Table 32-12. HLC Register Encoding

Bits	Register
000	R0
001	R1
010	R2
011	R3
100	C0
101	C1
110	C2

32.4.7.4 Operation of the PUSH and PULL Instructions (overflow and underflow detection)

The PUSH and PULL operations of the HLC are intended for data exchange with the host system. There are separate FIFO buffers for PUSH and PULL operations. For example, a series of PUSH operations will write to successive locations in a linearly mapped memory buffer. The PUSH buffer and the PULL buffer are mapped at address offsets shown in the registers section.

The CPU can read from and write to the PUSH and PULL buffers respectively, in order to exchange data with the HLC. Data pushed by the HLC is read by the CPU from the PUSH buffers. Data sent from the CPU to the HLC will be written by the CPU to the PULL buffer and will be read by the HLC using the PULL instruction.

There are separate PUSH and PULL address pointers that increment each time the HLC performs a PUSH or PULL operation. These address pointers are also memory-mapped so that the CPU can determine their value. They are also writable and can be reset by the CPU can at any time.

Overflow and underflow detection is done by simply reading the values of the PUSH and PULL address pointers.

In the CLB module of the device, the depth of the PUSH and PULL FIFOs is four 32-bit words each. If the CPU starts a fresh data transfer to the PULL buffers and sees the address pointer greater than four, then an underflow has occurred since the HLC has pulled more data than the number of words written by the CPU into the buffer.

32.5 CPU Interface

32.5.1 Register Description

There are three classes of registers that are used to control and configure the CLB tile. This specification only describes the offset addresses of the registers. The absolute register addresses will be different for each CLB tile. The three instances of the various blocks (LUT4, FSM, and Counter Block) are numbered 0, 1, and 2.

- **Logic configuration registers (0x000 – 0x0FF):** These registers control the core reconfiguration logic for the tile. All registers in this group are EALLOW protected and also protected by the LOCK register.
- **Top level control registers (0x100 – 0x1FF):** These registers are used for top level and device related control of the CLB. These registers typically control mux selects for inputs, global enables, and so forth, and are accessible by normal memory mapped access. Some of these registers have EALLOW and LOCK protection.
- **Data exchange registers (0x200 – 0x3FF):** These registers are used to exchange data between the CLB and the rest of the device. They are accessible by normal memory mapped access and no EALLOW or LOCK protection exists.

Note

EALLOW protection means that the write access to the register will be enabled only when the EALLOW instruction has been executed prior to the write access. The complementary EDIS instruction disables access to all registers protected in this way. For more information, see .

32.5.2 Non-Memory Mapped Registers

The memory mapped CLB registers are described later in this document; however, many of the CLB resources including counters, the instruction memory of the High Level Controller, and the HLC general-purpose registers (R0 through R3) are only indirectly accessible through a local interface bus and are not memory mapped. These registers are accessible through the two memory-mapped registers CLB_LOAD_DATA and CLB_LOAD_ADDR.

You load the data to be written into the CLB_LOAD_DATA register, then load the appropriate address into CLB_LOAD_ADDR to determine where this data is written. Writing a '1' to bit position 0 in the CLB_LOAD_EN register then causes an internal write operation to be triggered. The address allocation for the CLB_LOAD_ADDR register is shown in Table 32-13.

Table 32-13. Non-Memory Mapped Register Addresses

Address (Binary)	Resource
000000 to 000010	Counter 0 to 2 load value
000100 to 000110	Counter 0 to 2 Match1 value
001000 to 001010	Counter 0 to 2 Match2 value
001100 to 001111	R0 to R3 of High Level controller
100000 to 100111	Instructions for Event 0
101000 to 101111	Instructions for Event 1
110000 to 110111	Instructions for Event 2
111000 to 111111	Instructions for Event 3

Use the following steps to load the value 0x11223344 into the general purpose R0 register:

1. Write 0x11223344 to CLB_LOAD_DATA.
2. Write 0xc to CLB_LOAD_ADDR.
3. Write 0x1 to CLB_LOAD_EN.

Note

Even though HLC registers are accessible by the CPU, your application code needs to ensure that no other CLB internal logics are updating the same HLC register at the same time, causing a race condition.

32.6 CLB Data Export Through SPI RX Buffer

For a continuous export of data from the CLB peripherals, SPI RX buffers can be used. CLB data can be exported through the SPI RX buffers without CPU/CLA interventions.

For F28002x, CLB1 and CLB2 have access to SPIA and SPIB.

Table 32-14. CLB to SPI RX Access

CLB Instance	SPI Instance
CLB1	SPIA
CLB2	SPIB

When the CLB to SPI data exporting is enabled, 16-bit data can be exported from CLB to SPI RX buffers. The 32-bit HLC R0 register is the data which is exported to the SPI RX buffers. The user can select which 16-bit range of the HLC R0 is exported by configuring the **CLB_SPI_DATA_CTRL_HI.SHIFT**. The CLB also controls when HLC R0 data must be transferred to the SPI RX buffer through **CLB_SPI_DATA_CTRL_HI.STRB** that selects one of the HLC event signals from the static switch block.

When CLB to SPI data exporting is required, it is important to note:

- The selected SPI transmit functionality is not affected.
- Even though the data is being pushed into the SPI RX buffers by the CLB, the SPI RX interrupt and the DMA trigger for SPI RX in the respective peripherals must be configured.
- The SPI can resume normal operation when the CLB to SPI data exporting is disabled.

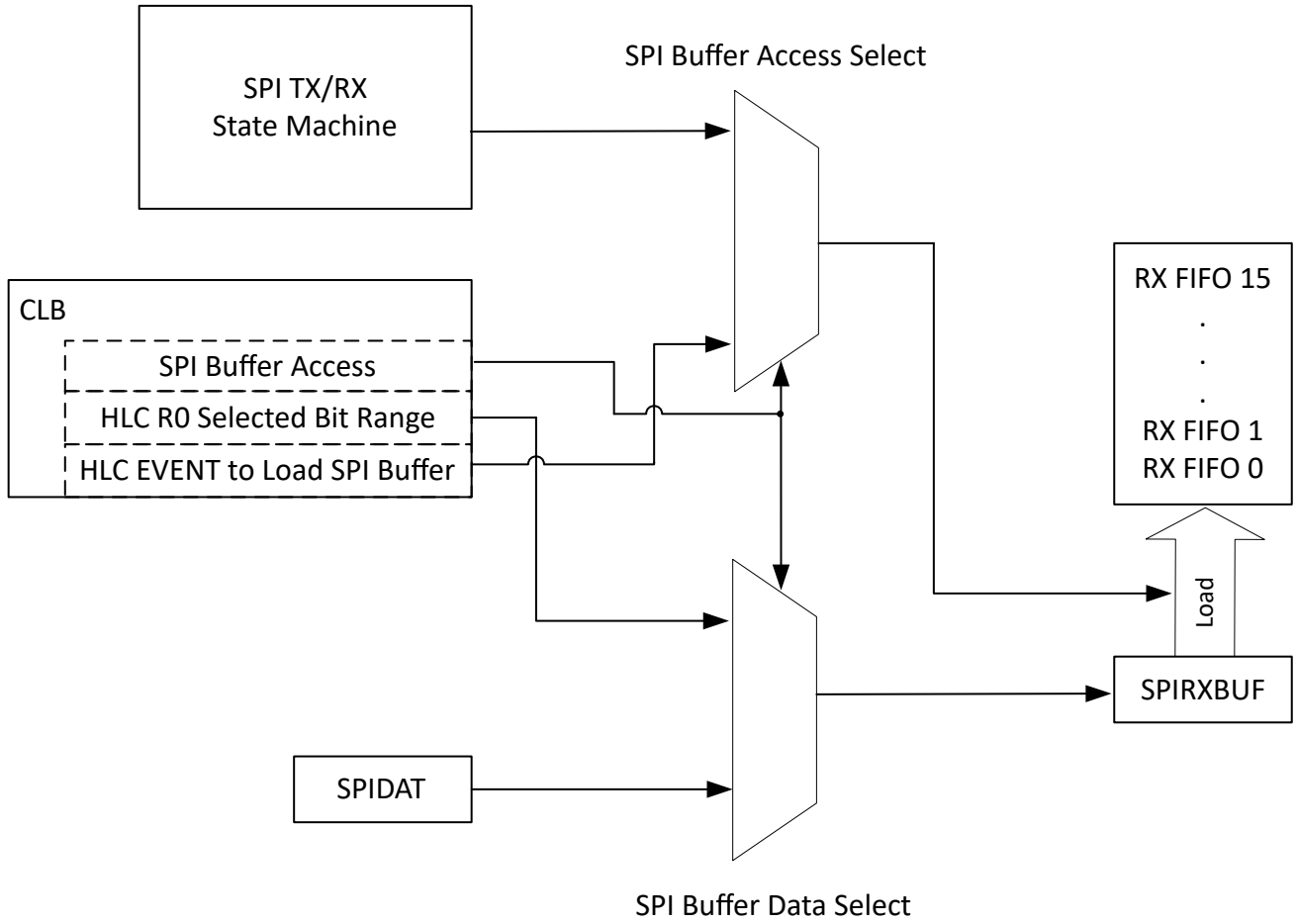


Figure 32-17. CLB Control of SPI RX Buffer

32.7 Software

32.7.1 CLB Examples

NOTE: These examples are located in the C2000Ware installation at the following location:
C2000Ware_VERSION#/driverlib/f28003x/examples/clb

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

32.7.1.1 CLB Timer Two States

FILE: clb_ex10_timer_two_states.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.2 CLB Interrupt Tag

FILE: clb_ex11_interrupt_tag.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.3 CLB Output Intersect

FILE: clb_ex12_output_intersect.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.4 CLB PUSH PULL

FILE: clb_ex13_push_pull.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.5 CLB Multi Tile

FILE: clb_ex14_multi_tile.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.6 CLB Glue Logic

FILE: clb_ex16_glue_logic.c

For the detailed description of this example, please refer to :
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.7 CLB based One-shot PWM

FILE: clb_ex17_one_shot_pwm.c

For the detailed description of this example, please refer to :
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.8 CLB AOC Control

FILE: clb_ex18_aoc.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.9 CLB AOC Release Control

FILE: clb_ex19_aoc_release_control.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.10 CLB Combinational Logic

FILE: clb_ex1_combinatorial_logic.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.11 CLB XBARs

FILE: clb_ex20_clbxbars.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.12 CLB AOC Control

FILE: clb_ex21_clockprescalar_nmi.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.13 CLB Serializer

FILE: clb_ex22_serializer.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.14 CLB LFSR

FILE: clb_ex23_lfsr.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.15 CLB Lock Output Mask

FILE: clb_ex24_lock_output_mask.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.16 CLB INPUT Pipeline Mode

FILE: clb_ex25_input_pipeline.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.17 CLB Clocking and PIPELINE Mode

FILE: clb_ex26_clocking_pipeline.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.18 CLB SPI Data Export

FILE: clb_ex27_spi_data_export.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.19 CLB SPI Data Export DMA

FILE: clb_ex28_spi_data_export_dma.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.20 CLB Trip Zone Timestamp

FILE: clb_ex29_timestamp.c

This example displays how to timestamp interrupts generated by the CLB. An interrupt is generated when ePWM1 is tripped.

ePWM1 is configured to be interrupted by TZ1 and TZ2, both one shot trip sources.

The CLB is configured as follows:
- COUNTER0 and COUNTER1 continually count when the program begins.
- COUNTER0 timestamps TZ1 and COUNTER1 timestamps TZ2.
- COUNTER2 increments once when COUNTER0/COUNTER1 overflows using LUT2.
- FSM0/1 are configured to sync counters and stop COUNTER0/1 when an interrupt is received.
- TZ1 (GPIO12) and TZ2 (GPIO13) are routed as inputs through CLBXBAR.
- BOUNDARY.in0 denotes TZ1. On rising edge, HLC issues an interrupt with tag 12.
- BOUDNARY.in1 denotes TZ2. On rising edge, HLC issues an interrupt with tag 13.
- BOUNDARY.in7 serves as a simultaneous enable for COUNTER0/1 to begin counting.

TZ1 is tripped when GPIO12 is connected to GND. TZ2 is tripped when GPIO13 is connected to GND. When an interrupt occurs, the interrupt handler determines the initial trip source and stores this value in a variable 'initialTripZone'.

View these variables in Debug Expressions tab:

initialTripZone: stores the first TZ to have been tripped tz1Counter64bit: stores the counter value at the instant that TZ1 is tripped. tz2Counter64bit: stores the counter value at the instant that TZ2 is tripped.

32.7.1.21 CLB GPIO Input Filter

FILE: clb_ex2_gpio_input_filter.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.22 CLB Auxilary PWM

FILE: clb_ex3_auxiliary_pwm.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.23 CLB PWM Protection

FILE: clb_ex4_pwm_protection.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.24 CLB Event Window

FILE: clb_ex5_event_window.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.25 CLB Signal Generator

FILE: clb_ex6_siggen.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.26 CLB State Machine

FILE: clb_ex7_state_machine.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.27 CLB External Signal AND Gate

FILE: clb_ex8_external_signal_AND_gate.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.28 CLB Timer

FILE: clb_ex9_timer.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.7.1.29 CLB Empty Project

FILE: empty.c

For the detailed description of this example, please refer to:
C2000Ware_PATH\utilities\clb_tool\clb_syscfg\doc\CLB Tool Users Guide.pdf

32.8 CLB Registers

This section describes the Configurable Logic Block Registers.

32.8.1 CLB Base Address Table

Table 32-15. CLB Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
Clb1LogicCfgRegs	CLB_LOGIC_CONFIG_REGS	CLB1_LOGICCFG_BASE	0x0000_3000	YES	-	YES	YES	-
Clb1LogicCtrlRegs	CLB_LOGIC_CONTROL_REGS	CLB1_LOGICCTRL_BASE	0x0000_3100	YES	-	YES	YES	-
Clb1DataExchRegs	CLB_DATA_EXCHANGE_REGS	CLB1_DATAEXCH_BASE	0x0000_3180	YES	-	YES	YES	-
Clb2LogicCfgRegs	CLB_LOGIC_CONFIG_REGS	CLB2_LOGICCFG_BASE	0x0000_3400	YES	-	YES	YES	-
Clb2LogicCtrlRegs	CLB_LOGIC_CONTROL_REGS	CLB2_LOGICCTRL_BASE	0x0000_3500	YES	-	YES	YES	-
Clb2DataExchRegs	CLB_DATA_EXCHANGE_REGS	CLB2_DATAEXCH_BASE	0x0000_3580	YES	-	YES	YES	-
Clb3LogicCfgRegs	CLB_LOGIC_CONFIG_REGS	CLB3_LOGICCFG_BASE	0x0000_3800	YES	-	YES	YES	-
Clb3LogicCtrlRegs	CLB_LOGIC_CONTROL_REGS	CLB3_LOGICCTRL_BASE	0x0000_3900	YES	-	YES	YES	-
Clb3DataExchRegs	CLB_DATA_EXCHANGE_REGS	CLB3_DATAEXCH_BASE	0x0000_3980	YES	-	YES	YES	-
Clb4LogicCfgRegs	CLB_LOGIC_CONFIG_REGS	CLB4_LOGICCFG_BASE	0x0000_3C00	YES	-	YES	YES	-
Clb4LogicCtrlRegs	CLB_LOGIC_CONTROL_REGS	CLB4_LOGICCTRL_BASE	0x0000_3D00	YES	-	YES	YES	-
Clb4DataExchRegs	CLB_DATA_EXCHANGE_REGS	CLB4_DATAEXCH_BASE	0x0000_3D80	YES	-	YES	YES	-

32.8.2 CLB_LOGIC_CONFIG_REGS Registers

Table 32-16 lists the memory-mapped registers for the CLB_LOGIC_CONFIG_REGS registers. All register offset addresses not listed in Table 32-16 should be considered as reserved locations and the register contents should not be modified.

Table 32-16. CLB_LOGIC_CONFIG_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
2h	CLB_COUNT_RESET	Counter Block RESET	EALLOW, LOCK	Go
4h	CLB_COUNT_MODE_1	Counter Block MODE_1	EALLOW, LOCK	Go
6h	CLB_COUNT_MODE_0	Counter Block MODE_0	EALLOW, LOCK	Go
8h	CLB_COUNT_EVENT	Counter Block EVENT	EALLOW, LOCK	Go
Ah	CLB_FSM_EXTRA_IN0	FSM Extra EXT_IN0	EALLOW, LOCK	Go
Ch	CLB_FSM_EXTERNAL_IN0	FSM EXT_IN0	EALLOW, LOCK	Go
Eh	CLB_FSM_EXTERNAL_IN1	FSM_EXT_IN1	EALLOW, LOCK	Go
10h	CLB_FSM_EXTRA_IN1	FSM Extra_EXT_IN1	EALLOW, LOCK	Go
12h	CLB_LUT4_IN0	LUT4_0/1/2 IN0 input source	EALLOW, LOCK	Go
14h	CLB_LUT4_IN1	LUT4_0/1/2 IN1 input source	EALLOW, LOCK	Go
16h	CLB_LUT4_IN2	LUT4_0/1/2 IN2 input source	EALLOW, LOCK	Go
18h	CLB_LUT4_IN3	LUT4_0/1/2 IN3 input source	EALLOW, LOCK	Go
1Ch	CLB_FSM_LUT_FN1_0	LUT function for FSM Unit 1 and Unit 0	EALLOW, LOCK	Go
1Eh	CLB_FSM_LUT_FN2	LUT function for FSM Unit 2	EALLOW, LOCK	Go
20h	CLB_LUT4_FN1_0	LUT function for LUT4 block of Unit 1 and 0	EALLOW, LOCK	Go
22h	CLB_LUT4_FN2	LUT function for LUT4 block of Unit 2	EALLOW, LOCK	Go
24h	CLB_FSM_NEXT_STATE_0	FSM Next state equations for Unit 0	EALLOW, LOCK	Go
26h	CLB_FSM_NEXT_STATE_1	FSM Next state equations for Unit 1	EALLOW, LOCK	Go
28h	CLB_FSM_NEXT_STATE_2	FSM Next state equations for Unit 2	EALLOW, LOCK	Go
2Ah	CLB_MISC_CONTROL	Static controls for Ctr,FSM	EALLOW, LOCK	Go
2Ch	CLB_OUTPUT_LUT_0	Inp Sel, LUT fns for Out0	EALLOW, LOCK	Go
2Eh	CLB_OUTPUT_LUT_1	Inp Sel, LUT fns for Out1	EALLOW, LOCK	Go
30h	CLB_OUTPUT_LUT_2	Inp Sel, LUT fns for Out2	EALLOW, LOCK	Go
32h	CLB_OUTPUT_LUT_3	Inp Sel, LUT fns for Out3	EALLOW, LOCK	Go
34h	CLB_OUTPUT_LUT_4	Inp Sel, LUT fns for Out4	EALLOW, LOCK	Go
36h	CLB_OUTPUT_LUT_5	Inp Sel, LUT fns for Out5	EALLOW, LOCK	Go
38h	CLB_OUTPUT_LUT_6	Inp Sel, LUT fns for Out6	EALLOW, LOCK	Go
3Ah	CLB_OUTPUT_LUT_7	Inp Sel, LUT fns for Out7	EALLOW, LOCK	Go
3Ch	CLB_HLC_EVENT_SEL	Event Selector register for the High Level controller	EALLOW, LOCK	Go
3Eh	CLB_COUNT_MATCH_TAP_SEL	Counter tap values for match1 and match2 outputs	EALLOW, LOCK	Go
40h	CLB_OUTPUT_COND_CTRL_0	Output conditioning control for output 0	EALLOW, LOCK	Go
42h	CLB_OUTPUT_COND_CTRL_1	Output conditioning control for output 1	EALLOW, LOCK	Go
44h	CLB_OUTPUT_COND_CTRL_2	Output conditioning control for output 2	EALLOW, LOCK	Go
46h	CLB_OUTPUT_COND_CTRL_3	Output conditioning control for output 3	EALLOW, LOCK	Go
48h	CLB_OUTPUT_COND_CTRL_4	Output conditioning control for output 4	EALLOW, LOCK	Go
4Ah	CLB_OUTPUT_COND_CTRL_5	Output conditioning control for output 5	EALLOW, LOCK	Go
4Ch	CLB_OUTPUT_COND_CTRL_6	Output conditioning control for output 6	EALLOW, LOCK	Go
4Eh	CLB_OUTPUT_COND_CTRL_7	Output conditioning control for output 7	EALLOW, LOCK	Go
50h	CLB_MISC_ACCESS_CTRL	Miscellaneous Access and enable control	EALLOW, LOCK	Go

Table 32-16. CLB_LOGIC_CONFIG_REGS Registers (continued)

Offset	Acronym	Register Name	Write Protection	Section
51h	CLB_SPI_DATA_CTRL_HI	CLB to SPI buffer control High	EALLOW, LOCK	Go

Complex bit access types are encoded to fit into small table cells. Table 32-17 shows the codes that are used for access types in this section.

Table 32-17. CLB_LOGIC_CONFIG_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

32.8.2.1 CLB_COUNT_RESET Register (Offset = 2h) [Reset = 0h]

CLB_COUNT_RESET is shown in Figure 32-18 and described in Table 32-18.

Return to the Summary Table.

Counter Block RESET

Figure 32-18. CLB_COUNT_RESET Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																	SEL_2					SEL_1					SEL_0				
R-0h																	R/W-0h					R/W-0h					R/W-0h				

Table 32-18. CLB_COUNT_RESET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Reserved
14-10	SEL_2	R/W	0h	Counter reset select inputs for unit 2. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	SEL_1	R/W	0h	Counter reset select inputs for unit 1. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	SEL_0	R/W	0h	Counter reset select inputs for unit 0. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

32.8.2.2 CLB_COUNT_MODE_1 Register (Offset = 4h) [Reset = 0h]

CLB_COUNT_MODE_1 is shown in Figure 32-19 and described in Table 32-19.

Return to the Summary Table.

Counter Block MODE_1

Figure 32-19. CLB_COUNT_MODE_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																	SEL_2					SEL_1					SEL_0				
R-0h																	R/W-0h					R/W-0h					R/W-0h				

Table 32-19. CLB_COUNT_MODE_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Reserved
14-10	SEL_2	R/W	0h	Counter MODE_1 select inputs for unit 2. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	SEL_1	R/W	0h	Counter MODE_1 select inputs for unit 1. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	SEL_0	R/W	0h	Counter MODE_1 select inputs for unit 0. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

32.8.2.3 CLB_COUNT_MODE_0 Register (Offset = 6h) [Reset = 0h]

CLB_COUNT_MODE_0 is shown in Figure 32-20 and described in Table 32-20.

Return to the Summary Table.

Counter Block MODE_0

Figure 32-20. CLB_COUNT_MODE_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																	SEL_2					SEL_1					SEL_0				
R-0h																	R/W-0h					R/W-0h					R/W-0h				

Table 32-20. CLB_COUNT_MODE_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Reserved
14-10	SEL_2	R/W	0h	Counter MODE_0 select inputs for unit 2. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	SEL_1	R/W	0h	Counter MODE_0 select inputs for unit 1. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	SEL_0	R/W	0h	Counter MODE_0 select inputs for unit 0. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

32.8.2.4 CLB_COUNT_EVENT Register (Offset = 8h) [Reset = 0h]

CLB_COUNT_EVENT is shown in Figure 32-21 and described in Table 32-21.

Return to the Summary Table.

Counter Block EVENT

Figure 32-21. CLB_COUNT_EVENT Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																	SEL_2					SEL_1					SEL_0				
R-0h																	R/W-0h					R/W-0h					R/W-0h				

Table 32-21. CLB_COUNT_EVENT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Reserved
14-10	SEL_2	R/W	0h	Counter event select inputs for unit 2. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	SEL_1	R/W	0h	Counter event select inputs for unit 1. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	SEL_0	R/W	0h	Counter event select inputs for unit 0. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

32.8.2.5 CLB_FSM_EXTRA_IN0 Register (Offset = Ah) [Reset = 0h]

CLB_FSM_EXTRA_IN0 is shown in Figure 32-22 and described in Table 32-22.

Return to the Summary Table.

FSM Extra EXT_IN0

Figure 32-22. CLB_FSM_EXTRA_IN0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																	SEL_2					SEL_1					SEL_0				
R-0h																	R/W-0h					R/W-0h					R/W-0h				

Table 32-22. CLB_FSM_EXTRA_IN0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Reserved
14-10	SEL_2	R/W	0h	FSM block extra external IN0 select inputs for unit 2. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	SEL_1	R/W	0h	FSM block extra external IN0 select inputs for unit 1. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	SEL_0	R/W	0h	FSM block extra external IN0 select inputs for unit 0. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

32.8.2.6 CLB_FSM_EXTERNAL_IN0 Register (Offset = Ch) [Reset = 0h]

CLB_FSM_EXTERNAL_IN0 is shown in Figure 32-23 and described in Table 32-23.

Return to the Summary Table.

FSM EXT_IN0

Figure 32-23. CLB_FSM_EXTERNAL_IN0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																	SEL_2					SEL_1					SEL_0				
R-0h																	R/W-0h					R/W-0h					R/W-0h				

Table 32-23. CLB_FSM_EXTERNAL_IN0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Reserved
14-10	SEL_2	R/W	0h	FSM block EXT_IN0 select input for unit 2. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	SEL_1	R/W	0h	FSM block EXT_IN0 select input for unit 1. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	SEL_0	R/W	0h	FSM block EXT_IN0 select input for unit 0. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

32.8.2.7 CLB_FSM_EXTERNAL_IN1 Register (Offset = Eh) [Reset = 0h]

CLB_FSM_EXTERNAL_IN1 is shown in Figure 32-24 and described in Table 32-24.

Return to the Summary Table.

FSM_EXT_IN1

Figure 32-24. CLB_FSM_EXTERNAL_IN1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																	SEL_2					SEL_1					SEL_0				
R-0h																	R/W-0h					R/W-0h					R/W-0h				

Table 32-24. CLB_FSM_EXTERNAL_IN1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Reserved
14-10	SEL_2	R/W	0h	FSM block EXT_IN1 select input for unit 2. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	SEL_1	R/W	0h	FSM block EXT_IN1 select input for unit 1. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	SEL_0	R/W	0h	FSM block EXT_IN1 select input for unit 0. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

32.8.2.8 CLB_FSM_EXTRA_IN1 Register (Offset = 10h) [Reset = 0h]

CLB_FSM_EXTRA_IN1 is shown in Figure 32-25 and described in Table 32-25.

Return to the Summary Table.

FSM Extra_EXT_IN1

Figure 32-25. CLB_FSM_EXTRA_IN1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																	SEL_2					SEL_1					SEL_0				
R-0h																	R/W-0h					R/W-0h					R/W-0h				

Table 32-25. CLB_FSM_EXTRA_IN1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Reserved
14-10	SEL_2	R/W	0h	FSM block extra external IN1 select inputs for unit 2. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	SEL_1	R/W	0h	FSM block extra external IN1 select inputs for unit 1. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	SEL_0	R/W	0h	FSM block extra external IN1 select inputs for unit 0. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

32.8.2.9 CLB_LUT4_IN0 Register (Offset = 12h) [Reset = 0h]

CLB_LUT4_IN0 is shown in Figure 32-26 and described in Table 32-26.

Return to the Summary Table.

LUT4_0/1/2 IN0 input source

Figure 32-26. CLB_LUT4_IN0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																	SEL_2					SEL_1					SEL_0				
R-0h																	R/W-0h					R/W-0h					R/W-0h				

Table 32-26. CLB_LUT4_IN0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Reserved
14-10	SEL_2	R/W	0h	LUT4 block IN0 select inputs for unit 2. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	SEL_1	R/W	0h	LUT4 block IN0 select inputs for unit 1. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	SEL_0	R/W	0h	LUT4 block IN0 select inputs for unit 0. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

32.8.2.10 CLB_LUT4_IN1 Register (Offset = 14h) [Reset = 0h]

CLB_LUT4_IN1 is shown in Figure 32-27 and described in Table 32-27.

Return to the Summary Table.

LUT4_0/1/2 IN1 input source

Figure 32-27. CLB_LUT4_IN1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																	SEL_2					SEL_1					SEL_0				
R-0h																	R/W-0h					R/W-0h					R/W-0h				

Table 32-27. CLB_LUT4_IN1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Reserved
14-10	SEL_2	R/W	0h	LUT4 block IN1 select inputs for unit 2. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	SEL_1	R/W	0h	LUT4 block IN1 select inputs for unit 1. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	SEL_0	R/W	0h	LUT4 block IN1 select inputs for unit 0. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

32.8.2.11 CLB_LUT4_IN2 Register (Offset = 16h) [Reset = 0h]

CLB_LUT4_IN2 is shown in Figure 32-28 and described in Table 32-28.

Return to the Summary Table.

LUT4_0/1/2 IN2 input source

Figure 32-28. CLB_LUT4_IN2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																	SEL_2					SEL_1					SEL_0				
R-0h																	R/W-0h					R/W-0h					R/W-0h				

Table 32-28. CLB_LUT4_IN2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Reserved
14-10	SEL_2	R/W	0h	LUT4 block IN2 select inputs for unit 2. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	SEL_1	R/W	0h	LUT4 block IN2 select inputs for unit 1. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	SEL_0	R/W	0h	LUT4 block IN2 select inputs for unit 0. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

32.8.2.12 CLB_LUT4_IN3 Register (Offset = 18h) [Reset = 0h]

CLB_LUT4_IN3 is shown in Figure 32-29 and described in Table 32-29.

Return to the Summary Table.

LUT4_0/1/2 IN3 input source

Figure 32-29. CLB_LUT4_IN3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																	SEL_2					SEL_1					SEL_0				
R-0h																	R/W-0h					R/W-0h					R/W-0h				

Table 32-29. CLB_LUT4_IN3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Reserved
14-10	SEL_2	R/W	0h	LUT4 block IN3 select inputs for unit 2. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	SEL_1	R/W	0h	LUT4 block IN3 select inputs for unit 1. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	SEL_0	R/W	0h	LUT4 block IN3 select inputs for unit 0. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

32.8.2.13 CLB_FSM_LUT_FN1_0 Register (Offset = 1Ch) [Reset = 0h]

CLB_FSM_LUT_FN1_0 is shown in Figure 32-30 and described in Table 32-30.

Return to the Summary Table.

LUT function for FSM Unit 1 and Unit 0

Figure 32-30. CLB_FSM_LUT_FN1_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FN1																FN0															
R/W-0h																R/W-0h															

Table 32-30. CLB_FSM_LUT_FN1_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	FN1	R/W	0h	FSM block LUT output function for unit 1 Reset type: SYSRSn
15-0	FN0	R/W	0h	FSM block LUT output function for unit 0 Reset type: SYSRSn

32.8.2.14 CLB_FSM_LUT_FN2 Register (Offset = 1Eh) [Reset = 0h]

CLB_FSM_LUT_FN2 is shown in Figure 32-31 and described in Table 32-31.

Return to the Summary Table.

LUT function for FSM Unit 2

Figure 32-31. CLB_FSM_LUT_FN2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																FN1															
R-0h																R/W-0h															

Table 32-31. CLB_FSM_LUT_FN2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	FN1	R/W	0h	LUT4 output function for unit 2 Reset type: SYSRSn

32.8.2.15 CLB_LUT4_FN1_0 Register (Offset = 20h) [Reset = 0h]

CLB_LUT4_FN1_0 is shown in Figure 32-32 and described in Table 32-32.

Return to the Summary Table.

LUT function for LUT4 block of Unit 1 and 0

Figure 32-32. CLB_LUT4_FN1_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FN1																FN0															
R/W-0h																R/W-0h															

Table 32-32. CLB_LUT4_FN1_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	FN1	R/W	0h	LUT4 output function for unit 1 Reset type: SYSRSn
15-0	FN0	R/W	0h	LUT4 output function for unit 0 Reset type: SYSRSn

32.8.2.16 CLB_LUT4_FN2 Register (Offset = 22h) [Reset = 0h]

CLB_LUT4_FN2 is shown in Figure 32-33 and described in Table 32-33.

Return to the Summary Table.

LUT function for LUT4 block of Unit 2

Figure 32-33. CLB_LUT4_FN2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																FN1															
R-0h																R/W-0h															

Table 32-33. CLB_LUT4_FN2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-0	FN1	R/W	0h	LUT4 output function for unit 2 Reset type: SYSRSn

32.8.2.17 CLB_FSM_NEXT_STATE_0 Register (Offset = 24h) [Reset = 0h]

CLB_FSM_NEXT_STATE_0 is shown in Figure 32-34 and described in Table 32-34.

Return to the Summary Table.

FSM Next state equations for Unit 0

Figure 32-34. CLB_FSM_NEXT_STATE_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S1																S0															
R/W-0h																R/W-0h															

Table 32-34. CLB_FSM_NEXT_STATE_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	S1	R/W	0h	FSM next state function for S1, unit0 Reset type: SYSRSn
15-0	S0	R/W	0h	FSM next state function for S0, unit0 Reset type: SYSRSn

32.8.2.18 CLB_FSM_NEXT_STATE_1 Register (Offset = 26h) [Reset = 0h]

CLB_FSM_NEXT_STATE_1 is shown in Figure 32-35 and described in Table 32-35.

Return to the Summary Table.

FSM Next state equations for Unit 1

Figure 32-35. CLB_FSM_NEXT_STATE_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S1																S0															
R/W-0h																R/W-0h															

Table 32-35. CLB_FSM_NEXT_STATE_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	S1	R/W	0h	FSM next state function for S1, unit1 Reset type: SYSRSn
15-0	S0	R/W	0h	FSM next state function for S0, unit1 Reset type: SYSRSn

32.8.2.19 CLB_FSM_NEXT_STATE_2 Register (Offset = 28h) [Reset = 0h]

CLB_FSM_NEXT_STATE_2 is shown in Figure 32-36 and described in Table 32-36.

Return to the Summary Table.

FSM Next state equations for Unit 2

Figure 32-36. CLB_FSM_NEXT_STATE_2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
S1																S0															
R/W-0h																R/W-0h															

Table 32-36. CLB_FSM_NEXT_STATE_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	S1	R/W	0h	FSM next state function for S1, unit2 Reset type: SYSRSn
15-0	S0	R/W	0h	FSM next state function for S0, unit2 Reset type: SYSRSn

32.8.2.20 CLB_MISC_CONTROL Register (Offset = 2Ah) [Reset = 0h]

CLB_MISC_CONTROL is shown in Figure 32-37 and described in Table 32-37.

Return to the Summary Table.

Static controls for Ctr,FSM

Figure 32-37. CLB_MISC_CONTROL Register

31	30	29	28	27	26	25	24
RESERVED					COUNT2_LFSR_EN	COUNT1_LFSR_EN	COUNT0_LFSR_EN
R-0h					R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
COUNT2_MATCH2_TAP_EN	COUNT1_MATCH2_TAP_EN	COUNT0_MATCH2_TAP_EN	COUNT2_MATCH1_TAP_EN	COUNT1_MATCH1_TAP_EN	COUNT0_MATCH1_TAP_EN	FSM_EXTRA_SEL1_2	FSM_EXTRA_SEL0_2
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
FSM_EXTRA_SEL1_1	FSM_EXTRA_SEL0_1	FSM_EXTRA_SEL1_0	FSM_EXTRA_SEL0_0	COUNT_SERIALIZER_2	COUNT_SERIALIZER_1	COUNT_SERIALIZER_0	COUNT_EVENT_CTRL_2
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
COUNT_DIR_2	COUNT_ADD_SHIFT_2	COUNT_EVENT_CTRL_1	COUNT_DIR_1	COUNT_ADD_SHIFT_1	COUNT_EVENT_CTRL_0	COUNT_DIR_0	COUNT_ADD_SHIFT_0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 32-37. CLB_MISC_CONTROL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-27	RESERVED	R	0h	Reserved
26	COUNT2_LFSR_EN	R/W	0h	Defines if Counter 2 should operate in LFSR mode. This should be set to 1 only if it is in the SERIALIZER mode. 0 = Selects normal serializer operation 1 = Selects LFSR mode of operation Reset type: SYSRSn
25	COUNT1_LFSR_EN	R/W	0h	Defines if Counter 1 should operate in LFSR mode. This should be set to 1 only if it is in the SERIALIZER mode. 0 = Selects normal serializer operation 1 = Selects LFSR mode of operation Reset type: SYSRSn
24	COUNT0_LFSR_EN	R/W	0h	Defines if Counter 0 should operate in LFSR mode. This should be set to 1 only if it is in the SERIALIZER mode. 0 = Selects normal serializer operation 1 = Selects LFSR mode of operation Reset type: SYSRSn
23	COUNT2_MATCH2_TAP_EN	R/W	0h	Defines if the Match2 output should come from the match unit or tapped from a bit position of the counter 0 = Selects Match2 comaprison output 1 = Selects Bit position defined by Match2_Tap_val Reset type: SYSRSn
22	COUNT1_MATCH2_TAP_EN	R/W	0h	Defines if the Match2 output should come from the match unit or tapped from a bit position of the counter 0 = Selects Match2 comaprison output 1 = Selects Bit position defined by Match2_Tap_val Reset type: SYSRSn

Table 32-37. CLB_MISC_CONTROL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	COUNT0_MATCH2_TAP_EN	R/W	0h	Defines if the Match2 output should come from the match unit or tapped from a bit position of the counter 0 = Selects Match2 comaprison output 1 = Selects Bit position defined by Match2_Tap_val Reset type: SYSRSn
20	COUNT2_MATCH1_TAP_EN	R/W	0h	Defines if the Match1 output should come from the match unit or tapped from a bit position of the counter 0 = Selects Match1 comaprison output 1 = Selects Bit position defined by Match1_Tap_val Reset type: SYSRSn
19	COUNT1_MATCH1_TAP_EN	R/W	0h	Defines if the Match1 output should come from the match unit or tapped from a bit position of the counter 0 = Selects Match1 comaprison output 1 = Selects Bit position defined by Match1_Tap_val Reset type: SYSRSn
18	COUNT0_MATCH1_TAP_EN	R/W	0h	Defines if the Match1 output should come from the match unit or tapped from a bit position of the counter 0 = Selects Match1 comaprison output 1 = Selects Bit position defined by Match1_Tap_val Reset type: SYSRSn
17	FSM_EXTRA_SEL1_2	R/W	0h	Defines which input should be selected for the FSM LUT of UNIT 2 0 = Selects State S1 for the FSM LUT 1 = Selects EXTRA_EXT_IN1 for the FSM LUT Reset type: SYSRSn
16	FSM_EXTRA_SEL0_2	R/W	0h	Defines which input should be selected for the FSM LUT of UNIT 2 0 = Selects State S0 for the FSM LUT 1 = Selects EXTRA_EXT_IN0 for the FSM LUT Reset type: SYSRSn
15	FSM_EXTRA_SEL1_1	R/W	0h	Defines which input should be selected for the FSM LUT of UNIT 1 0 = Selects State S1 for the FSM LUT 1 = Selects EXTRA_EXT_IN1 for the FSM LUT Reset type: SYSRSn
14	FSM_EXTRA_SEL0_1	R/W	0h	Defines which input should be selected for the FSM LUT of UNIT 1 0 = Selects State S0 for the FSM LUT 1 = Selects EXTRA_EXT_IN0 for the FSM LUT Reset type: SYSRSn
13	FSM_EXTRA_SEL1_0	R/W	0h	Defines which input should be selected for the FSM LUT of UNIT 0 0 = Selects State S1 for the FSM LUT 1 = Selects EXTRA_EXT_IN1 for the FSM LUT Reset type: SYSRSn
12	FSM_EXTRA_SEL0_0	R/W	0h	Defines which input should be selected for the FSM LUT of UNIT 0 0 = Selects State S0 for the FSM LUT 1 = Selects EXTRA_EXT_IN0 for the FSM LUT Reset type: SYSRSn
11	COUNT_SERIALIZER_2	R/W	0h	Controls if the Counter of UNIT 2 is the Serialzer mode or not. 0 = Normal mode 1 = Serializer mode Reset type: SYSRSn
10	COUNT_SERIALIZER_1	R/W	0h	Controls if the Counter of UNIT 1 is the Serialzer mode or not. 0 = Normal mode 1 = Serializer mode Reset type: SYSRSn
9	COUNT_SERIALIZER_0	R/W	0h	Controls if the Counter of UNIT 0 is the Serialzer mode or not. 0 = Normal mode 1 = Serializer mode Reset type: SYSRSn
8	COUNT_EVENT_CTRL_2	R/W	0h	Controls the actions on an EVENT for UNIT2. 0 = No add or shift, but load the predefined value 1 = Based on other bits, add/shift with the predefined value Reset type: SYSRSn

Table 32-37. CLB_MISC_CONTROL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
7	COUNT_DIR_2	R/W	0h	Controls add/shift direction for UNIT 2 0 = right shift or subtract 1 = left shift or add Reset type: SYSRSn
6	COUNT_ADD_SHIFT_2	R/W	0h	Controls whether the UNIT 2 counter will do an ADD or a SHIFT on an EVENT. 0 =Shift 1 = ADD Reset type: SYSRSn
5	COUNT_EVENT_CTRL_1	R/W	0h	Controls the actions on an EVENT for UNIT1. 0 = No add or shift, but load the predefined value 1 = Based on other bits, add/shift with the predefined value Reset type: SYSRSn
4	COUNT_DIR_1	R/W	0h	Controls add/shift direction for UNIT 1 0 = right shift or subtract 1 = left shift or add Reset type: SYSRSn
3	COUNT_ADD_SHIFT_1	R/W	0h	Controls whether the UNIT 1 counter will do an ADD or a SHIFT on an EVENT. 0 = Shift 1 = ADD Reset type: SYSRSn
2	COUNT_EVENT_CTRL_0	R/W	0h	Controls the actions on an EVENT for UNIT1. 0 = No add or shift, but load the predefined value 1 = Based on other bits, add/shift with the predefined value Reset type: SYSRSn
1	COUNT_DIR_0	R/W	0h	Controls add/shift direction for UNIT 0 0 = right shift or subtract 1 = left shift or add Reset type: SYSRSn
0	COUNT_ADD_SHIFT_0	R/W	0h	Controls whether the UNIT 0 counter will do an ADD or a SHIFT on an EVENT. 0 = Shift 1 = ADD Reset type: SYSRSn

32.8.2.21 CLB_OUTPUT_LUT_0 Register (Offset = 2Ch) [Reset = 0h]

CLB_OUTPUT_LUT_0 is shown in Figure 32-38 and described in Table 32-38.

Return to the Summary Table.

Inp Sel, LUT fns for Out0

Figure 32-38. CLB_OUTPUT_LUT_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED									FN								IN2					IN1					IN0				
R-0h									R/W-0h								R/W-0h					R/W-0h					R/W-0h				

Table 32-38. CLB_OUTPUT_LUT_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R	0h	Reserved
22-15	FN	R/W	0h	Output function for output LUT Reset type: SYSRSn
14-10	IN2	R/W	0h	Select value for IN2 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	IN1	R/W	0h	Select value for IN1 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	IN0	R/W	0h	Select value for IN0 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

32.8.2.22 CLB_OUTPUT_LUT_1 Register (Offset = 2Eh) [Reset = 0h]

CLB_OUTPUT_LUT_1 is shown in Figure 32-39 and described in Table 32-39.

Return to the Summary Table.

Inp Sel, LUT fns for Out1

Figure 32-39. CLB_OUTPUT_LUT_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED									FN								IN2					IN1					IN0				
R-0h									R/W-0h								R/W-0h					R/W-0h					R/W-0h				

Table 32-39. CLB_OUTPUT_LUT_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R	0h	Reserved
22-15	FN	R/W	0h	Output function for output LUT Reset type: SYSRSn
14-10	IN2	R/W	0h	Select value for IN2 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	IN1	R/W	0h	Select value for IN1 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	IN0	R/W	0h	Select value for IN0 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

32.8.2.23 CLB_OUTPUT_LUT_2 Register (Offset = 30h) [Reset = 0h]

CLB_OUTPUT_LUT_2 is shown in Figure 32-40 and described in Table 32-40.

Return to the Summary Table.

Inp Sel, LUT fns for Out2

Figure 32-40. CLB_OUTPUT_LUT_2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED									FN								IN2					IN1					IN0				
R-0h									R/W-0h								R/W-0h					R/W-0h					R/W-0h				

Table 32-40. CLB_OUTPUT_LUT_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R	0h	Reserved
22-15	FN	R/W	0h	Output function for output LUT Reset type: SYSRSn
14-10	IN2	R/W	0h	Select value for IN2 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	IN1	R/W	0h	Select value for IN1 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	IN0	R/W	0h	Select value for IN0 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

32.8.2.24 CLB_OUTPUT_LUT_3 Register (Offset = 32h) [Reset = 0h]

CLB_OUTPUT_LUT_3 is shown in Figure 32-41 and described in Table 32-41.

Return to the Summary Table.

Inp Sel, LUT fns for Out3

Figure 32-41. CLB_OUTPUT_LUT_3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED									FN								IN2					IN1					IN0				
R-0h									R/W-0h								R/W-0h					R/W-0h					R/W-0h				

Table 32-41. CLB_OUTPUT_LUT_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R	0h	Reserved
22-15	FN	R/W	0h	Output function for output LUT Reset type: SYSRSn
14-10	IN2	R/W	0h	Select value for IN2 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	IN1	R/W	0h	Select value for IN1 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	IN0	R/W	0h	Select value for IN0 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

32.8.2.25 CLB_OUTPUT_LUT_4 Register (Offset = 34h) [Reset = 0h]

CLB_OUTPUT_LUT_4 is shown in Figure 32-42 and described in Table 32-42.

Return to the Summary Table.

Inp Sel, LUT fns for Out4

Figure 32-42. CLB_OUTPUT_LUT_4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED									FN								IN2					IN1					IN0				
R-0h									R/W-0h								R/W-0h					R/W-0h					R/W-0h				

Table 32-42. CLB_OUTPUT_LUT_4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R	0h	Reserved
22-15	FN	R/W	0h	Output function for output LUT Reset type: SYSRSn
14-10	IN2	R/W	0h	Select value for IN2 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	IN1	R/W	0h	Select value for IN1 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	IN0	R/W	0h	Select value for IN0 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

32.8.2.26 CLB_OUTPUT_LUT_5 Register (Offset = 36h) [Reset = 0h]

CLB_OUTPUT_LUT_5 is shown in Figure 32-43 and described in Table 32-43.

Return to the Summary Table.

Inp Sel, LUT fns for Out5

Figure 32-43. CLB_OUTPUT_LUT_5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED									FN								IN2					IN1					IN0				
R-0h									R/W-0h								R/W-0h					R/W-0h					R/W-0h				

Table 32-43. CLB_OUTPUT_LUT_5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R	0h	Reserved
22-15	FN	R/W	0h	Output function for output LUT Reset type: SYSRSn
14-10	IN2	R/W	0h	Select value for IN2 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	IN1	R/W	0h	Select value for IN1 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	IN0	R/W	0h	Select value for IN0 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

32.8.2.27 CLB_OUTPUT_LUT_6 Register (Offset = 38h) [Reset = 0h]

CLB_OUTPUT_LUT_6 is shown in Figure 32-44 and described in Table 32-44.

Return to the Summary Table.

Inp Sel, LUT fns for Out6

Figure 32-44. CLB_OUTPUT_LUT_6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED									FN								IN2					IN1					IN0				
R-0h									R/W-0h								R/W-0h					R/W-0h					R/W-0h				

Table 32-44. CLB_OUTPUT_LUT_6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R	0h	Reserved
22-15	FN	R/W	0h	Output function for output LUT Reset type: SYSRSn
14-10	IN2	R/W	0h	Select value for IN2 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	IN1	R/W	0h	Select value for IN1 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	IN0	R/W	0h	Select value for IN0 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

32.8.2.28 CLB_OUTPUT_LUT_7 Register (Offset = 3Ah) [Reset = 0h]

CLB_OUTPUT_LUT_7 is shown in Figure 32-45 and described in Table 32-45.

Return to the Summary Table.

Inp Sel, LUT fns for Out7

Figure 32-45. CLB_OUTPUT_LUT_7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED									FN								IN2					IN1					IN0				
R-0h									R/W-0h								R/W-0h					R/W-0h					R/W-0h				

Table 32-45. CLB_OUTPUT_LUT_7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-23	RESERVED	R	0h	Reserved
22-15	FN	R/W	0h	Output function for output LUT Reset type: SYSRSn
14-10	IN2	R/W	0h	Select value for IN2 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	IN1	R/W	0h	Select value for IN1 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	IN0	R/W	0h	Select value for IN0 of output LUT. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

32.8.2.29 CLB_HLC_EVENT_SEL Register (Offset = 3Ch) [Reset = 0h]

CLB_HLC_EVENT_SEL is shown in Figure 32-46 and described in Table 32-46.

Return to the Summary Table.

Event Selector register for the High Level controller

Figure 32-46. CLB_HLC_EVENT_SEL Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
ALT_EVENT3_SEL	ALT_EVENT2_SEL	ALT_EVENT1_SEL	ALT_EVENT0_SEL	EVENT3_SEL			
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h			
15	14	13	12	11	10	9	8
EVENT3_SEL	EVENT2_SEL					EVENT1_SEL	
R/W-0h	R/W-0h					R/W-0h	
7	6	5	4	3	2	1	0
EVENT1_SEL			EVENT0_SEL				
R/W-0h			R/W-0h				

Table 32-46. CLB_HLC_EVENT_SEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23	ALT_EVENT3_SEL	R/W	0h	Defines selection of alternate inputs for EVENT3 Reset type: SYSRSn
22	ALT_EVENT2_SEL	R/W	0h	Defines selection of alternate inputs for EVENT2 Reset type: SYSRSn
21	ALT_EVENT1_SEL	R/W	0h	Defines selection of alternate inputs for EVENT1 Reset type: SYSRSn
20	ALT_EVENT0_SEL	R/W	0h	Defines selection of alternate inputs for EVENT0 Reset type: SYSRSn
19-15	EVENT3_SEL	R/W	0h	5 bit select value for EVENT3 of the High Level Controller. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
14-10	EVENT2_SEL	R/W	0h	5 bit select value for EVENT2 of the High Level Controller. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
9-5	EVENT1_SEL	R/W	0h	5 bit select value for EVENT1 of the High Level Controller. See the Static Switch Block Output Mux Table. Reset type: SYSRSn
4-0	EVENT0_SEL	R/W	0h	5 bit select value for EVENT0 of the High Level Controller. See the Static Switch Block Output Mux Table. Reset type: SYSRSn

32.8.2.30 CLB_COUNT_MATCH_TAP_SEL Register (Offset = 3Eh) [Reset = 0h]

CLB_COUNT_MATCH_TAP_SEL is shown in Figure 32-47 and described in Table 32-47.

Return to the Summary Table.

Counter tap values for match1 and match2 outputs

Figure 32-47. CLB_COUNT_MATCH_TAP_SEL Register

31	30	29	28	27	26	25	24
RESERVED	COUNT2_MATCH2					COUNT1_MATCH2	
R-0h	R/W-0h					R/W-0h	
23	22	21	20	19	18	17	16
COUNT1_MATCH2			COUNT0_MATCH2				
R/W-0h			R/W-0h				
15	14	13	12	11	10	9	8
RESERVED	COUNT2_MATCH1					COUNT1_MATCH1	
R-0h	R/W-0h					R/W-0h	
7	6	5	4	3	2	1	0
COUNT1_MATCH1			COUNT0_MATCH1				
R/W-0h			R/W-0h				

Table 32-47. CLB_COUNT_MATCH_TAP_SEL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R	0h	Reserved
30-26	COUNT2_MATCH2	R/W	0h	5 bit MUX Select for Match2 Tap for Counter Unit 2 Reset type: SYSRSn
25-21	COUNT1_MATCH2	R/W	0h	5 bit MUX Select for Match2 Tap for Counter Unit 1 Reset type: SYSRSn
20-16	COUNT0_MATCH2	R/W	0h	5 bit MUX Select for Match2 Tap for Counter Unit 0 Reset type: SYSRSn
15	RESERVED	R	0h	Reserved
14-10	COUNT2_MATCH1	R/W	0h	5 bit MUX Select for Match1 Tap for Counter Unit 2 Reset type: SYSRSn
9-5	COUNT1_MATCH1	R/W	0h	5 bit MUX Select for Match1 Tap for Counter Unit 1 Reset type: SYSRSn
4-0	COUNT0_MATCH1	R/W	0h	5 bit MUX Select for Match1 Tap for Counter Unit 0 Reset type: SYSRSn

32.8.2.31 CLB_OUTPUT_COND_CTRL_0 Register (Offset = 40h) [Reset = 0h]

CLB_OUTPUT_COND_CTRL_0 is shown in Figure 32-48 and described in Table 32-48.

Return to the Summary Table.

Output conditioning control for output 0

Figure 32-48. CLB_OUTPUT_COND_CTRL_0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	ASYNC_COND_EN	SEL_RAW_IN	HW_RLS_CTRL_SEL	HW_GATING_CTRL_SEL	SEL_RELEASE_CTRL		
R-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h		
7	6	5	4	3	2	1	0
SEL_GATING_CTRL			LEVEL_3_SEL		LEVEL_2_SEL		LEVEL_1_SEL
R/W1C-0h			R/W1C-0h		R/W1C-0h		R/W1C-0h

Table 32-48. CLB_OUTPUT_COND_CTRL_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Reserved
14	ASYNC_COND_EN	R/W1C	0h	Controls whether the output will pass through the asynchronous conditioning block or bypass it. 0 Bypass the asynchronous conditioning block 1 Enable the asynchronous conditioning path Reset type: SYSRSn
13	SEL_RAW_IN	R/W1C	0h	Controls whether the CELL outputs or inputs are sent to the output conditioning block logic. 0 = CELL output (internally delayed by 1 cycle) is used. 1 = CELL input (raw) is used. Reset type: SYSRSn
12	HW_RLS_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the release control 0 SW register value will act as release control 1 Selected CELL output will act as release control Reset type: SYSRSn
11	HW_GATING_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the gating control 0 SW register value will act as gating control 1 Selected CELL output will act as gating control Reset type: SYSRSn
10-8	SEL_RELEASE_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Release control. Reset type: SYSRSn
7-5	SEL_GATING_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Gating control. Reset type: SYSRSn

Table 32-48. CLB_OUTPUT_COND_CTRL_0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-3	LEVEL_3_SEL	R/W1C	0h	Controls Third level Mux select 00 Input Signal will be sent as is to the output 01 Rising edge of Input signal will cause asynchronous CLEAR of the output 10 Rising edge of Input signal will cause asynchronous SET of the output 11 Input Signal delayed by 1 clock cycle will be sent to the output Reset type: SYSRSn
2-1	LEVEL_2_SEL	R/W1C	0h	Controls Second level Mux select 00 Input Signal sent as output to next level 01 Input Signal AND Gating_control sent as output to next level 10 Input Signal OR Gating_control sent as output to next level 11 Input Signal XOR Gating_control sent as output to next level Reset type: SYSRSn
0	LEVEL_1_SEL	R/W1C	0h	First level MUX select value 0 Direct signal sent as output to next level 1 Inverted signal sent as output to the next level Reset type: SYSRSn

32.8.2.32 CLB_OUTPUT_COND_CTRL_1 Register (Offset = 42h) [Reset = 0h]

CLB_OUTPUT_COND_CTRL_1 is shown in Figure 32-49 and described in Table 32-49.

Return to the Summary Table.

Output conditioning control for output 1

Figure 32-49. CLB_OUTPUT_COND_CTRL_1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	ASYNC_COND_EN	SEL_RAW_IN	HW_RLS_CTRL_SEL	HW_GATING_CTRL_SEL	SEL_RELEASE_CTRL		
R-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h		
7	6	5	4	3	2	1	0
SEL_GATING_CTRL			LEVEL_3_SEL		LEVEL_2_SEL		LEVEL_1_SEL
R/W1C-0h			R/W1C-0h		R/W1C-0h		R/W1C-0h

Table 32-49. CLB_OUTPUT_COND_CTRL_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Reserved
14	ASYNC_COND_EN	R/W1C	0h	Controls whether the output will pass through the asynchronous conditioning block or bypass it. 0 Bypass the asynchronous conditioning block 1 Enable the asynchronous conditioning path Reset type: SYSRSn
13	SEL_RAW_IN	R/W1C	0h	Controls whether the CELL outputs or inputs are sent to the output conditioning block logic. 0 = CELL output (internally delayed by 1 cycle) is used. 1 = CELL input (raw) is used. Reset type: SYSRSn
12	HW_RLS_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the release control 0 SW register value will act as release control 1 Selected CELL output will act as release control Reset type: SYSRSn
11	HW_GATING_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the gating control 0 SW register value will act as gating control 1 Selected CELL output will act as gating control Reset type: SYSRSn
10-8	SEL_RELEASE_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Release control. Reset type: SYSRSn
7-5	SEL_GATING_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Gating control. Reset type: SYSRSn

Table 32-49. CLB_OUTPUT_COND_CTRL_1 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-3	LEVEL_3_SEL	R/W1C	0h	Controls Third level Mux select 00 Input Signal will be sent as is to the output 01 Rising edge of Input signal will cause asynchronous CLEAR of the output 10 Rising edge of Input signal will cause asynchronous SET of the output 11 Input Signal delayed by 1 clock cycle will be sent to the output Reset type: SYSRSn
2-1	LEVEL_2_SEL	R/W1C	0h	Controls Second level Mux select 00 Input Signal sent as output to next level 01 Input Signal AND Gating_control sent as output to next level 10 Input Signal OR Gating_control sent as output to next level 11 Input Signal XOR Gating_control sent as output to next level Reset type: SYSRSn
0	LEVEL_1_SEL	R/W1C	0h	First level MUX select value 0 Direct signal sent as output to next level 1 Inverted signal sent as output to the next level Reset type: SYSRSn

32.8.2.33 CLB_OUTPUT_COND_CTRL_2 Register (Offset = 44h) [Reset = 0h]

CLB_OUTPUT_COND_CTRL_2 is shown in Figure 32-50 and described in Table 32-50.

Return to the Summary Table.

Output conditioning control for output 2

Figure 32-50. CLB_OUTPUT_COND_CTRL_2 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	ASYNC_COND_EN	SEL_RAW_IN	HW_RLS_CTRL_SEL	HW_GATING_CTRL_SEL	SEL_RELEASE_CTRL		
R-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h		
7	6	5	4	3	2	1	0
SEL_GATING_CTRL			LEVEL_3_SEL		LEVEL_2_SEL		LEVEL_1_SEL
R/W1C-0h			R/W1C-0h		R/W1C-0h		R/W1C-0h

Table 32-50. CLB_OUTPUT_COND_CTRL_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Reserved
14	ASYNC_COND_EN	R/W1C	0h	Controls whether the output will pass through the asynchronous conditioning block or bypass it. 0 Bypass the asynchronous conditioning block 1 Enable the asynchronous conditioning path Reset type: SYSRSn
13	SEL_RAW_IN	R/W1C	0h	Controls whether the CELL outputs or inputs are sent to the output conditioning block logic. 0 = CELL output (internally delayed by 1 cycle) is used. 1 = CELL input (raw) is used. Reset type: SYSRSn
12	HW_RLS_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the release control 0 SW register value will act as release control 1 Selected CELL output will act as release control Reset type: SYSRSn
11	HW_GATING_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the gating control 0 SW register value will act as gating control 1 Selected CELL output will act as gating control Reset type: SYSRSn
10-8	SEL_RELEASE_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Release control. Reset type: SYSRSn
7-5	SEL_GATING_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Gating control. Reset type: SYSRSn

Table 32-50. CLB_OUTPUT_COND_CTRL_2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-3	LEVEL_3_SEL	R/W1C	0h	Controls Third level Mux select 00 Input Signal will be sent as is to the output 01 Rising edge of Input signal will cause asynchronous CLEAR of the output 10 Rising edge of Input signal will cause asynchronous SET of the output 11 Input Signal delayed by 1 clock cycle will be sent to the output Reset type: SYSRSn
2-1	LEVEL_2_SEL	R/W1C	0h	Controls Second level Mux select 00 Input Signal sent as output to next level 01 Input Signal AND Gating_control sent as output to next level 10 Input Signal OR Gating_control sent as output to next level 11 Input Signal XOR Gating_control sent as output to next level Reset type: SYSRSn
0	LEVEL_1_SEL	R/W1C	0h	First level MUX select value 0 Direct signal sent as output to next level 1 Inverted signal sent as output to the next level Reset type: SYSRSn

32.8.2.34 CLB_OUTPUT_COND_CTRL_3 Register (Offset = 46h) [Reset = 0h]

CLB_OUTPUT_COND_CTRL_3 is shown in Figure 32-51 and described in Table 32-51.

Return to the Summary Table.

Output conditioning control for output 3

Figure 32-51. CLB_OUTPUT_COND_CTRL_3 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	ASYNC_COND_EN	SEL_RAW_IN	HW_RLS_CTRL_SEL	HW_GATING_CTRL_SEL	SEL_RELEASE_CTRL		
R-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h		
7	6	5	4	3	2	1	0
SEL_GATING_CTRL			LEVEL_3_SEL		LEVEL_2_SEL		LEVEL_1_SEL
R/W1C-0h			R/W1C-0h		R/W1C-0h		R/W1C-0h

Table 32-51. CLB_OUTPUT_COND_CTRL_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Reserved
14	ASYNC_COND_EN	R/W1C	0h	Controls whether the output will pass through the asynchronous conditioning block or bypass it. 0 Bypass the asynchronous conditioning block 1 Enable the asynchronous conditioning path Reset type: SYSRSn
13	SEL_RAW_IN	R/W1C	0h	Controls whether the CELL outputs or inputs are sent to the output conditioning block logic. 0 = CELL output (internally delayed by 1 cycle) is used. 1 = CELL input (raw) is used. Reset type: SYSRSn
12	HW_RLS_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the release control 0 SW register value will act as release control 1 Selected CELL output will act as release control Reset type: SYSRSn
11	HW_GATING_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the gating control 0 SW register value will act as gating control 1 Selected CELL output will act as gating control Reset type: SYSRSn
10-8	SEL_RELEASE_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Release control. Reset type: SYSRSn
7-5	SEL_GATING_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Gating control. Reset type: SYSRSn

Table 32-51. CLB_OUTPUT_COND_CTRL_3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-3	LEVEL_3_SEL	R/W1C	0h	Controls Third level Mux select 00 Input Signal will be sent as is to the output 01 Rising edge of Input signal will cause asynchronous CLEAR of the output 10 Rising edge of Input signal will cause asynchronous SET of the output 11 Input Signal delayed by 1 clock cycle will be sent to the output Reset type: SYSRSn
2-1	LEVEL_2_SEL	R/W1C	0h	Controls Second level Mux select 00 Input Signal sent as output to next level 01 Input Signal AND Gating_control sent as output to next level 10 Input Signal OR Gating_control sent as output to next level 11 Input Signal XOR Gating_control sent as output to next level Reset type: SYSRSn
0	LEVEL_1_SEL	R/W1C	0h	First level MUX select value 0 Direct signal sent as output to next level 1 Inverted signal sent as output to the next level Reset type: SYSRSn

32.8.2.35 CLB_OUTPUT_COND_CTRL_4 Register (Offset = 48h) [Reset = 0h]

CLB_OUTPUT_COND_CTRL_4 is shown in Figure 32-52 and described in Table 32-52.

Return to the Summary Table.

Output conditioning control for output 4

Figure 32-52. CLB_OUTPUT_COND_CTRL_4 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	ASYNC_COND_EN	SEL_RAW_IN	HW_RLS_CTRL_SEL	HW_GATING_CTRL_SEL	SEL_RELEASE_CTRL		
R-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h		
7	6	5	4	3	2	1	0
SEL_GATING_CTRL			LEVEL_3_SEL		LEVEL_2_SEL		LEVEL_1_SEL
R/W1C-0h			R/W1C-0h		R/W1C-0h		R/W1C-0h

Table 32-52. CLB_OUTPUT_COND_CTRL_4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Reserved
14	ASYNC_COND_EN	R/W1C	0h	Controls whether the output will pass through the asynchronous conditioning block or bypass it. 0 Bypass the asynchronous conditioning block 1 Enable the asynchronous conditioning path Reset type: SYSRSn
13	SEL_RAW_IN	R/W1C	0h	Controls whether the CELL outputs or inputs are sent to the output conditioning block logic. 0 = CELL output (internally delayed by 1 cycle) is used. 1 = CELL input (raw) is used. Reset type: SYSRSn
12	HW_RLS_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the release control 0 SW register value will act as release control 1 Selected CELL output will act as release control Reset type: SYSRSn
11	HW_GATING_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the gating control 0 SW register value will act as gating control 1 Selected CELL output will act as gating control Reset type: SYSRSn
10-8	SEL_RELEASE_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Release control. Reset type: SYSRSn
7-5	SEL_GATING_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Gating control. Reset type: SYSRSn

Table 32-52. CLB_OUTPUT_COND_CTRL_4 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-3	LEVEL_3_SEL	R/W1C	0h	Controls Third level Mux select 00 Input Signal will be sent as is to the output 01 Rising edge of Input signal will cause asynchronous CLEAR of the output 10 Rising edge of Input signal will cause asynchronous SET of the output 11 Input Signal delayed by 1 clock cycle will be sent to the output Reset type: SYSRSn
2-1	LEVEL_2_SEL	R/W1C	0h	Controls Second level Mux select 00 Input Signal sent as output to next level 01 Input Signal AND Gating_control sent as output to next level 10 Input Signal OR Gating_control sent as output to next level 11 Input Signal XOR Gating_control sent as output to next level Reset type: SYSRSn
0	LEVEL_1_SEL	R/W1C	0h	First level MUX select value 0 Direct signal sent as output to next level 1 Inverted signal sent as output to the next level Reset type: SYSRSn

32.8.2.36 CLB_OUTPUT_COND_CTRL_5 Register (Offset = 4Ah) [Reset = 0h]

CLB_OUTPUT_COND_CTRL_5 is shown in Figure 32-53 and described in Table 32-53.

Return to the Summary Table.

Output conditioning control for output 5

Figure 32-53. CLB_OUTPUT_COND_CTRL_5 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	ASYNC_COND_EN	SEL_RAW_IN	HW_RLS_CTRL_SEL	HW_GATING_CTRL_SEL	SEL_RELEASE_CTRL		
R-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h		
7	6	5	4	3	2	1	0
SEL_GATING_CTRL			LEVEL_3_SEL		LEVEL_2_SEL		LEVEL_1_SEL
R/W1C-0h			R/W1C-0h		R/W1C-0h		R/W1C-0h

Table 32-53. CLB_OUTPUT_COND_CTRL_5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Reserved
14	ASYNC_COND_EN	R/W1C	0h	Controls whether the output will pass through the asynchronous conditioning block or bypass it. 0 Bypass the asynchronous conditioning block 1 Enable the asynchronous conditioning path Reset type: SYSRSn
13	SEL_RAW_IN	R/W1C	0h	Controls whether the CELL outputs or inputs are sent to the output conditioning block logic. 0 = CELL output (internally delayed by 1 cycle) is used. 1 = CELL input (raw) is used. Reset type: SYSRSn
12	HW_RLS_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the release control 0 SW register value will act as release control 1 Selected CELL output will act as release control Reset type: SYSRSn
11	HW_GATING_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the gating control 0 SW register value will act as gating control 1 Selected CELL output will act as gating control Reset type: SYSRSn
10-8	SEL_RELEASE_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Release control. Reset type: SYSRSn
7-5	SEL_GATING_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Gating control. Reset type: SYSRSn

Table 32-53. CLB_OUTPUT_COND_CTRL_5 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-3	LEVEL_3_SEL	R/W1C	0h	Controls Third level Mux select 00 Input Signal will be sent as is to the output 01 Rising edge of Input signal will cause asynchronous CLEAR of the output 10 Rising edge of Input signal will cause asynchronous SET of the output 11 Input Signal delayed by 1 clock cycle will be sent to the output Reset type: SYSRSn
2-1	LEVEL_2_SEL	R/W1C	0h	Controls Second level Mux select 00 Input Signal sent as output to next level 01 Input Signal AND Gating_control sent as output to next level 10 Input Signal OR Gating_control sent as output to next level 11 Input Signal XOR Gating_control sent as output to next level Reset type: SYSRSn
0	LEVEL_1_SEL	R/W1C	0h	First level MUX select value 0 Direct signal sent as output to next level 1 Inverted signal sent as output to the next level Reset type: SYSRSn

32.8.2.37 CLB_OUTPUT_COND_CTRL_6 Register (Offset = 4Ch) [Reset = 0h]

CLB_OUTPUT_COND_CTRL_6 is shown in Figure 32-54 and described in Table 32-54.

Return to the Summary Table.

Output conditioning control for output 6

Figure 32-54. CLB_OUTPUT_COND_CTRL_6 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	ASYNC_COND_EN	SEL_RAW_IN	HW_RLS_CTRL_SEL	HW_GATING_CTRL_SEL	SEL_RELEASE_CTRL		
R-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h		
7	6	5	4	3	2	1	0
SEL_GATING_CTRL			LEVEL_3_SEL		LEVEL_2_SEL		LEVEL_1_SEL
R/W1C-0h			R/W1C-0h		R/W1C-0h		R/W1C-0h

Table 32-54. CLB_OUTPUT_COND_CTRL_6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Reserved
14	ASYNC_COND_EN	R/W1C	0h	Controls whether the output will pass through the asynchronous conditioning block or bypass it. 0 Bypass the asynchronous conditioning block 1 Enable the asynchronous conditioning path Reset type: SYSRSn
13	SEL_RAW_IN	R/W1C	0h	Controls whether the CELL outputs or inputs are sent to the output conditioning block logic. 0 = CELL output (internally delayed by 1 cycle) is used. 1 = CELL input (raw) is used. Reset type: SYSRSn
12	HW_RLS_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the release control 0 SW register value will act as release control 1 Selected CELL output will act as release control Reset type: SYSRSn
11	HW_GATING_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the gating control 0 SW register value will act as gating control 1 Selected CELL output will act as gating control Reset type: SYSRSn
10-8	SEL_RELEASE_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Release control. Reset type: SYSRSn
7-5	SEL_GATING_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Gating control. Reset type: SYSRSn

Table 32-54. CLB_OUTPUT_COND_CTRL_6 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-3	LEVEL_3_SEL	R/W1C	0h	Controls Third level Mux select 00 Input Signal will be sent as is to the output 01 Rising edge of Input signal will cause asynchronous CLEAR of the output 10 Rising edge of Input signal will cause asynchronous SET of the output 11 Input Signal delayed by 1 clock cycle will be sent to the output Reset type: SYSRSn
2-1	LEVEL_2_SEL	R/W1C	0h	Controls Second level Mux select 00 Input Signal sent as output to next level 01 Input Signal AND Gating_control sent as output to next level 10 Input Signal OR Gating_control sent as output to next level 11 Input Signal XOR Gating_control sent as output to next level Reset type: SYSRSn
0	LEVEL_1_SEL	R/W1C	0h	First level MUX select value 0 Direct signal sent as output to next level 1 Inverted signal sent as output to the next level Reset type: SYSRSn

32.8.2.38 CLB_OUTPUT_COND_CTRL_7 Register (Offset = 4Eh) [Reset = 0h]

CLB_OUTPUT_COND_CTRL_7 is shown in Figure 32-55 and described in Table 32-55.

Return to the Summary Table.

Output conditioning control for output 7

Figure 32-55. CLB_OUTPUT_COND_CTRL_7 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED	ASYNC_COND_EN	SEL_RAW_IN	HW_RLS_CTRL_SEL	HW_GATING_CTRL_SEL	SEL_RELEASE_CTRL		
R-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h	R/W1C-0h		
7	6	5	4	3	2	1	0
SEL_GATING_CTRL			LEVEL_3_SEL		LEVEL_2_SEL		LEVEL_1_SEL
R/W1C-0h			R/W1C-0h		R/W1C-0h		R/W1C-0h

Table 32-55. CLB_OUTPUT_COND_CTRL_7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-15	RESERVED	R	0h	Reserved
14	ASYNC_COND_EN	R/W1C	0h	Controls whether the output will pass through the asynchronous conditioning block or bypass it. 0 Bypass the asynchronous conditioning block 1 Enable the asynchronous conditioning path Reset type: SYSRSn
13	SEL_RAW_IN	R/W1C	0h	Controls whether the CELL outputs or inputs are sent to the output conditioning block logic. 0 = CELL output (internally delayed by 1 cycle) is used. 1 = CELL input (raw) is used. Reset type: SYSRSn
12	HW_RLS_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the release control 0 SW register value will act as release control 1 Selected CELL output will act as release control Reset type: SYSRSn
11	HW_GATING_CTRL_SEL	R/W1C	0h	Controls whether the HW (CELL outputs) or software (GP_REG) will act as the gating control 0 SW register value will act as gating control 1 Selected CELL output will act as gating control Reset type: SYSRSn
10-8	SEL_RELEASE_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Release control. Reset type: SYSRSn
7-5	SEL_GATING_CTRL	R/W1C	0h	3 bit MUX selects which will select one of the 8 CELL outputs for Gating control. Reset type: SYSRSn

Table 32-55. CLB_OUTPUT_COND_CTRL_7 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
4-3	LEVEL_3_SEL	R/W1C	0h	Controls Third level Mux select 00 Input Signal will be sent as is to the output 01 Rising edge of Input signal will cause asynchronous CLEAR of the output 10 Rising edge of Input signal will cause asynchronous SET of the output 11 Input Signal delayed by 1 clock cycle will be sent to the output Reset type: SYSRSn
2-1	LEVEL_2_SEL	R/W1C	0h	Controls Second level Mux select 00 Input Signal sent as output to next level 01 Input Signal AND Gating_control sent as output to next level 10 Input Signal OR Gating_control sent as output to next level 11 Input Signal XOR Gating_control sent as output to next level Reset type: SYSRSn
0	LEVEL_1_SEL	R/W1C	0h	First level MUX select value 0 Direct signal sent as output to next level 1 Inverted signal sent as output to the next level Reset type: SYSRSn

32.8.2.39 CLB_MISC_ACCESS_CTRL Register (Offset = 50h) [Reset = 0h]

CLB_MISC_ACCESS_CTRL is shown in Figure 32-56 and described in Table 32-56.

Return to the Summary Table.

Miscellaneous Access and enable control

Figure 32-56. CLB_MISC_ACCESS_CTRL Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						BLKEN	SPIEN
R-0h						R/W1C-0h	R/W1C-0h

Table 32-56. CLB_MISC_ACCESS_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
15-2	RESERVED	R	0h	Reserved
1	BLKEN	R/W1C	0h	This bit is used to block writes to CLB_OUT_EN 0 Writes to CLB_OUT_EN are allowed 1 Writes to CLB_OUT_EN are blocked Reset type: SYSRSn
0	SPIEN	R/W1C	0h	This bit indicates the status of the SPI buffers ability to export CLB output data. 0 Feature Disabled 1 Feature Enabled Reset type: SYSRSn

32.8.2.40 CLB_SPI_DATA_CTRL_HI Register (Offset = 51h) [Reset = 0h]

CLB_SPI_DATA_CTRL_HI is shown in Figure 32-57 and described in Table 32-57.

Return to the Summary Table.

CLB to SPI buffer control High

Figure 32-57. CLB_SPI_DATA_CTRL_HI Register

15	14	13	12	11	10	9	8
RESERVED			SHIFT				
R-0h			R/W1C-0h				
7	6	5	4	3	2	1	0
RESERVED			STRB				
R-0h			R/W1C-0h				

Table 32-57. CLB_SPI_DATA_CTRL_HI Register Field Descriptions

Bit	Field	Type	Reset	Description
15-13	RESERVED	R	0h	Reserved
12-8	SHIFT	R/W1C	0h	This is a 5 bit value which denotes the first bit position of register R0 from which the Least Significant Bit of the output data should start. 00000 Output data is R0[15:0] 00001 Output data is R0[16:1] 00010 Output data is R0[17:2] 00011 Output data is R0[18:3] ... ... 10000 Output data is R0[31:16] Reset type: SYSRSn
7-5	RESERVED	R	0h	Reserved
4-0	STRB	R/W1C	0h	This is a 5 bit value which selects one of the HLC_EVENT inputs to be treated as the data_valid strobe Reset type: SYSRSn

32.8.3 CLB_LOGIC_CONTROL_REGS Registers

Table 32-58 lists the memory-mapped registers for the CLB_LOGIC_CONTROL_REGS registers. All register offset addresses not listed in Table 32-58 should be considered as reserved locations and the register contents should not be modified.

Table 32-58. CLB_LOGIC_CONTROL_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	CLB_LOAD_EN	Global enable & indirect load enable control	EALLOW, LOCK (only Global Enable Bit is LOCK protected)	Go
2h	CLB_LOAD_ADDR	Indirect address		Go
4h	CLB_LOAD_DATA	Data for indirect loads		Go
6h	CLB_INPUT_FILTER	Input filter selection for both edge detection and synchronizers	LOCK	Go
8h	CLB_IN_MUX_SEL_0	Input selection to decide between Signals and GP register	LOCK	Go
Ah	CLB_LCL_MUX_SEL_1	Input Mux selection for local mux	LOCK	Go
Ch	CLB_LCL_MUX_SEL_2	Input Mux selection for local mux	LOCK	Go
Eh	CLB_BUF_PTR	PUSH and PULL pointers		Go
10h	CLB_GP_REG	General purpose register for CELL inputs		Go
12h	CLB_OUT_EN	CELL output enable register		Go
14h	CLB_GLBL_MUX_SEL_1	Global Mux select for CELL inputs	LOCK	Go
16h	CLB_GLBL_MUX_SEL_2	Global Mux select for CELL inputs	LOCK	Go
18h	CLB_PRESCALE_CTRL	Prescaler register control	LOCK	Go
20h	CLB_INTR_TAG_REG	Interrupt Tag register		Go
22h	CLB_LOCK	Lock control register	EALLOW	Go
24h	CLB_HLC_INSTR_READ_PTR	HLC instruction read pointer		Go
26h	CLB_HLC_INSTR_VALUE	HLC instruction read value		Go
2Eh	CLB_DBG_OUT_2	Visibility for CLB inputs and final asynchronous outputs		Go
30h	CLB_DBG_R0	R0 of High level Controller		Go
32h	CLB_DBG_R1	R1 of High level Controller		Go
34h	CLB_DBG_R2	R2 of High level Controller		Go
36h	CLB_DBG_R3	R3 of High level Controller		Go
38h	CLB_DBG_C0	Count of Unit 0		Go
3Ah	CLB_DBG_C1	Count of Unit 1		Go
3Ch	CLB_DBG_C2	Count of Unit 2		Go
3Eh	CLB_DBG_OUT	Outputs of various units in the Cell		Go

Complex bit access types are encoded to fit into small table cells. Table 32-59 shows the codes that are used for access types in this section.

Table 32-59. CLB_LOGIC_CONTROL_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s

Table 32-59. CLB_LOGIC_CONTROL_REGS Access Type Codes (continued)

Access Type	Code	Description
R-1	R -1	Read Returns 1s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

32.8.3.1 CLB_LOAD_EN Register (Offset = 0h) [Reset = 0h]

CLB_LOAD_EN is shown in Figure 32-58 and described in Table 32-60.

Return to the Summary Table.

Global enable & indirect load enable control

Figure 32-58. CLB_LOAD_EN Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			PIPELINE_EN	NMI_EN	STOP	GLOBAL_EN	LOAD_EN
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 32-60. CLB_LOAD_EN Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R	0h	Reserved
4	PIPELINE_EN	R/W	0h	This bit controls pipelining of all the CLB operations in HLC and Counter blocks. Pipelined operation is enabled when this bit is set to 1. Reset type: SYSRSn
3	NMI_EN	R/W	0h	This bit controls the generation of NMI along with the interrupt whenever a INTR operation is executed by the HLC. NMI generation is disabled by default. It will be enabled when this bit is set to 1. Reset type: SYSRSn
2	STOP	R/W	0h	This bit defines the behaviour of the sequential elements in the CELL during debug HALTs of the CPU. If this bit is set to 0, the debug HALT condition is ignored. Reset type: SYSRSn
1	GLOBAL_EN	R/W	0h	This bit is a global enable signal for the logic in the CELL. This also acts as a soft reset for the CELL logic. This bit is normally set after all the other configuration settings are completed. This bit is LOCK protected. Reset type: SYSRSn
0	LOAD_EN	R/W	0h	A write with this bit set to 1 will pulse the Load Enable signal for the indirect register loads in the CELL. Reset type: SYSRSn

32.8.3.2 CLB_LOAD_ADDR Register (Offset = 2h) [Reset = 0h]

CLB_LOAD_ADDR is shown in Figure 32-59 and described in Table 32-61.

Return to the Summary Table.

Indirect address

Figure 32-59. CLB_LOAD_ADDR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																										ADDR					
R-0h																										R/W-0h					

Table 32-61. CLB_LOAD_ADDR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-6	RESERVED	R	0h	Reserved
5-0	ADDR	R/W	0h	These are the address bits used for writing to the indirect address space of the CELL. Reset type: SYSRSn

32.8.3.3 CLB_LOAD_DATA Register (Offset = 4h) [Reset = 0h]

CLB_LOAD_DATA is shown in Figure 32-60 and described in Table 32-62.

Return to the Summary Table.

Data for indirect loads

Figure 32-60. CLB_LOAD_DATA Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R/W-0h																															

Table 32-62. CLB_LOAD_DATA Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R/W	0h	This register holds the 32-bit data for writing to the indirect address space of the CELL. Reset type: SYSRSn

32.8.3.4 CLB_INPUT_FILTER Register (Offset = 6h) [Reset = 0h]

CLB_INPUT_FILTER is shown in Figure 32-61 and described in Table 32-63.

Return to the Summary Table.

Input filter selection for both edge detection and synchronizers

Figure 32-61. CLB_INPUT_FILTER Register

31	30	29	28	27	26	25	24
PIPE7	PIPE6	PIPE5	PIPE4	PIPE3	PIPE2	PIPE1	PIPE0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
SYNC7	SYNC6	SYNC5	SYNC4	SYNC3	SYNC2	SYNC1	SYNC0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
FIN7		FIN6		FIN5		FIN4	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	
7	6	5	4	3	2	1	0
FIN3		FIN2		FIN1		FIN0	
R/W-0h		R/W-0h		R/W-0h		R/W-0h	

Table 32-63. CLB_INPUT_FILTER Register Field Descriptions

Bit	Field	Type	Reset	Description
31	PIPE7	R/W	0h	Enable pipelining for Input 7 Reset type: SYSRSn
30	PIPE6	R/W	0h	Enable pipelining for Input 6 Reset type: SYSRSn
29	PIPE5	R/W	0h	Enable pipelining for Input 5 Reset type: SYSRSn
28	PIPE4	R/W	0h	Enable pipelining for Input 4 Reset type: SYSRSn
27	PIPE3	R/W	0h	Enable pipelining for Input 3 Reset type: SYSRSn
26	PIPE2	R/W	0h	Enable pipelining for Input 2 Reset type: SYSRSn
25	PIPE1	R/W	0h	Enable pipelining for Input 1 Reset type: SYSRSn
24	PIPE0	R/W	0h	Enable pipelining for Input 0 Reset type: SYSRSn
23	SYNC7	R/W	0h	Synchronizer Select Control for Input 7 Reset type: SYSRSn
22	SYNC6	R/W	0h	Synchronizer Select Control for Input 6 Reset type: SYSRSn
21	SYNC5	R/W	0h	Synchronizer Select Control for Input 5 Reset type: SYSRSn
20	SYNC4	R/W	0h	Synchronizer Select Control for Input 4 Reset type: SYSRSn
19	SYNC3	R/W	0h	Synchronizer Select Control for Input 3 Reset type: SYSRSn
18	SYNC2	R/W	0h	Synchronizer Select Control for Input 2 Reset type: SYSRSn
17	SYNC1	R/W	0h	Synchronizer Select Control for Input 1 Reset type: SYSRSn

Table 32-63. CLB_INPUT_FILTER Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
16	SYNC0	R/W	0h	Synchronizer Select Control for Input 0 Reset type: SYSRSn
15-14	FIN7	R/W	0h	Input filter selection for CELL Input 7 2 bits are used to define the edge filtering . 00 : No filtering 01 : Rising edge detect 10 : Falling edge detect 11 : Any edge detect Reset type: SYSRSn
13-12	FIN6	R/W	0h	Input filter selection for CELL Input 6 2 bits are used to define the edge filtering . 00 : No filtering 01 : Rising edge detect 10 : Falling edge detect 11 : Any edge detect Reset type: SYSRSn
11-10	FIN5	R/W	0h	Input filter selection for CELL Input 5 2 bits are used to define the edge filtering . 00 : No filtering 01 : Rising edge detect 10 : Falling edge detect 11 : Any edge detect Reset type: SYSRSn
9-8	FIN4	R/W	0h	Input filter selection for CELL Input 4 2 bits are used to define the edge filtering . 00 : No filtering 01 : Rising edge detect 10 : Falling edge detect 11 : Any edge detect Reset type: SYSRSn
7-6	FIN3	R/W	0h	Input filter selection for CELL Input 3 2 bits are used to define the edge filtering . 00 : No filtering 01 : Rising edge detect 10 : Falling edge detect 11 : Any edge detect Reset type: SYSRSn
5-4	FIN2	R/W	0h	Input filter selection for CELL Input 2 2 bits are used to define the edge filtering . 00 : No filtering 01 : Rising edge detect 10 : Falling edge detect 11 : Any edge detect Reset type: SYSRSn
3-2	FIN1	R/W	0h	Input filter selection for CELL Input 1 2 bits are used to define the edge filtering . 00 : No filtering 01 : Rising edge detect 10 : Falling edge detect 11 : Any edge detect Reset type: SYSRSn
1-0	FIN0	R/W	0h	Input filter selection for CELL Input 0 2 bits are used to define the edge filtering . 00 : No filtering 01 : Rising edge detect 10 : Falling edge detect 11 : Any edge detect Reset type: SYSRSn

32.8.3.5 CLB_IN_MUX_SEL_0 Register (Offset = 8h) [Reset = 0h]

CLB_IN_MUX_SEL_0 is shown in Figure 32-62 and described in Table 32-64.

Return to the Summary Table.

Input selection to decide between Signals and GP register

Figure 32-62. CLB_IN_MUX_SEL_0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
SEL_GP_IN_7	SEL_GP_IN_6	SEL_GP_IN_5	SEL_GP_IN_4	SEL_GP_IN_3	SEL_GP_IN_2	SEL_GP_IN_1	SEL_GP_IN_0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 32-64. CLB_IN_MUX_SEL_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	SEL_GP_IN_7	R/W	0h	Select control for Input 7 to decide between external input and CLB_GP_REG[7] 0 : Input comes from selected external input 1 : Input comes from CLB_GP_REG[7] Reset type: SYSRSn
6	SEL_GP_IN_6	R/W	0h	Select control for Input 6 to decide between external input and CLB_GP_REG[6] 0 : Input comes from selected external input 1 : Input comes from CLB_GP_REG[6] Reset type: SYSRSn
5	SEL_GP_IN_5	R/W	0h	Select control for Input 5 to decide between external input and CLB_GP_REG[5] 0 : Input comes from selected external input 1 : Input comes from CLB_GP_REG[5] Reset type: SYSRSn
4	SEL_GP_IN_4	R/W	0h	Select control for Input 4 to decide between external input and CLB_GP_REG[4] 0 : Input comes from selected external input 1 : Input comes from CLB_GP_REG[4] Reset type: SYSRSn
3	SEL_GP_IN_3	R/W	0h	Select control for Input 3 to decide between external input and CLB_GP_REG[3] 0 : Input comes from selected external input 1 : Input comes from CLB_GP_REG[3] Reset type: SYSRSn
2	SEL_GP_IN_2	R/W	0h	Select control for Input 2 to decide between external input and CLB_GP_REG[2] 0 : Input comes from selected external input 1 : Input comes from CLB_GP_REG[2] Reset type: SYSRSn

Table 32-64. CLB_IN_MUX_SEL_0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	SEL_GP_IN_1	R/W	0h	Select control for Input 1 to decide between external input and CLB_GP_REG[1] 0 : Input comes from selected external input 1 : Input comes from CLB_GP_REG[1] Reset type: SYSRSn
0	SEL_GP_IN_0	R/W	0h	Select control for Input 0 to decide between external input and CLB_GP_REG[0] 0 : Input comes from selected external input 1 : Input comes from CLB_GP_REG[0] Reset type: SYSRSn

32.8.3.6 CLB_LCL_MUX_SEL_1 Register (Offset = Ah) [Reset = 0h]

CLB_LCL_MUX_SEL_1 is shown in Figure 32-63 and described in Table 32-65.

Return to the Summary Table.

Input Mux selection for local mux

Figure 32-63. CLB_LCL_MUX_SEL_1 Register

31	30	29	28	27	26	25	24
MISC_INPUT_SEL_3	MISC_INPUT_SEL_2	MISC_INPUT_SEL_1	MISC_INPUT_SEL_0	RESERVED			
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h			
23	22	21	20	19	18	17	16
RESERVED				LCL_MUX_SEL_IN_3			
R-0h				R/W-0h			
15	14	13	12	11	10	9	8
LCL_MUX_SEL_IN_3	LCL_MUX_SEL_IN_2					LCL_MUX_SEL_IN_1	
R/W-0h	R/W-0h					R/W-0h	
7	6	5	4	3	2	1	0
LCL_MUX_SEL_IN_1			LCL_MUX_SEL_IN_0				
R/W-0h			R/W-0h				

Table 32-65. CLB_LCL_MUX_SEL_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MISC_INPUT_SEL_3	R/W	0h	Setting this bit to 1 will select MISC_INPUT instead of Local Mux for Input 3 Reset type: SYSRSn
30	MISC_INPUT_SEL_2	R/W	0h	Setting this bit to 1 will select MISC_INPUT instead of Local Mux for Input 2 Reset type: SYSRSn
29	MISC_INPUT_SEL_1	R/W	0h	Setting this bit to 1 will select MISC_INPUT instead of Local Mux for Input 1 Reset type: SYSRSn
28	MISC_INPUT_SEL_0	R/W	0h	Setting this bit to 1 will select MISC_INPUT instead of Local Mux for Input 0 Reset type: SYSRSn
27-20	RESERVED	R	0h	Reserved
19-15	LCL_MUX_SEL_IN_3	R/W	0h	5 bit MUX Select for Local MUX control for Input 3 See Local Signals and Mux Selection Table Reset type: SYSRSn
14-10	LCL_MUX_SEL_IN_2	R/W	0h	5 bit MUX Select for Local MUX control for Input 2 See Local Signals and Mux Selection Table Reset type: SYSRSn
9-5	LCL_MUX_SEL_IN_1	R/W	0h	5 bit MUX Select for Local MUX control for Input 1 See Local Signals and Mux Selection Table Reset type: SYSRSn
4-0	LCL_MUX_SEL_IN_0	R/W	0h	5 bit MUX Select for Local MUX control for Input 0 See Local Signals and Mux Selection Table Reset type: SYSRSn

32.8.3.7 CLB_LCL_MUX_SEL_2 Register (Offset = Ch) [Reset = 0h]

CLB_LCL_MUX_SEL_2 is shown in Figure 32-64 and described in Table 32-66.

Return to the Summary Table.

Input Mux selection for local mux

Figure 32-64. CLB_LCL_MUX_SEL_2 Register

31	30	29	28	27	26	25	24
MISC_INPUT_SEL_7	MISC_INPUT_SEL_6	MISC_INPUT_SEL_5	MISC_INPUT_SEL_4	RESERVED			
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h			
23	22	21	20	19	18	17	16
RESERVED				LCL_MUX_SEL_IN_7			
R-0h				R/W-0h			
15	14	13	12	11	10	9	8
LCL_MUX_SEL_IN_7	LCL_MUX_SEL_IN_6					LCL_MUX_SEL_IN_5	
R/W-0h	R/W-0h					R/W-0h	
7	6	5	4	3	2	1	0
LCL_MUX_SEL_IN_5			LCL_MUX_SEL_IN_4				
R/W-0h			R/W-0h				

Table 32-66. CLB_LCL_MUX_SEL_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	MISC_INPUT_SEL_7	R/W	0h	Setting this bit to 1 will select MISC_INPUT instead of Local Mux for Input 7 Reset type: SYSRSn
30	MISC_INPUT_SEL_6	R/W	0h	Setting this bit to 1 will select MISC_INPUT instead of Local Mux for Input 6 Reset type: SYSRSn
29	MISC_INPUT_SEL_5	R/W	0h	Setting this bit to 1 will select MISC_INPUT instead of Local Mux for Input 5 Reset type: SYSRSn
28	MISC_INPUT_SEL_4	R/W	0h	Setting this bit to 1 will select MISC_INPUT instead of Local Mux for Input 4 Reset type: SYSRSn
27-20	RESERVED	R	0h	Reserved
19-15	LCL_MUX_SEL_IN_7	R/W	0h	5 bit MUX Select for Local MUX control for Input 7 See Local Signals and Mux Selection Table Reset type: SYSRSn
14-10	LCL_MUX_SEL_IN_6	R/W	0h	5 bit MUX Select for Local MUX control for Input 6 See Local Signals and Mux Selection Table Reset type: SYSRSn
9-5	LCL_MUX_SEL_IN_5	R/W	0h	5 bit MUX Select for Local MUX control for Input 5 See Local Signals and Mux Selection Table Reset type: SYSRSn
4-0	LCL_MUX_SEL_IN_4	R/W	0h	5 bit MUX Select for Local MUX control for Input 4 See Local Signals and Mux Selection Table Reset type: SYSRSn

32.8.3.8 CLB_BUF_PTR Register (Offset = Eh) [Reset = 0h]

CLB_BUF_PTR is shown in Figure 32-65 and described in Table 32-67.

Return to the Summary Table.

PUSH and PULL pointers

Figure 32-65. CLB_BUF_PTR Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								PUSH								RESERVED								PULL							
R-0h								R/W-0h								R-0h								R/W-0h							

Table 32-67. CLB_BUF_PTR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	PUSH	R/W	0h	8 bit pointer which indicates the number of data values which have been pulled from the buffer by the High Level Controller. This counter will wrap around after 0xff. The Least significant 2 bits are used as the actual pointer for the operation. Reset type: SYSRSn
15-8	RESERVED	R	0h	Reserved
7-0	PULL	R/W	0h	8 bit pointer which indicates the number of data values that have been written by the High Level controller into the buffer. The Least significant 2 bits are used as the actual pointer for the operation. Reset type: SYSRSn

32.8.3.9 CLB_GP_REG Register (Offset = 10h) [Reset = 0h]

CLB_GP_REG is shown in Figure 32-66 and described in Table 32-68.

Return to the Summary Table.

General purpose register for CELL inputs

Figure 32-66. CLB_GP_REG Register

31	30	29	28	27	26	25	24
SW_RLS_CTRL_7	SW_RLS_CTRL_6	SW_RLS_CTRL_5	SW_RLS_CTRL_4	SW_RLS_CTRL_3	SW_RLS_CTRL_2	SW_RLS_CTRL_1	SW_RLS_CTRL_0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
SW_GATING_CTRL_7	SW_GATING_CTRL_6	SW_GATING_CTRL_5	SW_GATING_CTRL_4	SW_GATING_CTRL_3	SW_GATING_CTRL_2	SW_GATING_CTRL_1	SW_GATING_CTRL_0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
REG							
R/W-0h							

Table 32-68. CLB_GP_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SW_RLS_CTRL_7	R/W	0h	Software release control for output 7 of the asynchronous output conditioning block Reset type: SYSRSn
30	SW_RLS_CTRL_6	R/W	0h	Software release control for output 6 of the asynchronous output conditioning block Reset type: SYSRSn
29	SW_RLS_CTRL_5	R/W	0h	Software release control for output 5 of the asynchronous output conditioning block Reset type: SYSRSn
28	SW_RLS_CTRL_4	R/W	0h	Software release control for output 4 of the asynchronous output conditioning block Reset type: SYSRSn
27	SW_RLS_CTRL_3	R/W	0h	Software release control for output 3 of the asynchronous output conditioning block Reset type: SYSRSn
26	SW_RLS_CTRL_2	R/W	0h	Software release control for output 2 of the asynchronous output conditioning block Reset type: SYSRSn
25	SW_RLS_CTRL_1	R/W	0h	Software release control for output 1 of the asynchronous output conditioning block Reset type: SYSRSn
24	SW_RLS_CTRL_0	R/W	0h	Software release control for output 0 of the asynchronous output conditioning block Reset type: SYSRSn
23	SW_GATING_CTRL_7	R/W	0h	Software gating control for output 7 of the asynchronous output conditioning block Reset type: SYSRSn
22	SW_GATING_CTRL_6	R/W	0h	Software gating control for output 6 of the asynchronous output conditioning block Reset type: SYSRSn

Table 32-68. CLB_GP_REG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	SW_GATING_CTRL_5	R/W	0h	Software gating control for output 5 of the asynchronous output conditioning block Reset type: SYSRSn
20	SW_GATING_CTRL_4	R/W	0h	Software gating control for output 4 of the asynchronous output conditioning block Reset type: SYSRSn
19	SW_GATING_CTRL_3	R/W	0h	Software gating control for output 3 of the asynchronous output conditioning block Reset type: SYSRSn
18	SW_GATING_CTRL_2	R/W	0h	Software gating control for output 2 of the asynchronous output conditioning block Reset type: SYSRSn
17	SW_GATING_CTRL_1	R/W	0h	Software gating control for output 1 of the asynchronous output conditioning block Reset type: SYSRSn
16	SW_GATING_CTRL_0	R/W	0h	Software gating control for output 0 of the asynchronous output conditioning block Reset type: SYSRSn
15-8	RESERVED	R	0h	Reserved
7-0	REG	R/W	0h	8 bits which are directly connected to the 8 inputs of the CELL if that corresponding bit is selected in the CLB_IN_MUX_SEL_0 register Reset type: SYSRSn

32.8.3.10 CLB_OUT_EN Register (Offset = 12h) [Reset = 0h]

CLB_OUT_EN is shown in Figure 32-67 and described in Table 32-69.

Return to the Summary Table.

CELL output enable register

Figure 32-67. CLB_OUT_EN Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
OUT0																															
R/W-0h																															

Table 32-69. CLB_OUT_EN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	OUT0	R/W	0h	32 bits which are directly driven out as OUTPUT_EN signals. Enabling bit x (x = 0:31) will override the corresponding peripheral signal muxed on the CLB OUTx. Reset type: SYSRSn

32.8.3.11 CLB_GLBL_MUX_SEL_1 Register (Offset = 14h) [Reset = 0h]

CLB_GLBL_MUX_SEL_1 is shown in Figure 32-68 and described in Table 32-70.

Return to the Summary Table.

Global Mux select for CELL inputs

Figure 32-68. CLB_GLBL_MUX_SEL_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				GLBL_MUX_SEL_IN_3							GLBL_MUX_SEL_IN_2				
R-0h				R/W-0h							R/W-0h				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GLBL_MUX_SEL_IN_2				GLBL_MUX_SEL_IN_1							GLBL_MUX_SEL_IN_0				
R/W-0h				R/W-0h							R/W-0h				

Table 32-70. CLB_GLBL_MUX_SEL_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-21	GLBL_MUX_SEL_IN_3	R/W	0h	7 bit MUX Select for Global MUX control for Input 3 See Global Signals and Mux Selection Table Reset type: SYSRSn
20-14	GLBL_MUX_SEL_IN_2	R/W	0h	7 bit MUX Select for Global MUX control for Input 2 See Global Signals and Mux Selection Table Reset type: SYSRSn
13-7	GLBL_MUX_SEL_IN_1	R/W	0h	7 bit MUX Select for Global MUX control for Input 1 See Global Signals and Mux Selection Table Reset type: SYSRSn
6-0	GLBL_MUX_SEL_IN_0	R/W	0h	7 bit MUX Select for Global MUX control for Input 0 See Global Signals and Mux Selection Table Reset type: SYSRSn

32.8.3.12 CLB_GLBL_MUX_SEL_2 Register (Offset = 16h) [Reset = 0h]

CLB_GLBL_MUX_SEL_2 is shown in Figure 32-69 and described in Table 32-71.

Return to the Summary Table.

Global Mux select for CELL inputs

Figure 32-69. CLB_GLBL_MUX_SEL_2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED				GLBL_MUX_SEL_IN_7							GLBL_MUX_SEL_IN_6				
R-0h				R/W-0h							R/W-0h				
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
GLBL_MUX_SEL_IN_6				GLBL_MUX_SEL_IN_5							GLBL_MUX_SEL_IN_4				
R/W-0h				R/W-0h							R/W-0h				

Table 32-71. CLB_GLBL_MUX_SEL_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	RESERVED	R	0h	Reserved
27-21	GLBL_MUX_SEL_IN_7	R/W	0h	7 bit MUX Select for Global MUX control for Input 7 See Global Signals and Mux Selection Table Reset type: SYSRSn
20-14	GLBL_MUX_SEL_IN_6	R/W	0h	7 bit MUX Select for Global MUX control for Input 6 See Global Signals and Mux Selection Table Reset type: SYSRSn
13-7	GLBL_MUX_SEL_IN_5	R/W	0h	7 bit MUX Select for Global MUX control for Input 5 See Global Signals and Mux Selection Table Reset type: SYSRSn
6-0	GLBL_MUX_SEL_IN_4	R/W	0h	7 bit MUX Select for Global MUX control for Input 4 See Global Signals and Mux Selection Table Reset type: SYSRSn

32.8.3.13 CLB_PRESCALE_CTRL Register (Offset = 18h) [Reset = 0h]

CLB_PRESCALE_CTRL is shown in Figure 32-70 and described in Table 32-72.

Return to the Summary Table.

Prescaler register control

Figure 32-70. CLB_PRESCALE_CTRL Register

31	30	29	28	27	26	25	24
PRESCALE							
R/W-0h							
23	22	21	20	19	18	17	16
PRESCALE							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		TAP				STRB	CLKEN
R-0h		R/W-0h				R/W-0h	R/W-0h

Table 32-72. CLB_PRESCALE_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	PRESCALE	R/W	0h	16-bit Value of prescaler to be used for the counter as reference to reset when reaching this value. The counter is a simple incrementing counter which will count up to the reference value and reset to 0 and this cycle will continue as long as the counter is enabled. This 16-bit register value is used as a reference for the 16-bit counter to reset to zero whenever count reaches this value. Reset type: SYSRSn
15-6	RESERVED	R	0h	Reserved
5-2	TAP	R/W	0h	TAP Select value. These 4 bits will be used as a select to tap one of the 16 register bit position of the counter as the output. 0000 selects Counter Bit position 0 0001 selects Counter Bit position 1 .... 1111 selects Counter Bit position 15 Reset type: SYSRSn
1	STRB	R/W	0h	When set to 0, a strobe output will be sent out whenever the counter value matches the PRESCALE_VALUE. When set to 1, the output of the counter register bit position as selected by TAP_SELECT_VALUE will be sent out. Reset type: SYSRSn
0	CLKEN	R/W	0h	Enable the prescale clock/strobe generator. A 16-bit counter is used to either generate a strobe or send out a selected counter bit position to the CLB CELL. This is meant to be a general purpose strobe/prescaled clock which can be used by the CELL logic if needed. This will be sent to the CELL through one of the LCL_IN MUX ports. Reset type: SYSRSn

32.8.3.14 CLB_INTR_TAG_REG Register (Offset = 20h) [Reset = 0h]

CLB_INTR_TAG_REG is shown in Figure 32-71 and described in Table 32-73.

Return to the Summary Table.

Interrupt Tag register

Figure 32-71. CLB_INTR_TAG_REG Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED		TAG					
R-0h		R/W-0h					

Table 32-73. CLB_INTR_TAG_REG Register Field Descriptions

Bit	Field	Type	Reset	Description
15-6	RESERVED	R	0h	Reserved
5-0	TAG	R/W	0h	6 bits which are used by the High Level Controller to set a tag value on flagging interrupts. . This can be cleared through the VBUS interface since it is writeable through the VBUS. Reset type: SYSRSn

32.8.3.15 CLB_LOCK Register (Offset = 22h) [Reset = 0h]

CLB_LOCK is shown in Figure 32-72 and described in Table 32-74.

Return to the Summary Table.

Lock control register

Figure 32-72. CLB_LOCK Register

31	30	29	28	27	26	25	24
KEY							
WSonce-0h							
23	22	21	20	19	18	17	16
KEY							
WSonce-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							LOCK
R-0h							R/W-0h

Table 32-74. CLB_LOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	KEY	WSonce	0h	These 16 bits act as a key to enable writes to Bit 0 of this register. The only time a '1' can be written to Bit 0 is by a single 32-bit write where bits 31:16 equal 0x5a5a and bit 0 is '1'. All other writes are ignored including separate 16-bit writes. This is EALLOW protected. Reset type: SYSRSn
15-1	RESERVED	R	0h	Reserved
0	LOCK	R/W	0h	This bit is used as a one-time write bit (Set Once). Once it is set to '1', only a reset (SYSRSN 0) will clear this bit back to 0. Reset type: SYSRSn

32.8.3.16 CLB_HLC_INSTR_READ_PTR Register (Offset = 24h) [Reset = 0h]

CLB_HLC_INSTR_READ_PTR is shown in Figure 32-73 and described in Table 32-75.

Return to the Summary Table.

HLC instruction read pointer

Figure 32-73. CLB_HLC_INSTR_READ_PTR Register

15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			READ_PTR				
R-0h			R/W-0h				

Table 32-75. CLB_HLC_INSTR_READ_PTR Register Field Descriptions

Bit	Field	Type	Reset	Description
15-5	RESERVED	R	0h	Reserved
4-0	READ_PTR	R/W	0h	This is a 5 bit value which will be used as an address pointer to read out HLC instruction memory. Reset type: SYSRSn

32.8.3.17 CLB_HLC_INSTR_VALUE Register (Offset = 26h) [Reset = 0h]

CLB_HLC_INSTR_VALUE is shown in Figure 32-74 and described in Table 32-76.

Return to the Summary Table.

HLC instruction read value

Figure 32-74. CLB_HLC_INSTR_VALUE Register

15	14	13	12	11	10	9	8
RESERVED				INSTR			
R-0h				R-0h			
7	6	5	4	3	2	1	0
INSTR							
R-0h							

Table 32-76. CLB_HLC_INSTR_VALUE Register Field Descriptions

Bit	Field	Type	Reset	Description
15-12	RESERVED	R	0h	Reserved
11-0	INSTR	R	0h	This is a 12 bit value which will read the content of the HLC instruction memory address pointed by CLB_HLC_INSTR_READ_PTR register. Reset type: SYSRSn

32.8.3.18 CLB_DBG_OUT_2 Register (Offset = 2Eh) [Reset = 0h]

CLB_DBG_OUT_2 is shown in Figure 32-75 and described in Table 32-77.

Return to the Summary Table.

Visibility for CLB inputs and final asynchronous outputs

Figure 32-75. CLB_DBG_OUT_2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED																IN								OUT							
R-0h																R/W-0h								R/W-0h							

Table 32-77. CLB_DBG_OUT_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15-8	IN	R/W	0h	These bits reflect the state of the 8 inputs finally going to the CELL after selection and input conditioning. Reset type: SYSRSn
7-0	OUT	R/W	0h	These bits reflect the state of the 8 outputs of the Output Conditioning Block. Reset type: SYSRSn

32.8.3.19 CLB_DBG_R0 Register (Offset = 30h) [Reset = 0h]

CLB_DBG_R0 is shown in Figure 32-76 and described in Table 32-78.

Return to the Summary Table.

R0 of High level Controller

Figure 32-76. CLB_DBG_R0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBG																															
R-0h																															

Table 32-78. CLB_DBG_R0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DBG	R	0h	CLB_DBG_R0 Reset type: SYSRSn

32.8.3.20 CLB_DBG_R1 Register (Offset = 32h) [Reset = 0h]

CLB_DBG_R1 is shown in Figure 32-77 and described in Table 32-79.

Return to the Summary Table.

R1 of High level Controller

Figure 32-77. CLB_DBG_R1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBG																															
R-0h																															

Table 32-79. CLB_DBG_R1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DBG	R	0h	CLB_DBG_R1 Reset type: SYSRSn

32.8.3.21 CLB_DBG_R2 Register (Offset = 34h) [Reset = 0h]

CLB_DBG_R2 is shown in Figure 32-78 and described in Table 32-80.

Return to the Summary Table.

R2 of High level Controller

Figure 32-78. CLB_DBG_R2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBG																															
R-0h																															

Table 32-80. CLB_DBG_R2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DBG	R	0h	CLB_DBG_R2 Reset type: SYSRSn

32.8.3.22 CLB_DBG_R3 Register (Offset = 36h) [Reset = 0h]

CLB_DBG_R3 is shown in Figure 32-79 and described in Table 32-81.

Return to the Summary Table.

R3 of High level Controller

Figure 32-79. CLB_DBG_R3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBG																															
R-0h																															

Table 32-81. CLB_DBG_R3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DBG	R	0h	CLB_DBG_R3 Reset type: SYSRSn

32.8.3.23 CLB_DBG_C0 Register (Offset = 38h) [Reset = 0h]

CLB_DBG_C0 is shown in Figure 32-80 and described in Table 32-82.

Return to the Summary Table.

Count of Unit 0

Figure 32-80. CLB_DBG_C0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBG																															
R-0h																															

Table 32-82. CLB_DBG_C0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DBG	R	0h	CLB_DBG_C0 Reset type: SYSRSn

32.8.3.24 CLB_DBG_C1 Register (Offset = 3Ah) [Reset = 0h]

CLB_DBG_C1 is shown in Figure 32-81 and described in Table 32-83.

Return to the Summary Table.

Count of Unit 1

Figure 32-81. CLB_DBG_C1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBG																															
R-0h																															

Table 32-83. CLB_DBG_C1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DBG	R	0h	CLB_DBG_C1 Reset type: SYSRSn

32.8.3.25 CLB_DBG_C2 Register (Offset = 3Ch) [Reset = 0h]

CLB_DBG_C2 is shown in Figure 32-82 and described in Table 32-84.

Return to the Summary Table.

Count of Unit 2

Figure 32-82. CLB_DBG_C2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DBG																															
R-0h																															

Table 32-84. CLB_DBG_C2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DBG	R	0h	CLB_DBG_C2 Reset type: SYSRSn

32.8.3.26 CLB_DBG_OUT Register (Offset = 3Eh) [Reset = 00010100h]

CLB_DBG_OUT is shown in Figure 32-83 and described in Table 32-85.

Return to the Summary Table.

Outputs of various units in the Cell

Figure 32-83. CLB_DBG_OUT Register

31	30	29	28	27	26	25	24
OUT7	OUT6	OUT5	OUT4	OUT3	OUT2	OUT1	OUT0
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
23	22	21	20	19	18	17	16
LUT42_OUT	FSM2_LUTOUT	FSM2_S1	FSM2_S0	COUNT2_MATCH1	COUNT2_ZERO	COUNT2_MATCH2	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
15	14	13	12	11	10	9	8
LUT41_OUT	FSM1_LUTOUT	FSM1_S1	FSM1_S0	COUNT1_MATCH1	COUNT1_ZERO	COUNT1_MATCH2	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h
7	6	5	4	3	2	1	0
LUT40_OUT	FSM0_LUTOUT	FSM0_S1	FSM0_S0	COUNT0_MATCH1	COUNT0_ZERO	COUNT0_MATCH2	RESERVED
R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h	R-0h

Table 32-85. CLB_DBG_OUT Register Field Descriptions

Bit	Field	Type	Reset	Description
31	OUT7	R	0h	CELL Output 7 Reset type: SYSRSn
30	OUT6	R	0h	CELL Output 6 Reset type: SYSRSn
29	OUT5	R	0h	CELL Output 5 Reset type: SYSRSn
28	OUT4	R	0h	CELL Output 4 Reset type: SYSRSn
27	OUT3	R	0h	CELL Output 3 Reset type: SYSRSn
26	OUT2	R	0h	CELL Output 2 Reset type: SYSRSn
25	OUT1	R	0h	CELL Output 1 Reset type: SYSRSn
24	OUT0	R	0h	CELL Output 0 Reset type: SYSRSn
23	LUT42_OUT	R	0h	LUT4_OUT UNIT 2 Reset type: SYSRSn
22	FSM2_LUTOUT	R	0h	FSM_LUT_OUT UNIT 2 Reset type: SYSRSn
21	FSM2_S1	R	0h	FSM_S1 UNIT 2 Reset type: SYSRSn
20	FSM2_S0	R	0h	FSM_S0 UNIT 2 Reset type: SYSRSn
19	COUNT2_MATCH1	R	0h	COUNT_MATCH1 UNIT 2 Reset type: SYSRSn
18	COUNT2_ZERO	R	0h	COUNT_ZERO UNIT 2 Reset type: SYSRSn

Table 32-85. CLB_DBG_OUT Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
17	COUNT2_MATCH2	R	0h	COUNT_MATCH2 UNIT 2 Reset type: SYSRSn
16	RESERVED	R	0h	Reserved
15	LUT41_OUT	R	0h	LUT4_OUT UNIT 1 Reset type: SYSRSn
14	FSM1_LUTOUT	R	0h	FSM_LUT_OUT UNIT 1 Reset type: SYSRSn
13	FSM1_S1	R	0h	FSM_S1 UNIT 1 Reset type: SYSRSn
12	FSM1_S0	R	0h	FSM_S0 UNIT 1 Reset type: SYSRSn
11	COUNT1_MATCH1	R	0h	COUNT_MATCH1 UNIT 1 Reset type: SYSRSn
10	COUNT1_ZERO	R	0h	COUNT_ZERO UNIT 1 Reset type: SYSRSn
9	COUNT1_MATCH2	R	0h	COUNT_MATCH2 UNIT 1 Reset type: SYSRSn
8	RESERVED	R	0h	Reserved
7	LUT40_OUT	R	0h	LUT4_OUT UNIT 0 Reset type: SYSRSn
6	FSM0_LUTOUT	R	0h	FSM_LUT_OUT UNIT 0 Reset type: SYSRSn
5	FSM0_S1	R	0h	FSM_S1 UNIT 0 Reset type: SYSRSn
4	FSM0_S0	R	0h	FSM_S0 UNIT 0 Reset type: SYSRSn
3	COUNT0_MATCH1	R	0h	COUNT_MATCH1 UNIT 0 Reset type: SYSRSn
2	COUNT0_ZERO	R	0h	COUNT_ZERO UNIT 0 Reset type: SYSRSn
1	COUNT0_MATCH2	R	0h	COUNT_MATCH2 UNIT 0 Reset type: SYSRSn
0	RESERVED	R	0h	Reserved

32.8.4 CLB_DATA_EXCHANGE_REGS Registers

Table 32-86 lists the memory-mapped registers for the CLB_DATA_EXCHANGE_REGS registers. All register offset addresses not listed in Table 32-86 should be considered as reserved locations and the register contents should not be modified.

Table 32-86. CLB_DATA_EXCHANGE_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h + formula	CLB_PUSH_y	CLB_PUSH FIFO Registers (from HLC)		Go
40h + formula	CLB_PULL_y	CLB_PULL FIFO Registers (TO HLC)		Go

Complex bit access types are encoded to fit into small table cells. Table 32-87 shows the codes that are used for access types in this section.

Table 32-87. CLB_DATA_EXCHANGE_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

32.8.4.1 CLB_PUSH_y Register (Offset = 0h + formula) [Reset = 0h]

CLB_PUSH_y is shown in Figure 32-84 and described in Table 32-88.

Return to the Summary Table.

CLB_PUSH FIFO Registers (from HLC)

Offset = 0h + (y * 2h); where y = 0h to 3h

Figure 32-84. CLB_PUSH_y Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PUSH																															
R-0h																															

Table 32-88. CLB_PUSH_y Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PUSH	R	0h	FIFO TO System From CLB Reset type: SYSRSn

32.8.4.2 CLB_PULL_y Register (Offset = 40h + formula) [Reset = 0h]

CLB_PULL_y is shown in Figure 32-85 and described in Table 32-89.

Return to the Summary Table.

CLB_PULL FIFO Registers (TO HLC)

Offset = 40h + (y * 2h); where y = 0h to 3h

Figure 32-85. CLB_PULL_y Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PULL																															
R/W-0h																															

Table 32-89. CLB_PULL_y Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	PULL	R/W	0h	FIFO From system TO CLB Reset type: SYSRSn

32.8.5 CLB Registers to Driverlib Functions

Table 32-90. CLB Registers to Driverlib Functions

File	Driverlib Function
COUNT_RESET	
clb.h	CLB_selectCounterInputs
COUNT_MODE_1	
clb.h	CLB_selectCounterInputs
COUNT_MODE_0	
clb.h	CLB_selectCounterInputs
COUNT_EVENT	
clb.h	CLB_selectCounterInputs
FSM_EXTRA_IN0	
clb.h	CLB_selectFSMInputs
FSM_EXTERNAL_IN0	
clb.h	CLB_selectFSMInputs
FSM_EXTERNAL_IN1	
clb.h	CLB_selectFSMInputs
FSM_EXTRA_IN1	
clb.h	CLB_selectFSMInputs
LUT4_IN0	
clb.h	CLB_selectLUT4Inputs
LUT4_IN1	
clb.h	CLB_selectLUT4Inputs
LUT4_IN2	
clb.h	CLB_selectLUT4Inputs
LUT4_IN3	
clb.h	CLB_selectLUT4Inputs
FSM_LUT_FN1_0	
clb.h	CLB_configFSMLUTFunction
FSM_LUT_FN2	
clb.h	CLB_configFSMLUTFunction

Table 32-90. CLB Registers to Driverlib Functions (continued)

File	Driverlib Function
LUT4_FN1_0	
clb.h	CLB_configLUT4Function
LUT4_FN2	
clb.h	CLB_configLUT4Function
FSM_NEXT_STATE_0	
clb.h	CLB_configFSMNextState
FSM_NEXT_STATE_1	
clb.h	CLB_configFSMNextState
FSM_NEXT_STATE_2	
clb.h	CLB_configFSMNextState
MISC_CONTROL	
clb.h	CLB_configMiscCtrlModes
OUTPUT_LUT_0	
clb.h	CLB_configOutputLUT
OUTPUT_LUT_1	
-	See OUTPUT_LUT_0
OUTPUT_LUT_2	
-	See OUTPUT_LUT_0
OUTPUT_LUT_3	
-	See OUTPUT_LUT_0
OUTPUT_LUT_4	
-	See OUTPUT_LUT_0
OUTPUT_LUT_5	
-	See OUTPUT_LUT_0
OUTPUT_LUT_6	
-	See OUTPUT_LUT_0
OUTPUT_LUT_7	
-	See OUTPUT_LUT_0
HLC_EVENT_SEL	
clb.h	CLB_configHLCEventSelect
COUNT_MATCH_TAP_SEL	
clb.h	CLB_configCounterTapSelects
OUTPUT_COND_CTRL_0	
clb.h	CLB_configAOC
OUTPUT_COND_CTRL_1	
-	
OUTPUT_COND_CTRL_2	
-	
OUTPUT_COND_CTRL_3	
-	
OUTPUT_COND_CTRL_4	
-	
OUTPUT_COND_CTRL_5	
-	
OUTPUT_COND_CTRL_6	

Table 32-90. CLB Registers to Driverlib Functions (continued)

File	Driverlib Function
-	
OUTPUT_COND_CTRL_7	
-	
MISC_ACCESS_CTRL	
clb.h	CLB_disableOutputMaskUpdates
clb.h	CLB_enableOutputMaskUpdates
clb.h	CLB_disableSPIBufferAccess
clb.h	CLB_enableSPIBufferAccess
SPI_DATA_CTRL_HI	
clb.h	CLB_configSPIBufferLoadSignal
clb.h	CLB_configSPIBufferShift
LOAD_EN	
clb.h	CLB_enableCLB
clb.h	CLB_disableCLB
clb.h	CLB_enableNMI
clb.h	CLB_disableNMI
clb.h	CLB_writeInterface
clb.h	CLB_enablePipelineMode
clb.h	CLB_disablePipelineMode
LOAD_ADDR	
clb.h	CLB_writeInterface
LOAD_DATA	
clb.h	CLB_writeInterface
INPUT_FILTER	
clb.h	CLB_selectInputFilter
clb.h	CLB_enableSynchronization
clb.h	CLB_disableSynchronization
clb.h	CLB_enableInputPipelineMode
clb.h	CLB_disableInputPipelineMode
IN_MUX_SEL_0	
clb.h	CLB_configGPInputMux
LCL_MUX_SEL_1	
clb.h	CLB_configLocalInputMux
LCL_MUX_SEL_2	
clb.h	CLB_configLocalInputMux
BUF_PTR	
clb.c	CLB_clearFIFOs
GP_REG	
clb.h	CLB_writeSWReleaseControl
clb.h	CLB_writeSWGateControl
clb.h	CLB_setGPREG
clb.h	CLB_getGPREG
OUT_EN	
clb.h	CLB_setOutputMask
GLBL_MUX_SEL_1	

Table 32-90. CLB Registers to Driverlib Functions (continued)

File	Driverlib Function
clb.h	CLB_configGlobalInputMux
GLBL_MUX_SEL_2	
clb.h	CLB_configGlobalInputMux
PRESCALE_CTRL	
clb.h	CLB_configureClockPrescalar
clb.h	CLB_configureStrobeMode
INTR_TAG_REG	
clb.h	CLB_getInterruptTag
clb.h	CLB_clearInterruptTag
LOCK	
clb.h	CLB_enableLock
HLC_INSTR_READ_PTR	
-	
HLC_INSTR_VALUE	
-	
DBG_OUT_2	
-	
DBG_R0	
-	
DBG_R1	
-	
DBG_R2	
-	
DBG_R3	
-	
DBG_C0	
-	
DBG_C1	
-	
DBG_C2	
-	
DBG_OUT	
clb.h	CLB_getOutputStatus
PUSH	
clb.c	CLB_readFIFOs
PULL	
clb.c	CLB_clearFIFOs
clb.c	CLB_writeFIFOs

Chapter 33

Advance Encryption Standard (AES) Accelerator

This section describes the Advanced Encryption Standard (AES) cryptographic hardware-accelerated module.

33.1 Introduction

This section introduces the advanced encryption standard (AES) and describes the AES main functions and connections in the device.

The AES module provides hardware-accelerated data encryption and decryption operations based on a binary key. The AES is a symmetric cipher module that supports a 128-, 192-, or 256-bit key in hardware for encryption and decryption. The AES module is based on a symmetric algorithm, which means that the encryption and decryption keys are identical. To encrypt data means to convert it from plain text to an unintelligible form called cipher text. Decrypting cipher text converts previously encrypted data to its original plain text form. The main features of the AES accelerator are:

AES encrypt and decrypt operations are supported by:

- Galois/Counter mode (GCM), with basic GHASH operation
- Counter mode with CBC-MAC (CCM)
- XTS mode

The following feedback operating modes are available:

- Electronic code book mode (ECB)
- Cipher block chaining mode (CBC)
- Counter mode (CTR)
- Cipher feedback mode (CFB), 128-bit
- F8 mode
- Key sizes: 128, 192, and 256 bits
- Support for CBC_MAC and Fedora 9 (F9) authentication modes
- Basic GHASH operation (when selecting no encryption)
- Key scheduling in hardware
- Support for µDMA transfers
- Fully synchronous design

33.1.1 AES Block Diagram

Figure 33-1 shows the AES block diagram. A single-core or dual-interface architecture is used.

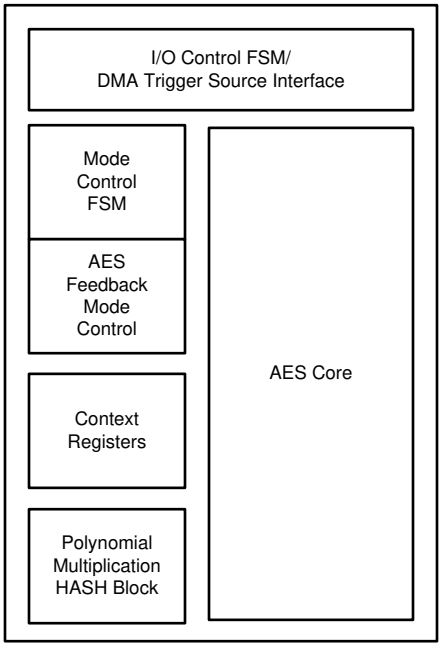


Figure 33-1. AES Block Diagram

AES is an efficient implementation of the Rijndael cipher (the AES algorithm) and a 128-bit polynomial multiplication (referred to here as GHASH, according to the AES-GCM specification). Rijndael is a block cipher in which each data block is 128 bits. The polynomial multiplication multiplies two 128-bit vectors using the smallest 128-bit irreducible polynomial, represented by the following 128-bit string: {0 120}||10000111. The two implementations are combined into the AES wide-bus engine.

Depending on the availability of context and data, the AES wide-bus engine is automatically triggered to process the data. The AES wide-bus engine is directly connected to the context and data registers, so that it can immediately start processing when all data is available. The AES wide-bus engine also interfaces to the I/O control FSM/µDMA request interface.

AES comprises the following major functional blocks:

- Global control FSM and µDMA interface
- Register interface module
- The AES wide-bus engine

The AES wide-bus engine, which is the major top-level component, comprises the following functional blocks:

- Mode control FSM: Manages the data flow to and from the AES wide-bus engine and starts each encryption or decryption operation
- Feedback modes: The logic that implements the various feedback modes supported by AES.
- GHASH core: The polynomial multiplication algorithm used for AES-GCM
- AES key scheduler: Generates AES encryption and decryption (round) keys
- AES encryption core: The AES encryption algorithm
- AES decryption core: The AES decryption algorithm
- Substitution-boxes (S-Boxes): Contain AES S-Box GF(2^8) implementations

AES encryption requires a specific number of rounds, depending on the key length. The supported key lengths are 128-, 192-, and 256-bit, which require 10, 12, and 14 rounds, respectively, or 32, 38, and 44 clock cycles, respectively, because {number of clock cycles} = 2 + 3 × {number of rounds}.

The larger key lengths provide greater encryption strength at the expense of additional rounds, and therefore reduced throughput. The overall throughput of the AES executing polynomial multiplication is adjusted based on the overall cryptographic performance. The AES module contains one ECB core and a dedicated 32-cycle polynomial multiplication module for performing GHASH operations. Polynomial multiplication operates in parallel with the AES core, if data is available for both modules.

Depending on the key size (128, 192, or 256 bits), this core requires 32, 38, or 44 clock cycles to process one 128-bit data block. While one data block processes, the next block can be preloaded immediately. When a block is preloaded, the previous block must finish before additional data can be loaded. Therefore, when the pipeline is full, sequential data blocks can be passed every 32, 38, or 44 clock cycles.

33.1.1.1 Interfaces

The interface signals to the AES module can be grouped into the following categories:

- Clock enable
- DMA and interrupt interface, used to request new context and packet data or to indicate available result data (encrypted or decrypted data, or authentication result)
- Functional register interface

33.1.1.2 AES Subsystem

The AES subsystem interfaces with the DMA module as shown in Table 33-1. Input/output context and data ports from the AES directly feed to the DMA trigger source. Two interrupt registers are included, AES_GLB_INT_FLG and AES_GLB_INT_CLR. AES_GLB_INT_FLG, that provide the status of the secure interrupt generated by the AES while AES_GLB_INT_CLR clears the flag. AES only allows word access. Non-word access (not 32-bit) generates an interrupt and is aggregated in SYS_ERR.

Table 33-1. AES Subsystem DMA Interface

Request	DMA Trigger Source
Context Output	AES_ContextIn
Context Input	AES_ContextOut
Data Output	AES_DataOut
Data Input	AES_DataIn

33.1.1.3 AES Wide-Bus Engine

The AES wide-bus engine performs the cryptographic operations. The composition of the AES core follows:

AES Key Scheduler

The AES key scheduler generates the round keys. During each round, a new subkey is generated from the input key to be XORed with the data. Round keys are generated arbitrarily and parallel to data processing to minimize register requirements.

For encryption operations, the key sequencer transfers the initial key data to the AES core. For decryption operations, the key scheduler must provide the final subkey to the AES core so it can generate the subkeys in reverse order.

AES Encryption Core

The AES encryption core implements the Rijndael algorithm as specified in [FIPS-197]. This core operates on the input block and performs the required substitution, shift, and mix operations. For each round, the encryption core receives the proper round key from the AES key scheduler. A fundamental component of the AES algorithm is the S-Box. The S-Box provides a unique 8-bit output for each 8-bit input. This implementation of the AES encryption core has a 64-bit data path.

AES Decryption Core

The architecture of the AES decryption core is generally the same as the architecture of the encryption core. One difference is that the generation of round keys for decryption requires an initial conversion of the input key (always supplied by the host in the form of an encryption key) to the corresponding decryption key. This conversion is done by performing a dummy encryption operation and storing the final round key as a decryption key. The key scheduler is then reversed to generate the round keys for the decryption operation. Consequently, for each sequence of decryption operations under the same key, a single throughput reduction equal to the time to encrypt a single block occurs. Once a decryption key is generated, subsequent decryption operations with the same key use this generated decryption key directly.

AES Feedback Mode Block

The AES feedback mode block buffers the feedback parameters and controls the various feedback modes. For more information about the ECB, CBC, CTR, and CFB modes of operation, see the NIST-SP800-38A specification.

CTR implements the standard incrementing function, as described in the NIST-SP800-38A specification, with *m* set to 16 or a multiple of 32.

AES-XTS mode requires a polynomial multiplication for initialization vector (IV) generation of the AES operation. This multiplication can be simplified when the first result is available due to the definition and use of the block number within a unit. The input for the polynomial multiplication is not directly j, but α^j, where $\alpha = x^2$ in the GF(2^{128}) domain.

In addition, f8 encryption/decryption mode and f9 and (X)CBC-MAC authentication modes are available.

GHASH Block

The data sequencer manages the data flow to and from the AES core. For data input, the data sequencer monitors the input buffer until a 16-byte block is available. If the AES core is idle, the data sequencer writes this data block to the internal working registers of the AES core, thus clearing the buffer for the next block.

After completing an encryption or decryption operation, the data sequencer writes the AES output to the output buffer. If the output buffer is full at the time of completion, the AES core is held until the buffer clears. Although the data sequencer is designed to support uninterrupted packet encryption, the host must properly manage the input and output packet buffers to achieve optimal performance.

33.1.2 AES Algorithm

The AES algorithm generates block ciphers. The AES block size is 16 bytes. The AES keys can be coded on 128, 192, or 256 bits. The larger key sizes provide a higher level of security, but at the cost of a moderate decrease in throughput.

For the AES algorithm:

- The length of the input and output blocks is 128 bits. The block length is represented by Nb = 4, which reflects the number of 32-bit words.
- The length of the cipher key (K) is 128, 192, or 256 bits. The key length is represented by Nk = 4, 6, or 8, which reflects the number of 32-bit words in the cipher key.
- The number of rounds to be performed during the execution of the algorithm depends on the key size. The number of rounds is represented by Nr, where Nr = 10 when Nk = 4 (128-bit key); Nr = 12 when Nk = 6 (192-bit key); and Nr = 14 when Nk = 8 (256-bit key).

Table 33-2 lists the combinations of keys, blocks, and rounds.

Table 33-2. Key-Block-Round Combinations

Key	Key Length (Nk)	Block Size (Nb)	Number of Rounds (Nr)
128 bits	4	4	10
192 bits	6	4	12
256 bits	8	4	14

The AES algorithm for cipher and inverse cipher uses a round function composed of four different byte-oriented transformations:

- Byte substitution using a substitution table (S-Box): This transformation is a nonlinear byte substitution that operates independently on each byte of the state (the state is an intermediate processed block of 128 bits inside the AES; the state is arranged as an array of [4 × Nk] bytes) using an S-Box. This S-Box transformation is reversible.
- Shifting rows of the state array by different offsets: In this transformation, the bytes in the last three rows of the state are cyclically shifted over different numbers of bytes (offsets). The first row ® = 0) is not shifted.
- Mixing the data within each column of the state array: This transformation operates on the state column-by-column, treating each column as a 4-term polynomial. The columns are considered polynomials over GF(2^8) and multiplied modulo $x^4 + 1$ with a fixed polynomial a(x).
- Adding a round key to the state: In this transformation, a round key is added to the state by a simple bitwise XOR operation. Each round key consists of Nb words from the key schedule.

The AES algorithm takes the cipher key (K) and performs a key expansion routine to generate a key schedule. The key expansion generates a total of Nb × (Nr + 1) words: The algorithm requires an initial set of Nb words, and each Nr round requires Nb words of key data. The resulting key schedule consists of a linear array of 4-byte words, denoted [w_I], with I in the range $0 \le I \le Nb \times (Nr + 1)$.

33.2 AES Operating Modes

33.2.1 GCM Operation

Figure 33-2 shows one round of a GCM operation for encryption and decryption. A 32-bit counter is used as IV (as it is for CTR mode). The data is encrypted in the same way as in CTR mode, by XORing the cryptographic core output with the input. After the encryption/decryption, the ciphertext is XORed with the intermediate authentication result. The XORed result is used as input for the polynomial multiplication to create the next (intermediate) authentication result. For more information about the GCM protocol, see Section 33.3.1.

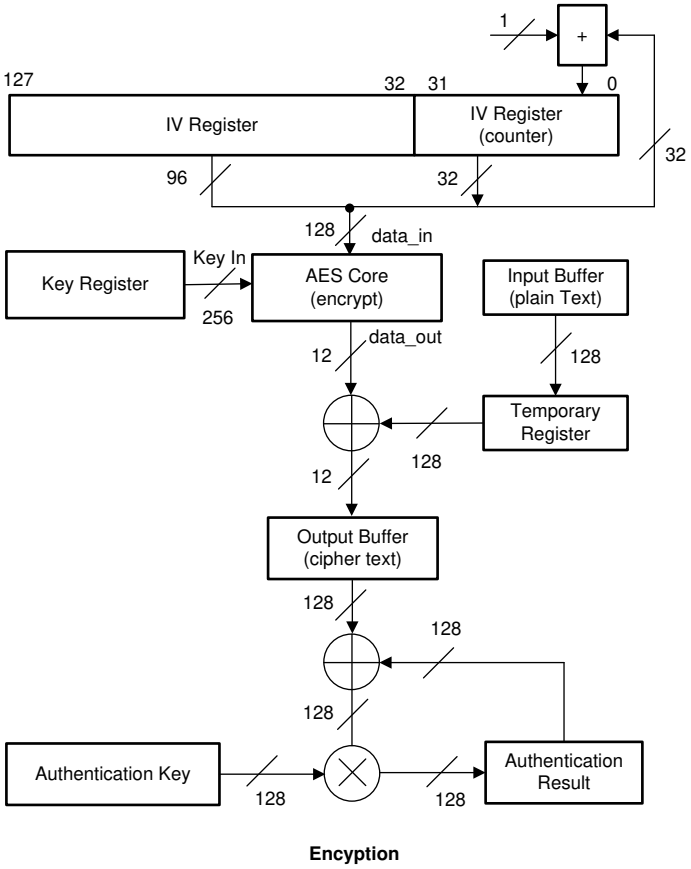


Figure 33-2. AES - GCM Operation

33.2.2 CCM Operation

Figure 33-3 shows one round of a CCM (counter with CBC-MAC) operation for encryption and decryption. A 32-bit counter is used as IV (as it is for CTR mode). The data is encrypted in the same way as in CTR mode, by XORing the cryptographic core output with the input. Immediately after the encryption operation, the plaintext is XORed with the intermediate authentication result. The XOR result is used as input for a second encryption operation to calculate the next (intermediate) authentication result.

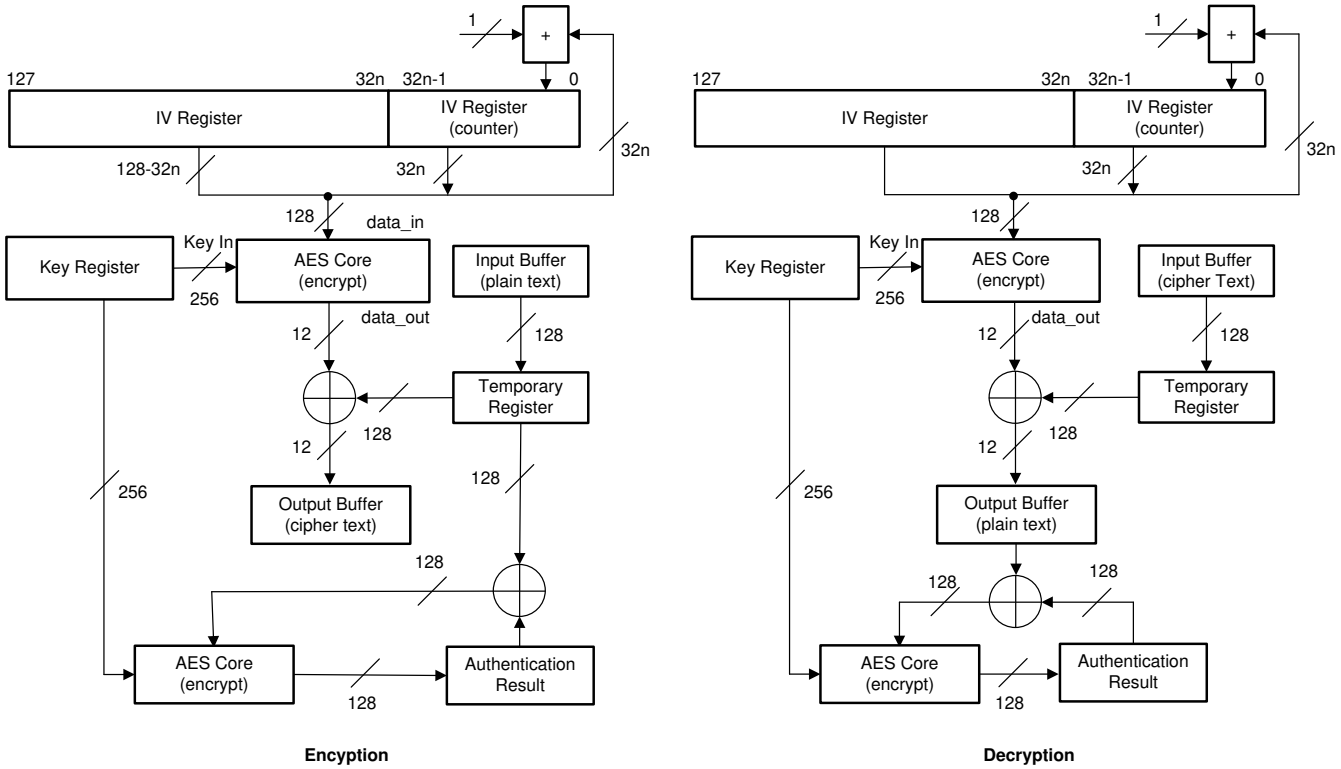


Figure 33-3. AES - CCM Operation

33.2.3 XTS Operation

Figure 33-4 shows the XTS mode of operation for encryption and decryption. The input to the cryptographic core is XORed with the IV; the output of the cryptographic core is XORed with the same IV. For decryption, the cryptographic core operates in reverse, but the XOR operations are the same.

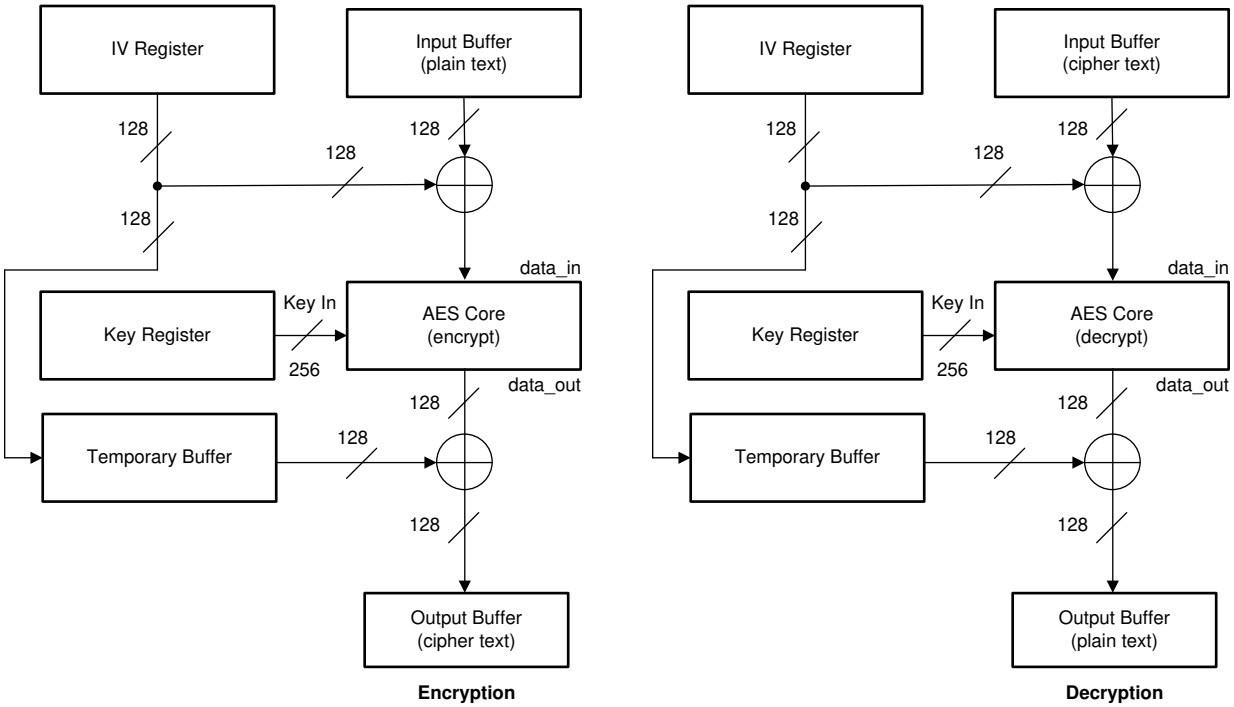


Figure 33-4. AES - XTS Operation

Note

The IV is created with an initial encryption, followed by an LFSR operation for each new block.

33.2.4 ECB Feedback Mode

Figure 33-5 shows the basic ECB feedback mode of operation, where the input data is passed directly to the basic cryptographic core and the output is passed directly to the output buffer. For decryption, the cryptographic core operates in reverse: the decryption data path is used for data processing, whereas encryption uses the encryption data path.

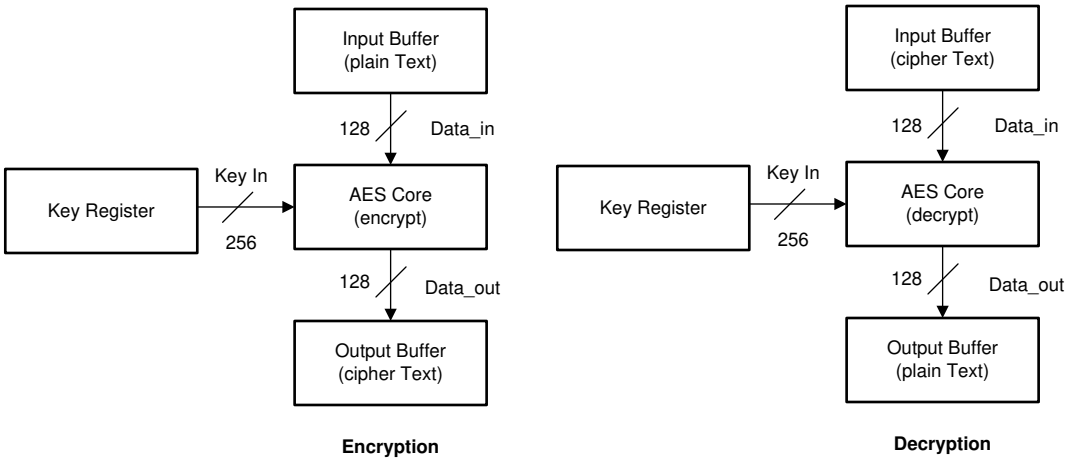


Figure 33-5. AES - ECB Feedback Mode

33.2.5 CBC Feedback Mode

Figure 33-6 shows the CBC feedback mode of operation, where the input data is XORed with the IV before it is passed to the basic cryptographic core. The output of the cryptographic core passes directly to the output buffer and becomes the next IV.

The operation is reversed for decryption, resulting in an XOR at the output of the cryptographic core. The input cipher text of the current operation is the IV for the next operation.

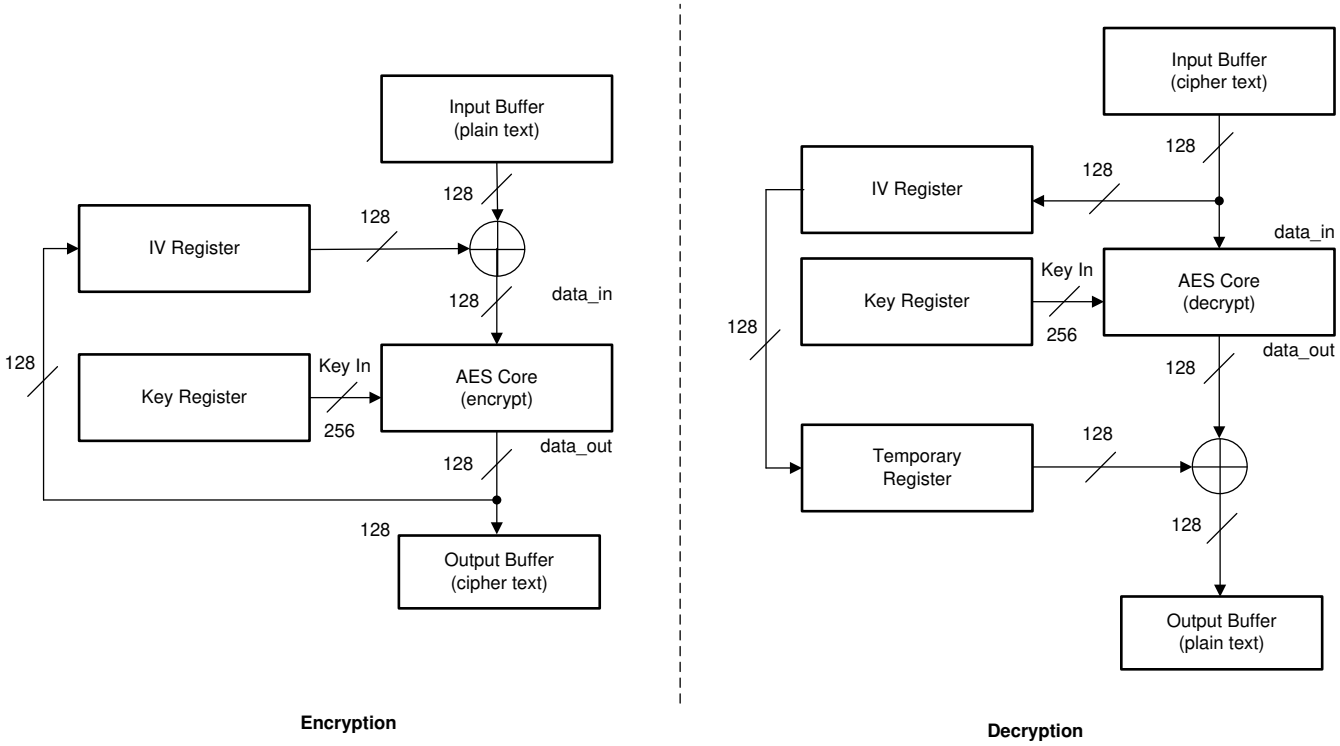


Figure 33-6. AES - CBC Feedback Mode

33.2.6 CTR and ICM Feedback Modes

Figure 33-7 shows the counter feedback (CTR/ICM) mode of operation. This operation encrypts the IV. The output of the cryptographic core (encrypted IV) is XORed with the data, thus creating the output result.

The IV is built out of two components: a fixed part and a counter part. The counter part is incremented with each block. The counter width is selectable per context and can be 16, 32, 64, 96, or 128 bits wide. In this mode, encryption and decryption use the same operation.

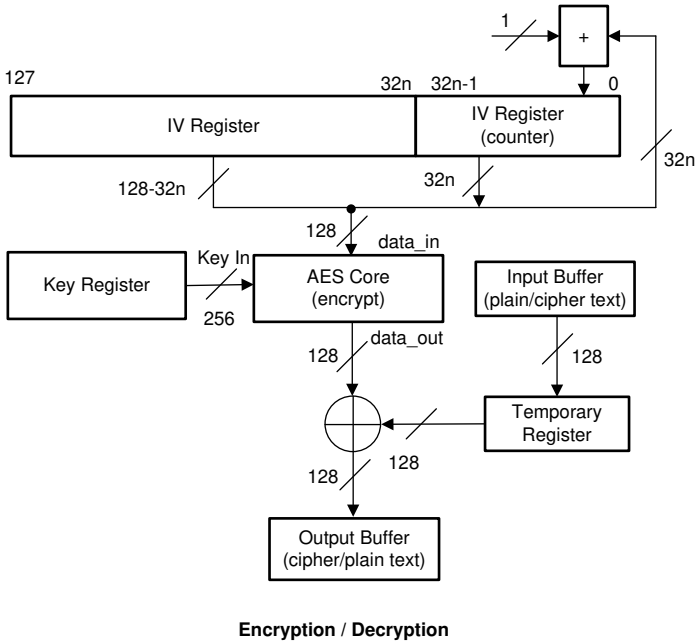


Figure 33-7. AES Encryption With CTR/ICM Mode

Note

The value for n can be 1, 2, 3, or 4 for CTR mode and is ½ for ICM mode.

33.2.7 CFB Mode

Figure 33-8 shows the full block (128 bits) CFB mode of operation for encryption and decryption. The input for the cryptographic core is the IV; the result is XORed with the data. The result is fed back through the IV register as the next input for the cryptographic core. The decryption operation is reversed, but the cryptographic core still performs encryption.

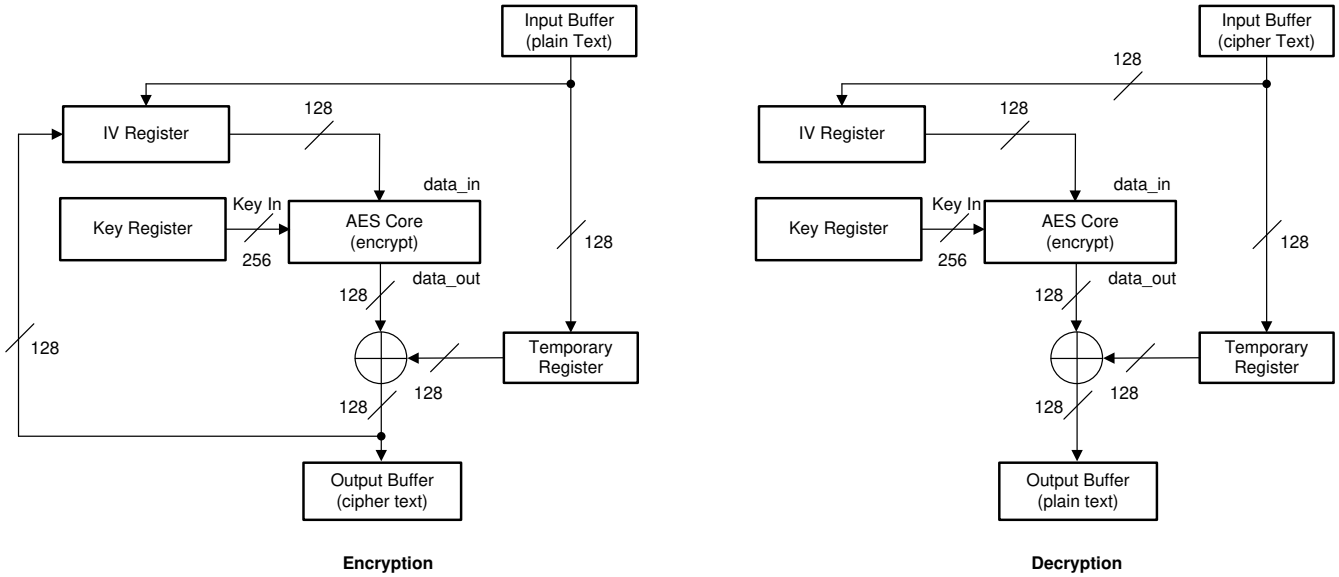


Figure 33-8. AES - CFB Feedback Mode

33.2.8 F8 Mode

Figure 33-9 shows the F8 feedback mode of operation for encryption and decryption. The input to the cryptographic core is the result of the XOR operation of the previous cryptographic core output, a constant IV, and a block counter. The output of the cryptographic core is XORed with the input to create the result. In this mode, encryption and decryption use the same operations.

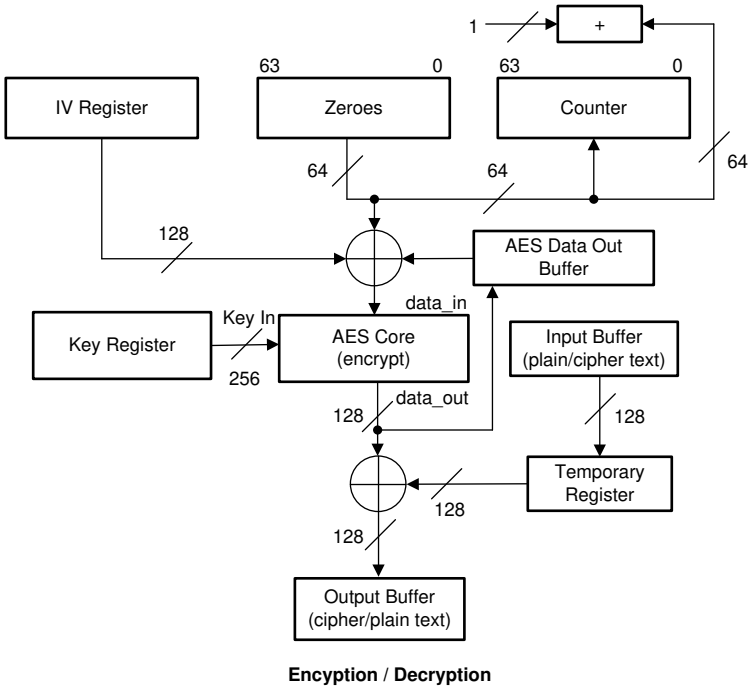


Figure 33-9. AES - F8 Mode

33.2.9 F9 Operation

Figure 33-10 shows the F9 authentication mode of operation, where the input to the cryptographic core is XORed with the IV, and the output is XORed with the previous result to create the next result. The cryptographic core output is fed back as IV for the next block. The result is the output of the last XOR operation of the cryptographic core output.

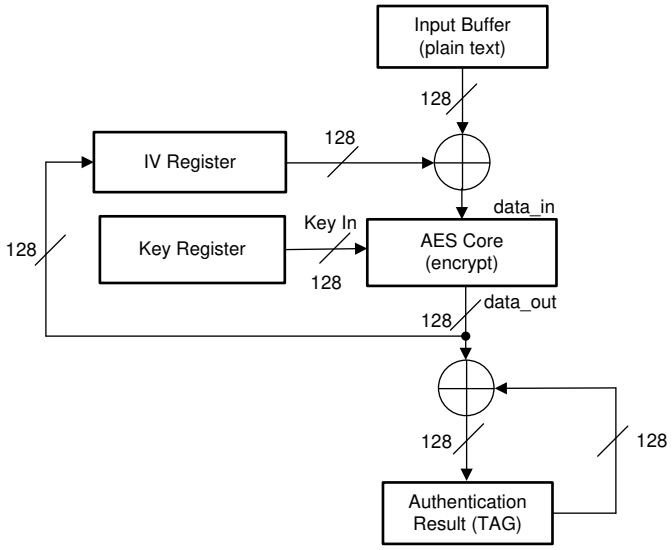


Figure 33-10. AES - F9 Operation

33.2.10 CBC-MAC Operation

Figure 33-11 shows the CBC-MAC authentication mode of operation, where the input to the cryptographic core is XORed with the IV. The cryptographic core output is then fed back as IV for the next block. The last data input block is XORed with an additional input value stored in the temporary buffer; this can be any precalculated value and is dependent on the alignment of the last input block. The result is the cryptographic core output of the last encryption operation.

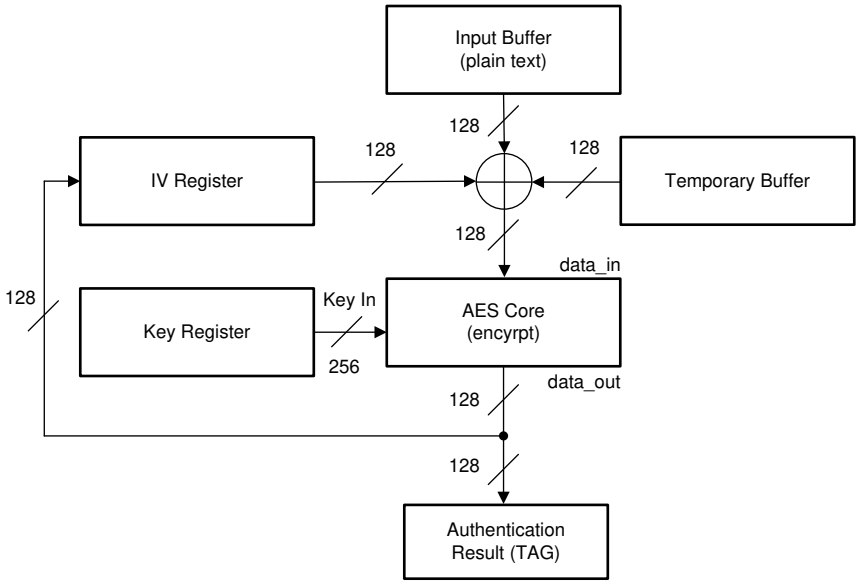


Figure 33-11. AES - CBC-MAC Authentication Mode

33.3 Extended and Combined Modes of Operations

This section describes the protocols (or autonomous precalculations) supported by the AES wide-bus engine.

33.3.1 GCM Protocol Operation

A GCM protocol operation is a combined operation consisting of encryption or decryption, and authentication. A part of the input data stream can be authenticated only, while normally most of the input data is encrypted or decrypted and authenticated. The authentication-only data must always be in front of the data requiring encryption. Within GCM, the authentication-only data is called the additional authentication data (AAD). The AAD is fetched independently of other data.

The intermediate (temporary) result data is used as input to the remaining authentication operation. Because the authentication operation does not require the cryptographic core but only the polynomial multiplication, encryption, decryption, and authentication can be performed in parallel. After encryption of the last data block, additional polynomial multiplication and encryption are required to authenticate a 128-bit-long vector and finally encrypt the authentication result.

33.3.2 CCM Protocol Operation

The CCM protocol operation is a combined operation consisting of encryption or decryption, and authentication. The authentication and encryption or decryption operations use the cryptographic core; these operations are executed sequentially on the AES core. A part of the data stream can require authentication only. The authentication-only data must always be in front of the data requiring encryption.

Authentication starts with the encryption of a predefined block B0. This block consists of flags, nonce, and message length. The next blocks contain the authentication data length concatenated with the authentication-only data. After processing the authentication-only data, the encryption or decryption operations are performed, each followed by the related authentication of the plaintext data block (which equals the input in the case of encryption, and the output in the case of decryption). The final authentication result must be encrypted using the output of the encryption of the IV block A0. This block contains the IV (consisting of flags and nonce) concatenated with the counter, which is zero for A0.

33.4 AES Module Programming Guide

33.4.1 AES Low-Level Programming Models

This section describes the low-level hardware programming sequences for configuring and using the AES module.

33.4.1.1 Global Initialization

The following list describes the requirements for initializing the AES and associated modules.

1. Configure the DMA Trigger Source corresponding to AES. For more information, refer to the DMA chapter.
2. Specify the size of the keys by programming the KEY_SIZE bit field in the AES_CTRL register.
3. Load the AES Key 1 (AES_KEY1_n) register.
4. Load the AES Key 2 (AES_KEY2_n) register if it is used by the configuration mode.
5. Configure the AES for the appropriate encryption or decryption mode (see Section 33.4.1.2).
6. Select encryption or decryption by programming the DIRECTION bit in the AES Control (AES_CTRL) register.

33.4.1.2 AES Operating Modes Configuration

The following sections list the initialization subsequences for the available encryption and decryption modes:

Subsequence: Initialize CCM AES Core Mode

The steps to initialize CCM mode follow:

1. Define the width of the length field and the length of the authentication field by programming the CCM_L and CCM_M bit fields in the AES_CTRL register.
2. Enable counter mode by setting the CTR bit in the AES_CTRL register.
3. Load the authentication data length in the AUTH field of the AES Authentication Data Length (AES_AUTH_LENGTH) register.
4. Select the IV counter by programming the CTR_WIDTH field in the AES_CTRL register.
5. Load the AES Initialization Vector Input n (AES_IV_IN_n) registers.

Subsequence: Initialize GCM AES Core Mode

The steps to enable GCM mode follow:

1. Enable counter mode by setting the CTR bit in the AES_CTRL register.
2. Load the authentication data length in the AUTH field of the AES Authentication Data Length (AES_AUTH_LENGTH) register.
3. Select the IV counter by programming the CTR_WIDTH field in the AES_CTRL register.
4. Load the AES Initialization Vector Input n (AES_IV_IN_n) registers.

Subsequence: Initialize CBC-MAC AES Core Mode

The steps to initialize CBC-MAC mode follow:
1. Enable CBC-MAC mode by setting the CBCMAC bit in the AES_CTRL register.
2. Select encryption mode by setting the DIRECTION bit in the AES_CTRL register.
3. Load the AES Initialization Vector Input n (AES_IV_IN_n) registers.

Subsequence: Initialize F9 AES Core Mode

The steps to configure the AES for F9 mode follow:
1. Enable F9 mode by setting the F9 bit in the AES_CTRL register.
2. Set the key size to 128 bits by programming the KEY_SIZE field to 0x1 in the AES_CTRL register.
3. Load the AES Initialization Vector Input n (AES_IV_IN_n) registers.

Subsequence: Initialize F8 AES Core Mode

The steps to configure the AES for F8 mode follow:
1. Enable F8 mode by setting the F8 bit in the AES_CTRL register.
2. Select the counter width by programming the CTR_WIDTH field in the AES_CTRL register.
3. Set the key size to 128 bits by setting the KEY_SIZE field to 0x1 in the AES_CTRL register.
4. Load the AES Initialization Vector Input n (AES_IV_IN_n) registers.

Subsequence: Initialize XTS AES Core Mode

The steps to configure XTS mode follow:
1. Enable XTS mode by configuring the XTS field in the AES_CTRL register.
2. If the XTS field in the AES_CTRL register indicates that the AAD length is required, load the AAD length in the AES_AUTH_LENGTH register.
3. Load the AES Initialization Vector Input n (AES_IV_IN_n) registers.

Subsequence: Initialize CFB AES Core Mode

The steps to initialize the AES code for CFB mode follow:
1. Enable CFB mode by setting the CFB bit in the AES_CTRL register.
2. Load the AES Initialization Vector Input n (AES_IV_IN_n) registers.

Subsequence: Initialize ICM AES Core Mode

The steps to initialize the AES code for ICM mode follow:
1. Enable ICM mode by setting the ICM bit in the AES_CTRL register.
2. Configure for a 16-bit counter by programming the CTR_WIDTH field to 0x0 in the AES_CTRL register.
3. Load the AES Initialization Vector Input n (AES_IV_IN_n) registers.

Subsequence: Initialize CTR AES Core Mode

The steps to initialize CTR mode follow:
1. Enable CTR mode by setting the CTR bit in the AES_CTRL register.
2. Select counter width by programming the CTR_WIDTH in the AES_CTRL register.
3. Load the AES Initialization Vector Input n (AES_IV_IN_n) registers.

Subsequence: Initialize CBC AES Core Mode

The steps to configure CBC mode follow:
1. Enable CBC mode by setting the MODE bit in the AES_CTRL register.
2. Load the AES Initialization Vector Input n (AES_IV_IN_n) registers.

33.4.1.3 AES Mode Configurations

AES Polling Mode

Main Sequence: AES Polling Mode – Figure 33-12 shows AES polling mode. The registers used in AES polling mode follow:

- AES Data RW Plaintext/Ciphertext 0 (AES_DATA_IN_OUT_0) registers
- AES Control (AES_CTRL) register
- AES Hash Tag Out 0 (AES_TAG_OUT_0) register

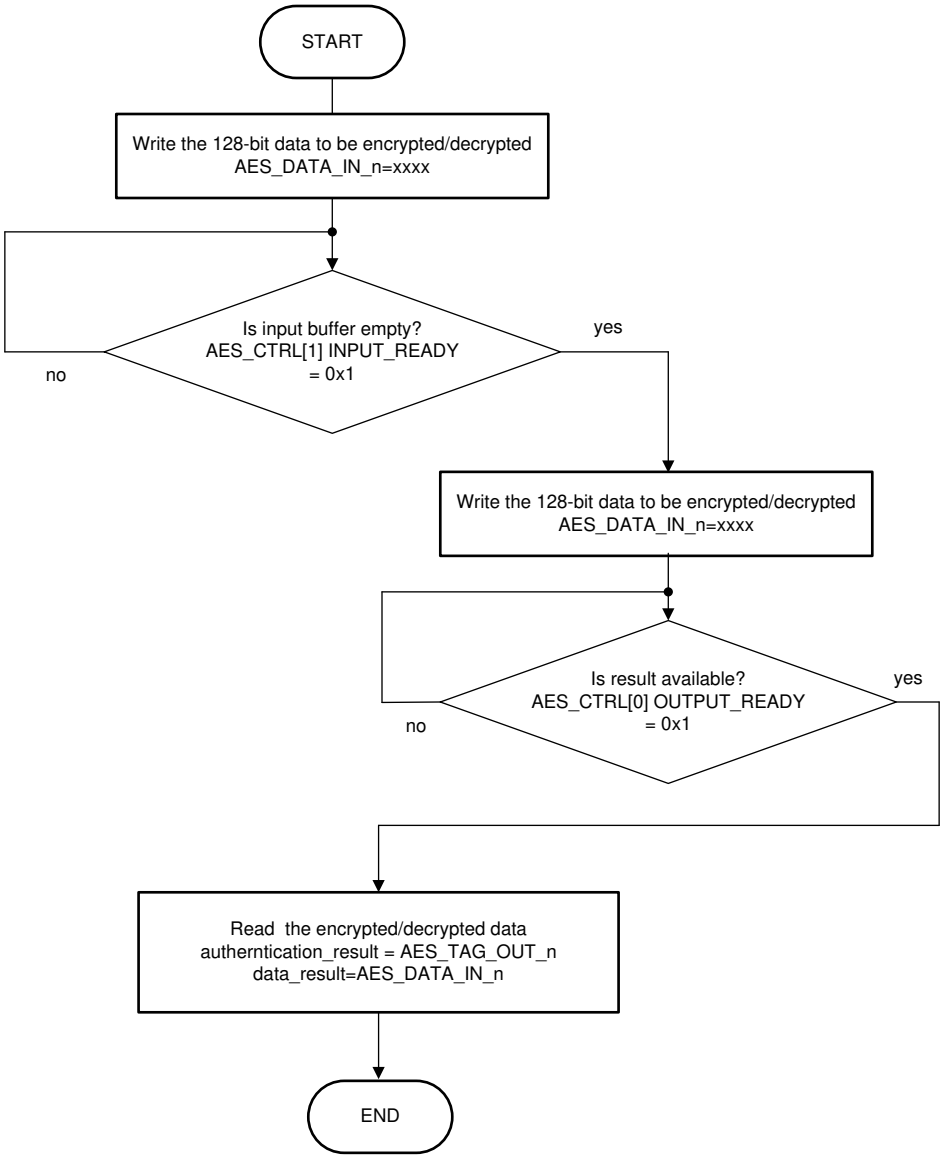


Figure 33-12. AES Polling Mode

AES Interrupt Mode

The application can use software interrupts to control the flow of Context In, Context Out, Data In, and Data Out requests. To enable these interrupts

1. First, initialize the device by following the initialization sequences described in Section 33.4.1.1 and Section 33.4.1.2.
2. When the device has been initialized, the application can enable the AES module interrupts through the AES Interrupt Enable (AES_IRQENABLE) register.
3. Load the input buffers, AES_DATA_IN_OUT_n, with data.

Note

If the application uses interrupt mode, an interrupt is generated for each block of processed data. To support larger data flow, DMA access mode should be used and the bits in the AES_IRQENABLE register should be cleared.

AES DMA Mode

When AES DMA mode is enabled, the AES_IRQENABLE register should be cleared. To enable the DMA to transfer data, follow these steps:

1. When the AES module is initialized, configure DMA sources and channels. Refer to the DMA chapter for the associated AES sources.
2. Configure the corresponding DMA enable and request bits in the AES System Configuration (AES_SYSCONFIG) register.

The input buffer registers, AES_DATA_IN_OUT_n, are now loaded.

33.4.1.4 AES Events Servicing

Interrupt Servicing

This section describes the event servicing of the module. Figure 33-13 shows the AES interrupt service. The registers used during event servicing follow:

- AES_IRQSTATUS
- AES_KEY1_n
- AES_KEY2_n
- AES_IV_IN_n
- AES_DATA_IN_OUT_n
- AES_TAG_OUT_n
- AES_IRQENABLE

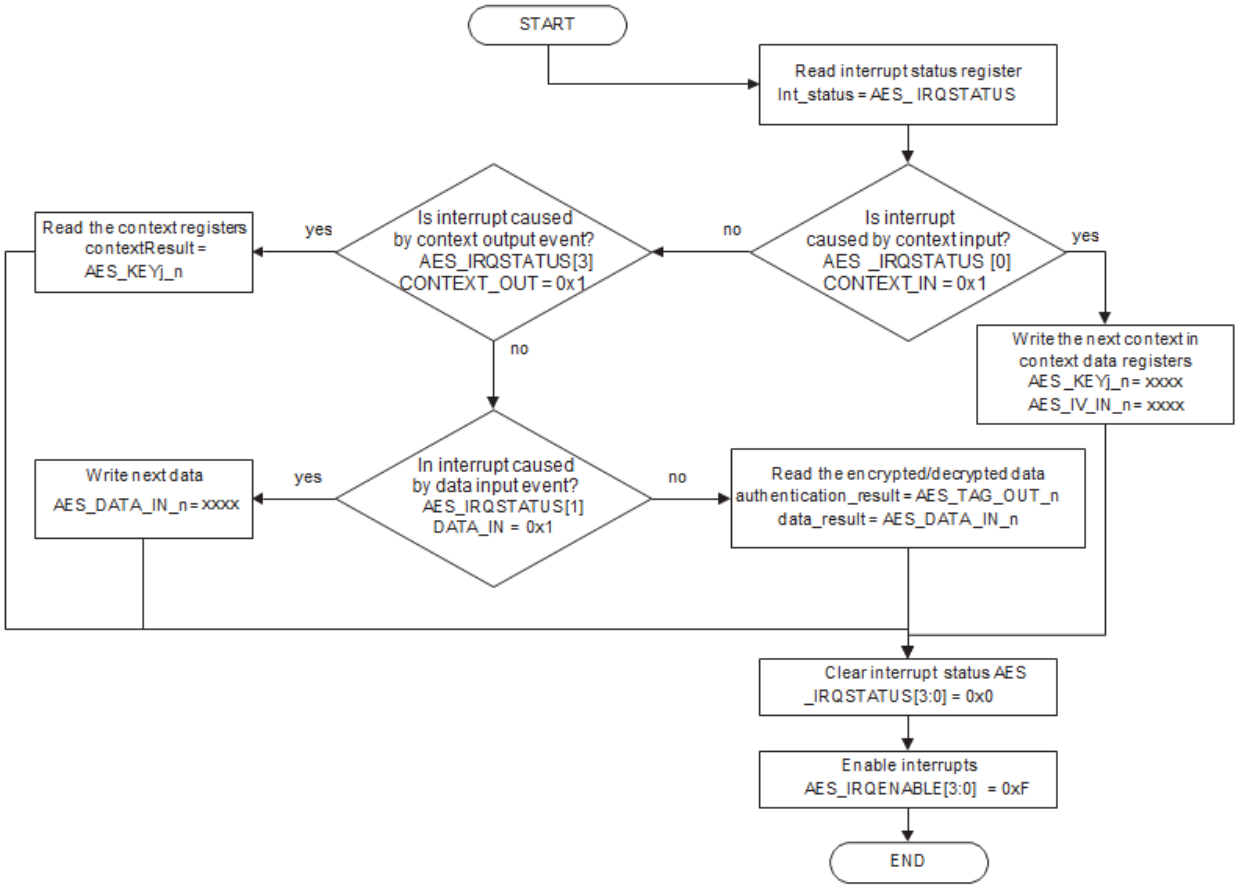


Figure 33-13. AES Interrupt Service

33.5 Software

33.5.1 AES Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/aes

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

33.5.1.1 AES ECB Encryption Example (CM)

FILE: aes_ex1_ecb_encrypt.c

This example encrypts block cipher-text using AES128 in ECB mode. It does the encryption first without uDMA and then with uDMA. The results are checked after each operation.

External Connections
- None

Watch Variables
- *errCountGlobal* - Error Counter. It should be zero.
- *testStatusGlobal* - Test status. It should be equal to PASS.

33.5.1.2 AES ECB De-cryption Example (CM)

FILE: aes_ex2_ecb_decrypt.c

This example de-crypts block cipher-text using AES128 in ECB mode. It does the de-cryption first without uDMA and then with uDMA. The results are checked after each operation.

External Connections
- None

Watch Variables
- *errCountGlobal* - Error Counter. It should be zero.
- *testStatusGlobal* - Test status. It should be equal to PASS.

33.5.1.3 AES GCM Encryption Example (CM)

FILE: aes_ex3_gcm_encrypt.c

This example encrypts block cipher-text using AES128 in GCM mode. It does the encryption first without uDMA and then with uDMA. The results are checked after each operation.

External Connections
- None

Watch Variables
- *errorCountGlobal* - Error Counter. It should be zero.
- *testStatusGlobal* - Test status. It should be equal to PASS.

33.5.1.4 AES GCM Decryption Example (CM)

FILE: aes_ex4_gcm_decrypt.c

This example decrypts block cipher-text using AES128 in GCM mode. It does the decryption first without uDMA and then with uDMA. The results are checked after each operation.

External Connections
- None

Watch Variables
- *errorCountGlobal* - Error Counter. It should be zero.
- *testStatusGlobal* - Test status. It should be equal to PASS.

33.6 AES Registers

The following sections are register descriptions for the AES Subsystem and AES Peripheral Core.

33.6.1 AES Base Address Table

Table 33-3. AES Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
AesaRegs	AES_IP_REGS	AESA_BASE	0x0004_2000	YES	YES	-	-	-
AesaSsRegs	AES_WRAPPER_REGS	AESA_SS_BASE	0x0004_2C00	YES	YES	-	-	-

33.6.2 AES_REGS Registers

Table 33-4 lists the memory-mapped registers for the AES_REGS registers. All register offset addresses not listed in Table 33-4 should be considered as reserved locations and the register contents should not be modified.

Table 33-4. AES_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	AES_KEY2_6	XTS Second Key or CBC-MAC Third Key		Go
4h	AES_KEY2_7	XTS Second Key or CBC-MAC Third Key		Go
8h	AES_KEY2_4	XTS/CCM Second Key or CBC-MAC Third Key		Go
Ch	AES_KEY2_5	XTS Second Key or CBC-MAC Third Key		Go
10h	AES_KEY2_2	XTS/CCM/CBC-MAC Second Key or Hash Key Input		Go
14h	AES_KEY2_3	XTS/CCM/CBC-MAC Second Key or Hash Key Input		Go
18h	AES_KEY2_0	XTS/CCM/CBC-MAC Second Key or Hash Key Input		Go
1Ch	AES_KEY2_1	XTS/CCM/CBC-MAC Second Key or Hash Key Input		Go
20h	AES_KEY1_6	Key		Go
24h	AES_KEY1_7	Key		Go
28h	AES_KEY1_4	Key		Go
2Ch	AES_KEY1_5	Key		Go
30h	AES_KEY1_2	Key		Go
34h	AES_KEY1_3	Key		Go
38h	AES_KEY1_0	Key		Go
3Ch	AES_KEY1_1	Key		Go
40h	AES_IV_IN_OUT_0	Initialization Vector 0		Go
44h	AES_IV_IN_OUT_1	Initialization Vector 1		Go
48h	AES_IV_IN_OUT_2	Initialization Vector 2		Go
4Ch	AES_IV_IN_OUT_3	Initialization Vector 3		Go
50h	AES_CTRL	Input/Output Buffer Control and Mode Selection		Go
54h	AES_C_LENGTH_0	Crypto Data Length 0		Go
58h	AES_C_LENGTH_1	Crypto Data Length 1		Go
5Ch	AES_AUTH_LENGTH	AAD Data Length		Go
60h	AES_DATA_IN_OUT_0	Data Word 0		Go
64h	AES_DATA_IN_OUT_1	Data Word 1		Go
68h	AES_DATA_IN_OUT_2	Data Word 2		Go
6Ch	AES_DATA_IN_OUT_3	Data Word 3		Go
70h	AES_TAG_OUT_0	Hash Result 0		Go
74h	AES_TAG_OUT_1	Hash Result 1		Go
78h	AES_TAG_OUT_2	Hash Result 2		Go
7Ch	AES_TAG_OUT_3	Hash Result 3		Go
80h	AES_REV	Module Revision Number		Go
84h	AES_SYSCONFIG	System Configuration		Go
88h	AES_SYSSTATUS	Reset Status		Go
8Ch	AES_IRQSTATUS	Interrupt Status		Go
90h	AES_IRQENABLE	Interrupt Enable		Go
94h	AES_DIRTY_BITS	Accessed / Dirty Bits		Go

Complex bit access types are encoded to fit into small table cells. Table 33-5 shows the codes that are used for access types in this section.

Table 33-5. AES_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1S	W 1S	Write 1 to set
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

33.6.2.1 AES_KEY2_6 Register (Offset = 0h) [Reset = 0h]

AES_KEY2_6 is shown in Figure 33-14 and described in Table 33-6.

Return to the Summary Table.

Secure Host Interface Bank

Figure 33-14. AES_KEY2_6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 33-6. AES_KEY2_6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	Key Data This register contains the 32-bit key data for the AES module. Initial key for XTS operations For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. The key size equals the AES_KEY1 size. For a Host write operation, these registers must be written with the new values to be subsequently transferred to the AES Engine. OR Pre-calculated CBC-MAC third key (K3), used to perform a final XOR operation on the last input data block. OR This register is used to store intermediate values and must be initialized with zeroes when writing a new GCM context. OR Used in f8/f9 algorithm Reset type: PER.RESET

33.6.2.2 AES_KEY2_7 Register (Offset = 4h) [Reset = 0h]

AES_KEY2_7 is shown in Figure 33-15 and described in Table 33-7.

Return to the Summary Table.

Secure Host Interface Bank MSW for CBC-MAC and 256-bit XTS

Figure 33-15. AES_KEY2_7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 33-7. AES_KEY2_7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	Key Data This register contains the 32-bit key data for the AES module. Initial key for XTS operations For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. The key size equals the AES_KEY1 size. For a Host write operation, these registers must be written with the new values to be subsequently transferred to the AES Engine. OR Pre-calculated CBC-MAC third key (K3), used to perform a final XOR operation on the last input data block. Reset type: PER.RESET

33.6.2.3 AES_KEY2_4 Register (Offset = 8h) [Reset = 0h]

AES_KEY2_4 is shown in Figure 33-16 and described in Table 33-8.

Return to the Summary Table.

Secure Host Interface Bank LSW for CBC-MAC

Figure 33-16. AES_KEY2_4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 33-8. AES_KEY2_4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	Key Data This register contains the 32-bit key data for the AES module. Initial key for XTS operations For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. The key size equals the AES_KEY1 size. For a Host write operation, these registers must be written with the new values to be subsequently transferred to the AES Engine. OR Pre-calculated CBC-MAC third key (K3), used to perform a final XOR operation on the last input data block. Reset type: PER.RESET

33.6.2.4 AES_KEY2_5 Register (Offset = Ch) [Reset = 0h]

AES_KEY2_5 is shown in Figure 33-17 and described in Table 33-9.

Return to the Summary Table.

Secure Host Interface Bank MSW for 192-bit XTS

Figure 33-17. AES_KEY2_5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 33-9. AES_KEY2_5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	Key Data This register contains the 32-bit key data for the AES module. Initial key for XTS operations For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. The key size equals the AES_KEY1 size. For a Host write operation, these registers must be written with the new values to be subsequently transferred to the AES Engine. OR Pre-calculated CBC-MAC third key (K3), used to perform a final XOR operation on the last input data block. Reset type: PER.RESET

33.6.2.5 AES_KEY2_2 Register (Offset = 10h) [Reset = 0h]

AES_KEY2_2 is shown in Figure 33-18 and described in Table 33-10.

Return to the Summary Table.

Secure Host Interface Bank

Figure 33-18. AES_KEY2_2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 33-10. AES_KEY2_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	Key Data This register contains the 32-bit key data for the AES module. Initial key for XTS operations For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. The key size equals the AES_KEY1 size. For a Host write operation, these registers must be written with the new values to be subsequently transferred to the AES Engine. OR Pre-calculated CBC-MAC second key (K2), used to perform a final XOR operation on the last input data block. OR Hash key, can be calculated internal or written via these registers. Only used for GHASH (GCM) modes. For a Host write operation, these registers must be written with the new values to be subsequently transferred to the AES Engine. OR These 128-bits are used to store the CKKM / IKKM value for f8 OR These 128-bits are used to store the CKKM / IKKM value for f9 Reset type: PER.RESET

33.6.2.6 AES_KEY2_3 Register (Offset = 14h) [Reset = 0h]

AES_KEY2_3 is shown in Figure 33-19 and described in Table 33-11.

Return to the Summary Table.

Secure Host Interface Bank MSW for CCM, CBC-MAC, Hash Key, and 128-bit XTS

Figure 33-19. AES_KEY2_3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 33-11. AES_KEY2_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	Key Data This register contains the 32-bit key data for the AES module. Initial key for XTS operations For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. The key size equals the AES_KEY1 size. For a Host write operation, these registers must be written with the new values to be subsequently transferred to the AES Engine. OR Pre-calculated CBC-MAC second key (K2), used to perform a final XOR operation on the last input data block. OR Hash key, can be calculated internal or written via these registers. Only used for GHASH (GCM) modes. For a Host write operation, these registers must be written with the new values to be subsequently transferred to the AES Engine. OR These 128-bits are used to store the CKKM / IKKM value for f8 OR These 128-bits are used to store the CKKM / IKKM value for f9 Reset type: PER.RESET

33.6.2.7 AES_KEY2_0 Register (Offset = 18h) [Reset = 0h]

AES_KEY2_0 is shown in Figure 33-20 and described in Table 33-12.

Return to the Summary Table.

Secure Host Interface Bank LSW for XTS, CCM, CBC-MAC, and Hash Key

Figure 33-20. AES_KEY2_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 33-12. AES_KEY2_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	Key Data This register contains the 32-bit key data for the AES module. Initial key for XTS operations For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. The key size equals the AES_KEY1 size. For a Host write operation, these registers must be written with the new values to be subsequently transferred to the AES Engine. OR Pre-calculated CBC-MAC second key (K2), used to perform a final XOR operation on the last input data block. OR Hash key, can be calculated internal or written via these registers. Only used for GHASH (GCM) modes. For a Host write operation, these registers must be written with the new values to be subsequently transferred to the AES Engine. OR These 128-bits are used to store the CKKM / IKKM value for f8 OR These 128-bits are used to store the CKKM / IKKM value for f9 Reset type: PER.RESET

33.6.2.8 AES_KEY2_1 Register (Offset = 1Ch) [Reset = 0h]

AES_KEY2_1 is shown in Figure 33-21 and described in Table 33-13.

Return to the Summary Table.

Secure Host Interface Bank

Figure 33-21. AES_KEY2_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 33-13. AES_KEY2_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	Key Data This register contains the 32-bit key data for the AES module. Initial key for XTS operations For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. The key size equals the AES_KEY1 size. For a Host write operation, these registers must be written with the new values to be subsequently transferred to the AES Engine. OR Pre-calculated CBC-MAC second key (K2), used to perform a final XOR operation on the last input data block. OR Hash key, can be calculated internal or written via these registers. Only used for GHASH (GCM) modes. For a Host write operation, these registers must be written with the new values to be subsequently transferred to the AES Engine. OR These 128-bits are used to store the CKKM / IKKM value for f8 OR These 128-bits are used to store the CKKM / IKKM value for f9 Reset type: PER.RESET

33.6.2.9 AES_KEY1_6 Register (Offset = 20h) [Reset = 0h]

AES_KEY1_6 is shown in Figure 33-22 and described in Table 33-14.

Return to the Summary Table.

Secure Host Interface Bank

Figure 33-22. AES_KEY1_6 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 33-14. AES_KEY1_6 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	AES Key register. For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. For a Host read operation, these registers return all-zeroes. The Host will typically write these registers in order, beginning with the AES_KEY_0 register. The key size (see the AES_MODE register) determines which key registers need to be populated, as indicated in the table below. Reset type: PER.RESET

33.6.2.10 AES_KEY1_7 Register (Offset = 24h) [Reset = 0h]

AES_KEY1_7 is shown in Figure 33-23 and described in Table 33-15.

Return to the Summary Table.

Secure Host Interface Bank MSW for 256-bit key

Figure 33-23. AES_KEY1_7 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 33-15. AES_KEY1_7 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	AES Key register. For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. For a Host read operation, these registers return all-zeroes. The Host will typically write these registers in order, beginning with the AES_KEY_0 register. The key size (see the AES_MODE register) determines which key registers need to be populated, as indicated in the table below. Reset type: PER.RESET

33.6.2.11 AES_KEY1_4 Register (Offset = 28h) [Reset = 0h]

AES_KEY1_4 is shown in Figure 33-24 and described in Table 33-16.

Return to the Summary Table.

Secure Host Interface Bank

Figure 33-24. AES_KEY1_4 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 33-16. AES_KEY1_4 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	AES Key register. For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. For a Host read operation, these registers return all-zeroes. The Host will typically write these registers in order, beginning with the AES_KEY_0 register. The key size (see the AES_MODE register) determines which key registers need to be populated, as indicated in the table below. Reset type: PER.RESET

33.6.2.12 AES_KEY1_5 Register (Offset = 2Ch) [Reset = 0h]

AES_KEY1_5 is shown in Figure 33-25 and described in Table 33-17.

Return to the Summary Table.

Secure Host Interface Bank MSW for 192-bit key

Figure 33-25. AES_KEY1_5 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 33-17. AES_KEY1_5 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	AES Key register. For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. For a Host read operation, these registers return all-zeroes. The Host will typically write these registers in order, beginning with the AES_KEY_0 register. The key size (see the AES_MODE register) determines which key registers need to be populated, as indicated in the table below. Reset type: PER.RESET

33.6.2.13 AES_KEY1_2 Register (Offset = 30h) [Reset = 0h]

AES_KEY1_2 is shown in Figure 33-26 and described in Table 33-18.

Return to the Summary Table.

Secure Host Interface Bank

Figure 33-26. AES_KEY1_2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 33-18. AES_KEY1_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	AES Key register. For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. For a Host read operation, these registers return all-zeroes. The Host will typically write these registers in order, beginning with the AES_KEY_0 register. The key size (see the AES_MODE register) determines which key registers need to be populated, as indicated in the table below. Reset type: PER.RESET

33.6.2.14 AES_KEY1_3 Register (Offset = 34h) [Reset = 0h]

AES_KEY1_3 is shown in Figure 33-27 and described in Table 33-19.

Return to the Summary Table.

Secure Host Interface Bank MSW for 128-bit key

Figure 33-27. AES_KEY1_3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 33-19. AES_KEY1_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	AES Key register. For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. For a Host read operation, these registers return all-zeroes. The Host will typically write these registers in order, beginning with the AES_KEY_0 register. The key size (see the AES_MODE register) determines which key registers need to be populated, as indicated in the table below. Reset type: PER.RESET

33.6.2.15 AES_KEY1_0 Register (Offset = 38h) [Reset = 0h]

AES_KEY1_0 is shown in Figure 33-28 and described in Table 33-20.

Return to the Summary Table.

Secure Host Interface Bank LSW

Figure 33-28. AES_KEY1_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 33-20. AES_KEY1_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	AES Key register. For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. For a Host read operation, these registers return all-zeroes. The Host will typically write these registers in order, beginning with the AES_KEY_0 register. The key size (see the AES_MODE register) determines which key registers need to be populated, as indicated in the table below. Reset type: PER.RESET

33.6.2.16 AES_KEY1_1 Register (Offset = 3Ch) [Reset = 0h]

AES_KEY1_1 is shown in Figure 33-29 and described in Table 33-21.

Return to the Summary Table.

Secure Host Interface Bank

Figure 33-29. AES_KEY1_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KEY																															
R-0/W-0h																															

Table 33-21. AES_KEY1_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	KEY	R-0/W	0h	AES Key register. For a Host write operation, these registers must be written with the 128, 192 or 256-bit key for a subsequent AES operation. For a Host read operation, these registers return all-zeroes. The Host will typically write these registers in order, beginning with the AES_KEY_0 register. The key size (see the AES_MODE register) determines which key registers need to be populated, as indicated in the table below. Reset type: PER.RESET

33.6.2.17 AES_IV_IN_OUT_0 Register (Offset = 40h) [Reset = 0h]

AES_IV_IN_OUT_0 is shown in Figure 33-30 and described in Table 33-22.

Return to the Summary Table.

Secure Host Interface Bank LSW

Figure 33-30. AES_IV_IN_OUT_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R/W-0h																															

Table 33-22. AES_IV_IN_OUT_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R/W	0h	For a Host write operation, these registers must be written with the new 128-bit IV to be subsequently transferred to the AES Engine. For a Host read operation, these registers contain the latest 128-bit IV output by the AES Engine. This value is incremented with 0x1, after first use when a new data block is submitted to the engine if CTR/ICM mode is selected. OR For XTS mode, this register must be written with the tweak location ("i") of the data unit or (intermediate) tweak value (Tj), For a Host read operation, the IV_OUT register holds the next tweak value that is used for the next data block provided via the DATA_IN registers. This value can be re-used for continuation with the next block. It must be provide together with a "j" of "0" via the IV_IN registers. OR For f8 this field must be written with zeroes. OR (In case of GCM) For a Host write operation, these registers must be written with the new 128-bit IV to be subsequently transferred to the AES Engine. For a Host read operation, these registers contain the latest 128-bit IV output by the AES Engine. Note that bits [127:96] of the IV represent the initial counter value (which is "1" for GCM) and must therefore be initialized to 0x01000000. This value is incremented with 0x1, after first use when a new data block is submitted to the engine. OR For CCM this field must be written with A0, this value constist of the concatenation of: A0-flags (5-bits of zero and 3-bits "L"), Nonce and counter value. "L" must be a copy from the "L" value of the AES_CTRL register. This "L" indicates the width of the Nonce and counter. The loaded counter must be initialized to zero. The total width of A0 is 128-bit. Reset type: PER.RESET

33.6.2.18 AES_IV_IN_OUT_1 Register (Offset = 44h) [Reset = 0h]

AES_IV_IN_OUT_1 is shown in Figure 33-31 and described in Table 33-23.

Return to the Summary Table.

Secure Host Interface Bank

Figure 33-31. AES_IV_IN_OUT_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R/W-0h																															

Table 33-23. AES_IV_IN_OUT_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R/W	0h	For a Host write operation, these registers must be written with the new 128-bit IV to be subsequently transferred to the AES Engine. For a Host read operation, these registers contain the latest 128-bit IV output by the AES Engine. This value is incremented with 0x1, after first use when a new data block is submitted to the engine if CTR/ICM mode is selected. OR For XTS mode, this register must be written with the tweak location ("i") of the data unit or (intermediate) tweak value (Tj), For a Host read operation, the IV_OUT register holds the next tweak value that is used for the next data block provided via the DATA_IN registers. This value can be re-used for continuation with the next block. It must be provide together with a "j" of "0" via the IV_IN registers. OR For f8 this field must be written with zeroes. OR (In case of GCM) For a Host write operation, these registers must be written with the new 128-bit IV to be subsequently transferred to the AES Engine. For a Host read operation, these registers contain the latest 128-bit IV output by the AES Engine. Note that bits [127:96] of the IV represent the initial counter value (which is "1" for GCM) and must therefore be initialized to 0x01000000. This value is incremented with 0x1, after first use when a new data block is submitted to the engine. OR For CCM this field must be written with A0, this value constist of the concatenation of: A0-flags (5-bits of zero and 3-bits "L"), Nonce and counter value. "L" must be a copy from the "L" value of the AES_CTRL register. This "L" indicates the width of the Nonce and counter. The loaded counter must be initialized to zero. The total width of A0 is 128-bit. Reset type: PER.RESET

33.6.2.19 AES_IV_IN_OUT_2 Register (Offset = 48h) [Reset = 0h]

AES_IV_IN_OUT_2 is shown in Figure 33-32 and described in Table 33-24.

Return to the Summary Table.

Secure Host Interface Bank

Figure 33-32. AES_IV_IN_OUT_2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R/W-0h																															

Table 33-24. AES_IV_IN_OUT_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R/W	0h	For a Host write operation, these registers must be written with the new 128-bit IV to be subsequently transferred to the AES Engine. For a Host read operation, these registers contain the latest 128-bit IV output by the AES Engine. This value is incremented with 0x1, after first use when a new data block is submitted to the engine if CTR/ICM mode is selected. OR For XTS mode, this register must be written with the tweak location ("i") of the data unit or (intermediate) tweak value (Tj), For a Host read operation, the IV_OUT register holds the next tweak value that is used for the next data block provided via the DATA_IN registers. This value can be re-used for continuation with the next block. It must be provide together with a "j" of "0" via the IV_IN registers. OR For f8 this field must be written with zeroes. OR (In case of GCM) For a Host write operation, these registers must be written with the new 128-bit IV to be subsequently transferred to the AES Engine. For a Host read operation, these registers contain the latest 128-bit IV output by the AES Engine. Note that bits [127:96] of the IV represent the initial counter value (which is "1" for GCM) and must therefore be initialized to 0x01000000. This value is incremented with 0x1, after first use when a new data block is submitted to the engine. OR For CCM this field must be written with A0, this value constist of the concatenation of: A0-flags (5-bits of zero and 3-bits "L"), Nonce and counter value. "L" must be a copy from the "L" value of the AES_CTRL register. This "L" indicates the width of the Nonce and counter. The loaded counter must be initialized to zero. The total width of A0 is 128-bit. Reset type: PER.RESET

33.6.2.20 AES_IV_IN_OUT_3 Register (Offset = 4Ch) [Reset = 0h]

AES_IV_IN_OUT_3 is shown in Figure 33-33 and described in Table 33-25.

Return to the Summary Table.

Secure Host Interface Bank MSW

Figure 33-33. AES_IV_IN_OUT_3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R/W-0h																															

Table 33-25. AES_IV_IN_OUT_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R/W	0h	For a Host write operation, these registers must be written with the new 128-bit IV to be subsequently transferred to the AES Engine. For a Host read operation, these registers contain the latest 128-bit IV output by the AES Engine. This value is incremented with 0x1, after first use when a new data block is submitted to the engine if CTR/ICM mode is selected. OR For XTS mode, this register must be written with the tweak location ("i") of the data unit or (intermediate) tweak value (Tj), For a Host read operation, the IV_OUT register holds the next tweak value that is used for the next data block provided via the DATA_IN registers. This value can be re-used for continuation with the next block. It must be provide together with a "j" of "0" via the IV_IN registers. OR For f8 this field must be written with zeroes. OR (In case of GCM) For a Host write operation, these registers must be written with the new 128-bit IV to be subsequently transferred to the AES Engine. For a Host read operation, these registers contain the latest 128-bit IV output by the AES Engine. Note that bits [127:96] of the IV represent the initial counter value (which is "1" for GCM) and must therefore be initialized to 0x01000000. This value is incremented with 0x1, after first use when a new data block is submitted to the engine. OR For CCM this field must be written with A0, this value constist of the concatenation of: A0-flags (5-bits of zero and 3-bits "L"), Nonce and counter value. "L" must be a copy from the "L" value of the AES_CTRL register. This "L" indicates the width of the Nonce and counter. The loaded counter must be initialized to zero. The total width of A0 is 128-bit. Reset type: PER.RESET

33.6.2.21 AES_CTRL Register (Offset = 50h) [Reset = 80000000h]

AES_CTRL is shown in Figure 33-34 and described in Table 33-26.

Return to the Summary Table.

Secure Host Interface Bank

Figure 33-34. AES_CTRL Register

31	30	29	28	27	26	25	24
CTXTRDY	SVCTXTRDY	SAVE_CONTEXT	RESERVED				CCM_M
R-1h	R-0h	R/W-0h	R-0h				R/W-0h
23	22	21	20	19	18	17	16
CCM_M			CCM_L		CCM	GCM	
R/W-0h			R/W-0h		R/W-0h	R/W-0h	
15	14	13	12	11	10	9	8
CBCMAC	F9	F8	XTS		CFB	ICM	CTR_WIDTH
R/W-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
CTR_WIDTH	CTR	MODE	KEY_SIZE		DIRECTION	INPUT_READY	OUTPUT_READY
R/W-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h	R-0h	R-0h

Table 33-26. AES_CTRL Register Field Descriptions

Bit	Field	Type	Reset	Description
31	CTXTRDY	R	1h	Context Data Registers Ready Value Description 0 The context data registers are not ready to be overwritten. 1 The context data registers can be overwritten and the host is permitted to write the next context. Reset type: PER.RESET
30	SVCTXTRDY	R	0h	AES TAG/IV Block(s) Ready This bit is only asserted if the SAVE_CONTEXT bit is set to 1. This bit is mutual exclusive with the CTXTRDY bit. Value Description 0 AES authentication TAG and/or IV block(s) is/are not available. 1 Indicates the AES authentication TAG and /or IV block(s) is/are available for the host to retrieve. Reset type: PER.RESET
29	SAVE_CONTEXT	R/W	0h	TAG or Result IV Save If this bit is set, the CONTEXT_OUT interrupt bit is set in the AES_IRQSTATUS register if the operation is finished and related signals are enabled. Value Description 0 No effect. 1 Indicates an authentication TAG of result IV needs to be stored as a result context. Reset type: PER.RESET
28-25	RESERVED	R	0h	Reserved

Table 33-26. AES_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
24-22	CCM_M	R/W	0h	Counter with CBC-MAC (CCM)Defines M which indicates the length of the authentication field for CCM operations the authentication field length equals two times the sum of CCM-M plus one. The AES Engine always returns a 128-bit authentication field, of which the M least significant bytes are valid. All values are supported. Reset type: PER.RESET
21-19	CCM_L	R/W	0h	Indicates the width of the length field for CCM operations the length field in bytes equals the value of CMM-L plus one. Supported values for L are: Value Description 0x0 width = 0 0x1 width = 2 0x2 reserved 0x3 width = 4 0x4 - 0x6 reserved 0x7 width = 8 Reset type: PER.RESET
18	CCM	R/W	0h	AES-CCM Mode Enable Value Description 0 AES-CCM mode is not enabled. 1 AES-CCM mode enabled. This is a combined mode, using AES for both authentication and encryption. No additional mode selection is required. Reset type: PER.RESET
17-16	GCM	R/W	0h	AES-GCM Mode Enable This is a combined mode, using the Galois field-multiplier GF(2^{128}) for authentication and AES-CTR mode for encryption the bits specify the GCM mode. Value Description 0x0 No operation 0x1 GHASH with H loaded and Y0-encrypted forced to zero 0x2 GHASH with H loaded and Y0-encrypted calculated internally 0x3 Autonomous GHASH (both H and Y0-encrypted calculated internally) Reset type: PER.RESET
15	CBCMAC	R/W	0h	AES-CBC MAC Enable The DIRECTION bit must be set to 1 for this mode. Value Description 0 AES-CBC MAC mode is not enabled. 1 AES-CBC MAC mode enabled. Reset type: PER.RESET
14	F9	R/W	0h	AES f9 Mode Enable The AES key size must be set to 128-bit for this mode. Value Description 0 f9 mode is not enabled 1 f9 mode is enabled. Reset type: PER.RESET
13	F8	R/W	0h	AES f8 Mode Enable, The KEY_SIZE must be set to 128-bit for this mode. Value Description 0 AES f8 mode is not enabled. 1 AES f8 mode is enabled. Reset type: PER.RESET

Table 33-26. AES_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
12-11	XTS	R/W	0h	AES-XTS Operation Enable The bits specify the XTS mode. Value Description 0x0 No operation 0x1 Previous/intermediate tweak value and j loaded (value is loaded via IV, j is loaded via the AAD length register) 0x2 Key2, n and j are loaded (n is loaded via IV, j is loaded via the AAD length register) 0x3 Key2 and n are loaded j=0 (n is loaded via IV) Reset type: PER.RESET
10	CFB	R/W	0h	Full block AES cipher feedback mode (CFB128) Enable Value Description 0 AES-CFB mode is not enabled. 1 AES-CFB mode is enabled. Reset type: PER.RESET
9	ICM	R/W	0h	AES Integer Counter Mode (ICM) Enable This is a counter mode with a 16-bit wide counter. Value Description 0 AES-ICM mode is not enabled. 1 AES-ICM mode is enabled. Reset type: PER.RESET
8-7	CTR_WIDTH	R/W	0h	AES-CTR Mode Counter Width Value Description 0x0 Counter is 32 bits 0x1 Counter is 64 bits 0x2 Counter is 96 bits 0x3 Counter is 128 bits Reset type: PER.RESET
6	CTR	R/W	0h	Counter Mode This bit must also be set for GCM and CCM mode, when encryption/decryption is required. Value Description 0 Counter mode is not enabled. 1 Counter mode is enabled. Reset type: PER.RESET
5	MODE	R/W	0h	ECB/CBC Mode Value Description 0 ECB mode 1 CBC mode Reset type: PER.RESET
4-3	KEY_SIZE	R/W	0h	Key Size Value Description 0x0 reserved 0x1 Key is 128 bits 0x2 Key is 192 bits 0x3 Key is 256 bits Reset type: PER.RESET
2	DIRECTION	R/W	0h	Encryption/Decryption Selection If set to =1, an encrypt operation is performed. If set to 0, a decrypt operation is performed. DIRECTION Value Description 0 Decryption is selected. 1 Encryption is selected. Reset type: PER.RESET

Table 33-26. AES_CTRL Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
1	INPUT_READY	R	0h	Input Ready Status Value Description 0 Input buffer is not empty. 1 Indicates that the 16-byte input buffer is empty, and the host is permitted to write the next block of data. Reset type: PER.RESET
0	OUTPUT_READY	R	0h	Output Ready Status Value Description 0 No AES output block is available. 1 An AES output block is available for the host to retrieve. Reset type: PER.RESET

33.6.2.22 AES_C_LENGTH_0 Register (Offset = 54h) [Reset = 0h]

AES_C_LENGTH_0 is shown in Figure 33-35 and described in Table 33-27.

Return to the Summary Table.

Secure Host Interface Bank LSW

Figure 33-35. AES_C_LENGTH_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LENGTH																															
R-0/W-0h																															

Table 33-27. AES_C_LENGTH_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	LENGTH	R-0/W	0h	Bits [60:0] of the crypto length registers (LSW and MSW) store the cryptographic data length in bytes for all modes. Once processing with this context is started, this length decrements to zero. Data lengths up to (2^{61} - 1) bytes are allowed. For GCM, any value up to 2^{36} - 32 bytes can be used. This is because a 32-bit counter mode is used the maximum number of 128-bit blocks is 2^{32} - 2, resulting in a maximum number of bytes of 2^{36} - 32. A write to this register triggers the engine to start using this context. This is valid for all modes except GCM and CCM. Note that for the combined modes, this length does not include the authentication only data the authentication length is specified in the AES_AUTH_LENGTH register below. All modes must have a length > 0. For the combined modes, it is allowed to have one of the lengths equal to zero. For the basic encryption modes (ECB/CBC/CTR/ICM/CFB128) it is allowed to program zero to the length field in that case the length is assumed infinite. All data must be byte (8-bit) aligned for stream cipher modes bit aligned data streams are not supported by the AES Engine. For block cipher modes, the data length must be programmed in multiples of the block cipher size, 16 bytes. For a Host read operation, these registers return all-zeroes. Reset type: PER.RESET

33.6.2.23 AES_C_LENGTH_1 Register (Offset = 58h) [Reset = 0h]

AES_C_LENGTH_1 is shown in Figure 33-36 and described in Table 33-28.

Return to the Summary Table.

Secure Host Interface Bank MSW

Figure 33-36. AES_C_LENGTH_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LENGTH																															
R-0/W-0h																															

Table 33-28. AES_C_LENGTH_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	LENGTH	R-0/W	0h	Bits [60:0] of the crypto length registers (LSW and MSW) store the cryptographic data length in bytes for all modes. Once processing with this context is started, this length decrements to zero. Data lengths up to (2^{61} - 1) bytes are allowed. For GCM, any value up to 2^{36} - 32 bytes can be used. This is because a 32-bit counter mode is used the maximum number of 128-bit blocks is 2^{32} - 2, resulting in a maximum number of bytes of 2^{36} - 32. A write to this register triggers the engine to start using this context. This is valid for all modes except GCM and CCM. Note that for the combined modes, this length does not include the authentication only data the authentication length is specified in the AES_AUTH_LENGTH register below. All modes must have a length > 0. For the combined modes, it is allowed to have one of the lengths equal to zero. For the basic encryption modes (ECB/CBC/CTR/ICM/CFB128) it is allowed to program zero to the length field in that case the length is assumed infinite. All data must be byte (8-bit) aligned for stream cipher modes bit aligned data streams are not supported by the AES Engine. For block cipher modes, the data length must be programmed in multiples of the block cipher size, 16 bytes. For a Host read operation, these registers return all-zeroes. Reset type: PER.RESET

33.6.2.24 AES_AUTH_LENGTH Register (Offset = 5Ch) [Reset = 0h]

AES_AUTH_LENGTH is shown in Figure 33-37 and described in Table 33-29.

Return to the Summary Table.

Secure Host Interface Bank

Figure 33-37. AES_AUTH_LENGTH Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
AUTH																															
R-0/W-0h																															

Table 33-29. AES_AUTH_LENGTH Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	AUTH	R-0/W	0h	Bits [31:0] of the authentication length register store the authentication data length in bytes for combined modes only (GCM or CCM) Supported AAD-lengths for CCM are from 0 to (2^{16} - 2^{8}) bytes. For GCM any value up to (2^{32} - 1) bytes can be used. Once processing with this context is started, this length decrements to zero. A write to this register triggers the engine to start using this context for GCM and CCM. For XTS this register is optionally used to load "j". Loading of "j" is only required if "j" != 0. "j" is a 28-bit value and must be written to bits [31-4] of this register. "j" represents the sequential number of the 128-bit block inside the data unit. For the first block in a unit, this value is zero. It is not required to provide a "j" for each new data block within a unit. Note that it is possible to start with a "j" unequal to zero refer to Table 4 for more details. For a Host read operation, these registers return all-zeroes. Reset type: PER.RESET

33.6.2.25 AES_DATA_IN_OUT_0 Register (Offset = 60h) [Reset = 0h]

AES_DATA_IN_OUT_0 is shown in Figure 33-38 and described in Table 33-30.

Return to the Summary Table.

Secure Host Interface Bank

Figure 33-38. AES_DATA_IN_OUT_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R/W-0h																															

Table 33-30. AES_DATA_IN_OUT_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R/W	0h	The Data Input/Output Registers buffer the input/output data blocks to/from the AES Engine. Notice that the data input buffer (AES_DATA_IN_n) and data output buffer (AES_DATA_OUT_n) are mapped to the same address locations. Writes to these addresses load the Input Buffer while reads pull from the Output Buffer. Therefore, for write access, the data input buffer is written for read access, the data output buffer is read. The data input buffer must be written prior to starting an operation. The data output buffer contains valid data on completion of an operation. All writes from, and reads to, these registers are tracked independently per direction and HIB. Therefore, any 128-bit data block can be split over multiple 32-bit word transfers, which can be mixed with other transfers over the external interface. Reset type: PER.RESET

33.6.2.26 AES_DATA_IN_OUT_1 Register (Offset = 64h) [Reset = 0h]

AES_DATA_IN_OUT_1 is shown in Figure 33-39 and described in Table 33-31.

Return to the Summary Table.

Secure Host Interface Bank

Figure 33-39. AES_DATA_IN_OUT_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R/W-0h																															

Table 33-31. AES_DATA_IN_OUT_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R/W	0h	The Data Input/Output Registers buffer the input/output data blocks to/from the AES Engine. Notice that the data input buffer (AES_DATA_IN_n) and data output buffer (AES_DATA_OUT_n) are mapped to the same address locations. Writes to these addresses load the Input Buffer while reads pull from the Output Buffer. Therefore, for write access, the data input buffer is written for read access, the data output buffer is read. The data input buffer must be written prior to starting an operation. The data output buffer contains valid data on completion of an operation. All writes from, and reads to, these registers are tracked independently per direction and HIB. Therefore, any 128-bit data block can be split over multiple 32-bit word transfers, which can be mixed with other transfers over the external interface. Reset type: PER.RESET

33.6.2.27 AES_DATA_IN_OUT_2 Register (Offset = 68h) [Reset = 0h]

AES_DATA_IN_OUT_2 is shown in Figure 33-40 and described in Table 33-32.

Return to the Summary Table.

Secure Host Interface Bank

Figure 33-40. AES_DATA_IN_OUT_2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R/W-0h																															

Table 33-32. AES_DATA_IN_OUT_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R/W	0h	The Data Input/Output Registers buffer the input/output data blocks to/from the AES Engine. Notice that the data input buffer (AES_DATA_IN_n) and data output buffer (AES_DATA_OUT_n) are mapped to the same address locations. Writes to these addresses load the Input Buffer while reads pull from the Output Buffer. Therefore, for write access, the data input buffer is written for read access, the data output buffer is read. The data input buffer must be written prior to starting an operation. The data output buffer contains valid data on completion of an operation. All writes from, and reads to, these registers are tracked independently per direction and HIB. Therefore, any 128-bit data block can be split over multiple 32-bit word transfers, which can be mixed with other transfers over the external interface. Reset type: PER.RESET

33.6.2.28 AES_DATA_IN_OUT_3 Register (Offset = 6Ch) [Reset = 0h]

AES_DATA_IN_OUT_3 is shown in Figure 33-41 and described in Table 33-33.

Return to the Summary Table.

Secure Host Interface Bank

Figure 33-41. AES_DATA_IN_OUT_3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA																															
R/W-0h																															

Table 33-33. AES_DATA_IN_OUT_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	DATA	R/W	0h	The Data Input/Output Registers buffer the input/output data blocks to/from the AES Engine. Notice that the data input buffer (AES_DATA_IN_n) and data output buffer (AES_DATA_OUT_n) are mapped to the same address locations. Writes to these addresses load the Input Buffer while reads pull from the Output Buffer. Therefore, for write access, the data input buffer is written for read access, the data output buffer is read. The data input buffer must be written prior to starting an operation. The data output buffer contains valid data on completion of an operation. All writes from, and reads to, these registers are tracked independently per direction and HIB. Therefore, any 128-bit data block can be split over multiple 32-bit word transfers, which can be mixed with other transfers over the external interface. Reset type: PER.RESET

33.6.2.29 AES_TAG_OUT_0 Register (Offset = 70h) [Reset = 0h]

AES_TAG_OUT_0 is shown in Figure 33-42 and described in Table 33-34.

Return to the Summary Table.

Secure Host Interface Bank

Figure 33-42. AES_TAG_OUT_0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HASH																															
R-0h																															

Table 33-34. AES_TAG_OUT_0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	HASH	R	0h	Bits [31:0] of the AES TAG registers store the authentication value for the combined and authentication only modes. For a Host read operation, these registers contain the last 128-bit TAG output of the AES Engine the TAG is available until the next context is written. This register will only contain valid data if the TAG is available, when the "store_ready" bit from AES_CTRL register is set. During processing or for operations/modes that do not return a TAG, reads from this register returns data from the IV register. For operations that do return a TAG in the IV register (e.g. XTS), the IV register must be accessed directly. Reset type: PER.RESET

33.6.2.30 AES_TAG_OUT_1 Register (Offset = 74h) [Reset = 0h]

AES_TAG_OUT_1 is shown in Figure 33-43 and described in Table 33-35.

Return to the Summary Table.

Secure Host Interface Bank

Figure 33-43. AES_TAG_OUT_1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HASH																															
R-0h																															

Table 33-35. AES_TAG_OUT_1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	HASH	R	0h	Bits [31:0] of the AES TAG registers store the authentication value for the combined and authentication only modes. For a Host read operation, these registers contain the last 128-bit TAG output of the AES Engine the TAG is available until the next context is written. This register will only contain valid data if the TAG is available, when the "store_ready" bit from AES_CTRL register is set. During processing or for operations/modes that do not return a TAG, reads from this register returns data from the IV register. For operations that do return a TAG in the IV register (e.g. XTS), the IV register must be accessed directly. Reset type: PER.RESET

33.6.2.31 AES_TAG_OUT_2 Register (Offset = 78h) [Reset = 0h]

AES_TAG_OUT_2 is shown in Figure 33-44 and described in Table 33-36.

Return to the Summary Table.

Secure Host Interface Bank

Figure 33-44. AES_TAG_OUT_2 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HASH																															
R-0h																															

Table 33-36. AES_TAG_OUT_2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	HASH	R	0h	Bits [31:0] of the AES TAG registers store the authentication value for the combined and authentication only modes. For a Host read operation, these registers contain the last 128-bit TAG output of the AES Engine the TAG is available until the next context is written. This register will only contain valid data if the TAG is available, when the "store_ready" bit from AES_CTRL register is set. During processing or for operations/modes that do not return a TAG, reads from this register returns data from the IV register. For operations that do return a TAG in the IV register (e.g. XTS), the IV register must be accessed directly. Reset type: PER.RESET

33.6.2.32 AES_TAG_OUT_3 Register (Offset = 7Ch) [Reset = 0h]

AES_TAG_OUT_3 is shown in Figure 33-45 and described in Table 33-37.

Return to the Summary Table.

Secure Host Interface Bank

Figure 33-45. AES_TAG_OUT_3 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
HASH																															
R-0h																															

Table 33-37. AES_TAG_OUT_3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	HASH	R	0h	Bits [31:0] of the AES TAG registers store the authentication value for the combined and authentication only modes. For a Host read operation, these registers contain the last 128-bit TAG output of the AES Engine the TAG is available until the next context is written. This register will only contain valid data if the TAG is available, when the "store_ready" bit from AES_CTRL register is set. During processing or for operations/modes that do not return a TAG, reads from this register returns data from the IV register. For operations that do return a TAG in the IV register (e.g. XTS), the IV register must be accessed directly. Reset type: PER.RESET

33.6.2.33 AES_REV Register (Offset = 80h) [Reset = 40000B02h]

AES_REV is shown in Figure 33-46 and described in Table 33-38.

Return to the Summary Table.

Secure Host Interface Bank

Figure 33-46. AES_REV Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
REVISION																															
R-40000B02h																															

Table 33-38. AES_REV Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	REVISION	R	40000B02h	Revision number: 31-30:SCHEME = 0b01 29-28:Reserved=0b00 27-16:FUNC = 0x000 15-11:RTL Revision = 0b00001 10-8:MAJOR = 0b011 7-2:CUSTOM = 0b000000 1-0:MINOR = 0b10 Reset type: PER.RESET

33.6.2.34 AES_SYSCONFIG Register (Offset = 84h) [Reset = 1h]

AES_SYSCONFIG is shown in Figure 33-47 and described in Table 33-39.

Return to the Summary Table.

Secure Host Interface Bank

Figure 33-47. AES_SYSCONFIG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED			RESERVED	RESERVED	RESERVED	MAP_CONTEXT_OUT_ON_DATA_OUT	DMA_REQ_CONTEXT_OUT_EN
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
DMA_REQ_CONTEXT_IN_EN	DMA_REQ_DATA_OUT_EN	DMA_REQ_DATA_IN_EN	RESERVED	SIDLE		SOFTRESET	AUTOIDLE
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h		R/W-0h	R/W-1h

Table 33-39. AES_SYSCONFIG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-13	RESERVED	R	0h	Reserved
12	RESERVED	R/W	0h	Reserved
11	RESERVED	R/W	0h	Reserved
10	RESERVED	R/W	0h	Reserved
9	MAP_CONTEXT_OUT_ON_DATA_OUT	R/W	0h	If set to '1' the two context out requests (dma_req_context_out_en, Bit [8] above, and context_out interrupt enable, Bit [3] of AES_IRQENABLE register) are mapped on the corresponding data output request bit. In this case, the original "context out" bit values are ignored. Therefore, when this bit is enabled and the dma_req_data_out_en (Bit [6]) is enabled, this DMA request is used for the context output and data output. Similarly if the data_out interrupt enable (Bit [2] of AES_IRQENABLE register) is enabled, this interrupt is used for context output and data output. Note that when context and data output request or interrupt are mapped on each other, the context output is always requested after the last data output. Reset type: PER.RESET
8	DMA_REQ_CONTEXT_OUT_EN	R/W	0h	DMA Request Context Out Enable If set to 1, the DMA context output request is enabled (for context data out, for example, TAG for authentication modes). Value Description 0 DMA disabled for context output request. 1 DMA enabled for context output request. Reset type: PER.RESET
7	DMA_REQ_CONTEXT_IN_EN	R/W	0h	DMA Request Context In Enable Value Description 0 DMA disabled for context input request. 1 DMA enabled for context input request. Reset type: PER.RESET

Table 33-39. AES_SYSCONFIG Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
6	DMA_REQ_DATA_OUT_EN	R/W	0h	DMA Request Data Out Enable Value Description 0 DMA disabled for data output request. 1 DMA enabled for data output request. Reset type: PER.RESET
5	DMA_REQ_DATA_IN_EN	R/W	0h	DMA Request Data In Enable Value Description 0 DMA disabled for data input request. 1 DMA enabled for data input request. Reset type: PER.RESET
4	RESERVED	R/W	0h	Reserved
3-2	SIDLE	R/W	0h	Slave Idle Mode Value Description 0x0 Force-idle mode 0x1 No-idle 0x2 Smart-idle 0x3 reserved Reset type: PER.RESET
1	SOFTRESET	R/W	0h	Soft reset Value Description 0 No operation 1 Start soft reset sequence Reset type: PER.RESET
0	AUTOIDLE	R/W	1h	If set to 1, the internal clocks are switched off when there is no processing to be done. This bit is only available on the sHIB. Reset type: PER.RESET

33.6.2.35 AES_SYSSTATUS Register (Offset = 88h) [Reset = 1h]

AES_SYSSTATUS is shown in Figure 33-48 and described in Table 33-40.

Return to the Summary Table.

Secure Host Interface Bank

Figure 33-48. AES_SYSSTATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							RESETDONE
R-0h							R-1h

Table 33-40. AES_SYSSTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	RESETDONE	R	1h	Reset Done Value Description 0 Reset is not complete. 1 Reset is has completed. Reset type: PER.RESET

33.6.2.36 AES_IRQSTATUS Register (Offset = 8Ch) [Reset = 0h]

AES_IRQSTATUS is shown in Figure 33-49 and described in Table 33-41.

Return to the Summary Table.

Secure Host Interface Bank

Figure 33-49. AES_IRQSTATUS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				CONTEXT_OUT	DATA_OUT	DATA_IN	CONTEXT_IN
R-0h				R-0h	R-0h	R-0h	R-0h

Table 33-41. AES_IRQSTATUS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	CONTEXT_OUT	R	0h	Context Output Interrupt Status Value Description 0 Authentication tag (and IV) interrupt(s) is/are not active. 1 Authentication tag (and IV) interrupt(s) is/are active and the interrupt output has been triggered. Reset type: PER.RESET
2	DATA_OUT	R	0h	Data Out Interrupt Status Value Description 0 The data out interrupt is not active. 1 The data out interrupt is active and the interrupt output has been triggered. Reset type: PER.RESET
1	DATA_IN	R	0h	Data In Interrupt Status Value Description 0 The data in interrupt is not active. 1 The data in interrupt is active and the interrupt output has been triggered. Reset type: PER.RESET
0	CONTEXT_IN	R	0h	Context In Interrupt Status Value Description 0 The context in interrupt is not active. 1 The context in interrupt is active and the interrupt output has been triggered. Reset type: PER.RESET

33.6.2.37 AES_IRQENABLE Register (Offset = 90h) [Reset = 0h]

AES_IRQENABLE is shown in Figure 33-50 and described in Table 33-42.

Return to the Summary Table.

Secure Host Interface Bank

Figure 33-50. AES_IRQENABLE Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				CONTEXT_OUT	DATA_OUT	DATA_IN	CONTEXT_IN
R-0h				R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 33-42. AES_IRQENABLE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	CONTEXT_OUT	R/W	0h	Context Out Interrupt Enable Value Description 0 Authentication tag (and IV) interrupt(s) is/are disabled. 1 Authentication tag (and IV) interrupt(s) is/are enabled. Reset type: PER.RESET
2	DATA_OUT	R/W	0h	Data Out Interrupt Enable Value Description 0 The data out interrupt is disabled. 1 The data out interrupt is enabled. Reset type: PER.RESET
1	DATA_IN	R/W	0h	Data In Interrupt Enable Value Description 0 The data in interrupt is disabled. 1 The data in interrupt is enabled. Reset type: PER.RESET
0	CONTEXT_IN	R/W	0h	Context In Interrupt Enable Value Description 0 The context in interrupt is disabled. 1 The context in interrupt is enabled. Reset type: PER.RESET

33.6.2.38 AES_DIRTY_BITS Register (Offset = 94h) [Reset = 0h]

AES_DIRTY_BITS is shown in Figure 33-51 and described in Table 33-43.

Return to the Summary Table.

Secure Host Interface Bank

Figure 33-51. AES_DIRTY_BITS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED				RESERVED	RESERVED	S_DIRTY	S_ACCESS
R-0h				R/W1S-0h	R/W1S-0h	R/W1S-0h	R/W1S-0h

Table 33-43. AES_DIRTY_BITS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-4	RESERVED	R	0h	Reserved
3	RESERVED	R/W1S	0h	Reserved
2	RESERVED	R/W1S	0h	Reserved
1	S_DIRTY	R/W1S	0h	AES Dirty Bit This bit must be written to a 1 to clear. Value Description 0 No AES registers have been written. 1 Indicates when any of the AES_x registers have been written (except for the AES_DIRTYBITS register). Reset type: PER.RESET
0	S_ACCESS	R/W1S	0h	AES Access Bit This bit must be written to a 1 to clear. Value Description 0 No AES registers have been read. 1 Indicates when any of the AES_x registers have been read (except for the AES_DIRTYBITS register). Reset type: PER.RESET

33.6.3 AES_SS_REGS Registers

Table 33-44 lists the memory-mapped registers for the AES_SS_REGS registers. All register offset addresses not listed in Table 33-44 should be considered as reserved locations and the register contents should not be modified.

Table 33-44. AES_SS_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
4h	AES_GLB_INT_FLG	AES Global Interrupt Flag Register		Go
8h	AES_GLB_INT_CLR	AES Global Interrupt Clear Register		Go

Complex bit access types are encoded to fit into small table cells. Table 33-45 shows the codes that are used for access types in this section.

Table 33-45. AES_SS_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

33.6.3.1 AES_GLB_INT_FLG Register (Offset = 4h) [Reset = 0h]

AES_GLB_INT_FLG is shown in Figure 33-52 and described in Table 33-46.

Return to the Summary Table.

The AES_GLB_INT_FLG register contains the current status of the AES interrupt

Figure 33-52. AES_GLB_INT_FLG Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INT_FLG
R-0h							R-0h

Table 33-46. AES_GLB_INT_FLG Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	INT_FLG	R	0h	Global Interrupt Flag for AES INT. This bit determines whether the SINTREQUEST is generated by AES This bit can be cleared by writing a 1 to the corresponding bit in the AES_GLB_INT_CLR register. Reset type: SYSRSn

33.6.3.2 AES_GLB_INT_CLR Register (Offset = 8h) [Reset = 0h]

AES_GLB_INT_CLR is shown in Figure 33-53 and described in Table 33-47.

Return to the Summary Table.

The AES_GLB_INT_CLR register is used to clear the interrupt flags in AES_GLB_INT_FLG register.

Figure 33-53. AES_GLB_INT_CLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED							INT_FLG_CLR
R-0h							R/W1C-0h

Table 33-47. AES_GLB_INT_CLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-1	RESERVED	R	0h	Reserved
0	INT_FLG_CLR	R/W1C	0h	Global Interrupt flag clear for AES INT. This bit is used to clear the corresponding bit in the AES_GLB_INT_FLG register. Write 1 to clear the INT_FLG bit. Writing 0 has no effect. Reset type: SYSRSn

33.6.4 Register to Driverlib Function Mapping

33.6.4.1 AES Registers to Driverlib Functions

Table 33-48. AES Registers to Driverlib Functions

File	Driverlib Function
KEY2_6	
aes.c	AES_setKey2
aes.c	AES_setKey3
KEY2_7	
aes.c	AES_setKey2
aes.c	AES_setKey3
KEY2_4	
aes.c	AES_setKey2
aes.c	AES_setKey3
KEY2_5	
aes.c	AES_setKey2
aes.c	AES_setKey3
KEY2_2	
aes.c	AES_setKey2
KEY2_3	
aes.c	AES_setKey2
KEY2_0	
aes.c	AES_setKey2
KEY2_1	
aes.c	AES_setKey2
KEY1_6	
aes.c	AES_setKey1
KEY1_7	
aes.c	AES_setKey1
KEY1_4	
aes.c	AES_setKey1
KEY1_5	
aes.c	AES_setKey1
KEY1_2	
aes.c	AES_setKey1
KEY1_3	
aes.c	AES_setKey1
KEY1_0	
aes.c	AES_setKey1
KEY1_1	
aes.c	AES_setKey1
IV_IN_OUT_0	
aes.c	AES_setInitializationVector
aes.c	AES_readInitializationVector
IV_IN_OUT_1	
aes.c	AES_setInitializationVector
aes.c	AES_readInitializationVector

Table 33-48. AES Registers to Driverlib Functions (continued)

File	Driverlib Function
IV_IN_OUT_2	
aes.c	AES_setInitializationVector
aes.c	AES_readInitializationVector
IV_IN_OUT_3	
aes.c	AES_setInitializationVector
aes.c	AES_readInitializationVector
CTRL	
aes.c	AES_configureModule
aes.c	AES_readTag
aes.c	AES_readDataNonBlocking
aes.c	AES_readDataBlocking
aes.c	AES_writeDataNonBlocking
aes.c	AES_writeDataBlocking
C_LENGTH_0	
aes.h	AES_setDataLength
C_LENGTH_1	
aes.h	AES_setDataLength
AUTH_LENGTH	
aes.h	AES_setAuthDataLength
DATA_IN_OUT_0	
aes.c	AES_readDataNonBlocking
aes.c	AES_readDataBlocking
aes.c	AES_writeDataNonBlocking
aes.c	AES_writeDataBlocking
DATA_IN_OUT_1	
aes.c	AES_readDataNonBlocking
aes.c	AES_readDataBlocking
aes.c	AES_writeDataNonBlocking
aes.c	AES_writeDataBlocking
DATA_IN_OUT_2	
aes.c	AES_readDataNonBlocking
aes.c	AES_readDataBlocking
aes.c	AES_writeDataNonBlocking
aes.c	AES_writeDataBlocking
DATA_IN_OUT_3	
aes.c	AES_readDataNonBlocking
aes.c	AES_readDataBlocking
aes.c	AES_writeDataNonBlocking
aes.c	AES_writeDataBlocking
TAG_OUT_0	
aes.c	AES_readTag
TAG_OUT_1	
aes.c	AES_readTag
TAG_OUT_2	
aes.c	AES_readTag

Table 33-48. AES Registers to Driverlib Functions (continued)

File	Driverlib Function
TAG_OUT_3	
aes.c	AES_readTag
REV	
-	
SYSCONFIG	
aes.h	AES_performSoftReset
aes.h	AES_enableDMARequest
aes.h	AES_disableDMARequest
SYSSTATUS	
aes.h	AES_performSoftReset
IRQSTATUS	
aes.c	AES_getInterruptStatus
IRQENABLE	
aes.c	AES_getInterruptStatus
aes.h	AES_enableInterrupt
aes.h	AES_disableInterrupt
DIRTY_BITS	
-	

33.6.4.2 AES_SS Registers to Driverlib Functions

Table 33-49. AES_SS Registers to Driverlib Functions

File	Driverlib Function
AES_GLB_INT_FLG	
aes.h	AES_enableGlobalInterrupt
aes.h	AES_disableGlobalInterrupt
aes.h	AES_getGlobalInterruptStatus
AES_GLB_INT_CLR	
aes.h	AES_clearGlobalInterrupt

This page intentionally left blank.

Chapter 34

Embedded Pattern Generator (EPG)

This chapter describes the Embedded Pattern Generator (EPG) module.

34.1 Introduction

The Embedded Pattern Generator (EPG) module is a customizable pattern and clock generator that could serve many test and application scenarios that require a simple pattern generator or a periodic clock generator. The EPG module can also be used to capture an incoming serial stream of data.

34.1.1 Features

Features of the EPG module are:

- Clock generation:
  - Independent clock generation and clock division
  - Synchronous clock generation with programmable offsets
- Pattern generation:
  - Independent serial data stream generation
  - Serial data stream and the associated clock generation
  - Ability to skew clock with respect to serial data
  - Synchronous data stream with programmable offset with respect to one another

The EPG output signals are connected to GPIOs or other peripherals inside the device. The EPG inputs act as shift register inputs to capture incoming serial data streams. This allows the EPG to be configured as a custom serial module. The EPG is also capable of generating interrupts that are used to supply the pattern generators with new data or signal the completion of a generated pattern.

34.1.2 EPG Block Diagram

Figure 34-1 shows the EPG overview block diagram.

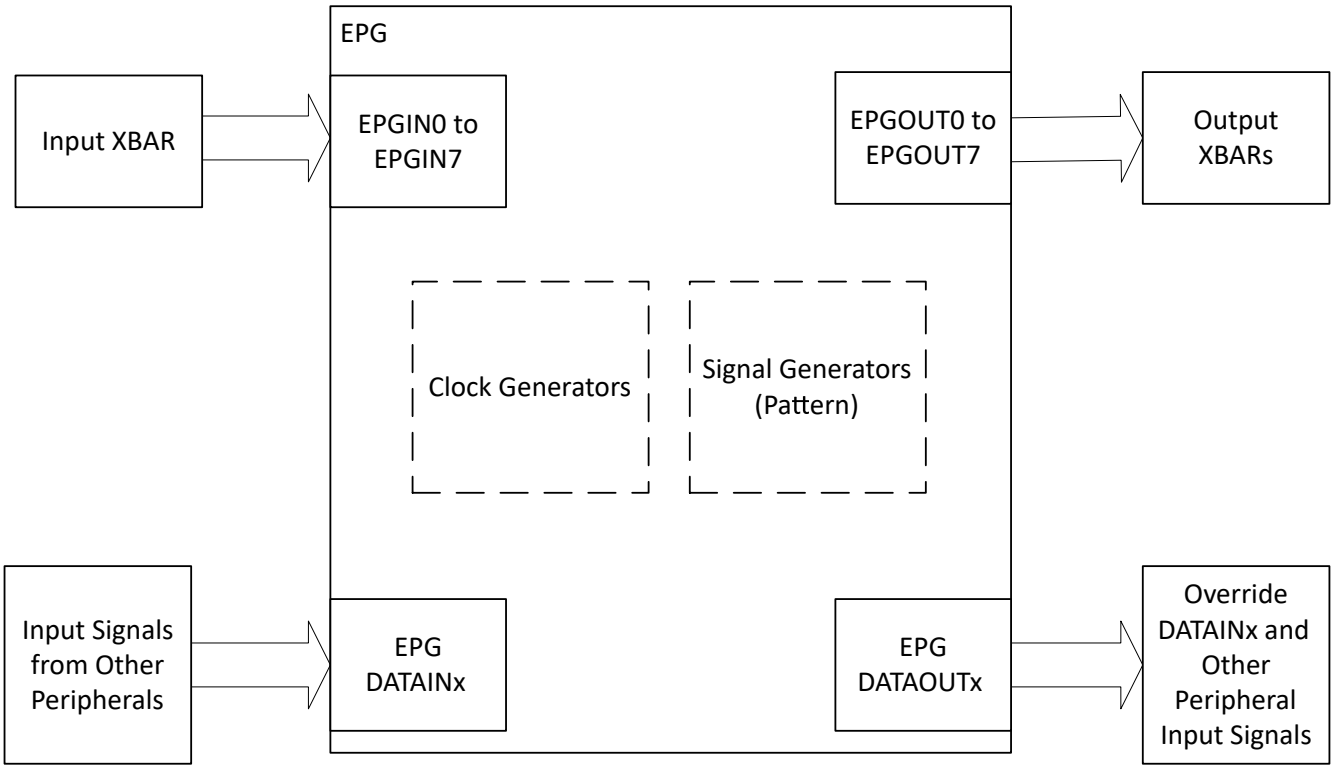


Figure 34-1. EPG Overview Block Diagram

Figure 34-2 shows the EPG block diagram.

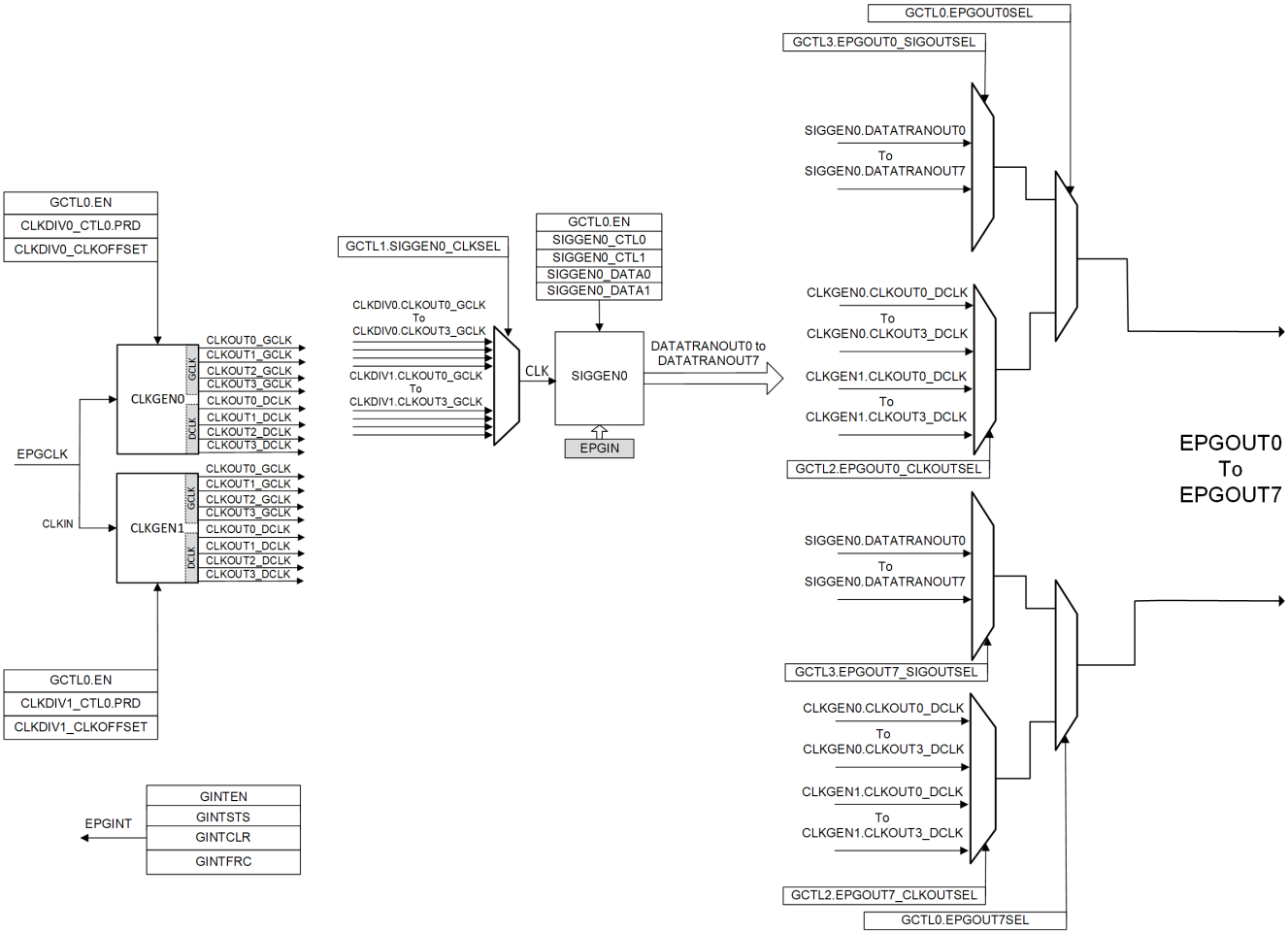


Figure 34-2. EPG Detailed Block Diagram

Note

The number of available EPGx SIGGEN modules for each device can be different. This diagram shows an EPG with 2 SIGGEN modules (SIGGEN0 and SIGGEN1). Refer to the EPG_REGS register description to see how many SIGGEN modules are available for a specific device.

Note

The EPG input clock (EPGCLK) is sourced from PERx.SYSCLK.

34.2 Clock Generator Modules

The clock generator modules use the EPG input clock and generate four output clocks, see Figure 34-3. Each of the four output clocks (CLKOUT0 to CLKOUT3) has a DCLK and a GCLK version. The EPG clock division results in a gated, GCLK, version of the EPG input clock which are named CLKOUT0_GCLK to CLKOUT3_GCLK. The other version, DCLK (CLKOUT0_DCLK to CLKOUT3_DCLK), have the same period but with an approximately 50% duty cycle. The clock generation takes place using the divider settings CLKDIVx_CTL0.PRD, and clock offset settings CLKDIVx_CLK.CLKyOFFSET. The RUNCLOCK signal generated by the clock stop logic determines when the clock generation is started and stopped.

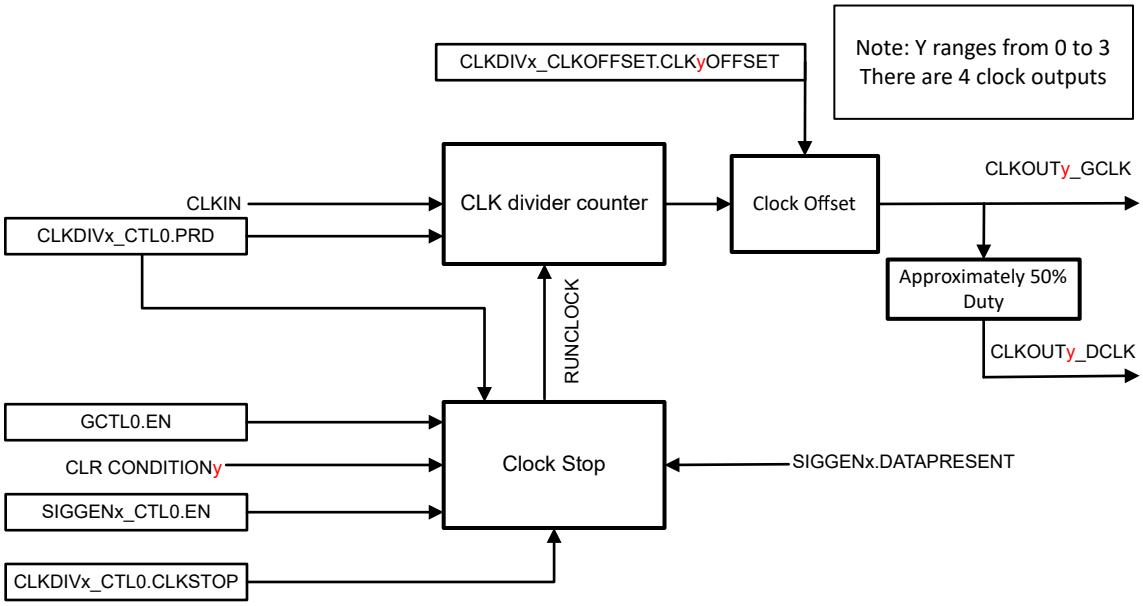


Figure 34-3. EPG Clock Generator

The clock divider counter is a simple up counter, which has a period determined by CLKDIVx_CTL0.PRD. This divider counter begins counting when RUNCLOCK is set. The CLKOUT0 to CLKOUT4 GCLKs are gated versions of the CLKIN (EPG input clock). The clock gates are enabled when the counter value matches the corresponding clock offsets. In the case where RUNCLOCK is cleared, the clock gates are disabled and the counter is set to zero.

Note

The CLKDIVx_CTL0.PRD register should only be written when GCTL0.EN is 0.

34.2.1 DCLK (50% duty cycle clock)

The CLKOUTy_DCLK is generated by setting the clock output for half the clock divider period. When CLKDIVx_CTL0.PRD is set to zero, CLKOUTy_DCLK is the same as the input clock.

34.2.2 Clock Stop

Figure 34-4 shows how the Clock Stop module sets and clears the RUNCLOCK signal.

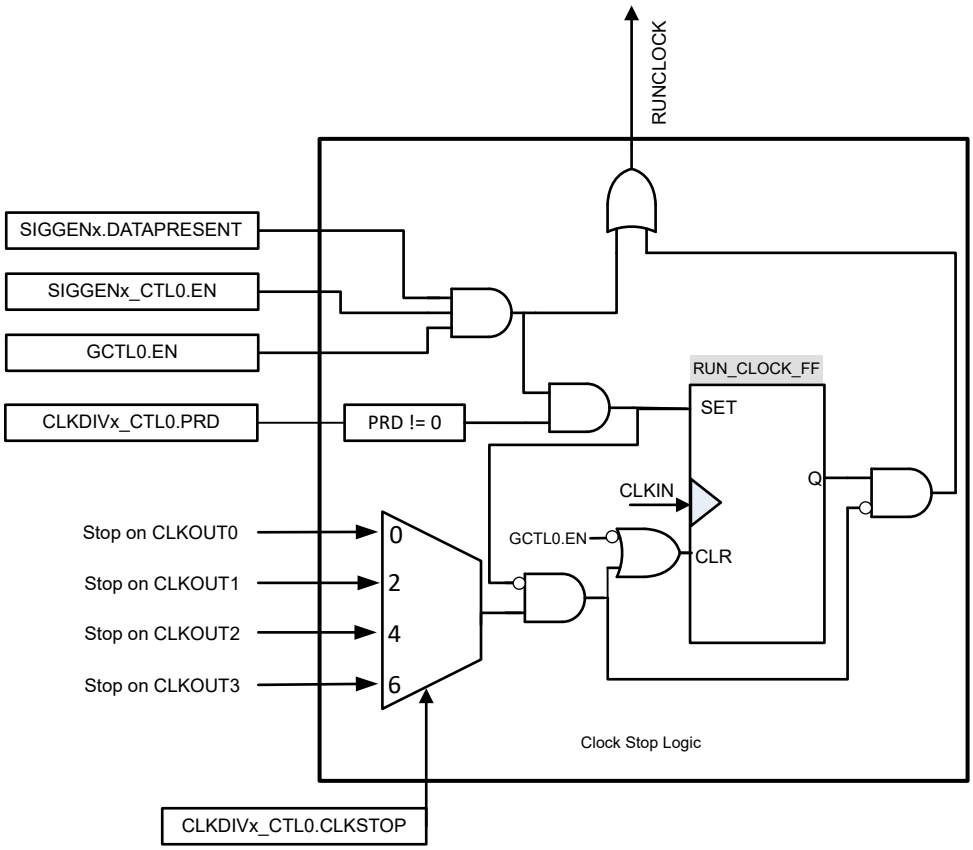


Figure 34-4. EPG Clock Stop

In signal generator modes, SIGGENx_CTL0.EN bit is cleared when BIT_LENGTH shifts (or rotates) is completed. This ensures that data is not shifted out. In addition, clock generation (generation of CLKOUT0 to CLKOUT3) also have to be stopped to ensure that sampling of input data does not continue after BIT_LENGTH shifts when SIGGENx_CTL0.EN is cleared to 0.

The RUNCLOCK signal has to be HIGH for the clock generation circuitry to be active. When SIGGENx_CTL0.EN is cleared, the clock generation can be selected to stop on the falling edge of CLKOUT0 to CLKOUT3.

The clock stop module operates as follow:

- RUNCLOCK is asserted as soon as both SIGGENx_CTL0.EN and GCTL0.EN are set.
- RUN_CLOCK_FF is set when both SIGGENx_CTL0.EN and GCTL0.EN are set and CLKDIVx_CTL0.PRD is not zero.
- When SIGGENx_CTL0.EN is cleared, RUN_CLOCK_FF is not cleared immediately. This ensures that RUNCLOCK remains set and the clock generation circuitry remains enabled.
- When SIGGENx.DATA0 and SIGGENx.DATA1 are not filled with data before the next shift sequence (repeat shift modes) and signal generator is waiting for data to be written, clocks will be stopped.
- The RUN_CLOCK_FF can be cleared on the falling edge of CLKOUT0 to CLKOUT3 based on the configuration of CLKSTOP field.
- Clearing GCTL0.EN clears RUN_CLOCK_FF unconditionally.

34.3 Signal Generator Module

The signal generator module is the main component of the EPG. It generates the data stream that follows custom patterns. Figure 34-5 shows the main components. This module has 8 output ports DATATRANOUT0 to DATATRANOUT7. The two registers SIGGENx_DATA1 and SIGGENx_DATA0 constitute a 64-bit bus named DATA[63:0]. DATATRANIN[63:0], which is used for all the data transform operations, could be DATA[63:0] or bit reversed DATA (when SIGGENx_CTL0.BRIN bit is set). The DATATRANOUT0 to DATATRANOUT7 are connected to DATATRANIN0 to DATATRANIN7 when the signal generator is **not** in BIT_BANG mode. In BIT_BANG mode, DATATRANOUT0 to DATATRANOUT7 are connected to DATATRANIN0, DATATRANIN8, and DATATRANIN16 to DATATRANIN56.

In addition to generating data outputs, one of the 8 EPGIN inputs can act as data input to SIGGENx_DATAy registers. In Figure 34-5, the EPGIN_MUX block illustrates the mechanism used to capture the input data stream. This enables one to capture a data input stream using EPG module.

Data transformation is done on the DATATRANIN bus, and is determined by the configured mode (SIGGENx_CTL0.MODE), provided GCTL0.EN and SIGGENx_CTL0.EN are both set. If either of these enable bits are 0, then the data output of the transform block is the same as the input. The transformed output is bit reversed when SIGGENx_CTL0.BROUT bit is set. Conditions under which the DATA active register (SIGGENx_DATA0_ACTIVE and SIGGENx_DATA1_ACTIVE) are:

- Loaded from SIGGENx_DATA1 and SIGGENx_DATA0
- Directly updated on a memory mapped write to SIGGENx_DATA1 and SIGGENx_DATA0
- Holds the current data

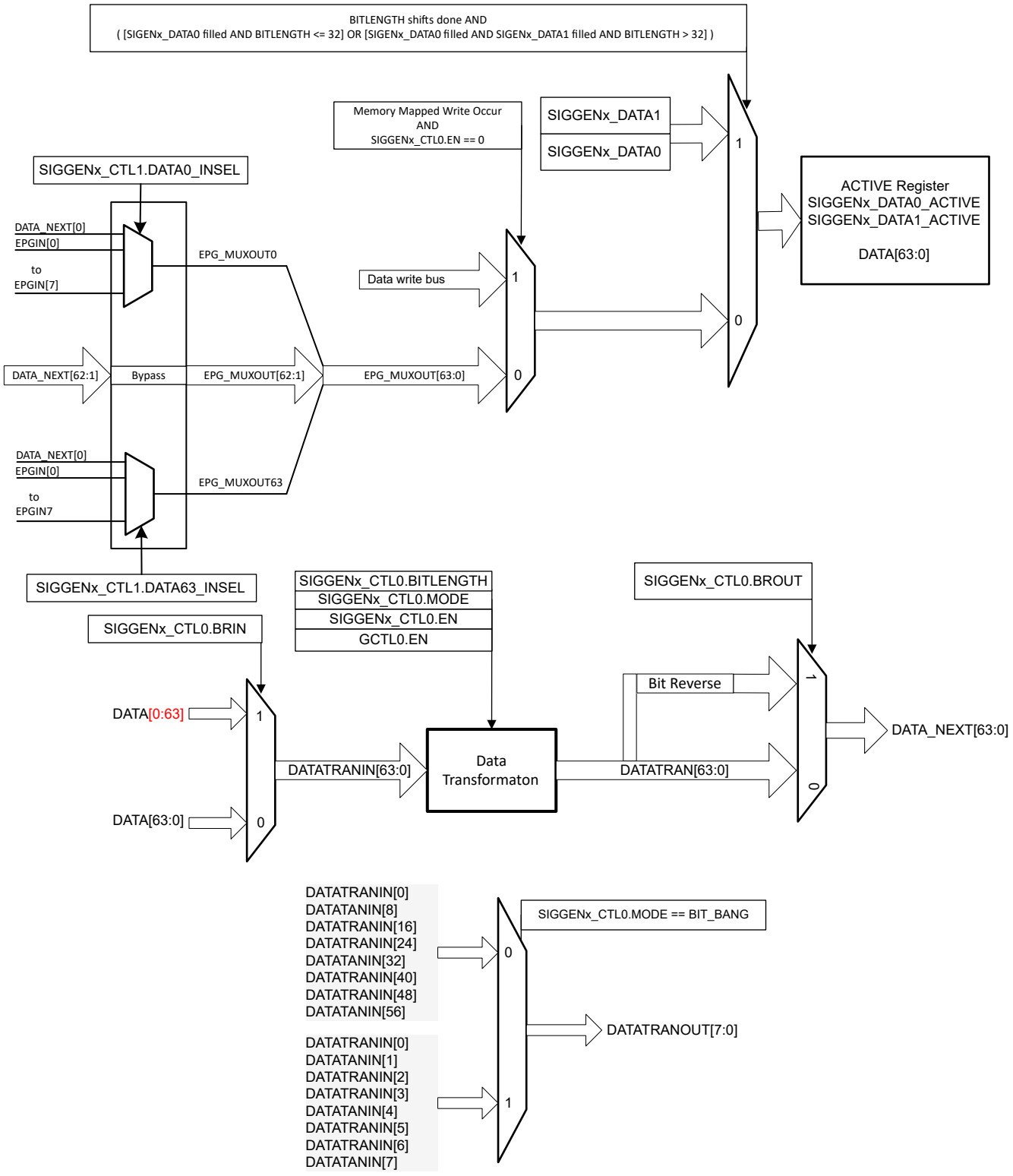


Figure 34-5. EPG Signal Generator Detailed Overview

Table 34-1. SIGGENx Active Register Loading

Condition	SIGGENx_DATA1_ACTIVE DATA ACTIVE Register [63:32]	SIGGENx_DATA0_ACTIVE DATA ACTIVE Register [31:0]
Memory mapped write to SIGGENx_DATA0 register and SIGGENx_CTL0.EN is 0	No updates	Updated with the value written to SIGGENx_DATA0 register
Memory mapped write to SIGGENx_DATA1 register and SIGGENx_CTL0.EN is 0.	Updated with the value written to SIGGENx_DATA1 register	No updates
“BITLENGTH” number of shifts are done, and Mode is SHIFT_RIGHT_REPEAT or SHIFT_LEFT_REPEAT, and BITLENGTH <= 32, and Either SIGGENx_DATA0 or SIGGENx_DATA1 has been updated	Copy SIGGENx_DATA1 register content	Copy SIGGENx_DATA0 register content
“BITLENGTH” number of shifts are done, and Mode is SHIFT_RIGHT_REPEAT or SHIFT_LEFT_REPEAT, and BITLENGTH >= 32, and Both SIGGENx_DATA0 and SIGGENx_DATA1 have been updated	Copy SIGGENx_DATA1 register contents	Copy SIGGENx_DATA0 register contents
“BITLENGTH” number of shifts are done, and Mode is SHIFT_RIGHT_REPEAT or SHIFT_LEFT_REPEAT, and BITLENGTH <= 32, and Neither SIGGENx_DATA0, nor SIGGENx_DATA1 has been updated	Hold the current value, no shifts	Hold the current value, no shifts
“BITLENGTH” number of shifts are done, and Mode is SHIFT_RIGHT_REPEAT or SHIFT_LEFT_REPEAT, and BITLENGTH >= 32, and Neither SIGGENx_DATA0, nor SIGGENx_DATA1 has not been updated	Hold the current value, no shifts	Hold the current value, no shifts
All other conditions	Updates based on current mode of operation	Updates based on current mode of operation

Following are the possible data transformations:

Bit-bang mode: In this mode, DATATRAN[63:0] is the same as DATATRANIN[63:0].

Shift right once mode: In this mode, DATATRAN[63:0] = {0,DATATRANIN[63:1]}. After SIGENx_CTL0.BITLENGTH shifts, SIGENx_CTL0.EN is cleared.

Shift right repeat mode: In this mode, DATATRAN[63:0] = {0,DATATRANIN[63:1]}. After SIGENx_CTL0.BITLENGTH shifts, load the data as per Table 34-1.

Rotate right once mode: In this mode, DATATRAN[63:0] = {DATATRANIN[0],DATATRANIN[63:1]}. After SIGENx_CTL0.BITLENGTH shifts, active register is loaded from the {DATA1,DATA0} register, and SIGENx_CTL0.EN is cleared.

Rotate right repeat: This mode is same as rotate right once, except that SIGENx_CTL0.EN is not cleared upon SIGENx_CTL0.BITLENGTH rotates.

Shift left once mode: In this mode, DATATRAN[63:0] = {DATATRANIN[62:1],0}. After SIGENx_CTL0.BITLENGTH shifts, SIGENx_CTL0.EN is cleared.

Shift left repeat mode: In this mode, DATATRAN[63:0] = {DATATRANIN[62:1],0}. After SIGENx_CTL0.BITLENGTH shifts, load the data as per Table 34-1.

Rotate left once mode: In this mode, DATATRAN[63:0] = {DATATRANIN[62:1],DATATRANIN[63]}. After SIGENx_CTL0.BITLENGTH shifts, active register is loaded from the {DATA1,DATA0} register, and SIGENx_CTL0.EN is cleared.

Rotate left repeat: This mode is same as rotate left once, except that SIGENx_CTL0.EN is not cleared upon SIGENx_CTL0.BITLENGTH rotates.

Note

SIGGENx_CTL0.MODE and SIGGENx_CTL.BITWIDTH should only be updated when SIGGENx_CTL0.EN is 0.

34.4 EPG Peripheral Signal Mux Selection

EPG DATAINx signals are connected to input signal sources from other peripherals. The EPG DATAINx signal sources are listed in Table 34-2.

Table 34-2. EPG Data Input Connections

EPG1 Data Connection (DATAINx)	Peripheral Input Signal Name
0	CANA RX
1	MCAN RX
2 to 18	Reserved
19	SD1 C1
20	SD1 C2
21	SD1 C3
22	SD1 C4
23	SD2 C1
24	SD2 C2
25	SD2 C3
26	SD2 C4
27 to 63	Reserved

EPGOUT0 to EPGOUT7 can override peripheral input signals connected to DATAINx. For example, the peripheral CAN RX input signal can be overwritten by EPGOUTx. DATAINx can be passed to DATAOUTx untouched, if EPG is not required, or the DATAINx can be replaced with EPGOUTy (y = x%8). The EPGMXSEL0 and EPGMXSEL1 registers must be configured to select the source of the DATAOUTx signals. Also, EPGOUTx signals can be connected to GPIOs through Output XBARs.

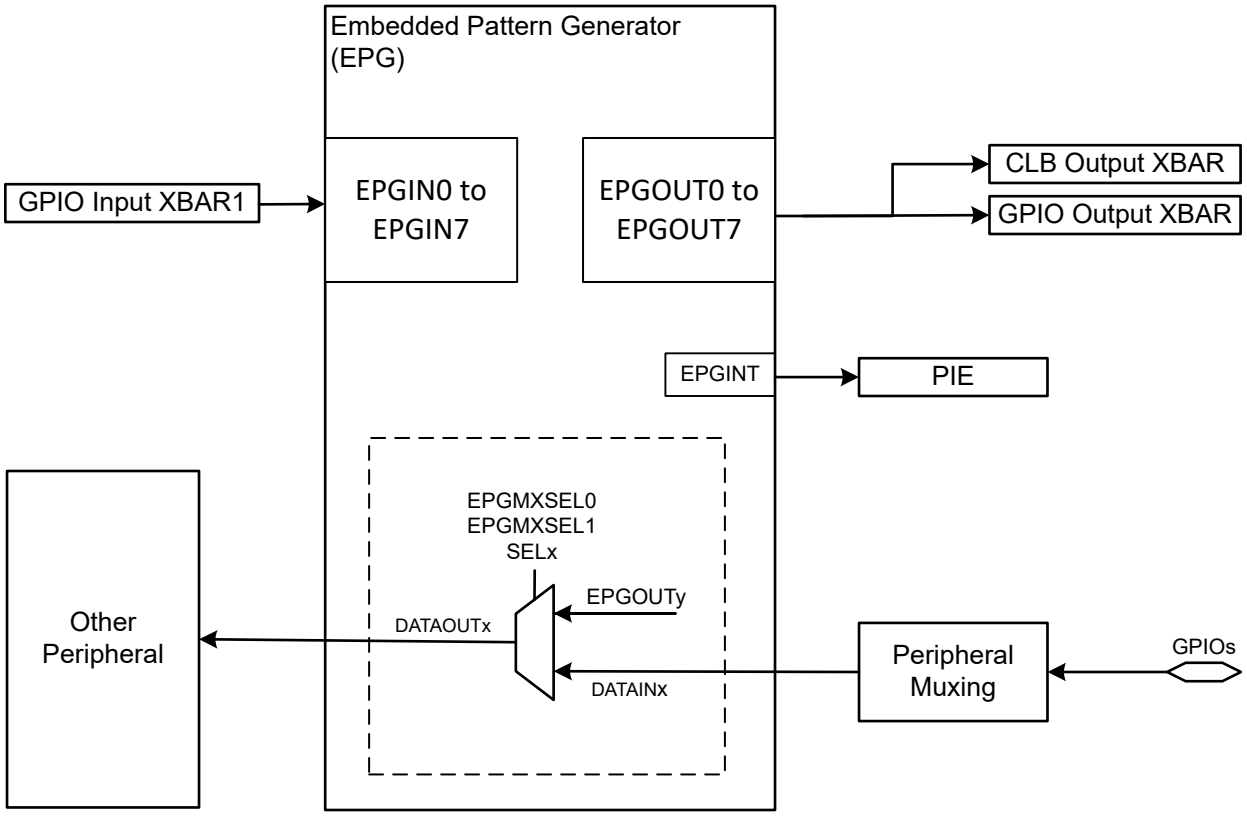


Figure 34-6. EPG Peripheral Signal Muxing

34.5 EPG Example Use Cases

This section provides example register configurations for different EPG use cases.

Note

Some examples may require more than one SIGGEN modules. Check the device registers or datasheet for the number available SIGGEN modules.

34.5.1 EPG Example 1: Synchronous Clocks with Offset

The Example 34-1 register configuration generates 4 clocks, all synchronous to one another with edges offset by 2 clock cycles. In Example 34-1, a clock divide value of 12 is used.

Example 34-1. Register Configuration

Register	Value	Selected Mode
Epg1MuxRegs		
EPGMXSEL0.SEL0	0x1	Select EPGOUT0 to drive DATAOUT[0]
EPGMXSEL0.SEL1	0x1	Select EPGOUT1 to drive DATAOUT[1]
EPGMXSEL0.SEL2	0x1	Select EPGOUT2 to drive DATAOUT[2]
EPGMXSEL0.SEL3	0x1	Select EPGOUT3 to drive DATAOUT[3]
Epg1Regs		
Global Settings		
GCTL0.EPGOUT0SEL	0x0	Selects signal mux output on EPGOUT0
GCTL0.EPGOUT1SEL	0x0	Selects signal mux output on EPGOUT1
GCTL0.EPGOUT2SEL	0x0	Selects signal mux output on EPGOUT2
GCTL0.EPGOUT3SEL	0x0	Selects signal mux output on EPGOUT3
GCTL3.EPGOUT0_SIGOUTSEL	0x0	Select SIGGEN0.OUT[0] on EPGOUT0
GCTL3.EPGOUT1_SIGOUTSEL	0x1	Select SIGGEN0.OUT[1] on EPGOUT1
GCTL3.EPGOUT2_SIGOUTSEL	0x2	Select SIGGEN0.OUT[2] on EPGOUT2
GCTL3.EPGOUT3_SIGOUTSEL	0x3	Select SIGGEN0.OUT[3] on EPGOUT3
GCTL1.SIGGEN0_CLKSEL	0x0	Select CLKOUT0 of CLKGEN0 as the clock source of SIGGEN0
CLKGEN0 Setting		
CLKDIV0_CTL0.PRD	0x0	Divide by 1
CLKDIV0_CLKOFFSET.CLK0OFFSET	0x0	No offset
SIGGEN0 Mode and Bit length configuration		
SIGGEN0_CTL0.BITLENGTH	0xC	Configure bit length to 12 to get a divide by 12 clock.
SIGGEN0_CTL0.MODE	0x3	Configure the mode to rotate right repeat mode to generate a periodic waveform.
SIGGEN0_DATA0[11:0]	"000001111110"	Initialize the 12 bits to 6 ones and 6 zeroes, which ensures that we get a 50 % duty cycle clock.
SIGGEN0_DATA0[27:16]	"000111111000"	Create a 2 cycle offset with respect to first clock.
SIGGEN0_DATA1[11:0]	"011111100000"	Create a 4 cycle offset with respect to first clock.
SIGGEN0_DATA1[27:16]	"111110000001"	Create a 6 cycle offset with respect to first clock.

34.5.2 EPG Example 2: Serial Data Bit Stream (LSB first)

The Example 34-2 register configuration shifts out a data word, the data rate is set to divide by 8 and LSB is shifted out first. After 32 shifts are completed, an interrupt is generated for further sequencing.

Example 34-2. Register Configuration

Register	Value	Selected Mode
Epg1MuxRegs		
EPGMXSEL0.SEL0	0x1	Select EPGOUT0 to drive DATAOUT[0]
Epg1Regs		
Global Settings		
GCTL1.SIGGEN0_CLKSEL	0x0	Select CLKOUT0 of CLKGEN0 as the clock source of SIGGEN0
CLKGEN0 Setting		
CLKDIV0_CTL0.PRD	0x7	Divide by 8
CLKDIV0_CLKOFFSET.CLK0OFFSET	0x0	No offset
SIGGEN0 Mode and Bit length configuration		
SIGGEN0_CTL0.BITLENGTH	0x20	Do 32 shifts.
SIGGEN0_CTL0.MODE	0x1	Configure the mode to shift right once mode. Generates an interrupt after 32 shifts.
SIGGEN0_DATA0[15:0]	0xaa55	Data to be shifted out
SIGGEN0_DATA0[31:16]	0xcccc	Data to be shifted out
SIGGEN0_DATA1[15:0]	0xaa55	Data to be shifted out
SIGGEN0_DATA1[31:16]	0x55aa	Data to be shifted out

34.5.3 EPG Example 3: Serial Data Bit Stream (MSB first)

The Example 34-3 register configuration shifts out a data word, the data rate is set to divide by 8 and MSB is shifted out first. After 32 shifts are complete, an interrupt is generated for further sequencing.

Example 34-3. Register Configuration

Register	Value	Selected Mode
Epg1MuxRegs		
EPGMXSEL0.SEL0	0x1	Select EPGOUT0 to drive DATAOUT[0]
Epg1Regs		
Global Settings		
GCTL0.EPGOUT0SEL	0x0	Selects signal mux output on EPGOUT0
GCTL3.EPGOUT0_SIGOUTSEL	0x4	Select SIGGEN0.OUT[4] on EPGOUT0, on 64 bit reversal, 31 bit will appear on 32 bit (hence configuring to 4).
GCTL1.SIGGEN0_CLKSEL	0x0	Select CLKOUT0 of CLKGEN0 as the clock source of SIGGEN0
CLKGEN0 Setting		
CLKDIV0_CTL0.PRD	0x7	Divide by 8
CLKDIV0_CLKOFFSET.CLK0OFFSET	0x0	No offset
SIGGEN0 Mode and Bit length configuration		
SIGGEN0_CTL0.BITLENGTH	0x20	Do 32 shifts.
SIGGEN0_CTL0.MODE	0x1	Configure the mode to shift right once mode. Generates an interrupt after 32 shifts.
SIGGEN0_CTL0.BRIN	0x1	Reverse the bits to the data transform block. This to ensure that MSB is transmitted first.
SIGGEN0_CTL0.BROUT	0x1	Reverse the data back and store in the register
SIGGEN0_DATA0[15:0]	0xaa55	Data to be shifted out
SIGGEN0_DATA0[31:16]	0xcccc	Data to be shifted out
SIGGEN0_DATA1[15:0]	0xaa55	Data to be shifted out
SIGGEN0_DATA1[31:16]	0x55aa	Data to be shifted out

34.5.4 EPG Example 4: Clock and Data Pair

The Example 34-4 register configuration shifts out a data word, also an associated clock is generated to latch the data (EPGOUT1), the data rate is set to divide by 8 and MSB is shifted out first. After 32 shifts are complete, an interrupt is generated for further sequencing.

Example 34-4. Register Configuration

Register	Value	Selected Mode
Epg1MuxRegs		
EPGMXSEL0.SEL0	0x1	Select EPGOUT0 to drive DATAOUT[0]
EPGMXSEL0.SEL1	0x1	Select EPGOUT1 to drive DATAOUT[1]
Epg1Regs		
Global Settings		
GCTL0.EPGOUT0SEL	0x0	Selects signal mux output on EPGOUT0
GCTL3.EPGOUT0_SIGOUTSEL	0x4	Select SIGGEN0.OUT[4] on EPGOUT0, on 64 bit reversal, 31 bit will appear on 32 bit (hence configuring to 4).
GCTL0.EPGOUT1SEL	0x0	Selects signal mux output on EPGOUT1
GCTL3.EPGOUT1_SIGOUTSEL	0x8	Select SIGGEN1.OUT[0] on EPGOUT1.
GCTL1.SIGGEN0_CLKSEL	0x0	Select CLKOUT0 of CLKGEN0 as the clock source of SIGGEN0
GCTL1.SIGGEN1_CLKSEL	0x4	Select CLKOUT0 of CLKGEN1 as the clock source of SIGGEN1
CLKGEN0 Setting		
CLKDIV0_CTL0.PRD	0x7	Divide by 8
CLKDIV0_CLKOFFSET.CLK0OFFSET	0x0	No offset
CLKDIV1_CTL0.PRD	0x3	Divide by 4
SIGGEN0 Mode and Bit length configuration		
SIGGEN0_CTL0.BITLENGTH	0x20	Do 32 shifts.
SIGGEN0_CTL0.MODE	0x1	Configure the mode to shift right once mode. Generates an interrupt after 32 shifts.
SIGGEN0_CTL0.BRIN	0x1	Reverse the bits to the data transform block. This to ensure that MSB is transmitted first.
SIGGEN0_CTL0.BROUT	0x1	Reverse the data back and store in the register
SIGGEN0_DATA0[15:0]	0xaa55	Data to be shifted out
SIGGEN0_DATA0[31:16]	0xcccc	Data to be shifted out
SIGGEN0_DATA1[15:0]	0xaa55	Data to be shifted out
SIGGEN0_DATA1[31:16]	0x55aa	Data to be shifted out
SIGGEN1 Mode and Bit length configuration		
SIGGEN1_CTL0.BITLENGTH	0x2	Set bit length to 2 to generate a 50% duty clock.
SIGGEN1_CTL0.MODE	0x3	Configure the mode to rotate right repeat mode. Generates a 50 % duty cycle clock.
SIGGEN1_DATA0[15:0]	0x2	Data to be shifted out

34.5.5 EPG Example 5: Clock and Skewed Data Pair

Example 34-5 is similar to Example 34-4; however, the data shifted out is skewed by few EPG input clock cycles.

Example 34-5. Register Configuration

Register	Value	Selected Mode
Epg1MuxRegs		
EPGMXSEL0.SEL0	0x1	Select EPGOUT0 to drive DATAOUT[0]
EPGMXSEL0.SEL1	0x1	Select EPGOUT1 to drive DATAOUT[1]
Epg1Regs		
Global Settings		
GCTL0.EPGOUT0SEL	0x0	Selects signal mux output on EPGOUT0
GCTL3.EPGOUT0_SIGOUTSEL	0x4	Select SIGGEN0.OUT[4] on EPGOUT0, on 64 bit reversal, 31 bit will appear on 32 bit (hence configuring to 4).
GCTL0.EPGOUT1SEL	0x0	Selects signal mux output on EPGOUT1
GCTL3.EPGOUT1_SIGOUTSEL	0x8	Select SIGGEN1.OUT[0] on EPGOUT1.
GCTL1.SIGGEN0_CLKSEL	0x0	Select CLKOUT0 of CLKGEN0 as the clock source of SIGGEN0
GCTL1.SIGGEN1_CLKSEL	0x4	Select CLKOUT0 of CLKGEN1 as the clock source of SIGGEN1
CLKGEN0 Setting		
CLKDIV0_CTL0.PRD	0x7	Divide by 8
CLKDIV0_CLKOFFSET.CLK0OFFSET	0x2	Offset of 2 cycles.
CLKDIV1_CTL0.PRD	0x3	Divide by 4
SIGGEN0 Mode and Bit length configuration		
SIGGEN0_CTL0.BITLENGTH	0x20	Do 32 shifts.
SIGGEN0_CTL0.MODE	0x1	Configure the mode to shift right once mode. Generates an interrupt after 32 shifts.
SIGGEN0_CTL0.BRIN	0x1	Reverse the bits to the data transform block. This to ensure that MSB is transmitted first.
SIGGEN0_CTL0.BROUT	0x1	Reverse the data back and store in the register
SIGGEN0_DATA0[15:0]	0xaa55	Data to be shifted out
SIGGEN0_DATA0[31:16]	0xcccc	Data to be shifted out
SIGGEN0_DATA1[15:0]	0xaa55	Data to be shifted out
SIGGEN0_DATA1[31:16]	0x55aa	Data to be shifted out
SIGGEN1 Mode and Bit length configuration		
SIGGEN1_DATA0[15:0]	0x2	Data to be shifted out
SIGGEN1_CTL0.BITLENGTH	0x2	Set bit length to 2 to generate a 50% duty clock.
SIGGEN1_CTL0.MODE	0x3	Configure the mode to rotate right repeat mode. Generates a 50 % duty cycle clock.

34.5.6 EPG Example 6: Capturing Serial Data with a Known Baud Rate

The Example 34-6 register configuration captures a 32-bit data stream. The data is generated from SIGGEN0 and captured in SIGGEN1 (EPGOUT0 looped back as EPGIN0). The clock to SIGEN1 is offset by a few cycles to reliably capture the data.

Example 34-6. Register Configuration

Register	Value	Selected Mode
Epg1MuxRegs		
EPGMXSEL0.SEL0	0x1	Select EPGOUT0 to drive DATAOUT[0]
EPGMXSEL0.SEL1	0x1	Select EPGOUT1 to drive DATAOUT[1]
Epg1Regs		
Global Settings		
GCTL0.EPGOUT0SEL	0x0	Selects signal mux output on EPGOUT0
GCTL3.EPGOUT0_SIGOUTSEL	0x4	Select SIGGEN0.OUT[4] on EPGOUT0, on 64 bit reversal, 31 bit will appear on 32 bit (hence configuring to 4).
GCTL0.EPGOUT1SEL	0x0	Selects signal mux output on EPGOUT1
GCTL3.EPGOUT1_SIGOUTSEL	0x8	Select SIGGEN1.OUT[0] on EPGOUT1.
GCTL1.SIGGEN0_CLKSEL	0x0	Select CLKOUT0 of CLKGEN0 as the clock source of SIGGEN0
GCTL1.SIGGEN1_CLKSEL	0x4	Select CLKOUT0 of CLKGEN1 as the clock source of SIGGEN1
CLKGEN0 Setting		
CLKDIV0_CTL0.PRD	0x7	Divide by 8
CLKDIV0_CLKOFFSET.CLK0OFFSET	0x0	No offset
CLKDIV1_CTL0.PRD	0x7	Divide by 8
CLKDIV1_CLKOFFSET.CLK0OFFSET	0x2	Offset the capture clock by a few cycles to capture the data reliably.
SIGGEN0 Mode and Bit length configuration		
SIGGEN0_CTL0.BITLENGTH	0x20	Do 32 shifts, data out looped back to EPGIN[0] to demonstrate the data capture function.
SIGGEN0_CTL0.MODE	0x1	Configure the mode to shift right once mode. Generates an interrupt after 32 shifts.
SIGGEN0_CTL0.BRIN	0x0	Reverse the bits to the data transform block. This to ensure that MSB is transmitted first.
SIGGEN0_CTL0.BROUT	0x0	Reverse the data back and store in the register
SIGGEN_DATA0[15:0]	0xaa55	Data to be shifted out
SIGGEN_DATA0[31:16]	0xcccc	Data to be shifted out
SIGGEN_DATA1[15:0]	0xaa55	Data to be shifted out
SIGGEN_DATA1[31:16]	0x55aa	Data to be shifted out
SIGGEN1 Mode and Bit length configuration		
SIGGEN1_CTL0.BITLENGTH	0x32	Set bit length to 2 to generate a 50% duty clock.
SIGGEN1_CTL0.MODE	0x4	Configure the mode to shift left once mode. To capture the data pattern on EPGIN[0].
SIGGEN1_CTL1.DATA0_INSEL	0x1	Select EPGIN[0] as the data input of 0'th bit.
SIGGEN1_DATA0	0x0	Clear the data contents. Will hold 0xAA553333 at the end of 32 shifts.

34.6 EPG Interrupt

EPG interrupt can be generated on “BIT_LENGTH" shifts or rotates, or “BIT_LENGTH/2” shifts or rotates. Interrupts can be generated in any of the signal generator modes.

The GINTSTS, GINTEN, GINTCLR, and GINTFRC registers are used to configure the EPG interrupt, as shown in Figure 34-7.

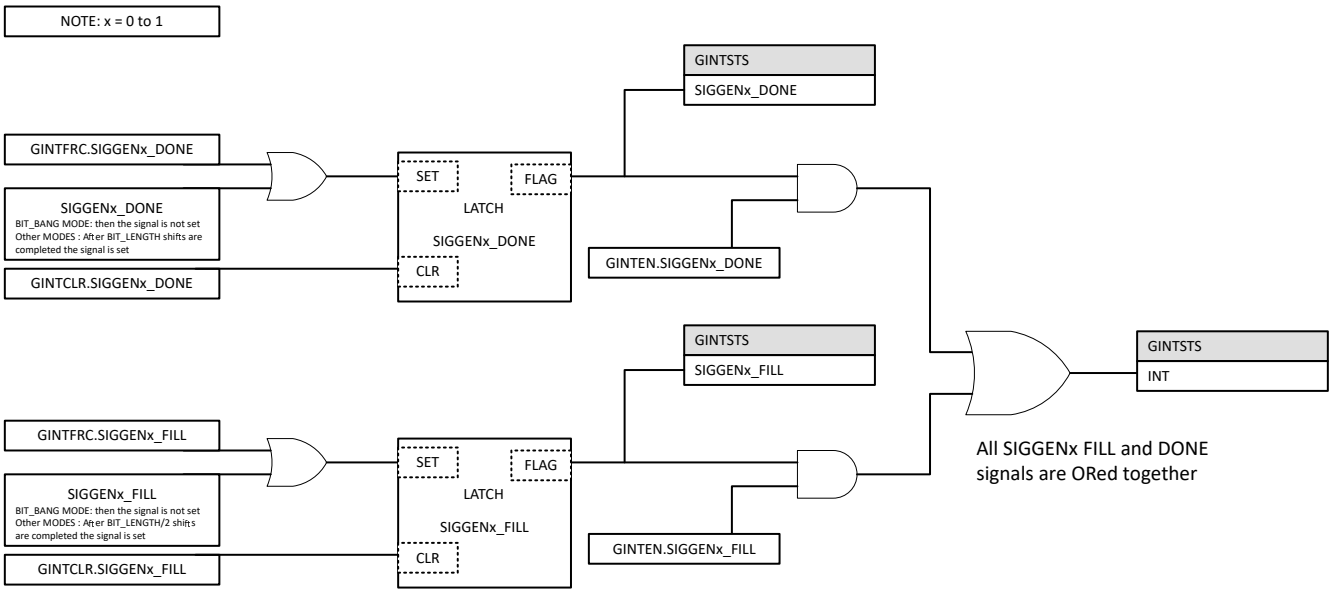


Figure 34-7. EPG Interrupt

34.7 Software

34.7.1 EPG Examples

NOTE: These examples are located in the C2000Ware installation at the following location: C2000Ware_VERSION#/driverlib/f28003x/examples/epg

Cloud access to these examples is available at the following link: dev.ti.com C2000Ware Examples.

34.7.1.1 EPG Generate Serial Data Shift Mode

FILE: epg_ex10_generate_serial_data_shift_mode.c

This example generates SPICLK and SPI DATA signals using the SIGGEN module in SHIFT mode. For more information on this example, visit: https://www.ti.com/lit/spracy7

External Connections

- None. Signal is generated on GPIO 58, 54. Can be visualized through oscilloscope.

34.7.1.2 EPG Generating Synchronous Clocks

FILE: epg_ex1_generate_clocks.c

This example shows how to generate 2 synchronous clocks with edges being offset by 2 clock cycles. It configures Signal Generator to shift a periodic data. Generated Clock has period EPG CLOCK/6.

External Connections

- None. Clock output is generated on GPIO 58, 54. Can be visualized through oscilloscope.

Watch Variables

- sigGenActiveData - Active Data of signal generator transform output

34.7.1.3 EPG Generating Two Offset Clocks

FILE: epg_ex7_generate_two_offset_clocks.c

This example generates two offset clocks using the CLKGEN (CLKDIV) modules. For more information on this example, visit: https://www.ti.com/lit/spracy7

External Connections

- None. Signal is generated on GPIO 58, 54. Can be visualized through oscilloscope.

34.7.1.4 EPG Generating Two Offset Clocks With SIGGEN

FILE: epg_ex8_generate_two_offset_clocks_with_siggen.c

This example generates two offset clocks using the SIGGEN module. For more information on this example, visit: https://www.ti.com/lit/spracy7

External Connections

- None. Signal is generated on GPIO 58, 54. Can be visualized through oscilloscope.

34.7.1.5 EPG Generate Serial Data

FILE: epg_ex9_generate_serial_data.c

This example generates SPICLK and SPI DATA signals using the SIGGEN module. For more information on this example, visit: https://www.ti.com/lit/spracy7

External Connections

- None. Signal is generated on GPIO 58, 54. Can be visualized through oscilloscope.

34.8 EPG Registers

The following sections are register descriptions for the EPG.

34.8.1 EPG Base Address Table

Table 34-3. EPG Base Address Table

Bit Field Name		DriverLib Name	Base Address	CPU1	DMA	HIC	CLA	Pipeline Protected
Instance	Structure							
Epg1Regs	EPG_REGS	EPG1_BASE	0x0005_EC00	YES	-	-	-	YES
Epg1MuxRegs	EPG_MUX_REGS	EPG1MUX_BASE	0x0005_ECD0	YES	-	-	-	YES

34.8.2 EPG_REGS Registers

Table 34-4 lists the memory-mapped registers for the EPG_REGS registers. All register offset addresses not listed in Table 34-4 should be considered as reserved locations and the register contents should not be modified.

Table 34-4. EPG_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	GCTL0	EPG Global control register 0		Go
2h	GCTL1	EPG Global control register 1		Go
4h	GCTL2	EPG Global control register 2		Go
6h	GCTL3	EPG Global control register 3		Go
8h	EPGLOCK	EPG LOCK Register		Go
Ah	EPGCOMMIT	EPG COMMIT register		Go
Ch	GINTSTS	EPG Global interrupt status register.		Go
Eh	GINTEN	EPG Global interrupt enable register.		Go
10h	GINTCLR	EPG Global interrupt clear register.		Go
12h	GINTFRC	EPG Global interrupt force register.		Go
18h	CLKDIV0_CTL0	Clock divider 0's control register 0		Go
1Eh	CLKDIV0_CLKOFFSET	Clock divider 0's clock offset value		Go
24h	CLKDIV1_CTL0	Clock divider 1's control register 0		Go
2Ah	CLKDIV1_CLKOFFSET	Clock divider 1's clock offset value		Go
30h	SIGGEN0_CTL0	Signal generator 0's control register 0		Go
32h	SIGGEN0_CTL1	Signal generator 0's control register 1		Go
38h	SIGGEN0_DATA0	Signal generator 0's data register 0		Go
3Ah	SIGGEN0_DATA1	Signal generator 0's data register 1		Go
3Ch	SIGGEN0_DATA0_ACTIVE	Signal generator 0's data active register 0		Go
3Eh	SIGGEN0_DATA1_ACTIVE	Signal generator 0's data active register 1		Go

Complex bit access types are encoded to fit into small table cells. Table 34-5 shows the codes that are used for access types in this section.

Table 34-5. EPG_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
R-0	R -0	Read Returns 0s
Write Type		
W	W	Write
W1C	W 1C	Write 1 to clear
W1S	W 1S	Write 1 to set
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		

Table 34-5. EPG_REGS Access Type Codes (continued)

Access Type	Code	Description
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

34.8.2.1 GCTL0 Register (Offset = 0h) [Reset = 0h]

GCTL0 is shown in Figure 34-8 and described in Table 34-6.

Return to the Summary Table.

EPG Global control register 0

Figure 34-8. GCTL0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
EPGOUT7SEL	EPGOUT6SEL	EPGOUT5SEL	EPGOUT4SEL	EPGOUT3SEL	EPGOUT2SEL	EPGOUT1SEL	EPGOUT0SEL
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
RESERVED							EN
R-0h							R/W-0h

Table 34-6. GCTL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-16	RESERVED	R	0h	Reserved
15	EPGOUT7SEL	R/W	0h	0 : Selects signal mux output 1 : Selects clock mux output Reset type: SYSRSn
14	EPGOUT6SEL	R/W	0h	0 : Selects signal mux output 1 : Selects clock mux output Reset type: SYSRSn
13	EPGOUT5SEL	R/W	0h	0 : Selects signal mux output 1 : Selects clock mux output Reset type: SYSRSn
12	EPGOUT4SEL	R/W	0h	0 : Selects signal mux output 1 : Selects clock mux output Reset type: SYSRSn
11	EPGOUT3SEL	R/W	0h	0 : Selects signal mux output 1 : Selects clock mux output Reset type: SYSRSn
10	EPGOUT2SEL	R/W	0h	0 : Selects signal mux output 1 : Selects clock mux output Reset type: SYSRSn
9	EPGOUT1SEL	R/W	0h	0 : Selects signal mux output 1 : Selects clock mux output Reset type: SYSRSn
8	EPGOUT0SEL	R/W	0h	0 : Selects signal mux output 1 : Selects clock mux output Reset type: SYSRSn
7-1	RESERVED	R	0h	Reserved
0	EN	R/W	0h	0 : EPG module is disabled 1 : EPG module is enabled, clock generators and signal generators are functional. Reset type: SYSRSn

34.8.2.2 GCTL1 Register (Offset = 2h) [Reset = 0h]

GCTL1 is shown in Figure 34-9 and described in Table 34-7.

Return to the Summary Table.

EPG Global control register 1

Figure 34-9. GCTL1 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	RESERVED			RESERVED	SIGGEN0_CLKSEL		
R/W-0h	R/W-0h			R/W-0h	R/W-0h		

Table 34-7. GCTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-8	RESERVED	R	0h	Reserved
7	RESERVED	R/W	0h	Reserved
6-4	RESERVED	R/W	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2-0	SIGGEN0_CLKSEL	R/W	0h	Clock source select of SIGGEN0: 0x0 : CLKGEN0.CLKOUT0_GCLK 0x1 : CLKGEN0.CLKOUT1_GCLK 0x2 : CLKGEN0.CLKOUT2_GCLK 0x3 : CLKGEN0.CLKOUT3_GCLK 0x4 : CLKGEN1.CLKOUT0_GCLK 0x5 : CLKGEN1.CLKOUT1_GCLK 0x6 : CLKGEN1.CLKOUT2_GCLK 0x7 : CLKGEN1.CLKOUT3_GCLK Reset type: SYSRSn

34.8.2.3 GCTL2 Register (Offset = 4h) [Reset = 0h]

GCTL2 is shown in Figure 34-10 and described in Table 34-8.

Return to the Summary Table.

EPG Global control register 2

Figure 34-10. GCTL2 Register

31	30	29	28	27	26	25	24
RESERVED	EPGOUT7_CLKOUTSEL			RESERVED	EPGOUT6_CLKOUTSEL		
R/W-0h	R/W-0h			R/W-0h	R/W-0h		
23	22	21	20	19	18	17	16
RESERVED	EPGOUT5_CLKOUTSEL			RESERVED	EPGOUT4_CLKOUTSEL		
R/W-0h	R/W-0h			R/W-0h	R/W-0h		
15	14	13	12	11	10	9	8
RESERVED	EPGOUT3_CLKOUTSEL			RESERVED	EPGOUT2_CLKOUTSEL		
R/W-0h	R/W-0h			R/W-0h	R/W-0h		
7	6	5	4	3	2	1	0
RESERVED	EPGOUT1_CLKOUTSEL			RESERVED	EPGOUT0_CLKOUTSEL		
R/W-0h	R/W-0h			R/W-0h	R/W-0h		

Table 34-8. GCTL2 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	RESERVED	R/W	0h	Reserved
30-28	EPGOUT7_CLKOUTSEL	R/W	0h	Output 7 signal source select: 0x0 : CLKGEN0.CLKOUT0_DCLK 0x1 : CLKGEN0.CLKOUT1_DCLK 0x2 : CLKGEN0.CLKOUT2_DCLK 0x3 : CLKGEN0.CLKOUT3_DCLK 0x4 : CLKGEN1.CLKOUT0_DCLK 0x5 : CLKGEN1.CLKOUT1_DCLK 0x6 : CLKGEN1.CLKOUT2_DCLK 0x7 : CLKGEN1.CLKOUT3_DCLK Reset type: SYSRSn
27	RESERVED	R/W	0h	Reserved
26-24	EPGOUT6_CLKOUTSEL	R/W	0h	Output 6 signal source select: 0x0 : CLKGEN0.CLKOUT0_DCLK 0x1 : CLKGEN0.CLKOUT1_DCLK 0x2 : CLKGEN0.CLKOUT2_DCLK 0x3 : CLKGEN0.CLKOUT3_DCLK 0x4 : CLKGEN1.CLKOUT0_DCLK 0x5 : CLKGEN1.CLKOUT1_DCLK 0x6 : CLKGEN1.CLKOUT2_DCLK 0x7 : CLKGEN1.CLKOUT3_DCLK Reset type: SYSRSn
23	RESERVED	R/W	0h	Reserved
22-20	EPGOUT5_CLKOUTSEL	R/W	0h	Output 5 signal source select: 0x0 : CLKGEN0.CLKOUT0_DCLK 0x1 : CLKGEN0.CLKOUT1_DCLK 0x2 : CLKGEN0.CLKOUT2_DCLK 0x3 : CLKGEN0.CLKOUT3_DCLK 0x4 : CLKGEN1.CLKOUT0_DCLK 0x5 : CLKGEN1.CLKOUT1_DCLK 0x6 : CLKGEN1.CLKOUT2_DCLK 0x7 : CLKGEN1.CLKOUT3_DCLK Reset type: SYSRSn
19	RESERVED	R/W	0h	Reserved

Table 34-8. GCTL2 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
18-16	EPGOUT4_CLKOUTSEL	R/W	0h	Output 4 signal source select: 0x0 : CLKGEN0.CLKOUT0_DCLK 0x1 : CLKGEN0.CLKOUT1_DCLK 0x2 : CLKGEN0.CLKOUT2_DCLK 0x3 : CLKGEN0.CLKOUT3_DCLK 0x4 : CLKGEN1.CLKOUT0_DCLK 0x5 : CLKGEN1.CLKOUT1_DCLK 0x6 : CLKGEN1.CLKOUT2_DCLK 0x7 : CLKGEN1.CLKOUT3_DCLK Reset type: SYSRSn
15	RESERVED	R/W	0h	Reserved
14-12	EPGOUT3_CLKOUTSEL	R/W	0h	Output 3 signal source select: 0x0 : CLKGEN0.CLKOUT0_DCLK 0x1 : CLKGEN0.CLKOUT1_DCLK 0x2 : CLKGEN0.CLKOUT2_DCLK 0x3 : CLKGEN0.CLKOUT3_DCLK 0x4 : CLKGEN1.CLKOUT0_DCLK 0x5 : CLKGEN1.CLKOUT1_DCLK 0x6 : CLKGEN1.CLKOUT2_DCLK 0x7 : CLKGEN1.CLKOUT3_DCLK Reset type: SYSRSn
11	RESERVED	R/W	0h	Reserved
10-8	EPGOUT2_CLKOUTSEL	R/W	0h	Output 2 signal source select: 0x0 : CLKGEN0.CLKOUT0_DCLK 0x1 : CLKGEN0.CLKOUT1_DCLK 0x2 : CLKGEN0.CLKOUT2_DCLK 0x3 : CLKGEN0.CLKOUT3_DCLK 0x4 : CLKGEN1.CLKOUT0_DCLK 0x5 : CLKGEN1.CLKOUT1_DCLK 0x6 : CLKGEN1.CLKOUT2_DCLK 0x7 : CLKGEN1.CLKOUT3_DCLK Reset type: SYSRSn
7	RESERVED	R/W	0h	Reserved
6-4	EPGOUT1_CLKOUTSEL	R/W	0h	Output 1 signal source select: 0x0 : CLKGEN0.CLKOUT0_DCLK 0x1 : CLKGEN0.CLKOUT1_DCLK 0x2 : CLKGEN0.CLKOUT2_DCLK 0x3 : CLKGEN0.CLKOUT3_DCLK 0x4 : CLKGEN1.CLKOUT0_DCLK 0x5 : CLKGEN1.CLKOUT1_DCLK 0x6 : CLKGEN1.CLKOUT2_DCLK 0x7 : CLKGEN1.CLKOUT3_DCLK Reset type: SYSRSn
3	RESERVED	R/W	0h	Reserved
2-0	EPGOUT0_CLKOUTSEL	R/W	0h	Output 0 signal source select: 0x0 : CLKGEN0.CLKOUT0_DCLK 0x1 : CLKGEN0.CLKOUT1_DCLK 0x2 : CLKGEN0.CLKOUT2_DCLK 0x3 : CLKGEN0.CLKOUT3_DCLK 0x4 : CLKGEN1.CLKOUT0_DCLK 0x5 : CLKGEN1.CLKOUT1_DCLK 0x6 : CLKGEN1.CLKOUT2_DCLK 0x7 : CLKGEN1.CLKOUT3_DCLK Reset type: SYSRSn

34.8.2.4 GCTL3 Register (Offset = 6h) [Reset = 0h]

GCTL3 is shown in Figure 34-11 and described in Table 34-9.

Return to the Summary Table.

EPG Global control register 3

Figure 34-11. GCTL3 Register

31	30	29	28	27	26	25	24
EPGOUT7_SIGOUTSEL				EPGOUT6_SIGOUTSEL			
R/W-0h				R/W-0h			
23	22	21	20	19	18	17	16
EPGOUT5_SIGOUTSEL				EPGOUT4_SIGOUTSEL			
R/W-0h				R/W-0h			
15	14	13	12	11	10	9	8
EPGOUT3_SIGOUTSEL				EPGOUT2_SIGOUTSEL			
R/W-0h				R/W-0h			
7	6	5	4	3	2	1	0
EPGOUT1_SIGOUTSEL				EPGOUT0_SIGOUTSEL			
R/W-0h				R/W-0h			

Table 34-9. GCTL3 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	EPGOUT7_SIGOUTSEL	R/W	0h	Output 7 source select: 0x0 : SIGGEN0.DATATRANOUT0 0x1 : SIGGEN0.DATATRANOUT1 0x2 : SIGGEN0.DATATRANOUT2 0x3 : SIGGEN0.DATATRANOUT3 0x4 : SIGGEN0.DATATRANOUT4 0x5 : SIGGEN0.DATATRANOUT5 0x6 : SIGGEN0.DATATRANOUT6 0x7 : SIGGEN0.DATATRANOUT7 0x8 : SIGGEN1.DATATRANOUT0 0x9 : SIGGEN1.DATATRANOUT1 0xA : SIGGEN1.DATATRANOUT2 0xB : SIGGEN1.DATATRANOUT3 0xC : SIGGEN1.DATATRANOUT4 0xD : SIGGEN1.DATATRANOUT5 0xE : SIGGEN1.DATATRANOUT6 0xF : SIGGEN1.DATATRANOUT7 Reset type: SYSRSn
27-24	EPGOUT6_SIGOUTSEL	R/W	0h	Output 6 source select: 0x0 : SIGGEN0.DATATRANOUT0 0x1 : SIGGEN0.DATATRANOUT1 0x2 : SIGGEN0.DATATRANOUT2 0x3 : SIGGEN0.DATATRANOUT3 0x4 : SIGGEN0.DATATRANOUT4 0x5 : SIGGEN0.DATATRANOUT5 0x6 : SIGGEN0.DATATRANOUT6 0x7 : SIGGEN0.DATATRANOUT7 0x8 : SIGGEN1.DATATRANOUT0 0x9 : SIGGEN1.DATATRANOUT1 0xA : SIGGEN1.DATATRANOUT2 0xB : SIGGEN1.DATATRANOUT3 0xC : SIGGEN1.DATATRANOUT4 0xD : SIGGEN1.DATATRANOUT5 0xE : SIGGEN1.DATATRANOUT6 0xF : SIGGEN1.DATATRANOUT7 Reset type: SYSRSn

Table 34-9. GCTL3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
23-20	EPGOUT5_SIGOUTSEL	R/W	0h	Output 5 source select: 0x0 : SIGGEN0.DATATRANOUT0 0x1 : SIGGEN0.DATATRANOUT1 0x2 : SIGGEN0.DATATRANOUT2 0x3 : SIGGEN0.DATATRANOUT3 0x4 : SIGGEN0.DATATRANOUT4 0x5 : SIGGEN0.DATATRANOUT5 0x6 : SIGGEN0.DATATRANOUT6 0x7 : SIGGEN0.DATATRANOUT7 0x8 : SIGGEN1.DATATRANOUT0 0x9 : SIGGEN1.DATATRANOUT1 0xA : SIGGEN1.DATATRANOUT2 0xB : SIGGEN1.DATATRANOUT3 0xC : SIGGEN1.DATATRANOUT4 0xD : SIGGEN1.DATATRANOUT5 0xE : SIGGEN1.DATATRANOUT6 0xF : SIGGEN1.DATATRANOUT7 Reset type: SYSRSn
19-16	EPGOUT4_SIGOUTSEL	R/W	0h	Output 4 source select: 0x0 : SIGGEN0.DATATRANOUT0 0x1 : SIGGEN0.DATATRANOUT1 0x2 : SIGGEN0.DATATRANOUT2 0x3 : SIGGEN0.DATATRANOUT3 0x4 : SIGGEN0.DATATRANOUT4 0x5 : SIGGEN0.DATATRANOUT5 0x6 : SIGGEN0.DATATRANOUT6 0x7 : SIGGEN0.DATATRANOUT7 0x8 : SIGGEN1.DATATRANOUT0 0x9 : SIGGEN1.DATATRANOUT1 0xA : SIGGEN1.DATATRANOUT2 0xB : SIGGEN1.DATATRANOUT3 0xC : SIGGEN1.DATATRANOUT4 0xD : SIGGEN1.DATATRANOUT5 0xE : SIGGEN1.DATATRANOUT6 0xF : SIGGEN1.DATATRANOUT7 Reset type: SYSRSn
15-12	EPGOUT3_SIGOUTSEL	R/W	0h	Output 3 source select: 0x0 : SIGGEN0.DATATRANOUT0 0x1 : SIGGEN0.DATATRANOUT1 0x2 : SIGGEN0.DATATRANOUT2 0x3 : SIGGEN0.DATATRANOUT3 0x4 : SIGGEN0.DATATRANOUT4 0x5 : SIGGEN0.DATATRANOUT5 0x6 : SIGGEN0.DATATRANOUT6 0x7 : SIGGEN0.DATATRANOUT7 0x8 : SIGGEN1.DATATRANOUT0 0x9 : SIGGEN1.DATATRANOUT1 0xA : SIGGEN1.DATATRANOUT2 0xB : SIGGEN1.DATATRANOUT3 0xC : SIGGEN1.DATATRANOUT4 0xD : SIGGEN1.DATATRANOUT5 0xE : SIGGEN1.DATATRANOUT6 0xF : SIGGEN1.DATATRANOUT7 Reset type: SYSRSn

Table 34-9. GCTL3 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
11-8	EPGOUT2_SIGOUTSEL	R/W	0h	Output 2 source select: 0x0 : SIGGEN0.DATATRANOUT0 0x1 : SIGGEN0.DATATRANOUT1 0x2 : SIGGEN0.DATATRANOUT2 0x3 : SIGGEN0.DATATRANOUT3 0x4 : SIGGEN0.DATATRANOUT4 0x5 : SIGGEN0.DATATRANOUT5 0x6 : SIGGEN0.DATATRANOUT6 0x7 : SIGGEN0.DATATRANOUT7 0x8 : SIGGEN1.DATATRANOUT0 0x9 : SIGGEN1.DATATRANOUT1 0xA : SIGGEN1.DATATRANOUT2 0xB : SIGGEN1.DATATRANOUT3 0xC : SIGGEN1.DATATRANOUT4 0xD : SIGGEN1.DATATRANOUT5 0xE : SIGGEN1.DATATRANOUT6 0xF : SIGGEN1.DATATRANOUT7 Reset type: SYSRSn
7-4	EPGOUT1_SIGOUTSEL	R/W	0h	Output 1 source select: 0x0 : SIGGEN0.DATATRANOUT0 0x1 : SIGGEN0.DATATRANOUT1 0x2 : SIGGEN0.DATATRANOUT2 0x3 : SIGGEN0.DATATRANOUT3 0x4 : SIGGEN0.DATATRANOUT4 0x5 : SIGGEN0.DATATRANOUT5 0x6 : SIGGEN0.DATATRANOUT6 0x7 : SIGGEN0.DATATRANOUT7 0x8 : SIGGEN1.DATATRANOUT0 0x9 : SIGGEN1.DATATRANOUT1 0xA : SIGGEN1.DATATRANOUT2 0xB : SIGGEN1.DATATRANOUT3 0xC : SIGGEN1.DATATRANOUT4 0xD : SIGGEN1.DATATRANOUT5 0xE : SIGGEN1.DATATRANOUT6 0xF : SIGGEN1.DATATRANOUT7 Reset type: SYSRSn
3-0	EPGOUT0_SIGOUTSEL	R/W	0h	Output 0 source select: 0x0 : SIGGEN0.DATATRANOUT0 0x1 : SIGGEN0.DATATRANOUT1 0x2 : SIGGEN0.DATATRANOUT2 0x3 : SIGGEN0.DATATRANOUT3 0x4 : SIGGEN0.DATATRANOUT4 0x5 : SIGGEN0.DATATRANOUT5 0x6 : SIGGEN0.DATATRANOUT6 0x7 : SIGGEN0.DATATRANOUT7 0x8 : SIGGEN1.DATATRANOUT0 0x9 : SIGGEN1.DATATRANOUT1 0xA : SIGGEN1.DATATRANOUT2 0xB : SIGGEN1.DATATRANOUT3 0xC : SIGGEN1.DATATRANOUT4 0xD : SIGGEN1.DATATRANOUT5 0xE : SIGGEN1.DATATRANOUT6 0xF : SIGGEN1.DATATRANOUT7 Reset type: SYSRSn

34.8.2.5 EPGLOCK Register (Offset = 8h) [Reset = 0h]

EPGLOCK is shown in Figure 34-12 and described in Table 34-10.

Return to the Summary Table.

EPG LOCK Register

Figure 34-12. EPGLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						RESERVED	RESERVED
R-0h						R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
SIGGEN0_CTL1	SIGGEN0_CTL0	CLKDIV1_CTL0	CLKDIV0_CTL0	GCTL3	GCTL2	GCTL1	GCTL0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 34-10. EPGLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9	RESERVED	R/W	0h	Reserved
8	RESERVED	R/W	0h	Reserved
7	SIGGEN0_CTL1	R/W	0h	0: Writes to SIGGEN0_CTL1 register is allowed. 1: Writes to SIGGEN0_CTL1 register is not allowed. Reset type: SYSRSn
6	SIGGEN0_CTL0	R/W	0h	0: Writes to SIGGEN0_CTL0 register is allowed. 1: Writes to SIGGEN0_CTL0 register is not allowed. Reset type: SYSRSn
5	CLKDIV1_CTL0	R/W	0h	0: Writes to CLKDIV1_CTL0 register is allowed. 1: Writes to CLKDIV1_CTL0 register is not allowed. Reset type: SYSRSn
4	CLKDIV0_CTL0	R/W	0h	0: Writes to CLKDIV0_CTL0 register is allowed. 1: Writes to CLKDIV0_CTL0 register is not allowed. Reset type: SYSRSn
3	GCTL3	R/W	0h	0: Writes to GCTL3 register is allowed. 1: Writes to GCTL3 register is not allowed. Reset type: SYSRSn
2	GCTL2	R/W	0h	0: Writes to GCTL2 register is allowed. 1: Writes to GCTL2 register is not allowed. Reset type: SYSRSn
1	GCTL1	R/W	0h	0: Writes to GCTL1 register is allowed. 1: Writes to GCTL1 register is not allowed. Reset type: SYSRSn
0	GCTL0	R/W	0h	0: Writes to GCTL0 register is allowed. 1: Writes to GCTL0 register is not allowed. Reset type: SYSRSn

34.8.2.6 EPGCOMMIT Register (Offset = Ah) [Reset = 0h]

EPGCOMMIT is shown in Figure 34-13 and described in Table 34-11.

Return to the Summary Table.

EPG COMMIT register

Figure 34-13. EPGCOMMIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED						RESERVED	RESERVED
R-0h						R/WSonce-0h	R/WSonce-0h
7	6	5	4	3	2	1	0
SIGGEN0_CTL1	SIGGEN0_CTL0	CLKDIV1_CTL0	CLKDIV0_CTL0	GCTL3	GCTL2	GCTL1	GCTL0
R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h	R/WSonce-0h

Table 34-11. EPGCOMMIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-10	RESERVED	R	0h	Reserved
9	RESERVED	R/WSonce	0h	Reserved
8	RESERVED	R/WSonce	0h	Reserved
7	SIGGEN0_CTL1	R/WSonce	0h	0: Writes to EPGLOCK.SIGGEN0_CTL1 field is allowed. 1: Writes to EPGLOCK.SIGGEN0_CTL1 field is not allowed. Reset type: SYSRSn
6	SIGGEN0_CTL0	R/WSonce	0h	0: Writes to EPGLOCK.SIGGEN0_CTL0 field is allowed. 1: Writes to EPGLOCK.SIGGEN0_CTL0 field is not allowed. Reset type: SYSRSn
5	CLKDIV1_CTL0	R/WSonce	0h	0: Writes to EPGLOCK.CLKDIV1_CTL0 field is allowed. 1: Writes to EPGLOCK.CLKDIV1_CTL0 field is not allowed. Reset type: SYSRSn
4	CLKDIV0_CTL0	R/WSonce	0h	0: Writes to EPGLOCK.CLKDIV0_CTL0 field is allowed. 1: Writes to EPGLOCK.CLKDIV0_CTL0 field is not allowed. Reset type: SYSRSn
3	GCTL3	R/WSonce	0h	0: Writes to EPGLOCK.GCTL3 field is allowed. 1: Writes to EPGLOCK.GCTL3 field is not allowed. Reset type: SYSRSn
2	GCTL2	R/WSonce	0h	0: Writes to EPGLOCK.GCTL2 field is allowed. 1: Writes to EPGLOCK.GCTL2 field is not allowed. Reset type: SYSRSn
1	GCTL1	R/WSonce	0h	0: Writes to EPGLOCK.GCTL1 field is allowed. 1: Writes to EPGLOCK.GCTL1 field is not allowed. Reset type: SYSRSn
0	GCTL0	R/WSonce	0h	0: Writes to EPGLOCK.GCTL0 field is allowed. 1: Writes to EPGLOCK.GCTL0 field is not allowed. Reset type: SYSRSn

34.8.2.7 GINTSTS Register (Offset = Ch) [Reset = 0h]

GINTSTS is shown in Figure 34-14 and described in Table 34-12.

Return to the Summary Table.

EPG Global interrupt status register.

Figure 34-14. GINTSTS Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			RESERVED	RESERVED	SIGGEN0_FILL	SIGGEN0_DONE	INT
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h

Table 34-12. GINTSTS Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	SIGGEN0_FILL	R/W	0h	0: Disable interrupt generation when SIGGEN0_FILL bits is set. 1: Enable interrupt generation when SIGGEN0_FILL bits is set. Reset type: SYSRSn
1	SIGGEN0_DONE	R/W	0h	0: Disable interrupt generation when SIGGEN0_DONE bits is set. 1: Enable interrupt generation when SIGGEN0_DONE bits is set. Reset type: SYSRSn
0	INT	R	0h	Global interrupt flag. 0 : If any of the enabled interrupt flags are set, generate an interrupt. 1 : Interrupts are not generated even if there are enabled interrupt flags. This bit is set when an interrupt is fired and cleared by writing to GINTCLR.INT bit. Reset type: SYSRSn

34.8.2.8 GINTEN Register (Offset = Eh) [Reset = 0h]

GINTEN is shown in Figure 34-15 and described in Table 34-13.

Return to the Summary Table.

EPG Global interrupt enable register.

Figure 34-15. GINTEN Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			RESERVED	RESERVED	SIGGEN0_FILL	SIGGEN0_DONE	RESERVED
R-0h			R/W-0h	R/W-0h	R/W-0h	R/W-0h	R-0h

Table 34-13. GINTEN Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reserved
4	RESERVED	R/W	0h	Reserved
3	RESERVED	R/W	0h	Reserved
2	SIGGEN0_FILL	R/W	0h	0: Disable interrupt generation when SIGGEN0_FILL bits is set. 1: Enable interrupt generation when SIGGEN0_FILL bits is set. Reset type: SYSRSn
1	SIGGEN0_DONE	R/W	0h	0: Disable interrupt generation when SIGGEN0_DONE bits is set. 1: Enable interrupt generation when SIGGEN0_DONE bits is set. Reset type: SYSRSn
0	RESERVED	R	0h	Reserved

34.8.2.9 GINTCLR Register (Offset = 10h) [Reset = 0h]

GINTCLR is shown in Figure 34-16 and described in Table 34-14.

Return to the Summary Table.

EPG Global interrupt clear register.

Figure 34-16. GINTCLR Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			RESERVED	RESERVED	SIGGEN0_FILL	SIGGEN0_DONE	INT
R-0h			R-0/W1C-0h	R-0/W1C-0h	R-0/W1C-0h	R-0/W1C-0h	R-0/W1C-0h

Table 34-14. GINTCLR Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reserved
4	RESERVED	R-0/W1C	0h	Reserved
3	RESERVED	R-0/W1C	0h	Reserved
2	SIGGEN0_FILL	R-0/W1C	0h	0: No effect 1: Clear SIGGEN0_FILL flag bit. Reset type: SYSRSn
1	SIGGEN0_DONE	R-0/W1C	0h	0: No effect 1: Clear SIGGEN0_DONE flag bit. Reset type: SYSRSn
0	INT	R-0/W1C	0h	0: No effect 1: Clear INT flag bit. Reset type: SYSRSn

34.8.2.10 GINTFRC Register (Offset = 12h) [Reset = 0h]

GINTFRC is shown in Figure 34-17 and described in Table 34-15.

Return to the Summary Table.

EPG Global interrupt force register.

Figure 34-17. GINTFRC Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED			RESERVED	RESERVED	SIGGEN0_FILL	SIGGEN0_DONE	RESERVED
R-0h			R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0/W1S-0h	R-0h

Table 34-15. GINTFRC Register Field Descriptions

Bit	Field	Type	Reset	Description
31-5	RESERVED	R	0h	Reserved
4	RESERVED	R-0/W1S	0h	Reserved
3	RESERVED	R-0/W1S	0h	Reserved
2	SIGGEN0_FILL	R-0/W1S	0h	0: No effect 1: set SIGGEN0_FILL flag bit. Reset type: SYSRSn
1	SIGGEN0_DONE	R-0/W1S	0h	0: No effect 1: set SIGGEN0_DONE flag bit. Reset type: SYSRSn
0	RESERVED	R	0h	Reserved

34.8.2.11 CLKDIV0_CTL0 Register (Offset = 18h) [Reset = 0h]

CLKDIV0_CTL0 is shown in Figure 34-18 and described in Table 34-16.

Return to the Summary Table.

Clock divider 0's control register 0

Figure 34-18. CLKDIV0_CTL0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED													CLKSTOP		
R-0h													R/W-0h		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								PRD							
R-0h								R/W-0h							

Table 34-16. CLKDIV0_CTL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Reserved
18-16	CLKSTOP	R/W	0h	Determines on which of the CLKOUTs edge clock generation is stopped following a clear of SIGGEN0_CTL0.EN. 000 : Stop on CLKOUT0 010 : Stop on CLKOUT1 100 : Stop on CLKOUT2 110 : Stop on CLKOUT3 Reset type: SYSRSn
15-8	RESERVED	R	0h	Reserved
7-0	PRD	R/W	0h	Clock divider period: Clock divider counter counts up to period (PRD) and snaps back to 0. Reset type: SYSRSn

34.8.2.12 CLKDIV0_CLKOFFSET Register (Offset = 1Eh) [Reset = 0h]

CLKDIV0_CLKOFFSET is shown in Figure 34-19 and described in Table 34-17.

Return to the Summary Table.

Clock divider 0's clock offset value

Figure 34-19. CLKDIV0_CLKOFFSET Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLK3OFFSET								CLK2OFFSET							
R/W-0h								R/W-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLK1OFFSET								CLK0OFFSET							
R/W-0h								R/W-0h							

Table 34-17. CLKDIV0_CLKOFFSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	CLK3OFFSET	R/W	0h	Number of source clock cycles by which the divided clock output 3 (CLKOUT3) is delayed. Reset type: SYSRSn
23-16	CLK2OFFSET	R/W	0h	Number of source clock cycles by which the divided clock output 2 (CLKOUT2) is delayed. Reset type: SYSRSn
15-8	CLK1OFFSET	R/W	0h	Number of source clock cycles by which the divided clock output 1 (CLKOUT1) is delayed. Reset type: SYSRSn
7-0	CLK0OFFSET	R/W	0h	Number of source clock cycles by which the divided clock output 0 (CLKOUT0) is delayed. Reset type: SYSRSn

34.8.2.13 CLKDIV1_CTL0 Register (Offset = 24h) [Reset = 0h]

CLKDIV1_CTL0 is shown in Figure 34-20 and described in Table 34-18.

Return to the Summary Table.

Clock divider 1's control register 0

Figure 34-20. CLKDIV1_CTL0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RESERVED													CLKSTOP		
R-0h													R/W-0h		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED								PRD							
R-0h								R/W-0h							

Table 34-18. CLKDIV1_CTL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-19	RESERVED	R	0h	Reserved
18-16	CLKSTOP	R/W	0h	Determines on which of the CLKOUTs edge clock generation is stopped following a clear of SIGGEN1_CTL0.EN. 000 : Stop on CLKOUT0 010 : Stop on CLKOUT1 100 : Stop on CLKOUT2 110 : Stop on CLKOUT3 Reset type: SYSRSn
15-8	RESERVED	R	0h	Reserved
7-0	PRD	R/W	0h	Clock divider period: Clock divider counter counts up to period (PRD) and snaps back to 0. Reset type: SYSRSn

34.8.2.14 CLKDIV1_CLKOFFSET Register (Offset = 2Ah) [Reset = 0h]

CLKDIV1_CLKOFFSET is shown in Figure 34-21 and described in Table 34-19.

Return to the Summary Table.

Clock divider 1's clock offset value

Figure 34-21. CLKDIV1_CLKOFFSET Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLK3OFFSET								CLK2OFFSET							
R/W-0h								R/W-0h							
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLK1OFFSET								CLK0OFFSET							
R/W-0h								R/W-0h							

Table 34-19. CLKDIV1_CLKOFFSET Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	CLK3OFFSET	R/W	0h	Number of source clock cycles by which the divided clock output 3 (CLKOUT3) is delayed. Reset type: SYSRSn
23-16	CLK2OFFSET	R/W	0h	Number of source clock cycles by which the divided clock output 2 (CLKOUT2) is delayed. Reset type: SYSRSn
15-8	CLK1OFFSET	R/W	0h	Number of source clock cycles by which the divided clock output 1 (CLKOUT1) is delayed. Reset type: SYSRSn
7-0	CLK0OFFSET	R/W	0h	Number of source clock cycles by which the divided clock output 0 (CLKOUT0) is delayed. Reset type: SYSRSn

34.8.2.15 SIGGEN0_CTL0 Register (Offset = 30h) [Reset = 0h]

SIGGEN0_CTL0 is shown in Figure 34-22 and described in Table 34-20.

Return to the Summary Table.

Signal generator 0's control register 0

Figure 34-22. SIGGEN0_CTL0 Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
BITLENGTH							
R/W-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED	BROUT	BRIN	EN	MODE			
R-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h			

Table 34-20. SIGGEN0_CTL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-24	RESERVED	R	0h	Reserved
23-16	BITLENGTH	R/W	0h	Defines the number bits which participates in the shift rotate operations. Reset type: SYSRSn
15-7	RESERVED	R	0h	Reserved
6	BROUT	R/W	0h	0 : No bit reversal on data output from data transform block 1 : Perform bit reversal on data output from data transform block Reset type: SYSRSn
5	BRIN	R/W	0h	0 : No bit reversal on data input of data transform block 1 : Perform bit reversal on data input of data transform block Reset type: SYSRSn
4	EN	R/W	0h	0 : Signal generator is disabled. 1 : Signal generator is enabled, signal generator functions as per the mode definition. Reset type: SYSRSn

Table 34-20. SIGGEN0_CTL0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
3-0	MODE	R/W	0h	0 : BIT_BANG mode, The value written into DATA0 and DATA1 registers appear on the signal generator outputs as is. 1 : SHIFT_RIGHT_ONCE mode, The data value written into (DATA1,DATA0) registers are shifted right by 1 on every clock. Shifting operations stops when BITLENGTH shifts are done and SIGGEN0_CTL0.EN bit is cleared. 2 : ROTATE_RIGHT_ONCE, The data value written into (DATA1,DATA0) registers are rotated right by 1 on every clock. Rotation happens within (DATA1,DATA0)[BITLENGTH-1:0] Rotate operations stops when BITLENGTH shifts are done and SIGGEN0_CTL0.EN bit is cleared. 3 : ROTATE_RIGHT_REPEAT, The data value written into (DATA1,DATA0) registers are rotated right by 1 on every clock. Rotation happens within (DATA1,DATA0)[BITLENGTH-1:0] Rotate operations continue until SIGGEN0_CTL0.EN bit is cleared. 4 : SHIFT_LEFT_ONCE mode, The data value written into (DATA1,DATA0) registers are shifted left by 1 on every clock. Shifting operations stops when BITLENGTH shifts are done and SIGGEN0_CTL0.EN bit is cleared. 5 : ROTATE_LEFT_ONCE, The data value written into (DATA1,DATA0) registers are rotated left by 1 on every clock. Rotation happens within (DATA1,DATA0)[BITLENGTH-1:0] Rotate operations stops when BITLENGTH shifts are done and SIGGEN0_CTL0.EN bit is cleared. 6 : ROTATE_LEFT_REPEAT, The data value written into (DATA1,DATA0) registers are rotated left by 1 on every clock. Rotation happens within (DATA1,DATA0)[BITLENGTH-1:0] Rotate operations continue until SIGGEN0_CTL0.EN bit is cleared. 7 : SHIFT_RIGHT_REPEAT mode, The data value written into (DATA1,DATA0) registers are shifted right by 1 on every clock. Shifting operations stops when SIGGEN0_CTL0.EN bit is cleared. 8 : SHIFT_LEFT_REPEAT mode, The data value written into (DATA1,DATA0) registers are shifted left by 1 on every clock. Shifting operations stops when SIGGEN0_CTL0.EN bit is cleared. Reset type: SYSRSn

34.8.2.16 SIGGEN0_CTL1 Register (Offset = 32h) [Reset = 0h]

SIGGEN0_CTL1 is shown in Figure 34-23 and described in Table 34-21.

Return to the Summary Table.

Signal generator 0's control register 1

Figure 34-23. SIGGEN0_CTL1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA63_INSEL				RESERVED											
R/W-0h				R-0h											
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RESERVED												DATA0_INSEL			
R-0h												R/W-0h			

Table 34-21. SIGGEN0_CTL1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-28	DATA63_INSEL	R/W	0h	Source input of bit 63 of Data register. If 0 delects DATA_NEXT[63] else, selects one of the EPGIN inputs. This provides the ability to capture the data. 0x0 : DATA_NEXT[63] 0x1 : EPGIN0 0x2 : EPGIN1 0x3 : EPGIN2 0x4 : EPGIN3 0x5 : EPGIN4 0x6 : EPGIN5 0x7 : EPGIN6 0x8 : EPGIN7 0x9-0xF : 0 Reset type: SYSRSn
27-4	RESERVED	R	0h	Reserved
3-0	DATA0_INSEL	R/W	0h	Source input of bit 0 of Data register. If 0 delects DATA_NEXT[0] else, selects one of the EPGIN inputs. This provides the ability to capture the data. 0x0 : DATA_NEXT[0] 0x1 : EPGIN0 0x2 : EPGIN1 0x3 : EPGIN2 0x4 : EPGIN3 0x5 : EPGIN4 0x6 : EPGIN5 0x7 : EPGIN6 0x8 : EPGIN7 0x9-0xF : 0 Reset type: SYSRSn

34.8.2.17 SIGGEN0_DATA0 Register (Offset = 38h) [Reset = 0h]

SIGGEN0_DATA0 is shown in Figure 34-24 and described in Table 34-22.

Return to the Summary Table.

Signal generator 0's data register 0

Figure 34-24. SIGGEN0_DATA0 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIGGEN_DATA0																															
R/W-0h																															

Table 34-22. SIGGEN0_DATA0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SIGGEN_DATA0	R/W	0h	Data used in signal bit stream. {SIGGEN_DATA1,SIGGEN_DATA0} together constitures a 64 bit data stream, which are modified as per the SIGGENx_CTL0.MODE configuration. Reset type: SYSRSn

34.8.2.18 SIGGEN0_DATA1 Register (Offset = 3Ah) [Reset = 0h]

SIGGEN0_DATA1 is shown in Figure 34-25 and described in Table 34-23.

Return to the Summary Table.

Signal generator 0's data register 1

Figure 34-25. SIGGEN0_DATA1 Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIGGEN_DATA1																															
R/W-0h																															

Table 34-23. SIGGEN0_DATA1 Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SIGGEN_DATA1	R/W	0h	Data used in signal bit stream. {SIGGEN_DATA1,SIGGEN_DATA0} together constitutes a 64 bit data stream, which are modified as per the SIGGENx_CTL0.MODE configuration. Reset type: SYSRSn

34.8.2.19 SIGGEN0_DATA0_ACTIVE Register (Offset = 3Ch) [Reset = 0h]

SIGGEN0_DATA0_ACTIVE is shown in Figure 34-26 and described in Table 34-24.

Return to the Summary Table.

Signal generator 0's data active register 0

Figure 34-26. SIGGEN0_DATA0_ACTIVE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIGEN_DATA0																															
R-0h																															

Table 34-24. SIGGEN0_DATA0_ACTIVE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SIGEN_DATA0	R	0h	This is the lower 32 bits of the 64 bit active register (used in data transformation) Reset type: SYSRSn

34.8.2.20 SIGGEN0_DATA1_ACTIVE Register (Offset = 3Eh) [Reset = 0h]

SIGGEN0_DATA1_ACTIVE is shown in Figure 34-27 and described in Table 34-25.

Return to the Summary Table.

Signal generator 0's data active register 1

Figure 34-27. SIGGEN0_DATA1_ACTIVE Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SIGGEN_DATA1																															
R-0h																															

Table 34-25. SIGGEN0_DATA1_ACTIVE Register Field Descriptions

Bit	Field	Type	Reset	Description
31-0	SIGGEN_DATA1	R	0h	This is the upper 32 bits of the 64 bit active register (used in data transformation) Reset type: SYSRSn

34.8.3 EPG_MUX_REGS Registers

Table 34-26 lists the memory-mapped registers for the EPG_MUX_REGS registers. All register offset addresses not listed in Table 34-26 should be considered as reserved locations and the register contents should not be modified.

Table 34-26. EPG_MUX_REGS Registers

Offset	Acronym	Register Name	Write Protection	Section
0h	EPGMXSEL0	EPG Mux select register 0		Go
Ch	EPGMXSELLOCK	EPG Mux select register lock		Go
Eh	EPGMXSELCOMMIT	EPG Mux select register commit		Go

Complex bit access types are encoded to fit into small table cells. Table 34-27 shows the codes that are used for access types in this section.

Table 34-27. EPG_MUX_REGS Access Type Codes

Access Type	Code	Description
Read Type		
R	R	Read
Write Type		
W	W	Write
WSonce	W Sonce	Write Set once
Reset or Default Value		
-n		Value after reset or the default value
Register Array Variables		
i,j,k,l,m,n		When these variables are used in a register name, an offset, or an address, they refer to the value of a register array where the register is part of a group of repeating registers. The register groups form a hierarchical structure and the array is represented with a formula.
y		When this variable is used in a register name, an offset, or an address it refers to the value of a register array.

34.8.3.1 EPGMXSEL0 Register (Offset = 0h) [Reset = 0h]

EPGMXSEL0 is shown in Figure 34-28 and described in Table 34-28.

Return to the Summary Table.

EPG Mux select register 0

Figure 34-28. EPGMXSEL0 Register

31	30	29	28	27	26	25	24
SEL31	SEL30	SEL29	SEL28	SEL27	SEL26	SEL25	SEL24
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
23	22	21	20	19	18	17	16
SEL23	SEL22	SEL21	SEL20	SEL19	SEL18	SEL17	SEL16
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
15	14	13	12	11	10	9	8
SEL15	SEL14	SEL13	SEL12	SEL11	SEL10	SEL9	SEL8
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h
7	6	5	4	3	2	1	0
SEL7	SEL6	SEL5	SEL4	SEL3	SEL2	SEL1	SEL0
R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h	R/W-0h

Table 34-28. EPGMXSEL0 Register Field Descriptions

Bit	Field	Type	Reset	Description
31	SEL31	R/W	0h	0: DATAIN[31] is selected 1: EPGOUT[7] is selected Reset type: SYSRSn
30	SEL30	R/W	0h	0: DATAIN[30] is selected 1: EPGOUT[6] is selected Reset type: SYSRSn
29	SEL29	R/W	0h	0: DATAIN[29] is selected 1: EPGOUT[5] is selected Reset type: SYSRSn
28	SEL28	R/W	0h	0: DATAIN[28] is selected 1: EPGOUT[4] is selected Reset type: SYSRSn
27	SEL27	R/W	0h	0: DATAIN[27] is selected 1: EPGOUT[3] is selected Reset type: SYSRSn
26	SEL26	R/W	0h	0: DATAIN[26] is selected 1: EPGOUT[2] is selected Reset type: SYSRSn
25	SEL25	R/W	0h	0: DATAIN[25] is selected 1: EPGOUT[1] is selected Reset type: SYSRSn
24	SEL24	R/W	0h	0: DATAIN[24] is selected 1: EPGOUT[0] is selected Reset type: SYSRSn
23	SEL23	R/W	0h	0: DATAIN[23] is selected 1: EPGOUT[7] is selected Reset type: SYSRSn
22	SEL22	R/W	0h	0: DATAIN[22] is selected 1: EPGOUT[6] is selected Reset type: SYSRSn

Table 34-28. EPGMXSEL0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
21	SEL21	R/W	0h	0: DATAIN[21] is selected 1: EPGOUT[5] is selected Reset type: SYSRSn
20	SEL20	R/W	0h	0: DATAIN[20] is selected 1: EPGOUT[4] is selected Reset type: SYSRSn
19	SEL19	R/W	0h	0: DATAIN[19] is selected 1: EPGOUT[3] is selected Reset type: SYSRSn
18	SEL18	R/W	0h	0: DATAIN[18] is selected 1: EPGOUT[2] is selected Reset type: SYSRSn
17	SEL17	R/W	0h	0: DATAIN[17] is selected 1: EPGOUT[1] is selected Reset type: SYSRSn
16	SEL16	R/W	0h	0: DATAIN[16] is selected 1: EPGOUT[0] is selected Reset type: SYSRSn
15	SEL15	R/W	0h	0: DATAIN[15] is selected 1: EPGOUT[7] is selected Reset type: SYSRSn
14	SEL14	R/W	0h	0: DATAIN[14] is selected 1: EPGOUT[6] is selected Reset type: SYSRSn
13	SEL13	R/W	0h	0: DATAIN[13] is selected 1: EPGOUT[5] is selected Reset type: SYSRSn
12	SEL12	R/W	0h	0: DATAIN[12] is selected 1: EPGOUT[4] is selected Reset type: SYSRSn
11	SEL11	R/W	0h	0: DATAIN[11] is selected 1: EPGOUT[3] is selected Reset type: SYSRSn
10	SEL10	R/W	0h	0: DATAIN[10] is selected 1: EPGOUT[2] is selected Reset type: SYSRSn
9	SEL9	R/W	0h	0: DATAIN[9] is selected 1: EPGOUT[1] is selected Reset type: SYSRSn
8	SEL8	R/W	0h	0: DATAIN[8] is selected 1: EPGOUT[0] is selected Reset type: SYSRSn
7	SEL7	R/W	0h	0: DATAIN[7] is selected 1: EPGOUT[7] is selected Reset type: SYSRSn
6	SEL6	R/W	0h	0: DATAIN[6] is selected 1: EPGOUT[6] is selected Reset type: SYSRSn
5	SEL5	R/W	0h	0: DATAIN[5] is selected 1: EPGOUT[5] is selected Reset type: SYSRSn
4	SEL4	R/W	0h	0: DATAIN[4] is selected 1: EPGOUT[4] is selected Reset type: SYSRSn
3	SEL3	R/W	0h	0: DATAIN[3] is selected 1: EPGOUT[3] is selected Reset type: SYSRSn

Table 34-28. EPGMXSEL0 Register Field Descriptions (continued)

Bit	Field	Type	Reset	Description
2	SEL2	R/W	0h	0: DATAIN[2] is selected 1: EPGOUT[2] is selected Reset type: SYSRSn
1	SEL1	R/W	0h	0: DATAIN[1] is selected 1: EPGOUT[1] is selected Reset type: SYSRSn
0	SEL0	R/W	0h	0: DATAIN[0] is selected 1: EPGOUT[0] is selected Reset type: SYSRSn

34.8.3.2 EPGMXSELLOCK Register (Offset = Ch) [Reset = 0h]

EPGMXSELLOCK is shown in Figure 34-29 and described in Table 34-29.

Return to the Summary Table.

EPG Mux select register lock

Figure 34-29. EPGMXSELLOCK Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						RESERVED	EPGMXSEL0
R-0h						R/W-0h	R/W-0h

Table 34-29. EPGMXSELLOCK Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	RESERVED	R/W	0h	Reserved
0	EPGMXSEL0	R/W	0h	0: Writes to EPGMXSEL0 registers are allowed. 1: Writes to EPGMXSEL0 registers are not allowed. Reset type: SYSRSn

34.8.3.3 EPGMXSELCOMMIT Register (Offset = Eh) [Reset = 0h]

EPGMXSELCOMMIT is shown in Figure 34-30 and described in Table 34-30.

Return to the Summary Table.

EPG Mux select register commit

Figure 34-30. EPGMXSELCOMMIT Register

31	30	29	28	27	26	25	24
RESERVED							
R-0h							
23	22	21	20	19	18	17	16
RESERVED							
R-0h							
15	14	13	12	11	10	9	8
RESERVED							
R-0h							
7	6	5	4	3	2	1	0
RESERVED						RESERVED	EPGMXSEL0
R-0h						R/WSonce-0h	R/WSonce-0h

Table 34-30. EPGMXSELCOMMIT Register Field Descriptions

Bit	Field	Type	Reset	Description
31-2	RESERVED	R	0h	Reserved
1	RESERVED	R/WSonce	0h	Reserved
0	EPGMXSEL0	R/WSonce	0h	0: Writes to EPGMXSELLOCK.EPGMXSEL12 field is allowed. 1: Writes to EPGMXSELLOCK.EPGMXSEL12 field is not allowed. Reset type: SYSRSn

34.8.4 EPG Registers to Driverlib Functions

Table 34-31. EPG Registers to Driverlib Functions

File	Driverlib Function
GCTL0	
epg.h	EPG_enableGlobal
epg.h	EPG_disableGlobal
epg.h	EPG_selectEPGOutput
GCTL1	
epg.h	EPG_selectSigGenClkSource
GCTL2	
epg.h	EPG_selectClkOutput
GCTL3	
epg.h	EPG_selectSignalOutput
LOCK	
epg.h	EPG_lockReg
epg.h	EPG_releaseLockReg
COMMIT	
epg.h	EPG_commitRegLock
GINTSTS	
epg.h	EPG_getInterruptStatus

Table 34-31. EPG Registers to Driverlib Functions (continued)

File	Driverlib Function
GINTEN	
epg.h	EPG_enableInterruptFlag
epg.h	EPG_disableInterruptFlag
GINTCLR	
epg.h	EPG_clearInterruptFlag
GINTFRC	
epg.h	EPG_forceInterruptFlag
CLKDIV0_CTL0	
epg.h	EPG_setClkGenPeriod
epg.h	EPG_setClkGenStopEdge
CLKDIV0_CLKOFFSET	
epg.h	EPG_setClkGenOffset
CLKDIV1_CTL0	
-	See CLKDIV0_CTL0
CLKDIV1_CLKOFFSET	
-	See CLKDIV0_CLKOFFSET
SIGGEN0_CTL0	
epg.h	EPG_enableSignalGen
epg.h	EPG_disableSignalGen
epg.h	EPG_setSignalGenMode
epg.h	EPG_enableBitRevOnDataIn
epg.h	EPG_disableBitRevOnDataIn
epg.h	EPG_enableBitRevOnDataOut
epg.h	EPG_disableBitRevOnDataOut
epg.h	EPG_setDataBitLen
SIGGEN0_CTL1	
epg.h	EPG_setData0In
epg.h	EPG_setData63In
SIGGEN0_DATA0	
epg.h	EPG_setData0Word
epg.h	EPG_getData0ActiveReg
SIGGEN0_DATA1	
epg.h	EPG_setData1Word
epg.h	EPG_getData1ActiveReg
SIGGEN0_DATA0_ACTIVE	
epg.h	EPG_getData0ActiveReg
SIGGEN0_DATA1_ACTIVE	
epg.h	EPG_getData1ActiveReg
MXSEL0	
epg.c	EPG_selectEPGDataOut
MXSELLOCK	
epg.h	EPG_lockMXSelReg
epg.h	EPG_releaseLockMXSelReg
MXSELCOMMIT	
epg.h	EPG_commitMXSelRegLock

Revision History

Changes from October 20, 2021 to March 4, 2022 (from Revision * (October 2020) to Revision A (March 2022)) **Page**

- Added Section 5.1.1 592
- Added Section 6.1.1 732
- Added Section 19.1.1 2018
- Added Section 32.1.1 3240
- Added Section 32.7.1 3273
- Added Section 33.5.1 3388

This page intentionally left blank.

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265