MC9S12XF512 Reference Manual

Covers
MC9S12XF512
MC9S12XF384
MC9S12XF256
MC9S12XF128

S12X
Microcontrollers

MC9S12XF512RMV1
Rev.1.20
10-Nov-2010

freescale.com

freescale™
semiconductor

To provide the most up-to-date information, the revision of our documents on the World Wide Web will be the most current. Your printed copy may be an earlier revision. To verify you have the latest information available, refer to:
http://freescale.com/

A full list of family members and options is included in the appendices.

The following revision history table summarizes changes contained in this document.

This document contains information for all constituent modules, with the exception of the S12 CPU. For S12 CPU information please refer to the CPU S12 Reference Manual.

Revision History

Date	Revision Level	Description
06-Dec-2007	1.12	Updated memory map description for family parts (1.1.4 MC9S12XF512 - Address Mapping) Updated derivative differences w.r.t. DFlash size (D.1 Memory Sizes and Package Options S12XF - Family)
12-Dec-2007	1.13	Add FTM BG (384K2/256K2)
08-Jan-2008	1.14	Remove table for Module Run Supply Currents (A-10) Remove 3.3V columns in Table A-27, 3.0V columns in Table A-28 Add FTM BG (128K2)
15-Jan-2008	1.15	Fixed typo in detailed register map (SPI1/SPI1DRH) Import updated BGs VREG, ECT, INT, DBG Fixed typo in Table 1-6
05-Mar-2008	1.16	Updated ordering info for 112 LQFP
02-Oct-2008	1.17	Updated NVM timing parameter section for brownout case Specified time delay from RESET to start of CPU code execution Added NVM patch Part IDs Enhanced ECT GPIO / timer function transitioning description CRG section updated
01-Mar-2010	1.18	Updated PIM,FTM,XGATE,MSCAN,DBG,BDM,ADC,CRG sections Corrected startup from reset min cycle count
18-May-2010	1.19	Updated ECT section (see ECT revision history table)
10-Nov-2010	1.20	Fixed PT typo in SCI section Corrected oscillator frequency reference in Flexray section Fixed maximum deadtime delay counts for PMFDTMA and PMFDTMB in PMF section Updated part number note in Appendix

Chapter 1
MC9S12XF-Family Reference Manual

Chapter 2
S12XE Clocks and Reset Generator (S12XECRG)

Chapter 3
Voltage Regulator (S12VREGL3V3V1)

Chapter 4
384 KByte Flash Module (S12XFTM384K2V1)

Chapter 5
Pierce Oscillator (S12OSCLCPV2)

Chapter 6
Security (S12XFSECV2)

Chapter 7Interrupt (S12XINTV2)

Chapter 8
256 KByte Flash Module (S12XFTM256K2XFV1)

Chapter 9
512 KByte Flash Module (S12XFTM512K3V1)

Chapter 10
128 KByte Flash Module (S12XFTM128K2XFV1)

Chapter 11
Memory Mapping Control (S12XMMCV4) SUPPORTING FLEXRAY

Chapter 12
Clock Generation Module using IPLL (CGMIPLL) Block Description

Chapter 13FlexRay Communication Controller (FLEXRAY)

Chapter 14
XGATE (S12XGATEV3)

Chapter 15
Background Debug Module (S12XBDMV2)

Chapter 16
S12X Debug (S12XDBGV3) Module

Chapter 17
Memory Protection Unit (S12XMPUV2)

Chapter 18
External Bus Interface (S12XEBIV4)

Chapter 19
Port Integration Module (S12XFPIMV2)

Chapter 20
Pulse Width Modulator with Fault Protection (PMF15B6C) Module

Chapter 21
Freescale's Scalable Controller Area Network (S12MSCANV2)

Chapter 22
Enhanced Programmable Interrupt Timer (S12XEPIT24B8CV1)

Chapter 23
Serial Communication Interface (S12SCIV5)

Chapter 24
Analog-to-Digital Converter (ADC12B16C)
Block Description

Chapter 25
Serial Peripheral Interface (S12SPIV5)

Chapter 26
Enhanced Capture Timer (ECT16B8CV3)

Appendix A
Electrical Characteristics

Appendix B
Package Physical Dimension Information.

Appendix C
PCB Layout Guidelines

Appendix D
Derivative Differences

Appendix E
Detailed Register Address Map

Appendix F
Ordering Information

Chapter 1
MC9S12XF-Family Reference Manual

1.1 Introduction

Targeted at actuators, sensors and other distributed nodes in the FlexRay network for Chassis and Body Electronics, the MC9S12XF-Family delivers 32-bit performance with all the advantages and efficiencies of a 16-bit MCU. The design goal was to retain the low cost, power consumption, EMC and code-size efficiency advantages currently enjoyed by users of Freescale Semiconductor's other 16-bit MC9S12 MCU families.

Based around an enhanced S12X core, the MC9S12XF-Family runs 16-bit wide accesses without wait states for all peripherals and memories. The MC9S12XF-Family also features a new flexible interrupt handler, which allows multilevel nested interrupts.

The MC9S12XF-Family features the performance boosting enhanced XGATE co-processor. The XGATE is programmable in “C” language and runs at twice the bus frequency of the S12. The XGATE instruction set is optimized for data movement, logic and bit manipulation instructions. Any peripheral module can be serviced by the XGATE.

The MC9S12XF-Family features a Memory Protection Unit (MPU).

The MC9S12XF-Family features a FlexRay module for high speed serial communication supporting various bit rates up to 10 Mbit/s. The FlexRay internal clock can be generated from crystals ranging from 4MHz to 40MHz[1]. The 64-pin LQFP allows interfacing to a single FlexRay channel. The 64-pin LQFP (10mm x 10mm) is intended for those applications challenged by the size constraint of some satellite FlexRay modules. The 112-pin LQFP offers an increase in the number of I/Os as well as 16 A/D channels. In addition to that the 144-pin LQFP provides a full 16-bit wide non-multiplexed external bus interface with the pins usable as general purpose I/O in single-chip modes.

The MC9S12XF-Family features the MSCAN module with a FIFO receiver buffer arrangement, and input filters optimized for Gateway applications handling numerous message identifiers.

The MC9S12XF-Family provides Flash memory sizes from 128K to 512K plus Data Flash and enhanced EEPROM functionality (EE-Emulation) with built in Error Correcting Code (ECC). The memory uses Freescale Semiconductor's industry-leading, full automotive qualified SG-Flash.

The inclusion of a frequency modulated PLL circuit allows power consumption and performance to be adjusted to suit operational requirements and allows optimization of the radiated emissions (EMC).

The ATD now offers 12 Bit resolution at a faster conversion rate down to 3µs per channel.

The MC9S12XF512 is available in 144-Pin LQFP, 112-Pin LQFP and 64-Pin LQFP package.

1. 8MHz - 16MHz recommended for low jitter

NOTE

The 144-Pin LQFP version will not be qualified for production and is intended to be used for emulation (development tools) only.

See Table 1-9 for information about port and peripheral availability by package option.

1.1.1 Features

Features of the MC9S12XF-Family are listed here. Please see Table 1-1 for memory options and Table 1-2 for the peripheral features that are available on the different family members.

The system includes these distinctive features:

- 16-Bit CPU12X
  - Upward compatible with MC9S12 instruction set with the exception of five Fuzzy instructions (MEM, WAV, WAVR, REV, REVW) which have been removed.
  - Enhanced indexed addressing
  - Additional (superset) instructions to improve 32-bit calculations and semaphore handling
  - Access large data segments independent of PPAGE
- Enhanced Interrupt Module
  - Eight levels of nested interrupts
  - Flexible assignment of interrupt sources to each interrupt level
  - One non-maskable high priority interrupt (XIRQ)
  - Wake-up Interrupt Inputs
- Memory Protection Unit (MPU)
  - 4 address regions definable per active program task
  - Address range granularity as low as 256-bytes
  - Protection Attributes
    - No write
    - No execute
  - Non-maskable interrupt on access violation
- XGATE
  - Programmable, high performance I/O coprocessor module – up to 100 MIPS RISC performance
  - Transfers data to or from all peripherals and RAM without CPU intervention or CPU wait states
  - Performs logical, shifts, arithmetic, and bit operations on data
  - Can interrupt the HCS12X CPU signalling transfer completion
  - Triggers from any hardware module as well as from the CPU possible
  - Two interrupt levels to service high priority tasks
  - Enables Full CAN capability when used in conjunction with MSCAN module

  - Full LIN master or slave capability when used in conjunction with the integrated LIN SCI module
- System Integrity Support
  - Power-on reset (POR)
  - Illegal address detection with reset
  - Low-voltage detection with interrupt or reset
  - Computer Operating Properly (COP) watchdog
    - Configurable as window COP for enhanced failure detection
    - Can be initialized out of reset using option bits located in Flash
  - Clock monitor supervising the correct function of the oscillator
- Memory Options
  - 128K, 256k, 384K and 512K byte Flash
  - 2K, 4K byte Emulated EEPROM
  - 16K, 20K, 24K and 32K Byte RAM
  - Program Flash General Features
    - 64 data bits plus 8 syndrome ECC (Error Correction Code) bits allow single bit fault correction and double fault detection
    - Erase sector size 1024 bytes
    - Automated program and erase algorithm
    - Security option to prevent unauthorized access
    - Sense-amp margin level setting for reads
  - Data Flash General Features
    - Up to 32K bytes of D-Flash memory with 256-byte sectors for user access.
    - Dedicated commands to access D-Flash memory over EEE operation
    - Single bit fault correction and double fault detection within a word during read operations
    - Automated program and erase algorithm with verify and generation of ECC parity bits
    - Fast sector erase and word program operation
    - Ability to program up to four words in a burst sequence
  - Emulated EEPROM General Features
    - Automatic EEE file handling using internal Memory Controller
    - Automatic transfer of valid EEE data from D-Flash memory to buffer RAM on reset
    - Ability to monitor the number of outstanding EEE related buffer RAM words left to be programmed into D-Flash memory
    - Ability to disable EEE operation and allow priority access to the D-Flash memory
    - Ability to cancel all pending EEE operations to allow priority access to the D-Flash memory
- Oscillator (OSC_LCP)
  - Loop Control Pierce oscillator utilizing a 4MHz to 16MHz crystal
  - Good noise immunity

  - Full-swing Pierce option utilizing a 2MHz to 40MHz crystal
  - Transconductance sized for optimum start-up margin for typical crystals
- Clock and Reset Generator (CRG)
  - Phase-locked-loop (IPLL) clock frequency multiplier
  - Internally filtered. No external components required
  - Configurable option to spread spectrum for reduced EMC radiation (frequency modulation)
  - Fast wake up from STOP in self clock mode for power saving and immediate program execution
- Non-Multiplexed External Bus (144 Pin package only)
  - 16 data bits wide
  - Support for external WAIT input or internal wait cycles to adapt MCU speed to peripheral speed requirements
  - Up to four chip select outputs to select 16K, 768K, 2M and 4MByte address spaces
  - Supports glue less interface to popular asynchronous RAMs and Flash devices
  - External address space 4MByte for Data and Program space
- FlexRay Module (FR)
  - FlexRay protocol implementation according to FlexRay V2.1 Protocol Implementation document
  - Optimized programmers model to fit small address footprint
  - Supports Data Rates of 2.5, 5, 8 and 10MBit/s
  - The FlexRay clock can be derived from crystals ranging from 4MHz to 40MHz for cost and EMC optimization
  - FlexRay clocking independent from the CPU and XGATE bus frequency. Clock is generated by a “dedicated” IPLL.
  - Up to two channels for fault tolerant systems (see Table 1-2 Peripheral Feature Summary of MC9S12XF-Family Members)
  - Single channel operation on channel A, configurable to run FlexRay channel A or channel B protocol
  - 32 configurable message buffers
  - Message buffers can be configured as Receive, single buffered Transmit or double buffered Transmit message buffer
  - Message buffer header, status and payload data stored in System RAM
  - 2 independent message buffer segments
  - Size of message buffer payload data section configurable from 0 up to 254 bytes
  - 2 independent receive FIFOs, 1 per channel
  - Six separate interrupt channels for Receive, receive FIFO channel A, receive FIFO channel B, Transmit, Error and Wake-up
  - Internal signals can be routed to I/O pins to ease debugging
- Analog-to-Digital Converter (ATD)

- 8/10/12 Bit resolution
- Multiplexer for 16 analog input channels
- 3µs, 12-bit single conversion time
- Left or right justified result data
- External and internal conversion trigger capability
- Internal oscillator for conversion in Stop Modes
- Wake-up from low power modes on analog comparison > or <= match

- Enhanced Capture Timer (ECT)
  - 8 x 16-bit channels for input capture or output compare
  - 16-bit free-running counter with 8-bit precision prescaler
  - 16-bit modulus down counter with 8-bit precision prescaler
  - 4 x 8-bit or 2 x 16-bit pulse accumulators
  - Four channels have enhanced input capture capabilities:
    - Delay counter for noise immunity
    - 16-bit capture buffer
    - 8-bit pulse accumulator buffer
- Enhanced Periodic Interrupt Timer (EPIT)
  - Up to 8 timers with independent time-out periods
  - Time-out periods selectable between 1 and 2^{24} bus clock cycles
  - Time-out interrupt and peripheral triggers
- Real Time Interrupt (RTI)
  - Real Time Interrupt for task scheduling purposes or cyclic wake-up
  - Can be active in Pseudo Stop mode for low power precision timing tasks
- Asynchronous Periodic Interrupt (API)
  - Available in all modes including Full Stop mode
  - Trimmable to +-5% accuracy
  - Time-out periods range from 0.2ms to ~13s with a 0.2ms resolution
- Pulse Width Modulator with Fault detection (PMF)
  - Six channel Pulse width Modulator with Fault protection (PMF) optimized for electrical motor control
  - Three independent 15-bit counters with synchronous mode
  - Complementary channel operation
  - Edge and center aligned PWM signals
  - Programmable dead time insertion
  - Integral reload rates from 1 to 16
  - Up to four fault protection shut down input pins depending on the package option
  - Up to three current sense input pins depending on the package option (see Table 1-9 Port and Peripheral Availability by Package Option)

- Multi-scalable Controller Area Networks (MSCAN)
  - CAN 2.0 A, B software compatible
    - Standard and extended data frames
    - 0 - 8 bytes data length
    - Programmable bit rate up to 1 Mbps
  - Five receive buffers with FIFO storage scheme
  - Three transmit buffers with internal prioritization
  - Flexible identifier acceptance filter programmable as:
    - 2 x 32-bit
    - 4 x 16-bit
    - 8 x 8-bit
  - Wake-up with integrated low pass filter option
  - Loop back for self test
  - Listen-only mode to monitor CAN bus
  - Bus-off recovery by software intervention or automatically
  - 16-bit time stamp of transmitted/received messages
- Serial Peripheral Interface (SPI)
  - Up to two SPI modules (see Table 1-2 Peripheral Feature Summary of MC9S12XF-Family Members)
  - Configurable 8 or 16-bit data size
  - Full-duplex or single-wire bidirectional
  - Double-buffered transmit and receive
  - Master or Slave mode
  - MSB-first or LSB-first shifting
  - Serial clock phase and polarity options
- Serial Communication Interfaces (SCI)
  - Up to two SCI modules (see Table 1-2 Peripheral Feature Summary of MC9S12XF-Family Members)
  - Full-duplex or single wire operation
  - Standard mark/space non-return-to-zero (NRZ) format
  - Selectable IrDA 1.4 return-to-zero-inverted (RZI) format with programmable pulse widths
  - 13-bit baud rate selection
  - Programmable character length
  - Programmable polarity for transmitter and receiver
  - Receive wakeup on active edge
  - Break detect and transmit collision detect supporting LIN
- Background Debug Module (BDM)
  - Background debug controller (BDM) with single-wire interface

  - Non-intrusive memory access commands
  - Supports in-circuit programming of on-chip non-volatile memory
  - Supports security

- Debugger (DBG)
  - Four comparators A, B, C and D
    - Each can monitor CPU or XGATE busses
    - A and C compares 23-bit address bus and 16-bit data bus with mask register
    - B and D compares 23-bit address bus only
    - Three modes: simple address/data match, inside address range or outside address range
  - 64 x 64-bit circular trace buffer to capture change-of-flow addresses or address and data of every access
  - Tag-type or force-type hardware breakpoint requests
- Input/Output
  - up to 110 general-purpose input/output (I/O) pins depending on the package option and 2 input-only pins
  - Hysteresis and configurable pull up/pull down device on all input pins
  - Configurable drive strength on all output pins
- Package Options
  - 144-pin low-profile quad flat-pack (LQFP)[1]
  - 112-pin low-profile quad flat-pack (LQFP)
  - 64-pin low-profile quad flat-pack (LQFP)
- On-Chip Voltage Regulator
  - Three parallel, linear voltage regulators with bandgap reference providing VDDPLL, 1.8V logic and 2.8V Flash supply
  - Low-voltage detect (LVD) with low-voltage interrupt (LVI)
  - Power-on reset (POR) circuit
  - 3.3V and 5V range operation
  - Low-voltage reset (LVR)
- Operating Conditions
  - Ambient temperature range –40 C to 85 C
  - Temperature Options:
    - –40 C to 105 C
    - –40 C to 125 C
- 50MHz maximum CPU bus frequency, 100MHz maximum XGATE bus frequency

1. The 144-Pin LQFP version will not be qualified for production and is intended to be used for emulation (development tools) only.

Table 1-1. Package and Memory Options of MC9S12XF-Family Members

Device	Package	Flash	RAM	EEEPROM
9S12XF512	144 LQFP	512K	32K	4K
	112 LQFP			
	64 LQFP			
9S12XF384	144 LQFP	384K	24K	4K
	112 LQFP			
	64 QFP			
9S12XF256	144 LQFP	256K	20K	2K
	112 LQFP			
	64 QFP			
9S12XF128	144 LQFP	128K	16K	2K
	112 LQFP			
	64 QFP			

1.1.2 Block Diagram

Figure 1-1 shows a block diagram of the MC9S12XF-Family devices

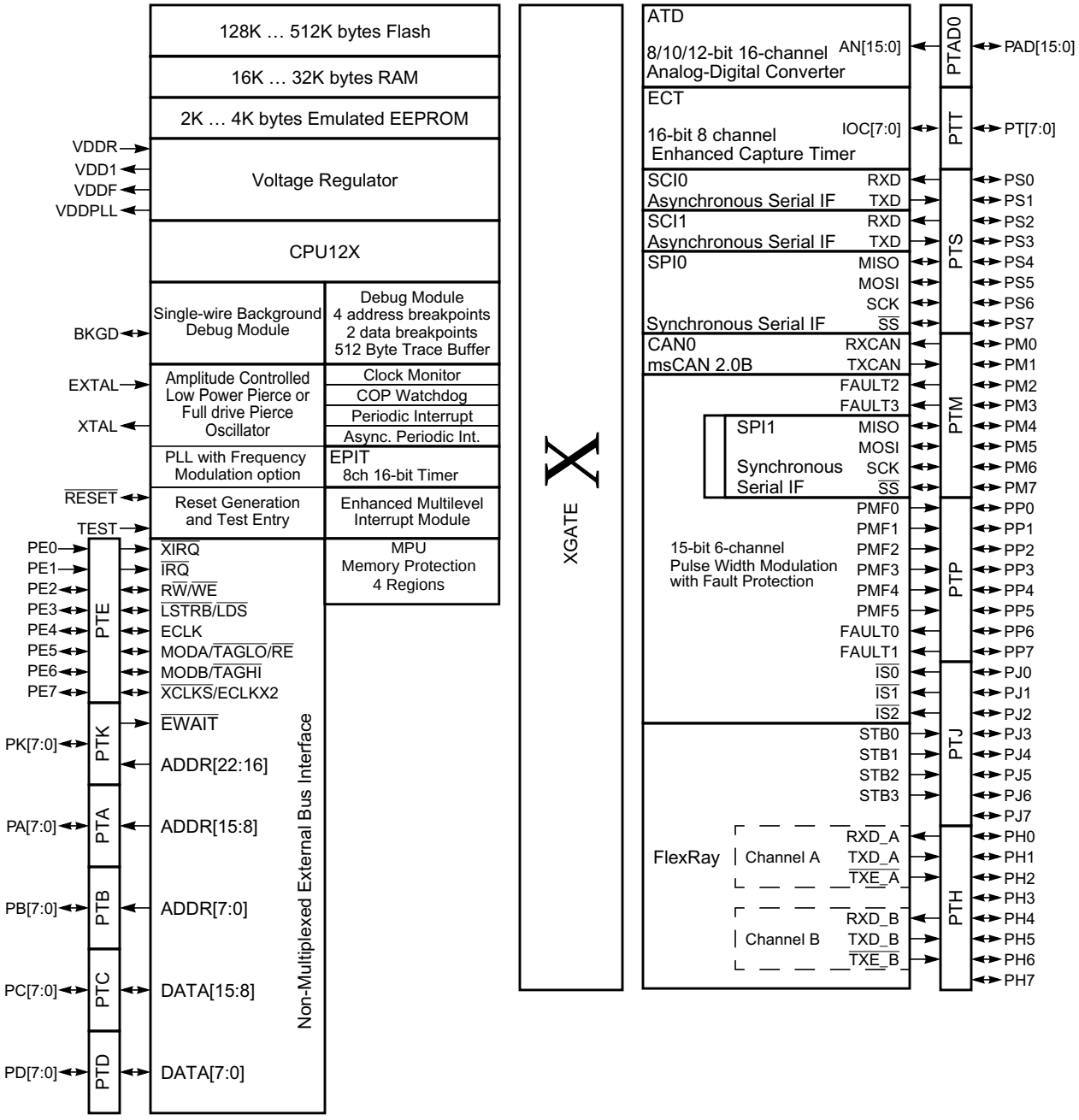


Figure 1-1. MC9S12XF-Family Block Diagram[1]

1. See Section 1.2, “Signal Description“ to get more information with regard to I/O muxing variants.

Table 1-2. Peripheral Feature Summary of MC9S12XF-Family Members

Package	FlexRay	ECT	PIT	CAN	SCI	SPI	A/D	PMF
144 LQFP	2-ch	8ch	8ch	1	2	2	16-ch	6-ch 4 Fault Inputs 3 Current Sense
112 LQFP	2-ch	8ch	8ch	1	2	2	16-ch	6-ch 4 Fault Inputs 3 Current Sense
64 LQFP	1-ch	8ch	8ch	1	1	1	8-ch	6-ch 0 Fault Inputs 0 Current Sense

1.1.3 Device Memory Map

Table 1-3 shows the device register memory map.

Table 1-3. Device Memory Map

Address	Module	Size (Bytes)
0x0000 - 0x0009	PIM (Port Integration Module)	10
0x000A - 0x000B	MMC (Memory Map Control)	2
0x000C - 0x000D	PIM (Port Integration Module)	2
0x000E - 0x000F	EBI (External Bus Interface)	2
0x0010 - 0x0017	MMC (Memory Map Control)	10
0x0018 - 0x0019	Reserved	2
0x001A - 0x001B	Device ID register	2
0x001C - 0x001F	PIM (Port Integration Module)	4
0x0020 - 0x002F	DBG (Debug Module)	16
0x0030 - 0x0031	Reserved	2
0x0032 - 0x0033	PIM (Port Integration Module)	2
0x0034 - 0x003F	CRG (Clock and Reset Generator)	12
0x0040 - 0x007F	ECT (Enhanced Capture Timer 16-bit 8 channel)s	64
0x0080 - 0x00AF	ATD (Analog to Digital Converter 10-bit 16 channel)	48
0x00B0 - 0x00C7	Reserved	24
0x00C8 - 0x00CF	SCI0 (Serial Communications Interface)	8
0x00D0 - 0x00D7	SCI1 (Serial Communications Interface)	8
0x00D8 - 0x00DF	Serial Peripheral Interface (SPI0)	8
0x00E0 - 0x00EF	Reserved	16
0x00F0 - 0x00F7	Serial Peripheral Interface (SPI1)	8
0x00F8 - 0x00FF	Reserved	8
0x0100–0x0113	FTM control register	20
0x0114–0x011F	MPU (memory protection unit)	12
0x0120 - 0x012F	INT (Interrupt Module)	16
0x0130 - 0x013F	Reserved	16
0x0140 - 0x017F	CAN (Freescale Scalable Can)	64
0x0180 - 0x01FF	Reserved	128
0x0200 - 0x023F	PMF (Pulse With Modulator with Fault Protection)	64
0x0240 - 0x027F	PIM (Port Integration Module)	64
0x0280 - 0x02EF	Reserved	112
0x02F0 - 0x02F7	Voltage Regulator	8
0x02F8 - 0x02FF	Reserved	8
0x0300 - 0x0307	FlexRay IPLL	8
0x0308 - 0x033F	Reserved	56
0x0340 - 0x036F	Enhanced Periodic Interrupt Timer	48
0x0370 - 0x037F	Reserved	16
0x0380 - 0x03BF	XGATE	64

Table 1-3. Device Memory Map

Address	Module	Size (Bytes)
0x03C0 - 0x03FF	Reserved	64
0x0400 - 0x05FF	FlexRay	512
0x0600 - 0x07FF	Reserved	512

NOTE

Reserved register space shown in Table 1-3 is not allocated to any module. This register space is reserved for future use. Writing to these locations has no effect. Read access to these locations returns zero.

1.1.4 MC9S12XF512 - Address Mapping

Figure 1-2 shows S12XE CPU & BDM local address translation to the global memory map. It indicates also the location of the internal resources in the memory map.

EEEPROM size is presented like a fixed 256 KByte in the memory map.

Table 1-4. 9S12XF512 Dependent Memory Parameters

Device	FLASH_LOW	PPAGE(1)	RAM_LOW	RPAGE(2)	DF_HIGH	EPAGE
9S12XF512	0x78_0000	32	0x0F_8000	8	0x10_7FFF	32

1. Number of 16K pages addressable via PPAGE register
2. Number of 4K pages addressing the RAM. RAM can also be mapped to 0x4000 - 0x7FFF

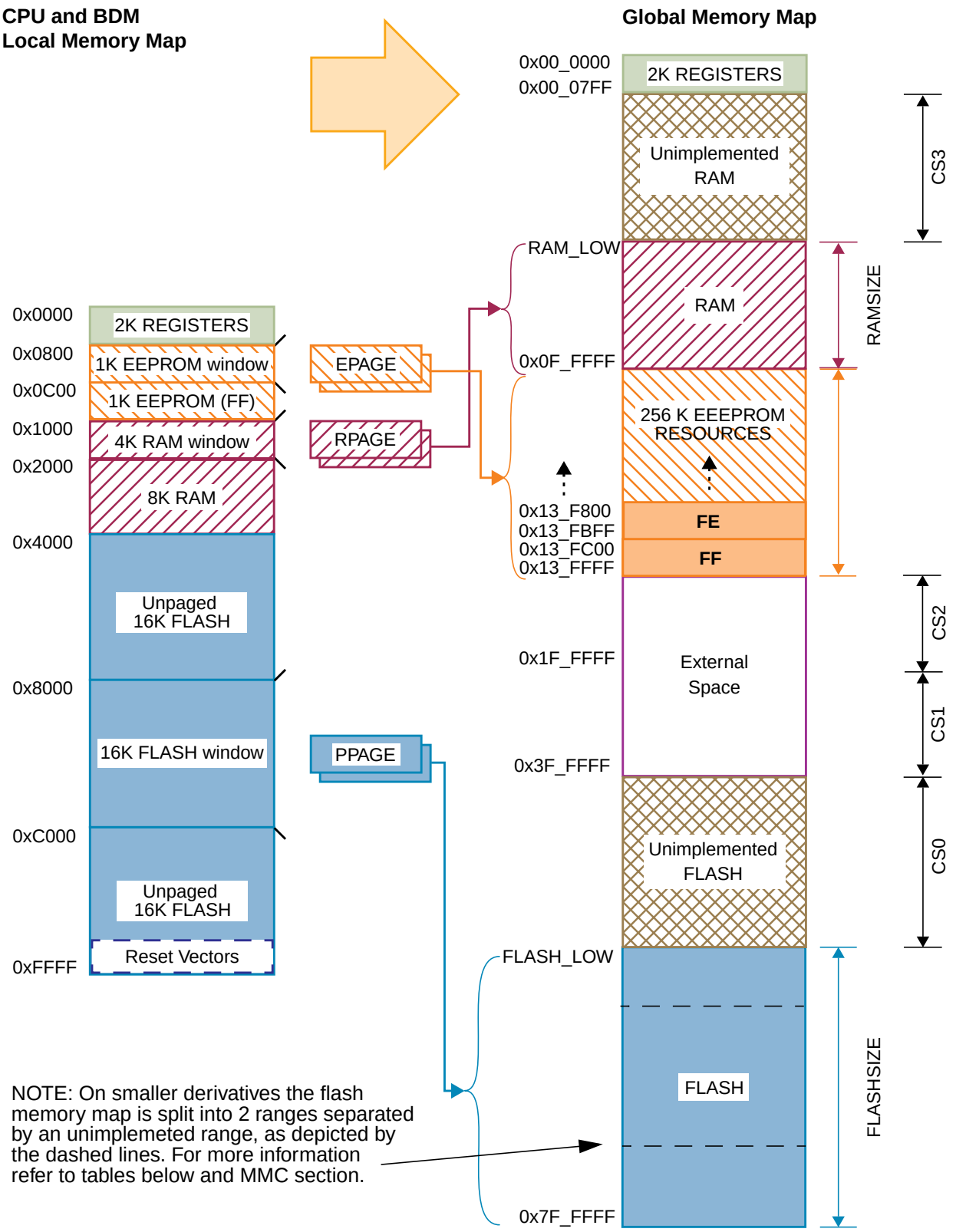


Figure 1-2. MC9S12XF512 Global Memory Map

Unimplemented RAM pages are mapped externally in expanded modes. Accessing unimplemented RAM pages in single chip modes causes an illegal address reset if the MPU is not configured to flag an MPU protection error in that range.

Accessing unimplemented FLASH pages in single chip modes causes an illegal address reset if the MPU is not configured to flag an MPU protection error in that range.

The range between 0x10_0000 and 0x13_FFFF is mapped to EEEPROM resources. The actual EEEPROM and dataflash block sizes are listed in Table 1-6 . Within EEEPROM resource range an address range exists which is neither used by EEEPROM resources nor remapped to external resources via chip selects (see the FTM/MMC descriptions for details).

The fixed 8K RAM default location in the global map is 0x0F_E000 - 0x0F_FFFF. This is subject to remapping when configuring the local address map for a larger RAM access range.

Memory map Figure 1-3 shows XGATE local address translation to the global memory map. It indicates also the location of used internal resources in the memory map.

Table 1-5. XGATE Resources (9S12XF512)

Internal Resource	Size /KByte	$Address
XGATE RAM	32K	XGRAM_LOW = 0x0F_8000
FLASH	30K(1)	XGFLASH_HIGH = 0x78_8000

1. This value is calculated by the following formula: (64K - 2K - XGRAMSIZE)

Table 1-6. Derivative Dependent Memory Parameters

Device	FLASH_LOW	PPAGE (1)	RAM_LOW	RPAGE (2)	EE_LOW	DF_HIGH	EPAGE
9S12XF512	0x78_0000	32	0x0F_8000	8	0x13_F000	0x10_7FFF	4(3) + 32(4)
9S12XF384	0x78_0000(5)	24	0x0F_A000	6	0x13_F000	0x10_7FFF	4 + 32
9S12XF256	0x78_0000(6)	16	0x0F_B000	5	0x13_F800	0x10_7FFF	2 + 32
9S12XF128	0x78_0000(7)	8	0x0F_C000	4	0x13_F800	0x10_7FFF	2 + 32

1. Number of 16K pages addressable via PPAGE register
2. Number of 4K pages addressing the RAM. RAM can also be mapped to 0x4000 - 0x7FFF
3. Number of 1K pages addressing the Cache RAM via the EPAGE register counting downwards from 0xFF
4. Number of 1K pages addressing the Data flash via the EPAGE register starting upwards from 0x00
5. The 384K memory map is split into a 128K block from 0x78_0000 to 0x79_FFFF and a 256K block from 0x7C_0000 to 0x7F_FFFF
6. The 256K memory map is split into a 128K block from 0x78_0000 to 0x79_FFFF and a 128K block from 0x7E_0000 to 0x7F_FFFF
7. The 128K memory map is split into a 64K block from 0x78_0000 to 0x78_FFFF and a 64K block from 0x7F_0000 to 0x7F_FFFF

Table 1-7. Derivative Dependent Flash Block Mapping

Device	0x78_0000	0x7A_0000	0x7C_0000	0x7E_0000
9S12XF512	B1S	B1N	B0	
9S12XF384	B1S	—	B0	
9S12XF256	B1S	—	—	B0(128K)
9S12XF128	B1S (64K)	—	—	B0 (64K)

Block B1 is divided into two 128K blocks. The XGATE is always mapped to block B1S.

On the 9S12XF128 the flash is divided into two 64K blocks B0 and B1S, the B1S range extending from 0x78_0000 to 0x78_FFFF, the B0 range extending from 0x7F_0000 to 0x7F_FFFF.

The block B0 is a reduced size 128K block on the 256K derivative. On the larger derivatives B0 is a 256K block. The block B0 is a reduced size 64K block on the 128K derivative.

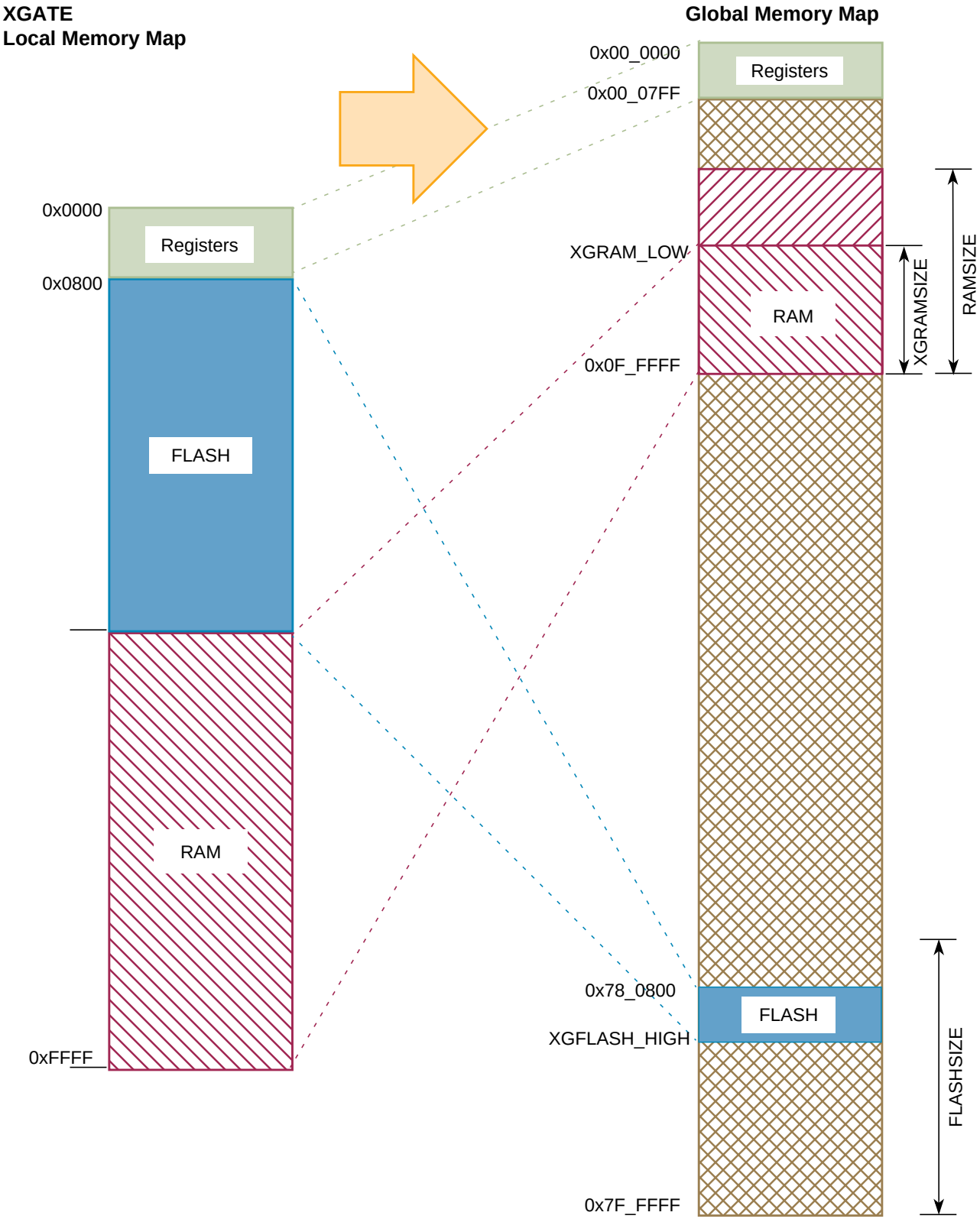


Figure 1-3. XGATE Global Address Mapping

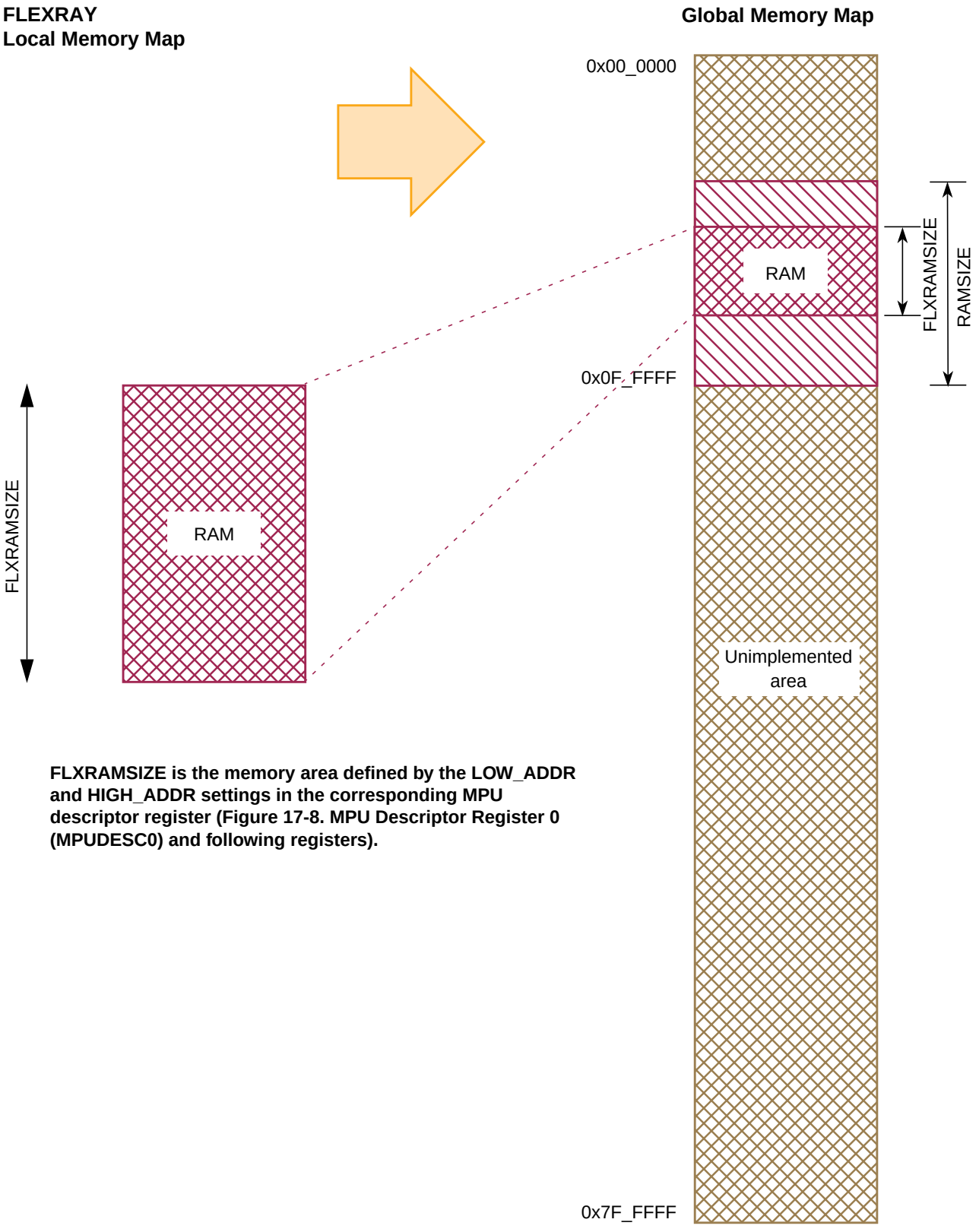
1.1.4.1 FlexRay Address Mapping

Memory map 1 Figure 1-4 shows FlexRay address mapping to the global memory map. It indicates also the location of the internal resources in the memory map.

The FlexRay address mapping can be configured via MPU descriptors. It is possible to set-up up to 4 descriptors giving the FlexRay module access to 4 different regions of RAM with different permissions. This is not reflected in Figure 1-4 for complexity reasons. For more details refer to the MPU section.

NOTE

The FlexRay module can only access system ram space.



FLXRAMSIZE is the memory area defined by the LOW_ADDR and HIGH_ADDR settings in the corresponding MPU descriptor register (Figure 17-8. MPU Descriptor Register 0 (MPUDESC0) and following registers).

Figure 1-4. FLEXRAY Global Address Mapping

1.1.5 Detailed Register Map

For the detailed register map refer to Appendix E Detailed Register Address Map.

1.1.6 Part ID Assignment

The part ID is located in two 8-bit registers PARTIDH and PARTIDL (addresses 0x001A and 0x001B). The read-only value is a unique part ID for each revision of the chip. Table 1-8 shows the assigned part ID number and Mask Set number.

The Version ID is a word located in a flash information row at 0x40_00E8. The version ID number indicates a specific version of internal NVM variables used to patch NVM errata. The default is no patch (0xFFFF).

Table 1-8. Part ID Assignment for MC9S12XF512

Device	Mask Set Number	Part ID[1]	Version ID
MC9S12XF512	0M64J	$D480	0xFFFF
MC9S12XF512	1M64J	$D481	0xFFFF
MC9S12XF512	2M64J	$D481	0x0006[2]

1. The coding is as follows:
   Bit 15-12: Major family identifier
   Bit 11-8: Minor family identifier
   Bit 7-4: Major mask set revision number including FAB transfers
   Bit 3-0: Minor — non full — mask set revision
2. See customer errata for more information w.r.t. patch code.

1.2 Signal Description

This section describes signals that connect off chip.

1.2.1 System Pinout

Figure 1-5 is a pinout diagram of the system. The diagram must include the names of all signals that are connected by system pins.

NOTE

Pin

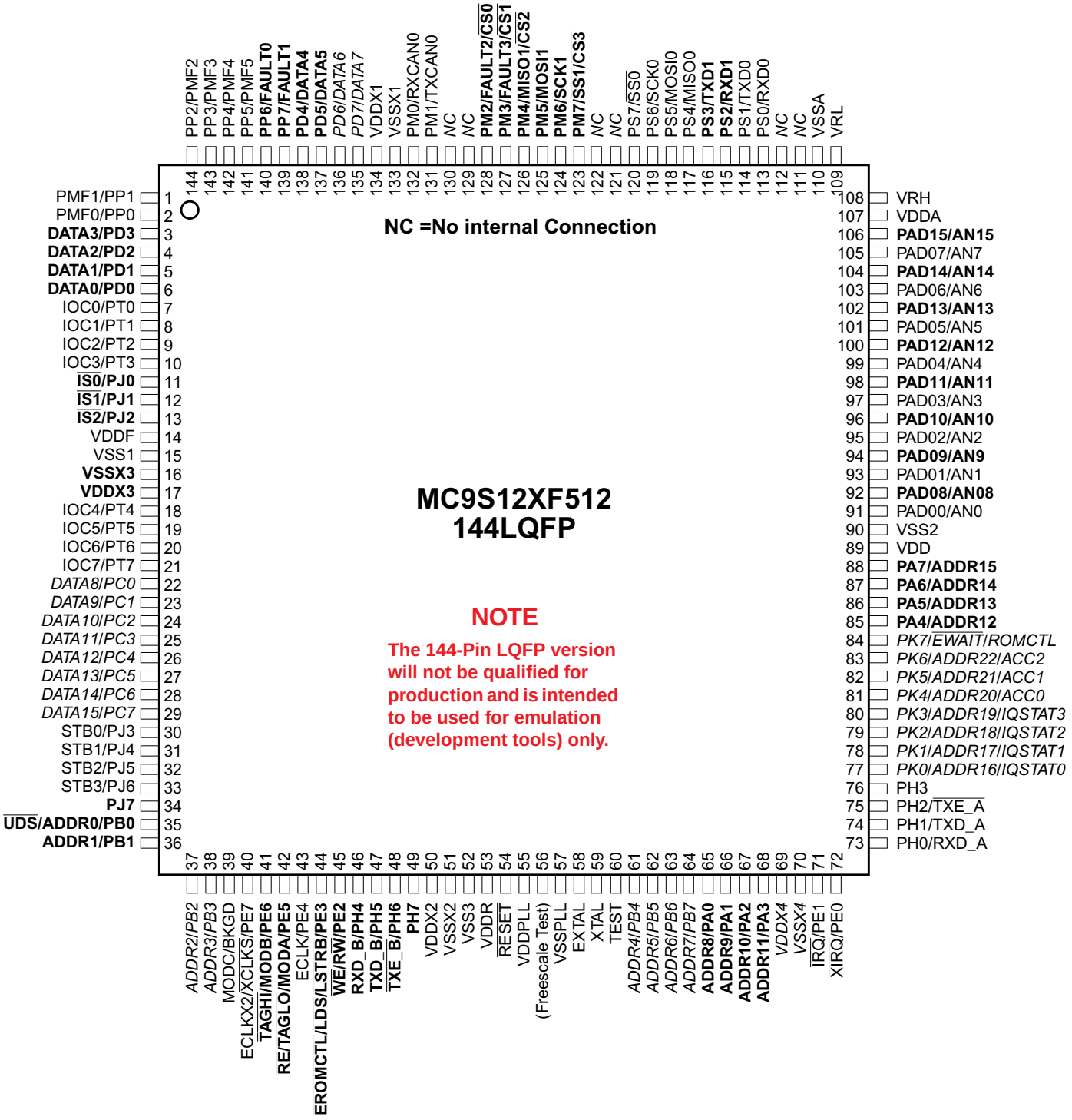
- 56 of the 144-Pin LQFP
- 46 of the 112-Pin LQFP
- 26 of the 64-Pin LQFP

must not be connected to VDD, VSS or any other component. Freescale test structures are bonded to this pin.

Keep this pin (according to package type) open.

NOTE

The VDD pins provide 1.8V derived from the **internal** voltage regulator. Usage of an **external** voltage regulator is **not** allowed.

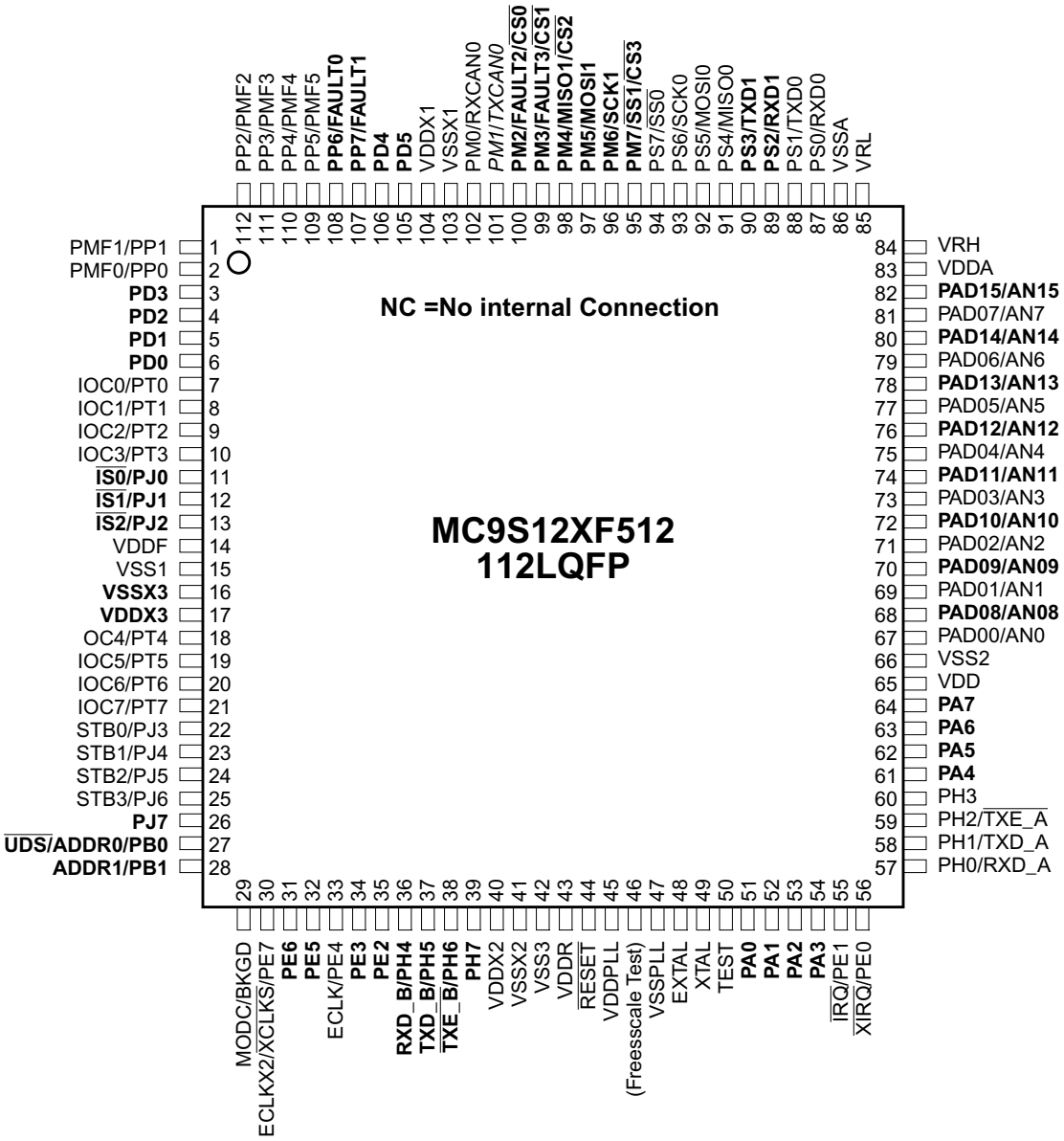


Pins shown in BOLD are not available on the 64-pin package option

Pins shown in *ITALICS* are not available on the 112-pin and 64-pin package options

Figure 1-5. MC9S12XF-Family Pin Assignments 144-pin LQFP Package[1]

1. The 144-Pin LQFP version will not be qualified for production and is intended to be used for emulation (development tools) only.



Pins shown in BOLD are not available on the 64-pin package option

Figure 1-6. MC9S12XF-Family Pin Assignments 112-pin LQFP Package

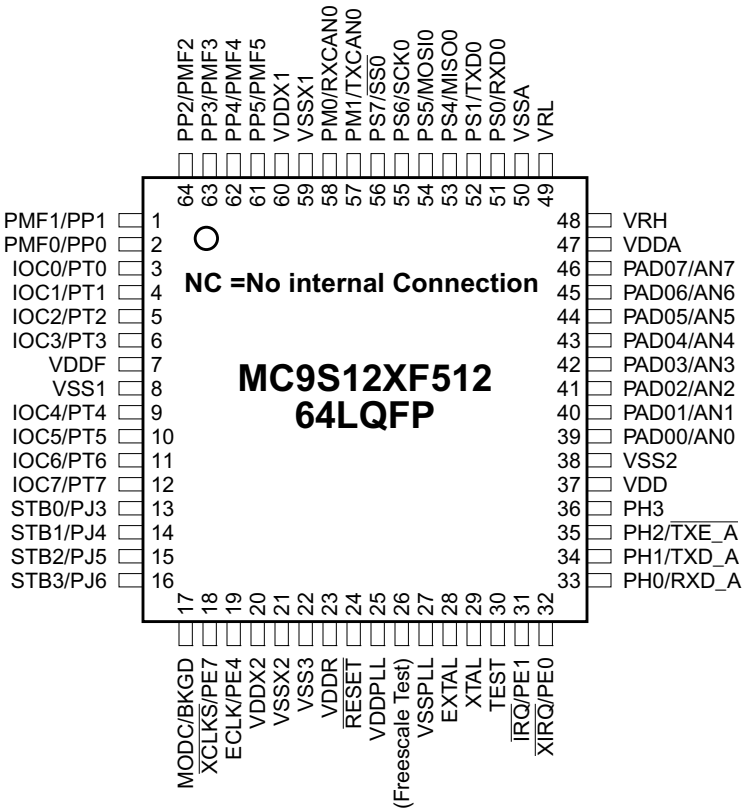


Figure 1-7. MC9S12XF-Family Pin Assignments 64-pin LQFP Package

Table 1-9. Port and Peripheral Availability by Package Option

Port	144 LQFP	112 LQFP	64 LQFP
Port AD/ADC Channels	16/16	16/16	8/8
Port A pins	8	8	0
Port B pins	8	2	0
Port C pins	8	0	0
Port D pins	8	6	0
Port E pins inc. IRQ/XIRQ input only	8	8	4
Port H/FlexRay Channels	8/A+B	8/A+B	4/A
Port J/PMF Current Sense	8/3	8/3	4/0
Port K pins	8	0	0
Port M/CAN/PMF Fault Inputs/SPI	8/1/2/1	8/1/2/1	2/1/0/0

Table 1-9. Port and Peripheral Availability by Package Option

Port	144 LQFP	112 LQFP	64 LQFP
Port P/PMF channels/PMF Fault Inputs	8/6/2	8/6/2	6/6/0
Port S/SCI/SPI	8/2/1	8/2/1	6/1/1
Port T/Timer Channels	8/8	8/8	8/8
VDDX/VSSX	4/4	3/3	2/2

1.2.2 Signal Properties Summary

Table 1-10. Signal Properties

Pin Number			Pin Name Funct. 1	Pin Name Funct. 2	Pin Name Funct. 3	Pin Name Funct. 4	Function	Power Supply	I/O	Termination out of Reset	
LQFP 144 (1)	LQFP 112	LQFP 64								CTRL(2)	Reset(3) State
1	1	1	PP1	PMF1	—	—	Port P I/O, PMF Channels 0/1	VDDX(4)	I/O	PERP PPSP	disabled
2	2	2	PP0	PMF0	—	—		VDDX	I/O		
3	3	—	PD3	DATA3	—	—	Port D I/O, Data Bus	VDDX	I/O	PUCR	disabled
4	4	—	PD2	DATA2	—	—		VDDX	I/O		
5	5	—	PD1	DATA1	—	—		VDDX	I/O		
6	6	—	PD0	DATA0	—	—		VDDX	I/O		
7	7	3	PT0	IOC0	—	—	Port T I/O, Timer channels	VDDX	I/O	PERT PPST	disabled
8	8	4	PT1	IOC1	—	—		VDDX	I/O		
9	9	5	PT2	IOC2	—	—		VDDX	I/O		
10	10	6	PT3	IOC3	—	—		VDDX	I/O		
11	11		PJ0	IS0	—	—	Port T I/O, Current status pins for top/bottom pulse width correction	VDDX	I/O	PERJ PPSJ	Up
12	12		PJ1	IS1	—	—		VDDX	I/O		
13	13		PJ2	IS2	—	—		VDDX	I/O		
14	14	7	VDDF(5)	—	—	—	NVM Power Supply 2.8V				
15	15	8	VSS1	—	—	—	Digital Ground Supply 1.8V				
16	16	—	VSSX3	—	—	—	I/O Ground Supply 3-5V				
17	17	—	VDDX3	—	—	—	I/O Power Supply 3-5V				
18	18	9	PT4	IOC4		—	Port T I/O, Timer channels	VDDX	I/O	PERT PPST	disabled
19	19	10	PT5	IOC5	—	—		VDDX	I/O		
20	20	11	PT6	IOC6	—	—		VDDX	I/O		
21	21	12	PT7	IOC7	—	—		VDDX	I/O		

Table 1-10. Signal Properties

Pin Number			Pin Name Funct. 1	Pin Name Funct. 2	Pin Name Funct. 3	Pin Name Funct. 4	Function	Power Supply	I/O	Termination out of Reset	
LQFP 144 (1)	LQFP 112	LQFP 64								CTRL(2)	Reset(3) State
22	—	—	PC0	DATA8	—	—	Port C I/O, Data Bus	VDDX	I/O	PUCR	disabled
23	—	—	PC1	DATA9	—	—		VDDX	I/O		
24	—	—	PC2	DATA10	—	—		VDDX	I/O		
25	—	—	PC3	DATA11	—	—		VDDX	I/O		
26	—	—	PC4	DATA12	—	—		VDDX	I/O		
27	—	—	PC5	DATA13	—	—		VDDX	I/O		
28	—	—	PC6	DATA14	—	—		VDDX	I/O		
29	—	—	PC7	DATA15	—	—		VDDX	I/O		
30	22	13	PJ3	STB0	—	—	Port J I/O, FR Strobe Signal 0	VDDX	I/O	PERJ PPSJ	Up
31	23	14	PJ4	STB1	—	—	Port J I/O, FR Strobe Signal 1	VDDX	I/O		
32	24	15	PJ5	STB2	—	—	Port J I/O, FR Strobe Signal 2	VDDX	I/O		
33	25	16	PJ6	STB3	—	—	Port J I/O, FR Strobe Signal 3	VDDX	I/O		
34	26	—	PJ7	—	—	—	Port J I/O	VDDX	I/O		
35	27	—	PB0	ADDR0	$\overline{UDS}$	IVD0(6)	Port B I/O, Address Bus, Internal Visibility Data	VDDX	I/O	PUCR	disabled
36	28	—	PB1	ADDR1	—	IVD1		VDDX	I/O		
37	—	—	PB2	ADDR2	—	IVD2		VDDX	I/O		
38	—	—	PB3	ADDR3	—	IVD3		VDDX	I/O		
39	29	17	BKGD	MODC	—	—	Background Debug,	VDDX	I/O	PUCR	Up
40	30	18	PE7	$\overline{XCLKS}$	ECLKX2	—	Port E I/O, Clock Select System clock output	VDDX	I/O	PUCR	Up
41	31	—	PE6	MODB	TAGHI	—	Port E I/O, System clock output, Clock Select	VDDX	I/O	While $\overline{RESET}$ pin is low: Down	
42	32	—	PE5	MODA	TAGLO	RE	Port E I/O, System clock output, Clock Select	VDDX	I/O	While RESET pin is low: Down	
43	33	19	PE4	ECLK	—	—	Port E I/O, Bus Clock Output	VDDX	I/O	PUCR	Up

Table 1-10. Signal Properties

Pin Number			Pin Name Funct. 1	Pin Name Funct. 2	Pin Name Funct. 3	Pin Name Funct. 4	Function	Power Supply	I/O	Termination out of Reset	
LQFP 144 (1)	LQFP 112	LQFP 64								CTRL(2)	Reset(3) State
44	34	—	PE3	$\overline{\text{LSTRB}}$	$\overline{\text{LDS}}$	EROMCTL	Port E I/O, Low Byte Data strobe, EROMON control	VDDX	I/O	PUCR	Up
45	35	—	PE2	R$\overline{\text{W}}$	$\overline{\text{WE}}$	—	Port E I/O, synchrounous Read/Write, asynchronous write	VDDX	I/O		
46	36	—	PH4	RXD_B	—	—	Port H I/O, FR Receive Data Channel B	VDDX	I/O	PERH PPSH	disabled
47	37	—	PH5	TXD_B	—	—	Port H I/O, FR Transmit Data Channel B	VDDX	I/O		
48	38	—	PH6	$\overline{\text{TXE\_B}}$	—	—	Port H I/O, FR Transmit Data Channel B	VDDX	I/O		
49	39	—	PH7	—	—	—	Port H I/O	VDDX	I/O		
50	40	20	VDDX2	—	—	—	I/O Power Supply 3-5V				
51	41	21	VSSX2	—	—	—	I/O Ground Supply 3-5V				
52	42	22	VSS3	—	—	—	Digital Ground Supply 1.8V				
53	43	23	VDDR	—	—	—	Voltage Regulator Power Supply 3-5V				
54	44	24	$\overline{\text{RESET}}$	—	—	—	External Reset	VDDX	I/O	Up	
55	45	25	VDDPLL	—	—	—	PLL & OSC Power Supply 1.8V				
56	46	26	NC	—	—	—	—	—		—	—
57	47	27	VSSPLL	—	—	—	PLL & OSC Ground Supply 1.8V				
58	48	28	EXTAL	—	—	—	Oscillator Pins	VDDPLL		NA	NA
59	49	29	XTAL	—	—	—		VDDPLL		NA	NA
60	50	30	TEST	—	—	—	Test Input	VDDX		RESET PIN	Down
61	—	—	PB4	ADDR4	IVD4	—	Port B I/O, Address Bus, Internal Visibility Data	VDDX	I/O	PUCR	disabled
62	—	—	PB5	ADDR5	IVD5	—		VDDX	I/O		
63	—	—	PB6	ADDR6	IVD6	—		VDDX	I/O		
64	—	—	PB7	ADDR7	IVD7	—		VDDX	I/O		

Table 1-10. Signal Properties

Pin Number			Pin Name Funct. 1	Pin Name Funct. 2	Pin Name Funct. 3	Pin Name Funct. 4	Function	Power Supply	I/O	Termination out of Reset	
LQFP 144 (1)	LQFP 112	LQFP 64								CTRL(2)	Reset(3) State
65	51	—	PA0	ADDR8	IVD8	—	Port A I/O, Address Bus, Internal Visibility Data	VDDX	I/O	PUCR	disabled
66	52	—	PA1	ADDR9	IVD9	—		VDDX	I/O		
67	53	—	PA2	ADDR10	IVD10	—		VDDX	I/O		
68	54	—	PA3	ADDR11	IVD11	—		VDDX	I/O		
69	—	—	VDDX4	—	—	—	I/O Power Supply 3-5V				
70	—	—	VSSX4	—	—	—	I/O Ground Supply 3-5V				
71	55	31	PE1	$\overline{\text{IRQ}}$	—	—	Port E Input, Maskable Interrupt	VDDX	I	PUCR	Up
72	56	32	PE0	$\overline{\text{XIRQ}}$	—	—	Port E Input, Non Maskable Interrupt	VDDX	I		
73	57	33	PH0	RXD_A	—	—	Port H I/O, FR Receive Data Channel A	VDDX	I/O	PERH PPSH	disabled
74	58	34	PH1	TXD_A	—	—	Port H I/O, FR Transmit Data Channel A	VDDX	I/O		
75	59	35	PH2	$\overline{\text{TXE\_A}}$	—	—	Port H I/O, FR Transmit Data Channel A	VDDX	I/O		
76	60	36	PH3	—	—	—	Port H I/O	VDDX	I/O		
77	—	—	PK0	ADDR16	IQSTAT0	—	Extended Address, PIPE status	VDDX	I/O	PUCR	Up
78	—	—	PK1	ADDR17	IQSTAT1	—		VDDX	I/O		
79	—	—	PK2	ADDR18	IQSTAT2	—		VDDX	I/O		
80	—	—	PK3	ADDR19	IQSTAT3	—		VDDX	I/O		
81	—	—	PK4	ADDR20	ACC0	—	Port K I/O, Extended Addresses,Access Source for external Access	VDDX	I/O		
82	—	—	PK5	ADDR21	ACC1	—		VDDX	I/O		
83	—	—	PK6	ADDR22	ACC2	—		VDDX	I/O		
84	—	—	PK7	$\overline{\text{EWAIT}}$	ROMCTL	—	Port K I/O, EWAIT input, ROM On Control	VDDX	I/O		
85	61	—	PA4	ADDR12	IVD12	—	Port A I/O, Address Bus, Internal Visibility Data	VDDX	I/O	PUCR	disabled
86	62	—	PA5	ADDR13	IVD13	—		VDDX	I/O		
87	63	—	PA6	ADDR14	IVD14	—		VDDX	I/O		
88	64	—	PA7	ADDR15	IVD15	—		VDDX	I/O		

Table 1-10. Signal Properties

Pin Number			Pin Name Funct. 1	Pin Name Funct. 2	Pin Name Funct. 3	Pin Name Funct. 4	Function	Power Supply	I/O	Termination out of Reset	
LQFP 144 (1)	LQFP 112	LQFP 64								CTRL(2)	Reset(3) State
89	65	37	VDD	—	—	—	Digital Power Supply 1.8V				
90	66	38	VSS2	—	—	—	Digital Ground Supply 1.8V				
91	67	39	PAD00	AN0	—	—	Port AD Inputs of ATD, Analog Inputs of ATD	VDDA	I/O	PER0AD PER1AD	disabled
92	68	—	PAD08	AN8	—	—		VDDA	I/O		
93	69	40	PAD01	AN1	—	—		VDDA	I/O		
94	70	—	PAD09	AN9	—	—		VDDA	I/O		
95	71	41	PAD02	AN2	—	—		VDDA	I/O		
96	72	—	PAD10	AN10	—	—		VDDA	I/O		
97	73	42	PAD03	AN3	—	—		VDDA	I/O		
98	74	—	PAD11	AN11	—	—		VDDA	I/O		
99	75	43	PAD04	AN4	—	—		VDDA	I/O		
100	76	—	PAD12	AN12	—	—		VDDA	I/O		
101	77	44	PAD05	AN5	—	—		VDDA	I/O		
102	78	—	PAD13	AN13	—	—		VDDA	I/O		
103	79	45	PAD06	AN6	—	—		VDDA	I/O		
104	80	—	PAD14	AN14	—	—		VDDA	I/O		
105	81	46	PAD07	AN7	—	—		VDDA	I/O		
106	82	—	PAD15	AN15	—	—		VDDA	I/O		
107	83	47	VDDA	—	—	—	Analog Power Supply 5V				
108	84	48	VRH	—	—	—	5V Reference voltages for the analog-to-digital converter.				
109	85	49	VRL	—	—	—	0V Reference voltages for the analog-to-digital converter.				
110	86	50	VSSA	—	—	—	Analog Power Ground 5V				
111	—	—	NC	—	—	—	Not connected	—	—	—	—
112	—	—	NC	—	—	—	Not connected	—	—	—	—

Table 1-10. Signal Properties

Pin Number			Pin Name Funct. 1	Pin Name Funct. 2	Pin Name Funct. 3	Pin Name Funct. 4	Function	Power Supply	I/O	Termination out of Reset	
LQFP 144 (1)	LQFP 112	LQFP 64								CTRL(2)	Reset(3) State
113	87	51	PS0	RXD0	—	—	Port S I/O, TXD of SCI0	VDDX	I/O	PERS PPSS	Up
114	88	52	PS1	TXD0	—	—	Port S I/O, RXD of SCI0	VDDX	I/O		
115	89	—	PS2	RXD1	—	—	Port S I/O, TXD of SCI1	VDDX	I/O		
116	90	—	PS3	TXD1	—	—	Port S I/O, RXD of SCI1	VDDX	I/O		
117	91	53	PS4	MISO0	—	—	Port S I/O, MISO of SPI0	VDDX	I/O		
118	92	54	PS5	MOSI0	—	—	Port S I/O, MOSI of SPI0	VDDX	I/O		
119	93	55	PS6	SCK0	—	—	Port S I/O, SCK of SPI0	VDDX	I/O		
120	94	56	PS7	$\overline{SS0}$	—	—	Port S I/O, $\overline{SS}$ of SP0	VDDX	I/O		
121	—	—	NC	—	—	—	Not connected	—	—	—	—
122	—	—	NC	—	—	—	Not connected	—	—	—	—
123	95	—	PM7	$\overline{SS1}$	$\overline{CS3}$	—	Port M I/O, $\overline{SS}$ of SPI1, Chip Select 3	VDDX	I/O	PERM PPSM	disabled
124	96	—	PM6	SCK1	—	—	Port M I/O, SCK of SPI1	VDDX	I/O		
125	97	—	PM5	MOSI1	—	—	Port M I/O, MOSI of SPI1	VDDX	I/O		
126	98	—	PM4	MISO1	$\overline{CS2}$	—	Port M I/O, MISO of SPI1, Chip Select 2	VDDX	I/O		
127	99	—	PM3	FAULT3	$\overline{CS1}$	—	Port M I/O, PMF Fault 3, Chip Select 1	VDDX	I/O		
128	100	—	PM2	FAULT2	$\overline{CS0}$	—	Port M I/O, PMF Fault 2, Chip Select 0	VDDX	I/O		
129	—	—	NC	—	—	—	Not connected	—	—	—	—
130	—	—	NC	—	—	—	Not connected	—	—	—	—
131	101	57	PM1	TXCAN0	—	—	Port M I/O, CAN TX	VDDX	I/O	PERM PPSM	disabled
132	102	58	PM0	RXCAN0	—	—	Port M I/O, CAN RX	VDDX	I/O		
133	103	59	VSSX1	—	—	—	I/O Ground Supply 3-5V				
134	104	60	VDDX1	—	—	—	5V power sypply				

Table 1-10. Signal Properties

Pin Number			Pin Name Funct. 1	Pin Name Funct. 2	Pin Name Funct. 3	Pin Name Funct. 4	Function	Power Supply	I/O	Termination out of Reset	
LQFP 144 (1)	LQFP 112	LQFP 64								CTRL(2)	Reset(3) State
135	—	—	PD7	DATA7	—	—	Port D I/O, Data Bus	VDDX	I/O	PUCR	disabled
136	—	—	PD6	DATA6	—	—		VDDX	I/O		
137	105	—	PD5	DATA5	—	—		VDDX	I/O		
138	106	—	PD4	DATA4	—	—		VDDX	I/O		
139	107	—	PP7	FAULT1	—	—	Port P I/O, PMF Fault 1	VDDX	I/O	PERP PPSP	disabled
140	108	—	PP6	FAULT0	—	—	Port P I/O, PMF Fault 0	VDDX	I/O		
141	109	61	PP5	PMF5	—	—	Port P I/O, PMF Channel 5	VDDX	I/O		
142	110	62	PP4	PMF4	—	—	Port P I/O, PMF Channel 4	VDDX	I/O		
143	111	63	PP3	PMF3	—	—	Port P I/O, PMF Channel 3	VDDX	I/O		
144	112	64	PP2	PMF2	—	—	Port P I/O, PMF Channel 2	VDDX	I/O		

1. The 144-Pin LQFP version will not be qualified for production and is intended to be used for emulation (development tools) only.
2. Register bit in the Port Integration Module which controls the behavior
3. State after reset (disabled, pull Up, pull Down)
4. VDDX = VDDX1,VDDX2,VDDX3,VDDX4
5. VDDF must not be connected to VDD
6. Internal visability is only available on the 144-LQFP Package

NOTE

For devices assembled in 144-pin, 112-pin and 64-pin packages all non-bonded out pins should be configured as outputs after reset in order to avoid current leakage through I/O structures of floating inputs. Refer to Table 1-10 for affected pins.

NOTE

VDDF must not be connected to VDD.

1.2.3 Detailed Signal Descriptions

1.2.4 EXTAL, XTAL — Oscillator Pins

EXTAL and XTAL are the crystal driver and external clock pins. On reset all the device clocks are derived from the EXTAL input frequency. XTAL is the crystal driver output.

1.2.5 $\overline{\text{RESET}}$ — External Reset Pin

The $\overline{\text{RESET}}$ pin is an active low bidirectional control signal. It acts as an input to initialize the MCU to a known start-up state, and an output when an internal MCU function causes a reset. The $\overline{\text{RESET}}$ pin has an internal pullup device.

1.2.6 TEST — Test Pin

This input only pin is reserved for test. This pin has a pulldown device.

NOTE

The TEST pin must be tied to VSS in all applications.

1.2.7 BKGD / MODC — Background Debug and Mode Pin

The BKGD/MODC pin is used as a pseudo-open-drain pin for the background debug communication. It is used as a MCU operating mode select pin during reset. The state of this pin is latched to the MODC bit at the rising edge of $\overline{\text{RESET}}$. The BKGD pin has a pullup device.

NOTE

An additional lower-ohm pullup is required as the on-chip pullup device is not intended to drive the debug line in the case of BDM communication.

1.2.8 Port Pins

1.2.8.1 PAD00 - PAD15 / AN00 - AN15 — Port AD I/0 Pin of ATD

PAD00 - PAD15 are general purpose inputs or outputs and analog inputs AN00 - AN15 of the analog to digital converter ATD.

1.2.8.2 PA[7:0] / ADDR[15:8] / IVD[15:8] — Port A I/O Pins

PA7-PA0 are general purpose input or output pins. In MCU expanded modes of operation, these pins are used for the external address bus. In MCU emulation modes of operation, these pins are used for external address bus and internal visibility read data.

1.2.8.3 PB[7:1] / ADDR[7:1] / IVD[7:1] — Port B I/O Pins

PB7-PB1 are general purpose input or output pins. In MCU expanded modes of operation, these pins are used for the external address bus. In MCU emulation modes of operation, these pins are used for external address bus and internal visibility read data.

1.2.8.4 PB0 / ADDR0 / $\overline{\text{UDS}}$ / IVD[0] — Port B I/O Pin

PB0 is a general purpose input or output pin. In MCU expanded modes of operation, this pin is used for the external address bus ADDR0 or as upper data strobe signal. In MCU emulation modes of operation, this pin is used for external address bus ADDR0 and internal visibility read data IVD0.

1.2.8.5 PC[7:0] / DATA [15:8] — Port C I/O Pins

PC7-PC0 are general purpose input or output pins. In MCU expanded modes of operation, these pins are used for the external data bus.

The input voltage thresholds for PC[7:0] can be configured to reduced levels, to allow data from an external 3.3V peripheral to be read by the MCU operating at 5.0V. The input voltage thresholds for PC[7:0] are configured to reduced levels out of reset in expanded and emulation modes. The input voltage thresholds for PC[7:0] are configured to 5V levels out of reset in normal modes.

1.2.8.6 PD[7:0] / DATA [7:0] — Port D I/O Pins

PD7-PD0 are general purpose input or output pins. In MCU expanded modes of operation, these pins are used for the external data bus.

The input voltage thresholds for PD[7:0] can be configured to reduced levels, to allow data from an external 3.3V peripheral to be read by the MCU operating at 5.0V. The input voltage thresholds for PD[7:0] are configured to reduced levels out of reset in expanded and emulation modes. The input voltage thresholds for PC[7:0] are configured to 5V levels out of reset in normal modes.

1.2.8.7 PE7 / ECLKX2 / $\overline{\text{XCLKS}}$ — Port E I/O

PE7 is a general-purpose input or output pin. ECLKX2 is a free running clock of twice the internal bus frequency, available by default in emulation modes and when enabled in other modes. The XCLKS is an input signal which controls whether a crystal in combination with the internal loop controlled Pierce oscillator is used or whether full swing Pierce oscillator/external clock circuitry is used (refer to Oscillator Configuration). An internal pullup is enabled during reset.

1.2.8.8 PE6 / MODB / $\overline{\text{TAGHI}}$ — Port E I/O

PE6 is a general purpose input or output pin. It is used as a MCU operating mode select pin during reset. The state of this pin is latched to the MODB bit at the rising edge of $\overline{\text{RESET}}$. This pin is an input with a pull-down device which is only active when $\overline{\text{RESET}}$ is low. $\overline{\text{TAGHI}}$ is used to tag the high half of the instruction word being read into the instruction queue.

The input voltage threshold for PE6 can be configured to reduced levels, to allow data from an external 3.3V peripheral to be read by the MCU operating at 5.0V. The input voltage threshold for PE6 is configured to reduced levels out of reset in expanded and emulation modes.

1.2.8.9 PE5 / MODA / $\overline{\text{TAGLO}}$ / $\overline{\text{RE}}$ — Port E I/O

PE5 is a general purpose input or output pin. It is used as a MCU operating mode select pin during reset. The state of this pin is latched to the MODA bit at the rising edge of $\overline{\text{RESET}}$. This pin is shared with the Read Enable $\overline{\text{RE}}$ output. This pin is an input with a pull-down device which is only active when $\overline{\text{RESET}}$ is low. $\overline{\text{TAGLO}}$ is used to tag the low half of the instruction word being read into the instruction queue.

The input voltage threshold for PE5 can be configured to reduced levels, to allow data from an external 3.3V peripheral to be read by the MCU operating at 5.0V. The input voltage threshold for PE5 is configured to reduced levels out of reset in expanded and emulation modes.

1.2.8.10 PE4 / ECLK — Port E I/O

PE4 is a general purpose input or output pin. It can be configured to drive the internal bus clock ECLK. ECLK can be used as a timing reference.

1.2.8.11 PE3 / $\overline{\text{LSTRB}}$ / $\overline{\text{LDS}}$ / EROMCTL — Port E I/O

PE3 is a general purpose input or output pin. In MCU expanded modes of operation, $\overline{\text{LSTRB}}$ or $\overline{\text{LDS}}$ can be used for the low byte strobe function to indicate the type of bus access. At the rising edge of $\overline{\text{RESET}}$ the state of this pin is latched to the EROMON bit.

1.2.8.12 PE2 / R/$\overline{\text{W}}$ / $\overline{\text{WE}}$— Port E I/O

PE2 is a general purpose input or output pin. In MCU expanded modes of operations, this pin drives the read/write output signal or write enable output signal for the external bus. It indicates the direction of data on the external bus.

1.2.8.13 PE1 / $\overline{\text{IRQ}}$ — Port E Input

PE1 is a general purpose input pin and the maskable interrupt request input that provides a means of applying asynchronous interrupt requests. This will wake up the MCU from STOP or WAIT mode.

1.2.8.14 PE0 / $\overline{\text{XIRQ}}$ — Port E Input

PE0 is a general purpose input pin and the non-maskable interrupt request input that provides a means of applying asynchronous interrupt requests. This will wake up the MCU from STOP or WAIT mode.

1.2.8.15 PH7— Port H I/O

PH7 is a general purpose input or output pin.

1.2.8.16 PH6 / $\overline{\text{TXE\_B}}$ — Port H I/O

PH6 is a general purpose input or output pin. It can be configured as FlexRay TXEN_B pin which indicates to the Bus Driver that the FlexRay module is attempting to transmit data on channel B.

1.2.8.17 PH5 / TXD_B — Port H I/O

PH5 is a general purpose input or output pin. It can be configured as FlexRay data transmit channel B.

1.2.8.18 PH4 / RXD_B — Port H I/O

PH4 is a general purpose input or output pin. It can be configured as FlexRay data receive channel B.

1.2.8.19 PH3 — Port H I/O

PH3 is a general purpose input or output pin.

1.2.8.20 PH2 / $\overline{\text{TXE\_A}}$ — Port H I/O

PH6 is a general purpose input or output pin. It can be configured as FlexRay TXEN_A pin which indicates to the Bus Driver that the FlexRay module is attempting to transmit data on channel A.

1.2.8.21 PH1 / TXD_A — Port H I/O

PH1 is a general purpose input or output pin. It can be configured as FlexRay data transmit channel A.

1.2.8.22 PH0 / RXD_A — Port H I/O

PH0 is a general purpose input or output pin. It can be configured as FlexRay data receive channel A.

1.2.8.23 PJ7 — PORT J I/O

PJ7 is a general purpose input or output pin.

1.2.8.24 PJ6 / STB3 — PORT J I/O

PJ6 is a general purpose input or output pin. It can be configured as FlexRay Strobe Signal 3 (STB3).

1.2.8.25 PJ5 / STB2 — PORT J I/O

PJ5 is a general purpose input or output pin. It can be configured as FlexRay Strobe Signal 2 (STB2).

1.2.8.26 PJ4 / STB1— PORT J I/O

PJ4 is a general purpose input or output pin. It can be configured as FlexRay Strobe Signal 1 (STB1).

1.2.8.27 PJ3 / STB0 — PORT J I/O

PJ4 is a general purpose input or output pin. It can be configured as FlexRay Strobe Signal 1 (STB0).

1.2.8.28 PJ2 / $\overline{IS2}$ — PORT J I/O

PJ2 is a general purpose input or output pin. It can be configured as PMF current status bit for top/bottom pulse width correction ($\overline{IS2}$).

1.2.8.29 PJ1 / $\overline{IS1}$ — PORT J I/O

PJ1 is a general purpose input or output pin. It can be configured as PMF current status bit for top/bottom pulse width correction ($\overline{IS1}$).

1.2.8.30 PJ0 / $\overline{IS0}$ — PORT J I/O

PJ0 is a general purpose input or output pin. It can be configured as PMF current status bit for top/bottom pulse width correction ($\overline{IS0}$).

1.2.8.31 PK7 / $\overline{EWAIT}$ / ROMCTL — Port K I/O

PK7 is a general purpose input or output pin. During MCU emulation modes and normal expanded modes of operation, this pin is used to enable the Flash EEEPROM memory in the memory map (ROMCTL). At the rising edge of $\overline{RESET}$, the state of this pin is latched to the ROMON bit. The $\overline{EWAIT}$ input signal maintains the external bus access until the external device is ready to capture data (write) or provide data (read).

The input voltage threshold for PK7 can be configured to reduced levels, to allow data from an external 3.3V peripheral to be read by the MCU operating at 5.0V.

1.2.8.32 PK[6:4] / ADDR[22:20] / ACC[2:0] — Port K I/O

PK[6:4] are general purpose input or output pins. During MCU expanded modes of operation, the ACC[2:0] signals are used to indicate the access source of the bus cycle . This pins also provide the expanded addresses ADDR[22:20] for the external bus. In Emulation modes ACC[2:0] is available and is time multiplexed with the high addresses

1.2.8.33 PK[3:0] / ADDR[19:16] / IQSTAT[3:0] — Port K I/O

PK3-PK0 are general purpose input or output pins. In MCU expanded modes of operation, these pins provide the expanded address ADDR[19:16] for the external bus and carry instruction pipe information.

1.2.8.34 PM7 / $\overline{SS1}$ / $\overline{CS3}$ — Port M I/O

PM7 is a general purpose input or output pin. It can be configured as $\overline{SS}$ pin for SPI1 ($\overline{SS1}$). It can be configured as Chip Select 3 ($\overline{CS3}$).

1.2.8.35 PM6 / SCK1 — Port M I/O

PM6 is a general purpose input or output pin. It can be configured as the serial clock pin SCK of the Serial Peripheral Interface 1 (SPI1).

1.2.8.36 PM5 / MOSI1 — Port M I/O

PM5 is a general purpose input or output pin. It can be configured as master output (during master mode) or slave input pin (during slave mode) MOSI of the Serial Peripheral Interface 1 (SPI1).

1.2.8.37 PM4 / MISO1 / $\overline{\text{CS2}}$ — Port M I/O

PM4 is a general purpose input or output pin. It can be configured as master input (during master mode) or slave output pin (during slave mode) MISO of the Serial Peripheral Interface 1 (SPI1). It can be configured as Chip Select 2 ($\overline{\text{CS2}}$).

1.2.8.38 PM3 / FAULT3 / $\overline{\text{CS1}}$ — Port M I/O

PM3 is a general purpose input or output pin. It can be configured as PMF FAULT3 input pin. The FAULT inputs are used to disable selected PWM outputs. It can be configured as Chip Select 1 ($\overline{\text{CS1}}$).

1.2.8.39 PM2 / FAULT2 / $\overline{\text{CS0}}$ — Port M I/O

PM2 is a general purpose input or output pin. It can be configured as PMF FAULT2 input pin. The FAULT inputs are used to disable selected PWM outputs. It can be configured as Chip Select 0 ($\overline{\text{CS0}}$).

1.2.8.40 PM1 / TXCAN0 — Port M I/O

PM1 is a general purpose input or output pin. It can be configured as the transmit pin TXCAN of the Freescale Scalable Controller Area Network controller (MCAN).

1.2.8.41 PM0 / RXCAN0 — Port M I/O

PM0 is a general purpose input or output pin. It can be configured as the receive pin RXCAN of the Freescale Scalable Controller Area Network controller (MSCAN).

1.2.8.42 PP7 / FAULT1 — Port P I/O

PP7 is a general purpose input or output pin. I It can be configured as PMF FAULT1 input pin. The FAULT inputs are used to disable selected PWM outputs.

1.2.8.43 PP6 / FAULT0 — Port P I/O

PP6 is a general purpose input or output pin. It can be configured as PMF FAULT0 input pin. The FAULT inputs are used to disable selected PWM outputs.

1.2.8.44 PP5 / PMF5 — Port P I/O

PP5 is a general purpose input or output pin. It can be configured to work as PWM output channel 5 of the PMF module.

1.2.8.45 PP4 / PMF4 — Port P I/O

PP4 is a general purpose input or output pin. It can be configured to work as PWM output channel 4 of the PMF module.

1.2.8.46 PP3 / PMF3 — Port P I/O

PP3 is a general purpose input or output pin. It can be configured to work as PWM output channel 3 of the PMF module.

1.2.8.47 PP2 / PMF2 — Port P I/O

PP2 is a general purpose input or output pin. It can be configured to work as PWM output channel 2 of the PMF module.

1.2.8.48 PP1 / PMF1 — Port P I/O

PP1 is a general purpose input or output pin. It can be configured to work as PWM output channel 1 of the PMF module.

1.2.8.49 PP0 / PMF0 — Port P I/O

PP0 is a general purpose input or output pin. It can be configured to work as PWM output channel 0 of the PMF module.

1.2.8.50 PS7 / $\overline{SS0}$ — Port S I/O

PS7 is a general purpose input or output pin. It can be configured as the slave select pin $\overline{SS}$ of the Serial Peripheral Interface (SPI0).

1.2.8.51 PS6 / SCK0 — Port S I/O

PS6 is a general purpose input or output pin. It can be configured as the serial clock pin SCK of the Serial Peripheral Interface 0 (SPI0).

1.2.8.52 PS5 / MOSI0 — Port S I/O

PS5 is a general purpose input or output pin. It can be configured as master output (during master mode) or slave input pin (during slave mode) MOSI of the Serial Peripheral Interface 0 (SPI0).

1.2.8.53 PS4 / MISO0 — Port S I/O

PS4 is a general purpose input or output pin. It can be configured as master input (during master mode) or slave output pin (during slave mode) MISO of the Serial Peripheral Interface 0 (SPI0).

1.2.8.54 PS3 / TXD1 — Port S I/O

PS3 is a general purpose input or output pin. It can be configured as the transmit pin TXD of Serial Communication Interface 1 (SCI1).

1.2.8.55 PS2 / RXD1 — Port S I/O

PS2 is a general purpose input or output pin. It can be configured as the receive pin RXD of Serial Communication Interface 1 (SCI1).

1.2.8.56 PS1 / TXD0 — Port S I/O

PS1 is a general purpose input or output pin. It can be configured as the transmit pin TXD of Serial Communication Interface 0 (SCI0).

1.2.8.57 PS0 / RXD0 — Port S I/O

PS0 is a general purpose input or output pin. It can be configured as the receive pin RXD of Serial Communication Interface 0 (SCI0).

1.2.8.58 PT7 / IOC7 — Port T I/O

PT7 is a general purpose input or output pin. It can be configured as input capture or output compare pin IOC7 of the Enhanced Capture Timer (ECT).

1.2.8.59 PT6-PT3 / IOC6-IOC3 — Port T I/O

PT6-PT3 are general purpose input or output pins. They can be configured as input capture or output compare pins IOC6-IOC3 of the Enhanced Capture Timer (ECT).

1.2.8.60 PT2-PT0 / IOC2-IOC0 — Port T I/O

PT2-PT0 are general purpose input or output pins. They can be configured as input capture or output compare pins IOC2-IOC0 of the Enhanced Capture Timer (ECT).

1.2.9 Power Supply Pins

The power and ground pins of the MC9S12XF512 are described below.

Because fast signal transitions place high, short-duration current demands on the power supply, use bypass capacitors with high-frequency characteristics and place them as close to the MCU as possible.

NOTE

All VSS pins must be connected together in the application.

1.2.9.1 VDDX1 - VDDX4 / VSSX1 - VSSX4 — Power & Ground Pins for I/O Drivers

External power and ground for I/O drivers. Because fast signal transitions place high, short-duration current demands on the power supply, use bypass capacitors with high-frequency characteristics and place them as close to the MCU as possible. Bypass requirements depend on how heavily the MCU pins are loaded.

1.2.9.2 VDDR — Power Pin for Internal Voltage Regulator

Input to the internal voltage regulator. Because fast signal transitions place high, short-duration current demands on the power supply, use bypass capacitors with high-frequency characteristics and place them as close to the MCU as possible. Bypass requirements depend on how heavily the MCU pins are loaded.

NOTE

Usage of an external voltage regulator is not allowed.

1.2.9.3 VDDF - NVM Power Pin

Power is supplied to the MCU NVM through VDDF. The voltage supply of nominally 2.8V is derived from the internal voltage regulator when enabled. Connecting additional load to this pin is not permitted when the internal regulator is enabled.

NOTE

VDDF must not be connected to VDD.

1.2.9.4 VDD / VSS1 - VSS2 - VSS3 — Core Power Pins

Use bypass capacitors with high-frequency characteristics because fast signal transitions place high, short-duration current demands on the power supply, and place them as close to the MCU as possible. This 1.8V supply is derived from the internal voltage regulator when enabled. Connecting additional load to this pin is not permitted when the internal regulator is enabled.

NOTE

VDD must not be connected to VDDF.

1.2.9.5 VDDA, VSSA — Power Supply Pins for ATD and VREG

VDDA, VSSA are the power supply and ground input pins for the voltage regulator and the analog to digital converters.

1.2.9.6 VRH, VRL — ATD Reference Voltage Input Pins

VRH and VRL are the reference voltage input pins for the analog to digital converter.

1.2.9.7 VDDPLL, VSSPLL — Power Supply Pins for PLL

These pins provide operating voltage and ground for the oscillator and the phased-locked loop. The voltage supply of nominally 1.8V is derived from the internal voltage regulator when enabled. This allows the supply voltage to the oscillator and PLL to be bypassed independently. This voltage is generated by the internal voltage regulator. No static external loading of these pins is permitted.

NOTE

Connecting additional load to this pin is not permitted when the internal regulator is enabled.

Table 1-11. MC9S12XF512 Power and Ground Connection Summary

Mnemonic	Pin Number			Nominal Voltage	Description
	144-pin LQFP[1]	112-pin LQFP	64-pin LQFP		
VDDF	14	14	7	2.8 V	Internal power and ground generated by internal regulator for the internal NVM.
VDD	89	65	37	1.8 V	Internal power and ground generated by internal regulator for the internal core.
VSS1, 2, 3	15, 90, 52	15, 66, 42	8, 38, 22	0V	
VDDR	53	43	23	5.0 V	External power supply internal voltage regulator
VDDX1	134	104	60	5.0 V	External power and ground, supply to pin drivers
VSSX1	133	103	59	0 V	
VDDX2	50	40	20	5.0 V	External power and ground, supply to pin drivers
VSSX2	52	41	21	0 V	
VDDX3	17	17	—	5.0 V	External power and ground, supply to pin drivers
VSSX3	16	16	—	0 V	
VDDX4	69	—	—	5.0 V	External power and ground, supply to pin drivers
VSSX4	70	—	—	0 V	
VDDA	107	83	47	5.0 V	Operating voltage and ground for the analog-to-digital converters and the reference for the internal voltage regulator, allows the supply voltage to the A/D to be bypassed independently.
VSSA	110	86	50	0 V	

Mnemonic	Pin Number			Nominal Voltage	Description
	144-pin LQFP[1]	112-pin LQFP	64-pin LQFP		
VRL	109	85	49	0 V	Reference voltages for the analog-to-digital converter.
VRH	108	84	48	5.0 V	
VDDPLL	55	45	25	1.8 V	Provides operating voltage and ground for the phased-locked loop. This allows the supply voltage to the PLL to be bypassed independently. Internal power and ground generated by internal regulator.
VSSPLL	57	47	27	0 V	

1. The 144-Pin LQFP version will not be qualified for production and is intended to be used for emulation (development tools) only.

1.3 System Clock Description

The Clock and Reset Generator module (CRG) provides the internal clock signals for the core and all peripheral modules. shows the clock connections from the CRG to all modules.

Consult the CRG Block User Guide for details on clock generation.

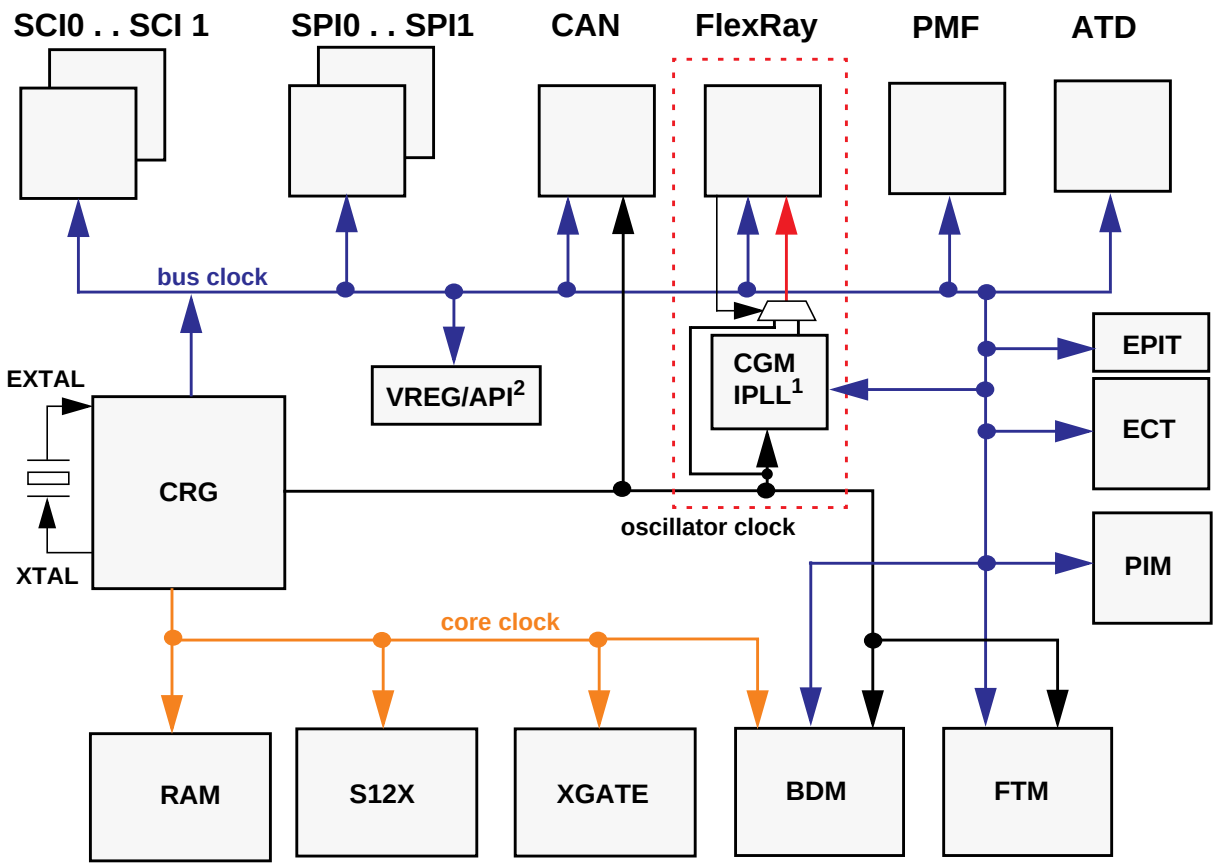


Figure 1-8. Clock Connections

1 PLL for FlexRay protocol engine. This PLL is independent from the system PLL (CRG) and has to be configured accordingly. Refer to Chapter 12, “Clock Generation Module using IPLL (CGMIPLL) Block Description“, Chapter 13, “FlexRay Communication Controller (FLEXRAY)“ and Section 1.12, “FlexRay IPLL (CGMIPLL) Configuration“ for details how to configure the FlexRay IPLL.

2 Internal Oscillator for API (see 3.4.8 Autonomous Periodical Interrupt (API)).

The system clock can be supplied in several ways enabling a range of system operating frequencies to be supported:

- The on-chip phase locked loop (PLL)

- the PLL self clocking
- the oscillator

The clock generated by the PLL or oscillator provides the main system clock frequencies core clock and bus clock. As shown in Figure 1-8, these system clocks are used throughout the MCU to drive the core, the memories, and the peripherals.

The Program Flash memory and the Data Flash are supplied by the bus clock and the oscillator clock. The oscillator clock is used as a time base to derive the program and erase times for the NVM's.

The CAN modules may be configured to have their clock sources derived either from the bus clock or directly from the oscillator clock. This allows the user to select its clock based on the required jitter performance.

In order to ensure the presence of the clock the MCU includes an on-chip clock monitor connected to the output of the oscillator. The clock monitor can be configured to invoke the PLL self-clocking mode or to generate a system reset if it is allowed to time out as a result of no oscillator clock being present.

In addition to the clock monitor, the MCU also provides a clock quality checker which performs a more accurate check of the clock. The clock quality checker counts a predetermined number of clock edges within a defined time window to insure that the clock is running. The checker can be invoked following specific events such as on wake-up or clock monitor failure.

1.4 Modes of Operation

The MCU can operate in different modes associated with MCU resource mapping and bus interface configuration. These are described in 1.4.1 Chip Configuration Summary.

The MCU can operate in different power modes to facilitate power saving when full system performance is not required. These are described in 1.4.2 Power Modes.

Some modules feature a software programmable option to freeze the module status whilst the background debug module is active to facilitate debugging. This is described in 1.4.3 Freeze Mode.

The “system state” for the XCPU is always Supervisor State (see Chapter 17 Memory Protection Unit (S12XMPUV2)).

1.4.1 Chip Configuration Summary

The MCU can operate in six different modes associated with resource configuration. The different modes, the state of ROMCTL and EROMCTL signal on rising edge of $\overline{\text{RESET}}$ and the security state of the MCU affect the following device characteristics:

- External bus interface configuration
- Flash in memory map, or not
- Debug features enabled or disabled

The operating mode out of reset is determined by the states of the MODC, MODB, and MODA signals during reset (see Table 1-12). The MODC, MODB, and MODA bits in the MODE register show the current operating mode and provide limited mode switching during operation. The states of the MODC, MODB, and MODA signals are latched into these bits on the rising edge of $\overline{\text{RESET}}$.

In normal expanded mode and in emulation modes the ROMON and the EROMON bits in the MISC register defines if the on chip flash memory is the memory map, or not. (See Table 1-12 .) For a detailed explanation of the ROMON and EROMON bits refer to the MMC description.

The state of the ROMCTL signal is latched into the ROMON bit in the MMCCTL1 register on the rising edge of $\overline{\text{RESET}}$. The state of the EROMCTL signal is latched into the EROMON bit in the MMCCTL1 register on the rising edge of $\overline{\text{RESET}}$.

Table 1-12. Chip Modes and Data Sources

Chip Modes	MODC	MODB	MODA	ROMCTL	EROMCTL	Data Source[1]
Normal single chip	1	0	0	X	X	Internal
Special single chip	0	0	0			
Emulation single chip	0	0	1	X	0	Emulation memory
				X	1	Internal Flash
Normal expanded	1	0	1	0	X	External application
				1	X	Internal Flash
Emulation expanded	0	1	1	0	X	External application
				1	0	Emulation memory
				1	1	Internal Flash
Special test	0	1	0	0	X	External application
				1	X	Internal Flash

1. Internal means resources inside the MCU are read/written.
Internal Flash means Flash resources inside the MCU are read/written.
Emulation memory means resources inside the emulator are read/written (PRU registers, Flash replacement, RAM, EEEPROM, and register space are always considered internal).
External application means resources residing outside the MCU are read/written.

1.4.1.1 Normal Expanded Mode

Ports K, A, and B are configured as a 23-bit address bus, ports C and D are configured as a 16-bit data bus, and port E provides bus control and status signals. This mode allows 16-bit external memory and peripheral devices to be interfaced to the system. The fastest external bus rate is one half of the internal bus rate.

1.4.1.2 Normal Single-Chip Mode

There is no external bus in this mode. The processor program is executed from internal memory. Ports A, B,C,D, K, and most pins of port E are available as general-purpose I/Os.

1.4.1.3 Special Single-Chip Mode

This mode is used for debugging single-chip operation, boot-strapping, or security related operations. The background debug module BDM is active in this mode. The CPU executes a monitor program located in an on-chip ROM. BDM firmware waits for additional serial commands through the BKGD pin. There is no external bus after reset in this mode.

1.4.1.4 Emulation of Expanded Mode

Developers use this mode for emulation systems in which the users target application is normal expanded mode. Code is executed from external memory or from internal memory depending on the state of ROMON and EROMON bit. In this mode the internal operation is visible on external bus interface.

1.4.1.5 Emulation of Single-Chip Mode

Developers use this mode for emulation systems in which the user's target application is normal single-chip mode. Code is executed from external memory or from internal memory depending on the state of ROMON and EROMON bit. In this mode the internal operation is visible on external bus interface.

1.4.1.6 Special Test Mode

Freescale internal use only.

1.4.2 Power Modes

The MCU features two main low-power modes. Consult the respective module description for module specific behavior in system stop, system pseudo stop, and system wait mode. An important source of information about the clock system is the Clock and Reset Generator description (CRG).

1.4.2.1 System Stop Modes

The system stop modes are entered if the CPU executes the STOP instruction and the S bit in the CCR register is cleared unless either the XGATE is active or an NVM command is active. The XGATE is active if it executes a thread or the XGFACT bit in the XGMCTL register is set. Depending on the state of the PSTP bit in the CLKSEL register the MCU goes into pseudo stop mode or full stop mode. Please refer to CRG description. Asserting $\overline{\text{RESET}}$, $\overline{\text{XIRQ}}$, $\overline{\text{IRQ}}$ or any other interrupt that is not masked causes the system to exit the stop mode. System stop modes can be exited by XGATE or CPU activity independently, depending on the configuration of the interrupt request. If System-Stop is exited on an XGATE request then, as long as the XGATE does not set an interrupt flag on the CPU and the XGATE fake activity bit (FACT) remains cleared, once XGATE activity is completed System Stop mode will automatically be re-entered.

If the CPU executes the STOP instruction whilst XGATE is active or an NVM command is being processed, then the system clocks continue running until XGATE/NVM activity is completed. If a non-masked CPU-serviced interrupt occurs within this time then the system does not effectively enter stop mode although the STOP instruction has been executed.

1.4.2.2 Full Stop Mode

The oscillator is stopped in this mode. By default all clocks are switched off and all counters and dividers remain frozen. The Autonomous Periodic Interrupt (API) and ADC module may be enabled to self wake the device. A Fast wake up mode is available to allow the device to wake from Full Stop mode immediately on the PLL internal clock without starting the oscillator clock.

1.4.2.3 Pseudo Stop Mode

In this mode the system clocks are stopped but the oscillator is still running and the real time interrupt (RTI) and watchdog (COP), API and ATD modules may be enabled. Other peripherals are turned off. This mode consumes more current than system stop mode but, as the oscillator continues to run, the full speed wake up time from this mode is significantly shorter.

1.4.2.4 XGATE Fake Activity Mode

This mode is entered if the CPU executes the STOP instruction when the XGATE is not executing a thread and the XGFACT bit in the XGMCTL register is set. The oscillator remains active and any enabled peripherals continue to function.

1.4.2.5 Wait Mode

This mode is entered when the CPU executes the WAI instruction. In this mode the CPU will not execute instructions. The internal CPU clock is switched off. All peripherals and the XGATE can be active in system wait mode. For further power consumption the peripherals can individually turn off their local clocks. Asserting $\overline{\text{RESET}}$, $\overline{\text{XIRQ}}$, $\overline{\text{IRQ}}$ or any other interrupt that is not masked and is not routed to XGATE ends system wait mode.

1.4.2.6 Run Mode

Although this is not a low-power mode, unused peripheral modules should not be enabled in order to save power.

1.4.3 Freeze Mode

The enhanced capture timer, COP, pulse width modulator, analog-to-digital converter, and the periodic interrupt timer provide a software programmable option to freeze the module status when the background debug module is active. This is useful when debugging application software. For detailed description of the behavior of the ADC, ECT, COP and EPIT when the background debug module is active consult the corresponding Block Guides.

1.5 Security

The MCU security feature allows the protection of on chip NVM memories and RAM. For a detailed description of the security features refer to the S12X9SEC description.

1.6 Resets and Interrupts

Consult the S12XCPU manual and the S12XINT description for information on exception processing.

1.6.1 Resets

Resets are explained in detail in the Clock Reset Generator (CRG) description.

1.6.2 Vectors

Table 1-13 lists all interrupt sources and vectors in the default order of priority. The interrupt module (S12XINT) provides an interrupt vector base register (IVBR) to relocate the vectors. Associated with each I-bit maskable service request is a configuration register. It selects if the service request is enabled, the service request priority level and whether the service request is handled either by the S12X CPU or by the XGATE module. IRQ is I-bit maskable and cannot be serviced by the XGATE.

Table 1-13. Interrupt Vector Locations (Sheet 1 of 4)

Vector Address[1]	XGATE Channel ID[2]	Interrupt Source	CCR Mask	Local Enable
$FFFE	—	System reset or illegal access reset	None	None
$FFFC	—	Clock monitor reset	None	PLLCTL (CME, SCME)
$FFFA	—	COP watchdog reset	None	COP rate select
Vector base + $F8	—	Unimplemented instruction trap	None	None
Vector base+ $F6	—	SWI	None	None
Vector base+ $F4	—	$\overline{\text{XIRQ}}$	X Bit	None
Vector base+ $F2	—	$\overline{\text{IRQ}}$	I bit	IRQCR (IRQEN)
Vector base+ $F0	$78	Real time interrupt	I bit	CRGINT (RTIE)
Vector base+ $EE	$77	Enhanced capture timer channel 0	I bit	TIE (C0I)
Vector base + $EC	$76	Enhanced capture timer channel 1	I bit	TIE (C1I)
Vector base+ $EA	$75	Enhanced capture timer channel 2	I bit	TIE (C2I)
Vector base+ $E8	$74	Enhanced capture timer channel 3	I bit	TIE (C3I)
Vector base+ $E6	$73	Enhanced capture timer channel 4	I bit	TIE (C4I)
Vector base+ $E4	$72	Enhanced capture timer channel 5	I bit	TIE (C5I)
Vector base + $E2	$71	Enhanced capture timer channel 6	I bit	TIE (C6I)
Vector base+ $E0	$70	Enhanced capture timer channel 7	I bit	TIE (C7I)
Vector base+ $DE	$6F	Enhanced capture timer overflow	I bit	TSRC2 (TOF)
Vector base+ $DC	$6E	Pulse accumulator A overflow	I bit	PACTL (PAOVI)
Vector base + $DA	$6D	Pulse accumulator input edge	I bit	PACTL (PAI)
Vector base + $D8	$6C	SPI0	I bit	SPI0CR1 (SPIE, SPTIE)
Vector base+ $D6	$6B	SCI0	I bit	SCI0CR2 (TIE, TCIE, RIE, ILIE)
Vector base + $D4	$6A	SCI1	I bit	SCI1CR2 (TIE, TCIE, RIE, ILIE)
Vector Base + $D2	Reserved			
Vector base + $D0	$68	ATD	I bit	ATDCTL2 (ASCIE)
Vector Base + $CE	Reserved			
Vector Base + $CC	Reserved			
Vector base + $CA	$65	Modulus down counter underflow	I bit	MCCTL (MCZI)
Vector base + $C8	$64	Pulse accumulator B overflow	I bit	PBCTL (PBOVI)
Vector base + $C6	$63	CRG PLL lock	I bit	CRGINT (LOCKIE)
Vector base + $C4	$62	CRG self-clock mode	I bit	CRGINT (SCMIE)
Vector base + $C2	$61	CGM IPLL change of lock	I bit	CGMFLG (LOCKIE)
Vector base + $C0	Reserved			
Vector base + $BE	$5F	SPI1	I bit	SPI1CR1 (SPIE, SPTIE)

Table 1-13. Interrupt Vector Locations (Sheet 2 of 4)

Vector Address[1]	XGATE Channel ID[2]	Interrupt Source	CCR Mask	Local Enable
Vector base + $BC	$5E	Reserved		
Vector base + $BA	$5D	FLASH Fault Detect	I bit	FCNFG2 (FDIE)
Vector base + $B8	$5C	FLASH	I bit	FCNFG (CCIE, CBEIE)
Vector base + $B6	$5B	CAN wake-up	I bit	CANRIER (WUPIE)
Vector base + $B4	$5A	CAN errors	I bit	CANRIER (CSCIE, OVRIE)
Vector base + $B2	$59	CAN receive	I bit	CANRIER (RXFIE)
Vector base + $B0	$58	CAN transmit	I bit	CANTIER (TXEIE[2:0])
Vector base + $AE	$57	Reserved		
Vector base + $AC	$56	Reserved		
Vector base + $AA	$55	Reserved		
Vector base + $A8	$54	Reserved		
Vector Base + $A6	$53	FlexRay Transmit Message Buffer Interrupt	I-Bit	GIFER (TBIE)
Vector Base + $A4	$52	FlexRay Receive Message Buffer Interrupt	I-Bit	GIFER (RBIE)
Vector Base + $A2	$51	FlexRay Receive FIFO channel A Not Empty Interrupt	I-Bit	GIFER (FNEAIE)
Vector Base + $A0	$50	FlexRay Receive FIFO channel B Not Empty Interrupt	I-Bit	GIFER (FNEBIE)
Vector Base + $9E	$4F	FlexRay Wakeup Interrupt	I-Bit	GIFER (WUPIE)
Vector Base+ $9C	$4E	FlexRay CHI Interrupt	I-Bit	GIFER (CHIE)
Vector Base+ $9A	$4D	FlexRay Protocol Interrupt	I-Bit	GIFER (PRIE)
Vector Base + $98	$4C	PMF Generator A Reload	I-Bit	PMFENCA (PWMRIEA)
Vector Base + $96	$4B	PMF Generator B Reload	I-Bit	PMFENCB (PWMRIEB)
Vector Base + $94	$4A	PMF Generator C Reload	I-Bit	PMFENCC (PWMRIEC)
Vector Base + $92	$49	PMF Fault 0	I-Bit	PMFFCTL (FIE0)
Vector Base + $90	$48	PMF Fault 1	I-Bit	PMFFCTL (FIE1)
Vector Base + $8E	$47	PMF Fault 2	I-Bit	PMFFCTL (FIE2)
Vector Base+ $8C	$46	PMF Fault 3	I-Bit	PMFFCTL (FIE3)
Vector Base + $8A	Reserved			
Vector Base + $88	Reserved			
Vector Base + $86	Reserved			
Vector Base + $84	Reserved			
Vector Base + $82	Reserved			
Vector base + $80	$40	Low-voltage interrupt (LVI)	I bit	VREGCTRL (LVIE)
Vector base + $7E	$3F	Autonomous periodical interrupt (API)	I bit	VREGAPICTRL (APIE)
Vector base + $7C	Reserved			

Table 1-13. Interrupt Vector Locations (Sheet 3 of 4)

Vector Address[1]	XGATE Channel ID[2]	Interrupt Source	CCR Mask	Local Enable
Vector base + $7A	$3D	Periodic interrupt timer channel 0	I bit	PITINTE (PINTE0)
Vector base + $78	$3C	Periodic interrupt timer channel 1	I bit	PITINTE (PINTE1)
Vector base + $76	$3B	Periodic interrupt timer channel 2	I bit	PITINTE (PINTE2)
Vector base + $74	$3A	Periodic interrupt timer channel 3	I bit	PITINTE (PINTE3)
Vector base + $72	$39	XGATE software trigger 0	I bit	XGMCTL (XGIE)
Vector base + $70	$38	XGATE software trigger 1	I bit	XGMCTL (XGIE)
Vector base + $6E	$37	XGATE software trigger 2	I bit	XGMCTL (XGIE)
Vector base + $6C	$36	XGATE software trigger 3	I bit	XGMCTL (XGIE)
Vector base + $6A	$35	XGATE software trigger 4	I bit	XGMCTL (XGIE)
Vector base + $68	$34	XGATE software trigger 5	I bit	XGMCTL (XGIE)
Vector base + $66	$33	XGATE software trigger 6	I bit	XGMCTL (XGIE)
Vector base + $64	$32	XGATE software trigger 7	I bit	XGMCTL (XGIE)
Vector base + $62	Reserved			
Vector base + $60	Reserved			
Vector base + $5E	$2F	Periodic interrupt timer channel 4	I bit	PITINTE (PINTE4)
Vector base + $5C	$2E	Periodic interrupt timer channel 5	I bit	PITINTE (PINTE5)
Vector base + $5A	$2D	Periodic interrupt timer channel 6	I bit	PITINTE (PINTE6)
Vector base + $58	$2C	Periodic interrupt timer channel 7	I bit	PITINTE (PINTE7)
Vector base + $56	$2B	Input Trigger Interrupt	I bit	PITTRIGIE
Vector base + $54	Reserved			
Vector base + $52	Reserved			
Vector base + $50	Reserved			
Vector base+ $4E	Reserved			
Vector base + $4C	Reserved			
Vector base+ $4A	Reserved			
Vector base+ $48	Reserved			
Vector base+ $46	Reserved			
Vector base+ $44	Reserved			
Vector base + $42	Reserved			
Vector base+ $40	Reserved			
Vector base+ $3E	Reserved			
Vector base + $3C	$1E	ATD Compare Interrupt	I bit	ATDCTL2 (ACMPIE)
Vector base + $18 to Vector base + $3A	Reserved			

Table 1-13. Interrupt Vector Locations (Sheet 4 of 4)

Vector Address[1]	XGATE Channel ID[2]	Interrupt Source	CCR Mask	Local Enable
Vector base + $16	—	XGATE software error interrupt	None	None
Vector base + $14	—	MPU Access Error	None	None
Vector base + $12	—	System Call Interrupt (SYS)	—	None
Vector base + $10	—	Spurious interrupt	—	None

1. 16 bits vector address based
2. For detailed description of XGATE channel ID refer to XGATE Block Guide

1.6.3 Effects of Reset

When a reset occurs, MCU registers and control bits are initialized. Refer to the respective block descriptions for register reset states.

On each reset, the Flash module executes a reset sequence to load Flash configuration registers and initialize the buffer RAM EEE partition, if required.

1.6.3.1 Flash Configuration Reset Sequence Phase (Core Hold Phase)

On each reset, the Flash module will hold CPU activity while loading Flash module registers and configuration from the Flash memory. The duration of this phase is given as tRST in the device electrical parameter specification. If double faults are detected in the reset phase, Flash module protection and security may be active on leaving reset. This is explained in more detail in the Flash (FTM) module section.

1.6.3.2 EEE Reset Sequence Phase (Core Active Phase)

During this phase of the reset sequence (following on from the core hold phase) the CPU can execute instructions while the FTM initialization completes and, if configured for EEE operation, the EEE RAM is loaded with valid data from the D-Flash EEE partition. Completion of this phase is indicated by the CCIF flag in the FTM FSTAT register becoming set. If the CPU accesses any EEE RAM location before the CCIF flag is set, the CPU is stalled until the FTM reset sequence is complete and the EEE RAM data is valid. Once the CCIF flag is set, indicating the end of this phase, the EEE RAM can be accessed without impacting the CPU and FTM commands can be executed.

1.6.3.3 Reset While Flash Command Active

If a reset occurs while any Flash command is in progress, that command will be immediately aborted. The state of the word being programmed or the sector/block being erased is not guaranteed.

1.6.3.4 I/O Pins

Refer to the PIM block description for reset configurations of all peripheral module ports.

1.6.3.5 Memory

The RAM arrays are not initialized out of reset with exception of the EEE buffer RAM (providing the EEE functionality is enabled).

1.6.3.6 COP Configuration

The COP timeout rate bits CR[2:0] and the WCOP bit in the COPCTL register are loaded on rising edge of $\overline{\text{RESET}}$ from the Flash register FOPT. See Table 1-14 and Table 1-15 for coding. The FOPT register is loaded from the Flash configuration field byte at global address $7FFF0E during the reset sequence.

If the MCU is secured and COP is enabled, the COP timeout rate is always set to the longest period (CR[2:0] = 111) after COP reset and after any reset into Special Single Chip mode.

Table 1-14. Initial COP Rate Configuration

NV[2:0] in FCTL Register	CR[2:0] in COPCTL Register
000	111
001	110
010	101
011	100
100	011
101	010
110	001
111	000

Table 1-15. Initial WCOP Configuration

NV[3] in FCTL Register	WCOP in COPCTL Register
1	0
0	1

1.7 EPIT External Trigger Input

The start of the timer channels can be aligned to an external trigger event. Four trigger event sources can be connected. The MC9S12XF Family uses two sources for external trigger events. See Table 1-16 and the EPIT block guide for more details.

Table 1-16. External Trigger Input Sources

External Trigger Input	Connectivity	EPIT Register Settings PITTRIGSRC[1:0]
TRIGIN0	EPIT - Hardware Trigger 0	00
TRIGIN1	Start of PWM Cycle Channel A	01
TRIGIN2	ECT Input Capture/Output Compare Interrupt 0	10
TRIGIN3	Not Connected	11 (1)

1. No effect as external trigger input is tied.

1.8 ATD External Trigger Input Connection

The ATD module includes four external trigger inputs ETRIG0, ETRIG1, ETRIG2, and ETRIG3. The external trigger allows the user to synchronize ATD conversion to external trigger events. Table 1-17 shows the connection of the external trigger inputs.

Table 1-17. ATD0 External Trigger Sources

External Trigger Input	Connectivity
ETRIG0	Start of PWM Cycle Channel A(1)
ETRIG1	EPIT - Combined Trigger(2)
ETRIG2	EPIT - Hardware Trigger 0(3)
ETRIG3	EPIT - Hardware Trigger 1(4)

1. Indicates start of new PWM cycle.
2. Selectable hardware trigger. One of eight EPIT channels can be selected. The trigger interval is started via a PMF output.
3. Interrupt timer hardware trigger channel 0
4. Interrupt timer hardware trigger channel 1

Consult the EPIT block description for more information about hardware trigger generation.

Consult the ATD block description for information about the analog-to-digital converter module. ATD block description refererences to freeze mode are equivalent to active BDM mode.

1.9 MPU Configuration

The MPU can handle 3 bus masters (CPU + XGATE + FlexRay). The MPU covers the system ram address space. See MPU documentation for more details.

Table 1-18. MPU Configuration

Parameter	Parameter Value
Number of Descriptors	4
Descripter Granularity(1)	8

1. Number of least significant address bits to treat as constant.

NOTE

- The FlexRay module is Master 3 (MSTR3)
- CPU user state is not supported

on this device.

1.10 VREG Configuration

The VREGEN connection of the voltage regulator is tied internally to VDDR such that the voltage regulator is always enabled with VDDR connected to a positive supply voltage. The device must be configured with the internal voltage regulator enabled. Operation in conjunction with an external voltage regulator is not supported.

The autonomous periodic interrupt clock output is mapped to PortT[5].

The API trimming register APITR is loaded on rising edge of $\overline{\text{RESET}}$ from the Flash IFR option field at global address 0x40_00F0 bits[5:0] during the reset sequence. Currently factory programming of this IFR range is not supported.

1.10.1 Temperature Sensor Configuration

The VREG high temperature trimming register bits VREGHTTR[3:0] are loaded from the Flash IFR option field at global address 0x40_00F0 bits[11:8] during the reset sequence. To use the high temperature interrupt within the specified limits (T_{HTIA} and T_{HTID}) these bits must be programmed to 0x8. Currently factory programming of this IFR range is not supported. Note that the API trimming bits are also loaded from 0x40_00F0[5:0].

The device temperature can be monitored on ADC0 channel[17].
The internal bandgap reference voltage can also be mapped to ADC0 analog input channel[17]. The voltage regulator VSEL bit when set, maps the bandgap and, when clear, maps the temperature sensor to ADC0 channel[17].

Read access to reserved VREG register space returns “0”. Write accesses have no effect. This device does not support access abort of reserved VREG register space.

1.11 BDM Configuration

The BDM alternative clock corresponds to the oscillator clock.

1.12 FlexRay IPLL (CGMIPLL) Configuration

MC9S12XF512 features a dedicated internal PLL for the FlexRay protocol engine. The IPLL hard IP and the register map for the configuration registers is identical to the system IPLL.

The usage of an dedicated internal PLL allows to use cheaper crystal devices and to achieve lower radiation.

1.12.1 CGMIPLL function

The CGMIPLL module supplies the clock to the FlexRay controller. The FlexRay controller can only operate according to FlexRay specification when it is supplied with stable 80MHz clock. The CGMIPLL must be configured to provide an 80MHz clock on its output (see Chapter 12, “Clock Generation Module using IPLL (CGMIPLL) Block Description“ for more details) and the FlexRay controller must be

configured to use the CGMIPLL as its clock source (bit CLKSEL in register MCR, see section 13.5.2.4, “Module Configuration Register (MCR)“ for more details).

It is the responsibility of the software to ensure that a stable 80MHz clock is supplied to the FlexRay controller while it is enabled.

NOTE

FlexRay applications have to use the FlexRay IPLL as clock source for the FlexRay protocol engine. The option to use the crystal as clock source is only intended for test purposes.

The CGMIPLL has to be configured for 80MHz to guarantee FlexRay functionality.

FlexRay needs a stable clock. Make sure the PLL is locked before enabling FlexRay and make sure it remains locked while FlexRay is running.

Frequency modulation should be turned off on the FlexRay IPLL.

1.12.2 Entry into and exit from low power modes

To ensure correct entry into stop mode the software should perform the following steps:

1. Shut down the FlexRay Controller (see 13.7.2 Shut Down Sequence for more details)
2. Turn off the CGMIPLL module by clearing the PLLON bit in the CGMCTL register (see section 12.3.2.4 CGMIPLL Control Register (CGMCTL) for more details)
3. Perform additional application specific tasks and enter low power mode

Once the microcontroller is woken-up from the low power mode the firmware should perform the following steps:

1. Turn on the CGMIPLL module by setting the PLLON bit in the CGMCTL register
2. Wait for the CGMIPLL to achieve lock by waiting for the LOCK bit in the CGMCTL register to become set
3. Re-initialize the FlexRay controller (see 13.7.1 Initialization Sequence for more details)

1.13 Oscillator Configuration

The $\overline{\text{XCLKS}}$ is an input signal which controls whether a crystal in combination with the internal loop controlled (low power) Pierce oscillator is used or whether full swing Pierce oscillator/external clock circuitry is used. For this device $\overline{\text{XCLKS}}$ is mapped to PE7.

The $\overline{\text{XCLKS}}$ signal selects the oscillator configuration during reset low phase while a clock quality check is ongoing. This is the case for:

- Power on reset or low-voltage reset
- Clock monitor reset
- Any reset while in self-clock mode or full stop mode

The selected oscillator configuration is frozen with the rising edge of the RESET pin in any of these above described reset cases.

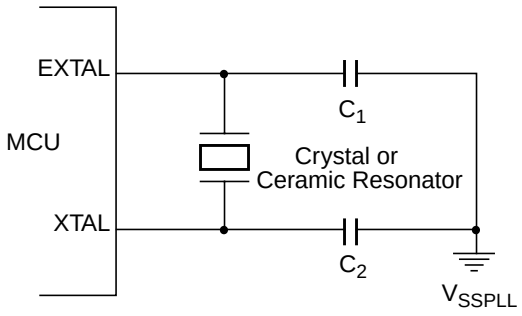


Figure 1-9. Loop Controlled Pierce Oscillator Connections ($\overline{\text{XCLKS}}$ = 1)

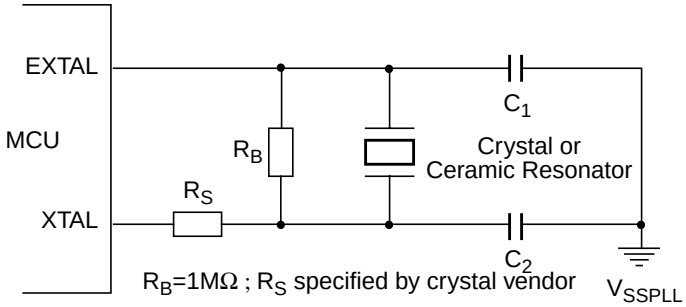


Figure 1-10. Full Swing Pierce Oscillator Connections ($\overline{\text{XCLKS}}$ = 0)

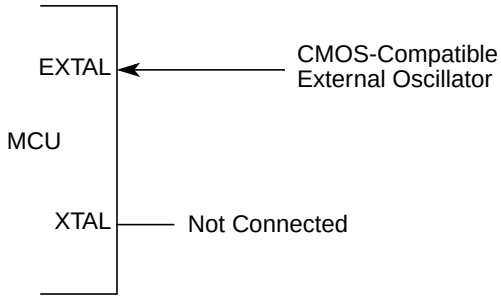


Figure 1-11. External Clock Connections ($\overline{\text{XCLKS}}$ = 0)

Chapter 2
S12XE Clocks and Reset Generator (S12XECRG)

Table 2-1. Revision History

Revision Number	Revision Date	Sections Affected	Description of Changes
V01.00	26 Oct. 2005		Initial release
V01.01	02 Nov 2006	2.4.1.1/2-100	Table “Examples of IPLL Divider settings”: corrected $32 to $31
V01.02	4 Mar. 2008	2.4.1.4/2-103 2.4.3.3/2-107	Corrected details
V01.03	1 Sep. 2008	Table 2-14	added 100MHz example for PLL
V01.04	20 Nov. 2008	2.3.2.4/2-89	S12XECRG Flags Register: corrected address to Module Base + 0x0003
V01.05	19. Sep 2009	2.5.1/2-109	Modified Note below Table 2-17./2-109

2.1 Introduction

This specification describes the function of the Clocks and Reset Generator (S12XECRG).

2.1.1 Features

The main features of this block are:

- Phase Locked Loop (IPLL) frequency multiplier with internal filter
  - Reference divider
  - Post divider
  - Configurable internal filter (no external pin)
  - Optional frequency modulation for defined jitter and reduced emission
  - Automatic frequency lock detector
  - Interrupt request on entry or exit from locked condition
  - Self Clock Mode in absence of reference clock
- System Clock Generator
  - Clock Quality Check
  - User selectable fast wake-up from Stop in Self-Clock Mode for power saving and immediate program execution
  - Clock switch for either Oscillator or PLL based system clocks
- Computer Operating Properly (COP) watchdog timer with time-out clear window.
- System Reset generation from the following possible sources:
  - Power on reset

  - Low voltage reset
  - Illegal address reset
  - COP reset
  - Loss of clock reset
  - External pin reset
- Real-Time Interrupt (RTI)

2.1.2 Modes of Operation

This subsection lists and briefly describes all operating modes supported by the S12XECRG.

- Run Mode

  All functional parts of the S12XECRG are running during normal Run Mode. If RTI or COP functionality is required the individual bits of the associated rate select registers (COPCTL, RTICTL) have to be set to a non zero value.
- Wait Mode

  In this mode the IPLL can be disabled automatically depending on the PLLWAI bit.
- Stop Mode

  Depending on the setting of the PSTP bit Stop Mode can be differentiated between Full Stop Mode (PSTP = 0) and Pseudo Stop Mode (PSTP = 1).
  - Full Stop Mode

    The oscillator is disabled and thus all system and core clocks are stopped. The COP and the RTI remain frozen.
  - Pseudo Stop Mode

    The oscillator continues to run and most of the system and core clocks are stopped. If the respective enable bits are set the COP and RTI will continue to run, else they remain frozen.
- Self Clock Mode

  Self Clock Mode will be entered if the Clock Monitor Enable Bit (CME) and the Self Clock Mode Enable Bit (SCME) are both asserted and the clock monitor in the oscillator block detects a loss of clock. As soon as Self Clock Mode is entered the S12XECRG starts to perform a clock quality check. Self Clock Mode remains active until the clock quality check indicates that the required quality of the incoming clock signal is met (frequency and amplitude). Self Clock Mode should be used for safety purposes only. It provides reduced functionality to the MCU in case a loss of clock is causing severe system conditions.

2.1.3 Block Diagram

Figure 2-1 shows a block diagram of the S12XECRG.

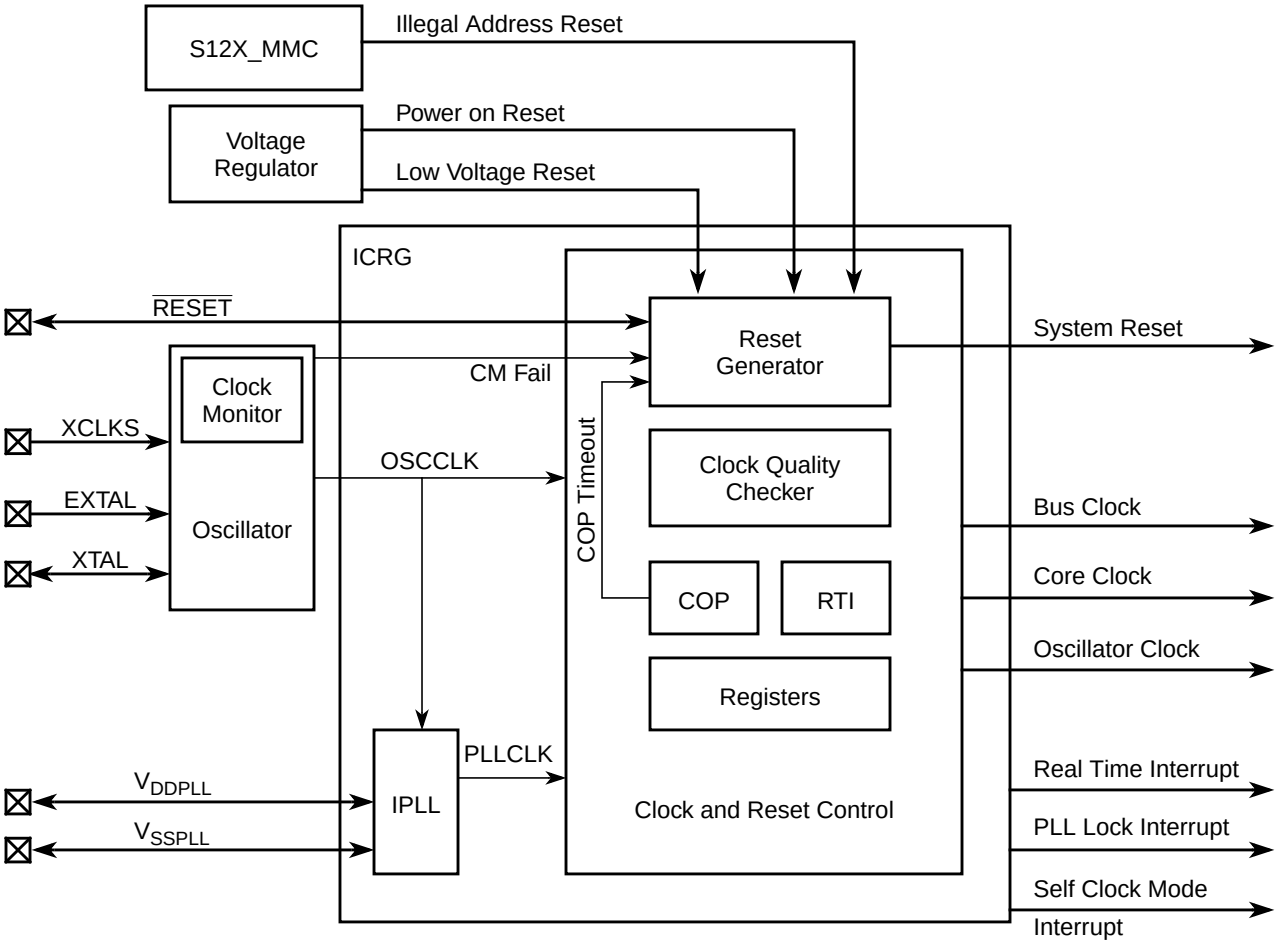


Figure 2-1. Block diagram of S12XECRG

2.2 Signal Description

This section lists and describes the signals that connect off chip.

2.2.1 V_{DDPLL}, V_{SSPLL}

These pins provides operating voltage (V_{DDPLL}) and ground (V_{SSPLL}) for the IPLL circuitry. This allows the supply voltage to the IPLL to be independently bypassed. Even if IPLL usage is not required V_{DDPLL} and V_{SSPLL} must be connected to properly.

2.2.2 $\overline{\text{RESET}}$

$\overline{\text{RESET}}$ is an active low bidirectional reset pin. As an input it initializes the MCU asynchronously to a known start-up state. As an open-drain output it indicates that an system reset (internal to MCU) has been triggered.

2.3 Memory Map and Registers

This section provides a detailed description of all registers accessible in the S12XECRG.

2.3.1 Module Memory Map

Figure 2-2 gives an overview on all S12XECRG registers.

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000	SYNR	R	VCOFRQ[1:0]		SYNDIV[5:0]					
		W								
0x0001	REFDV	R	REFFRQ[1:0]		REFDIV[5:0]					
		W								
0x0002	POSTDIV	R	0	0	0	POSTDIV[4:0]				
		W								
0x0003	CRGFLG	R	RTIF	PORF	LVRF	LOCKIF	LOCK	ILAF	SCMIF	SCM
		W								
0x0004	CRGINT	R	RTIE	0	0	LOCKIE	0	0	SCMIE	0
		W								
0x0005	CLKSEL	R	PLLSEL	PSTP	XCLKS	0	PLLWAI	0	RTIWAI	COPWAI
		W								
0x0006	PLLCTL	R	CME	PLLON	FM1	FM0	FSTWKP	PRE	PCE	SCME
		W								
0x0007	RTICTL	R	RTDEC	RTR6	RTR5	RTR4	RTR3	RTR2	RTR1	RTR0
		W								
0x0008	COPCTL	R	WCOP	RSBCK	0	0	0	CR2	CR1	CR0
		W			WRTMASK					
0x0009	FORBYP[2]	R	0	0	0	0	0	0	0	0
		W								
0x000A	CTCTL[2]	R	0	0	0	0	0	0	0	0
		W								
0x000B	ARMCOP	R	0	0	0	0	0	0	0	0
		W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

2. FORBYP and CTCTL are intended for factory test purposes only.

= Unimplemented or Reserved

Figure 2-2. CRG Register Summary

NOTE

Register Address = Base Address + Address Offset, where the Base Address is defined at the MCU level and the Address Offset is defined at the module level.

2.3.2 Register Descriptions

This section describes in address order all the S12XECRG registers and their individual bits.

2.3.2.1 S12XECRG Synthesizer Register (SYNR)

The SYNR register controls the multiplication factor of the IPLL and selects the VCO frequency range.

Module Base + 0x0000

	7	6	5	4	3	2	1	0
R W	VCOFRQ[1:0]		SYNDIV[5:0]					
Reset	0	0	0	0	0	0	0	0

Figure 2-3. S12XECRG Synthesizer Register (SYNR)

Read: Anytime

Write: Anytime except if PLLSEL = 1

NOTE

Write to this register initializes the lock detector bit.

$$f_{VCO} = 2 \times f_{OSC} \times \frac{(SYNDIV + 1)}{(REFDIV + 1)}$$

$$f_{PLL} = \frac{f_{VCO}}{2 \times POSTDIV}$$

$$f_{BUS} = \frac{f_{PLL}}{2}$$

NOTE

f_{VCO} must be within the specified VCO frequency lock range. $F._{BUS}$ (Bus Clock) must not exceed the specified maximum. If POSTDIV = $00 then f_{PLL} is same as f_{VCO} (divide by one).

The VCOFRQ[1:0] bit are used to configure the VCO gain for optimal stability and lock time. For correct IPLL operation the VCOFRQ[1:0] bits have to be selected according to the actual target VCOCLK frequency as shown in Table 2-2. Setting the VCOFRQ[1:0] bits wrong can result in a non functional IPLL (no locking and/or insufficient stability).

Table 2-2. VCO Clock Frequency Selection

VCOCLK Frequency Ranges	VCOFRQ[1:0]
32MHz <= f_{VCO}<= 48MHz	00
48MHz < f_{VCO}<= 80MHz	01
Reserved	10
80MHz < f_{VCO} <= 120MHz	11

2.3.2.2 S12XECRG Reference Divider Register (REFDV)

The REFDV register provides a finer granularity for the IPLL multiplier steps.

Module Base + 0x0001

	7	6	5	4	3	2	1	0
R W	REFFRQ[1:0]		REFDIV[5:0]					
Reset	0	0	0	0	0	0	0	0

Figure 2-4. S12XECRG Reference Divider Register (REFDV)

Read: Anytime

Write: Anytime except when PLLSEL = 1

NOTE

Write to this register initializes the lock detector bit.

$$f_{REF} = \frac{f_{OSC}}{(REFDIV + 1)}$$

The REFFRQ[1:0] bit are used to configure the internal PLL filter for optimal stability and lock time. For correct IPLL operation the REFFRQ[1:0] bits have to be selected according to the actual REFCLK frequency as shown in Figure 2-3. Setting the REFFRQ[1:0] bits wrong can result in a non functional IPLL (no locking and/or insufficient stability).

Table 2-3. Reference Clock Frequency Selection

REFCLK Frequency Ranges	REFFRQ[1:0]
1MHz <= f_{REF} <= 2MHz	00
2MHz < f_{REF} <= 6MHz	01
6MHz < f_{REF} <= 12MHz	10
f_{REF} >12MHz	11

2.3.2.3 S12XECRG Post Divider Register (POSTDIV)

The POSTDIV register controls the frequency ratio between the VCOCLK and PLLCLK. The count in the final divider divides VCOCLK frequency by 1 or 2*POSTDIV. Note that if POSTDIV = $00 f_{PLL}= f_{VCO} (divide by one).

Module Base + 0x0002

	7	6	5	4	3	2	1	0
R	0	0	0	POSTDIV[4:0]				
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 2-5. S12XECRG Post Divider Register (POSTDIV)

Read: Anytime

Write: Anytime except if PLLSEL = 1

$$f_{PLL} = \frac{f_{VCO}}{(2 \times POSTDIV)}$$

NOTE

If POSTDIV = $00 then f_{PLL} is identical to f_{VCO} (divide by one).

2.3.2.4 S12XECRG Flags Register (CRGFLG)

This register provides S12XECRG status bits and flags.

Module Base + 0x0003

	7	6	5	4	3	2	1	0
R	RTIF	PORF	LVRF	LOCKIF	LOCK	ILAF	SCMIF	SCM
W								
Reset	0	Note 1	Note 2	Note 3	0	0	0	0

1. PORF is set to 1 when a power on reset occurs. Unaffected by system reset.
2. LVRF is set to 1 when a low voltage reset occurs. Unaffected by system reset.
3. ILAF is set to 1 when an illegal address reset occurs. Unaffected by system reset. Cleared by power on or low voltage reset.

= Unimplemented or Reserved

Figure 2-6. S12XECRG Flags Register (CRGFLG)

Read: Anytime

Write: Refer to each bit for individual write conditions

Table 2-4. CRGFLG Field Descriptions

Field	Description
7 RTIF	**Real Time Interrupt Flag** — RTIF is set to 1 at the end of the RTI period. This flag can only be cleared by writing a 1. Writing a 0 has no effect. If enabled (RTIE=1), RTIF causes an interrupt request. 0 RTI time-out has not yet occurred. 1 RTI time-out has occurred.
6 PORF	**Power on Reset Flag** — PORF is set to 1 when a power on reset occurs. This flag can only be cleared by writing a 1. Writing a 0 has no effect. 0 Power on reset has not occurred. 1 Power on reset has occurred.
5 LVRF	**Low Voltage Reset Flag** — LVRF is set to 1 when a low voltage reset occurs. This flag can only be cleared by writing a 1. Writing a 0 has no effect. 0 Low voltage reset has not occurred. 1 Low voltage reset has occurred.
4 LOCKIF	IPLL **Lock Interrupt Flag** — LOCKIF is set to 1 when LOCK status bit changes. This flag can only be cleared by writing a 1. Writing a 0 has no effect.If enabled (LOCKIE=1), LOCKIF causes an interrupt request. 0 No change in LOCK bit. 1 LOCK bit has changed.
3 LOCK	**Lock Status Bit** — LOCK reflects the current state of IPLL lock condition. This bit is cleared in Self Clock Mode. Writes have no effect. 0 VCOCLK is not within the desired tolerance of the target frequency. 1 VCOCLK is within the desired tolerance of the target frequency.
2 ILAF	**Illegal Address Reset Flag** — ILAF is set to 1 when an illegal address reset occurs. Refer to S12XMMC Block Guide for details. This flag can only be cleared by writing a 1. Writing a 0 has no effect. 0 Illegal address reset has not occurred. 1 Illegal address reset has occurred.
1 SCMIF	**Self Clock Mode Interrupt Flag** — SCMIF is set to 1 when SCM status bit changes. This flag can only be cleared by writing a 1. Writing a 0 has no effect. If enabled (SCMIE=1), SCMIF causes an interrupt request. 0 No change in SCM bit. 1 SCM bit has changed.
0 SCM	**Self Clock Mode Status Bit** — SCM reflects the current clocking mode. Writes have no effect. 0 MCU is operating normally with OSCCLK available. 1 MCU is operating in Self Clock Mode with OSCCLK in an unknown state. All clocks are derived from PLLCLK running at its minimum frequency f_{SCM}.

2.3.2.5 S12XECRG Interrupt Enable Register (CRGINT)

This register enables S12XECRG interrupt requests.

Module Base + 0x0004

	7	6	5	4	3	2	1	0
R	RTIE	0	0	LOCKIE	0	0	SCMIE	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 2-7. S12XECRG Interrupt Enable Register (CRGINT)

Read: Anytime

Write: Anytime

Table 2-5. CRGINT Field Descriptions

Field	Description
7 RTIE	**Real Time Interrupt Enable Bit** 0 Interrupt requests from RTI are disabled. 1 Interrupt will be requested whenever RTIF is set.
4 LOCKIE	**Lock Interrupt Enable Bit** 0 LOCK interrupt requests are disabled. 1 Interrupt will be requested whenever LOCKIF is set.
1 SCMIE	**Self Clock Mode Interrupt Enable Bit** 0 SCM interrupt requests are disabled. 1 Interrupt will be requested whenever SCMIF is set.

2.3.2.6 S12XECRG Clock Select Register (CLKSEL)

This register controls S12XECRG clock selection. Refer toFigure 2-16 for more details on the effect of each bit.

Module Base + 0x0005

	7	6	5	4	3	2	1	0
R	PLLSEL	PSTP	XCLKS	0	PLLWAI	0	RTIWAI	COPWAI
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 2-8. S12XECRG Clock Select Register (CLKSEL)

Read: Anytime

Write: Refer to each bit for individual write conditions

Table 2-6. CLKSEL Field Descriptions

Field	Description
7 PLLSEL	**PLL Select Bit** Write: Anytime. Writing a one when LOCK=0 has no effect. This prevents the selection of an unstable PLLCLK as SYSCLK. PLLSEL bit is cleared when the MCU enters Self Clock Mode, Stop Mode or Wait Mode with PLLWAI bit set. **It is recommended to read back the PLLSEL bit to make sure PLLCLK has really been selected as SYSCLK, as LOCK status bit could theoretically change at the very moment writing the PLLSEL bit.** 0 System clocks are derived from OSCCLK ($f_{BUS} = f_{OSC} / 2$). 1 System clocks are derived from PLLCLK ($f_{BUS} = f_{PLL} / 2$).
6 PSTP	**Pseudo Stop Bit** Write: Anytime This bit controls the functionality of the oscillator during Stop Mode. 0 Oscillator is disabled in Stop Mode. 1 Oscillator continues to run in Stop Mode (Pseudo Stop). **Note:** Pseudo Stop Mode allows for faster STOP recovery and reduces the mechanical stress and aging of the resonator in case of frequent STOP conditions at the expense of a slightly increased power consumption.
5 XCLKS	**Oscillator Configuration Status Bit** — This read-only bit shows the oscillator configuration status. 0 Loop controlled Pierce Oscillator is selected. 1 External clock / full swing Pierce Oscillator is selected.
3 PLLWAI	**PLL Stops in Wait Mode Bit** Write: Anytime If PLLWAI is set, the S12XECRG will clear the PLLSEL bit before entering Wait Mode. The PLLON bit remains set during Wait Mode but the IPLL is powered down. Upon exiting Wait Mode, the PLLSEL bit has to be set manually if PLL clock is required. 0 IPLL keeps running in Wait Mode. 1 IPLL stops in Wait Mode.
1 RTIWAI	**RTI Stops in Wait Mode Bit** Write: Anytime 0 RTI keeps running in Wait Mode. 1 RTI stops and initializes the RTI dividers whenever the part goes into Wait Mode.
0 COPWAI	**COP Stops in Wait Mode Bit** Normal modes: Write once Special modes: Write anytime 0 COP keeps running in Wait Mode. 1 COP stops and initializes the COP counter whenever the part goes into Wait Mode.

2.3.2.7 S12XECRG IPLL Control Register (PLLCTL)

This register controls the IPLL functionality.

Module Base + 0x0006

	7	6	5	4	3	2	1	0
R W	CME	PLLON	FM1	FM0	FSTWKP	PRE	PCE	SCME
Reset	1	1	0	0	0	0	0	1

Figure 2-9. S12XECRG IPLL Control Register (PLLCTL)

Read: Anytime

Write: Refer to each bit for individual write conditions

Table 2-7. PLLCTL Field Descriptions

Field	Description
7 CME	**Clock Monitor Enable Bit** — CME enables the clock monitor. Write anytime except when SCM = 1. 0 Clock monitor is disabled. 1 Clock monitor is enabled. Slow or stopped clocks will cause a clock monitor reset sequence or Self Clock Mode. **Note:** Operating with CME=0 will not detect any loss of clock. In case of poor clock quality this could cause unpredictable operation of the MCU! In Stop Mode (PSTP=0) the clock monitor is disabled independently of the CME bit setting and any loss of external clock will not be detected. Also after wake-up from stop mode (PSTP = 0) with fast wake-up enabled (FSTWKP = 1) the clock monitor is disabled independently of the CME bit setting and any loss of external clock will not be detected.
6 PLLON	**Phase Lock Loop On Bit** — PLLON turns on the IPLL circuitry. In Self Clock Mode, the IPLL is turned on, but the PLLON bit reads the last written value. Write anytime except when PLLSEL = 1. 0 IPLL is turned off. 1 IPLL is turned on.
5, 4 FM1, FM0	IPLL **Frequency Modulation Enable Bit** — FM1 and FM0 enable additional frequency modulation on the VCOCLK. This is to reduce noise emission. The modulation frequency is f_{ref} divided by 16. Write anytime except when PLLSEL = 1. See Table 2-8 for coding.
3 FSTWKP	**Fast Wake-up from Full Stop Bit** — FSTWKP enables fast wake-up from full stop mode. Write anytime. If Self-Clock Mode is disabled (SCME = 0) this bit has no effect. 0 Fast wake-up from full stop mode is disabled. 1 Fast wake-up from full stop mode is enabled. When waking up from full stop mode the system will immediately resume operation in Self-Clock Mode (see Section 2.4.1.4, “Clock Quality Checker”). The SCMIF flag will not be set. The system will remain in Self-Clock Mode with oscillator and clock monitor disabled until FSTWKP bit is cleared. The clearing of FSTWKP will start the oscillator, the clock monitor and the clock quality check. If the clock quality check is successful, the S12XECRG will switch all system clocks to OSCCLK. The SCMIF flag will be set. See application examples in Figure 2-19 and Figure 2-20.
2 PRE	**RTI Enable During Pseudo Stop Bit** — PRE enables the RTI during Pseudo Stop Mode. Write anytime. 0 RTI stops running during Pseudo Stop Mode. 1 RTI continues running during Pseudo Stop Mode. **Note:** If the PRE bit is cleared the RTI dividers will go static while Pseudo Stop Mode is active. The RTI dividers will not initialize like in Wait Mode with RTIWAI bit set.
1 PCE	**COP Enable During Pseudo Stop Bit** — PCE enables the COP during Pseudo Stop Mode. Write anytime. 0 COP stops running during Pseudo Stop Mode 1 COP continues running during Pseudo Stop Mode **Note:** If the PCE bit is cleared the COP dividers will go static while Pseudo Stop Mode is active. The COP dividers will not initialize like in Wait Mode with COPWAI bit set.
0 SCME	**Self Clock Mode Enable Bit** Normal modes: Write once Special modes: Write anytime SCME can not be cleared while operating in Self Clock Mode (SCM = 1). 0 Detection of crystal clock failure causes clock monitor reset (see Section 2.5.1.1, “Clock Monitor Reset”). 1 Detection of crystal clock failure forces the MCU in Self Clock Mode (see Section 2.4.2.2, “Self Clock Mode”).

Table 2-8. FM Amplitude selection

FM1	FM0	FM Amplitude / f_{VCO} Variation
0	0	FM off
0	1	±1%
1	0	±2%
1	1	±4%

2.3.2.8 S12XECRG RTI Control Register (RTICTL)

This register selects the timeout period for the Real Time Interrupt.

Module Base + 0x0007

	7	6	5	4	3	2	1	0
R W	RTDEC	RTR6	RTR5	RTR4	RTR3	RTR2	RTR1	RTR0
Reset	0	0	0	0	0	0	0	0

Figure 2-10. S12XECRG RTI Control Register (RTICTL)

Read: Anytime

Write: Anytime

NOTE

A write to this register initializes the RTI counter.

Table 2-9. RTICTL Field Descriptions

Field	Description
7 RTDEC	**Decimal or Binary Divider Select Bit** — RTDEC selects decimal or binary based prescaler values. 0 Binary based divider value. See Table 2-10 1 Decimal based divider value. See Table 2-11
6–4 RTR[6:4]	**Real Time Interrupt Prescale Rate Select Bits** — These bits select the prescale rate for the RTI. See Table 2-10 and Table 2-11.
3–0 RTR[3:0]	**Real Time Interrupt Modulus Counter Select Bits** — These bits select the modulus counter target value to provide additional granularity.Table 2-10 and Table 2-11 show all possible divide values selectable by the RTICTL register. The source clock for the RTI is OSCCLK.

Table 2-10. RTI Frequency Divide Rates for RTDEC = 0

RTR[3:0]	RTR[6:4] =							
	000 (OFF)	001 (2^{10})	010 (2^{11})	011 (2^{12})	100 (2^{13})	101 (2^{14})	110 (2^{15})	111 (2^{16})
0000 (÷1)	OFF(1)	2^{10}	2^{11}	2^{12}	2^{13}	2^{14}	2^{15}	2^{16}

Table 2-10. RTI Frequency Divide Rates for RTDEC = 0

RTR[3:0]	RTR[6:4] =							
	000 (OFF)	001 (2^{10})	010 (2^{11})	011 (2^{12})	100 (2^{13})	101 (2^{14})	110 (2^{15})	111 (2^{16})
0001 (÷2)	OFF	$2x2^{10}$	$2x2^{11}$	$2x2^{12}$	$2x2^{13}$	$2x2^{14}$	$2x2^{15}$	$2x2^{16}$
0010 (÷3)	OFF	$3x2^{10}$	$3x2^{11}$	$3x2^{12}$	$3x2^{13}$	$3x2^{14}$	$3x2^{15}$	$3x2^{16}$
0011 (÷4)	OFF	$4x2^{10}$	$4x2^{11}$	$4x2^{12}$	$4x2^{13}$	$4x2^{14}$	$4x2^{15}$	$4x2^{16}$
0100 (÷5)	OFF	$5x2^{10}$	$5x2^{11}$	$5x2^{12}$	$5x2^{13}$	$5x2^{14}$	$5x2^{15}$	$5x2^{16}$
0101 (÷6)	OFF	$6x2^{10}$	$6x2^{11}$	$6x2^{12}$	$6x2^{13}$	$6x2^{14}$	$6x2^{15}$	$6x2^{16}$
0110 (÷7)	OFF	$7x2^{10}$	$7x2^{11}$	$7x2^{12}$	$7x2^{13}$	$7x2^{14}$	$7x2^{15}$	$7x2^{16}$
0111 (÷8)	OFF	$8x2^{10}$	$8x2^{11}$	$8x2^{12}$	$8x2^{13}$	$8x2^{14}$	$8x2^{15}$	$8x2^{16}$
1000 (÷9)	OFF	$9x2^{10}$	$9x2^{11}$	$9x2^{12}$	$9x2^{13}$	$9x2^{14}$	$9x2^{15}$	$9x2^{16}$
1001 (÷10)	OFF	$10x2^{10}$	$10x2^{11}$	$10x2^{12}$	$10x2^{13}$	$10x2^{14}$	$10x2^{15}$	$10x2^{16}$
1010 (÷11)	OFF	$11x2^{10}$	$11x2^{11}$	$11x2^{12}$	$11x2^{13}$	$11x2^{14}$	$11x2^{15}$	$11x2^{16}$
1011 (÷12)	OFF	$12x2^{10}$	$12x2^{11}$	$12x2^{12}$	$12x2^{13}$	$12x2^{14}$	$12x2^{15}$	$12x2^{16}$
1100 (÷13)	OFF	$13x2^{10}$	$13x2^{11}$	$13x2^{12}$	$13x2^{13}$	$13x2^{14}$	$13x2^{15}$	$13x2^{16}$
1101 (÷14)	OFF	$14x2^{10}$	$14x2^{11}$	$14x2^{12}$	$14x2^{13}$	$14x2^{14}$	$14x2^{15}$	$14x2^{16}$
1110 (÷15)	OFF	$15x2^{10}$	$15x2^{11}$	$15x2^{12}$	$15x2^{13}$	$15x2^{14}$	$15x2^{15}$	$15x2^{16}$
1111 (÷16)	OFF	$16x2^{10}$	$16x2^{11}$	$16x2^{12}$	$16x2^{13}$	$16x2^{14}$	$16x2^{15}$	$16x2^{16}$

1. Denotes the default value out of reset.This value should be used to disable the RTI to ensure future backwards compatibility.

Table 2-11. RTI Frequency Divide Rates for RTDEC=1

RTR[3:0]	RTR[6:4] =							
	000 ($1x10^3$)	001 ($2x10^3$)	010 ($5x10^3$)	011 ($10x10^3$)	100 ($20x10^3$)	101 ($50x10^3$)	110 ($100x10^3$)	111 ($200x10^3$)
0000 (÷1)	$1x10^3$	$2x10^3$	$5x10^3$	$10x10^3$	$20x10^3$	$50x10^3$	$100x10^3$	$200x10^3$
0001 (÷2)	$2x10^3$	$4x10^3$	$10x10^3$	$20x10^3$	$40x10^3$	$100x10^3$	$200x10^3$	$400x10^3$
0010 (÷3)	$3x10^3$	$6x10^3$	$15x10^3$	$30x10^3$	$60x10^3$	$150x10^3$	$300x10^3$	$600x10^3$
0011 (÷4)	$4x10^3$	$8x10^3$	$20x10^3$	$40x10^3$	$80x10^3$	$200x10^3$	$400x10^3$	$800x10^3$
0100 (÷5)	$5x10^3$	$10x10^3$	$25x10^3$	$50x10^3$	$100x10^3$	$250x10^3$	$500x10^3$	$1x10^6$
0101 (÷6)	$6x10^3$	$12x10^3$	$30x10^3$	$60x10^3$	$120x10^3$	$300x10^3$	$600x10^3$	$1.2x10^6$

Table 2-11. RTI Frequency Divide Rates for RTDEC=1

RTR[3:0]	RTR[6:4] =							
	000 ($1x10^3$)	001 ($2x10^3$)	010 ($5x10^3$)	011 ($10x10^3$)	100 ($20x10^3$)	101 ($50x10^3$)	110 ($100x10^3$)	111 ($200x10^3$)
0110 (÷7)	$7x10^3$	$14x10^3$	$35x10^3$	$70x10^3$	$140x10^3$	$350x10^3$	$700x10^3$	$1.4x10^6$
0111 (÷8)	$8x10^3$	$16x10^3$	$40x10^3$	$80x10^3$	$160x10^3$	$400x10^3$	$800x10^3$	$1.6x10^6$
1000 (÷9)	$9x10^3$	$18x10^3$	$45x10^3$	$90x10^3$	$180x10^3$	$450x10^3$	$900x10^3$	$1.8x10^6$
1001 (÷10)	$10 x10^3$	$20x10^3$	$50x10^3$	$100x10^3$	$200x10^3$	$500x10^3$	$1x10^6$	$2x10^6$
1010 (÷11)	$11 x10^3$	$22x10^3$	$55x10^3$	$110x10^3$	$220x10^3$	$550x10^3$	$1.1x10^6$	$2.2x10^6$
1011 (÷12)	$12x10^3$	$24x10^3$	$60x10^3$	$120x10^3$	$240x10^3$	$600x10^3$	$1.2x10^6$	$2.4x10^6$
1100 (÷13)	$13x10^3$	$26x10^3$	$65x10^3$	$130x10^3$	$260x10^3$	$650x10^3$	$1.3x10^6$	$2.6x10^6$
1101 (÷14)	$14x10^3$	$28x10^3$	$70x10^3$	$140x10^3$	$280x10^3$	$700x10^3$	$1.4x10^6$	$2.8x10^6$
1110 (÷15)	$15x10^3$	$30x10^3$	$75x10^3$	$150x10^3$	$300x10^3$	$750x10^3$	$1.5x10^6$	$3x10^6$
1111 (÷16)	$16x10^3$	$32x10^3$	$80x10^3$	$160x10^3$	$320x10^3$	$800x10^3$	$1.6x10^6$	$3.2x10^6$

2.3.2.9 S12XECRG COP Control Register (COPCTL)

This register controls the COP (Computer Operating Properly) watchdog.

Module Base + 0x0008

	7	6	5	4	3	2	1	0
R	WCOP	RSBCK	0	0	0	CR2	CR1	CR0
W			WRTMASK					
Reset[1]	0	0	0	0	0	0	0	0

1. Refer to Device User Guide (Section: S12XECRG) for reset values of WCOP, CR2, CR1 and CR0.

= Unimplemented or Reserved

Figure 2-11. S12XECRG COP Control Register (COPCTL)

Read: Anytime

Write:

1. RSBCK: anytime in special modes; write to “1” but not to “0” in all other modes
2. WCOP, CR2, CR1, CR0:
   - — Anytime in special modes
   - — Write once in all other modes
     - – Writing CR[2:0] to “000” has no effect, but counts for the “write once” condition.
     - – Writing WCOP to “0” has no effect, but counts for the “write once” condition.

The COP time-out period is restarted if one these two conditions is true:

1. Writing a non zero value to CR[2:0] (anytime in special modes, once in all other modes) with WRTMASK = 0.

   or

2. Changing RSBCK bit from “0” to “1”.

Table 2-12. COPCTL Field Descriptions

Field	Description
7 WCOP	**Window COP Mode Bit** — When set, a write to the ARMCOP register must occur in the last 25% of the selected period. A write during the first 75% of the selected period will reset the part. As long as all writes occur during this window, $55 can be written as often as desired. Once $AA is written after the $55, the time-out logic restarts and the user must wait until the next window before writing to ARMCOP. Table 2-13 shows the duration of this window for the seven available COP rates. 0 Normal COP operation 1 Window COP operation
6 RSBCK	**COP and RTI Stop in Active BDM Mode Bit** 0 Allows the COP and RTI to keep running in Active BDM mode. 1 Stops the COP and RTI counters whenever the part is in Active BDM mode.
5 WRTMASK	**Write Mask for WCOP and CR[2:0] Bit** — This write-only bit serves as a mask for the WCOP and CR[2:0] bits while writing the COPCTL register. It is intended for BDM writing the RSBCK without touching the contents of WCOP and CR[2:0]. 0 Write of WCOP and CR[2:0] has an effect with this write of COPCTL 1 Write of WCOP and CR[2:0] has no effect with this write of COPCTL. (Does not count for “write once”.)
2–0 CR[2:0]	**COP Watchdog Timer Rate Select** — These bits select the COP time-out rate (see Table 2-13). Writing a nonzero value to CR[2:0] enables the COP counter and starts the time-out period. A COP counter time-out causes a system reset. This can be avoided by periodically (before time-out) reinitialize the COP counter via the ARMCOP register. While all of the following four conditions are true the CR[2:0], WCOP bits are ignored and the COP operates at highest time-out period (2^{24} cycles) in normal COP mode (Window COP mode disabled): 1) COP is enabled (CR[2:0] is not 000) 2) BDM mode active 3) RSBCK = 0 4) Operation in emulation or special modes

Table 2-13. COP Watchdog Rates(1)

CR2	CR1	CR0	OSCCLK Cycles to Timeout
0	0	0	COP disabled
0	0	1	2^{14}
0	1	0	2^{16}
0	1	1	2^{18}
1	0	0	2^{20}
1	0	1	2^{22}
1	1	0	2^{23}

Table 2-13. COP Watchdog Rates(1)

CR2	CR1	CR0	OSCCLK Cycles to Timeout
1	1	1	2^{24}

1. OSCCLK cycles are referenced from the previous COP time-out reset (writing $55/$AA to the ARMCOP register)

2.3.2.10 Reserved Register (FORBYP)

NOTE

This reserved register is designed for factory test purposes only, and is not intended for general user access. Writing to this register when in special modes can alter the S12XECRG's functionality.

Module Base + 0x0009

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 2-12. Reserved Register (FORBYP)

Read: Always read $00 except in special modes

Write: Only in special modes

2.3.2.11 Reserved Register (CTCTL)

NOTE

This reserved register is designed for factory test purposes only, and is not intended for general user access. Writing to this register when in special test modes can alter the S12XECRG's functionality.

Module Base + 0x000A

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 2-13. Reserved Register (CTCTL)

Read: Always read $00 except in special modes

Write: Only in special modes

2.3.2.12 S12XECRG COP Timer Arm/Reset Register (ARMCOP)

This register is used to restart the COP time-out period.

Module Base + 0x000B

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset	0	0	0	0	0	0	0	0

Figure 2-14. S12XECRG ARMCOP Register Diagram

Read: Always reads $00

Write: Anytime

When the COP is disabled (CR[2:0] = “000”) writing to this register has no effect.

When the COP is enabled by setting CR[2:0] nonzero, the following applies:

Writing any value other than $55 or $AA causes a COP reset. To restart the COP time-out period you must write $55 followed by a write of $AA. Other instructions may be executed between these writes but the sequence ($55, $AA) must be completed prior to COP end of time-out period to avoid a COP reset. Sequences of $55 writes or sequences of $AA writes are allowed. When the WCOP bit is set, $55 and $AA writes must be done in the last 25% of the selected time-out period; writing any value in the first 75% of the selected period will cause a COP reset.

2.4 Functional Description

2.4.1 Functional Blocks

2.4.1.1 Phase Locked Loop with Internal Filter (IPLL)

The IPLL is used to run the MCU from a different time base than the incoming OSCCLK. Figure 2-15 shows a block diagram of the IPLL.

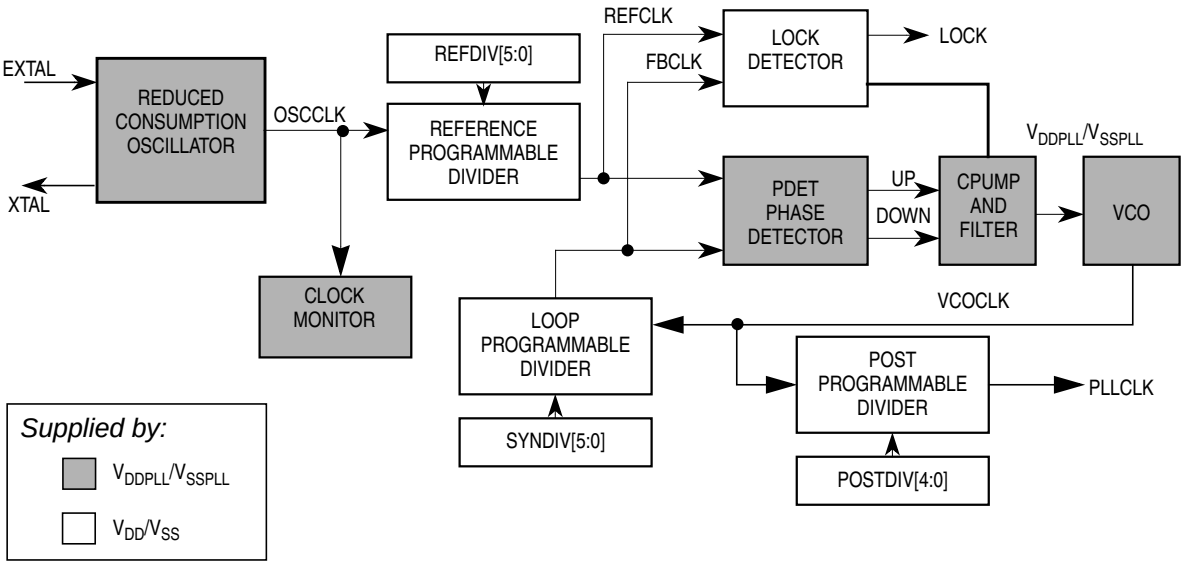


Figure 2-15. IPLL Functional Diagram

For increased flexibility, OSCCLK can be divided in a range of 1 to 64 to generate the reference frequency REFCLK using the REFDIV[5:0] bits. This offers a finer multiplication granularity. Based on the SYNDIV[5:0] bits the IPLL generates the VCOCLK by multiplying the reference clock by a multiple of 2, 4, 6,... 126, 128. Based on the POSTDIV[4:0] bits the VCOCLK can be divided in a range of 1,2,4,6,8,... to 62 to generate the PLLCLK.

$$f_{PLL} = 2 \times f_{OSC} \times \frac{SYNDIV + 1}{[REFDIV + 1][2 \times POSTDIV]}$$

NOTE

Although it is possible to set the dividers to command a very high clock frequency, do not exceed the specified bus frequency limit for the MCU.

If (PLLSEL = 1) then $f_{BUS} = f_{PLL} / 2$.

IF POSTDIV = $00 the f_{PLL} is identical to f_{VCO} (divide by one)

Several examples of IPLL divider settings are shown in Table 2-14. Shaded rows indicated that these settings are not recommended. The following rules help to achieve optimum stability and shortest lock time:

- Use lowest possible f_{VCO} / f_{REF} ratio (SYNDIV value).
- Use highest possible REFCLK frequency f_{REF}.

Table 2-14. Examples of IPLL Divider Settings

f_{OSC}	REFDIV[5:0]	f_{REF}	REFFRQ[1:0]	SYNDIV[5:0]	f_{VCO}	VCOFRQ[1:0]	POSTDIV[4:0]	f_{PLL}	f_{BUS}
4MHz	$01	2MHz	01	$18	100MHz	11	$00	100MHz	50 MHz
8MHz	$03	2MHz	01	$18	100MHz	11	$00	100MHz	50 MHz
4MHz	$00	4MHz	01	$09	80MHz	01	$00	80MHz	40MHz
8MHz	$00	8MHz	10	$04	80MHz	01	$00	80MHz	40MHz
4MHz	$00	4MHz	01	$03	32MHz	00	$01	16MHz	8MHz
4MHz	$01	2MHz	01	$18	100MHz	11	$01	50MHz	25MHz
4MHz	$03	1MHz	00	$18	50MHz	01	$00	50MHz	25MHz
4MHz	$03	1MHz	00	$31	100MHz	11	$01	50MHz	25MHz

2.4.1.1.1 IPLL Operation

The oscillator output clock signal (OSCCLK) is fed through the reference programmable divider and is divided in a range of 1 to 64 (REFDIV+1) to output the REFCLK. The VCO output clock, (VCOCLK) is fed back through the programmable loop divider and is divided in a range of 2 to 128 in increments of [2 x (SYNDIV +1)] to output the FBCLK. The VCOCLK is fed to the final programmable divider and is divided in a range of 1,2,4,6,8,... to 62 (2*POSTDIV) to output the PLLCLK. See Figure 2-15.

The phase detector then compares the FBCLK, with the REFCLK. Correction pulses are generated based on the phase difference between the two signals. The loop filter then slightly alters the DC voltage on the internal filter capacitor, based on the width and direction of the correction pulse.

The user must select the range of the REFCLK frequency and the range of the VCOCLK frequency to ensure that the correct IPLL loop bandwidth is set.

The lock detector compares the frequencies of the FBCLK, and the REFCLK. Therefore, the speed of the lock detector is directly proportional to the reference clock frequency. The circuit determines the lock condition based on this comparison.

If IPLL LOCK interrupt requests are enabled, the software can wait for an interrupt request and then check the LOCK bit. If interrupt requests are disabled, software can poll the LOCK bit continuously (during IPLL start-up, usually) or at periodic intervals. In either case, only when the LOCK bit is set, the PLLCLK can be selected as the source for the system and core clocks. If the IPLL is selected as the source for the system and core clocks and the LOCK bit is clear, the IPLL has suffered a severe noise hit and the software must take appropriate action, depending on the application.

- The LOCK bit is a read-only indicator of the locked state of the IPLL.
- The LOCK bit is set when the VCO frequency is within a certain tolerance, Δ_{Lock}, and is cleared when the VCO frequency is out of a certain tolerance, Δ_{unl}.
- Interrupt requests can occur if enabled (LOCKIE = 1) when the lock condition changes, toggling the LOCK bit.

2.4.1.2 System Clocks Generator

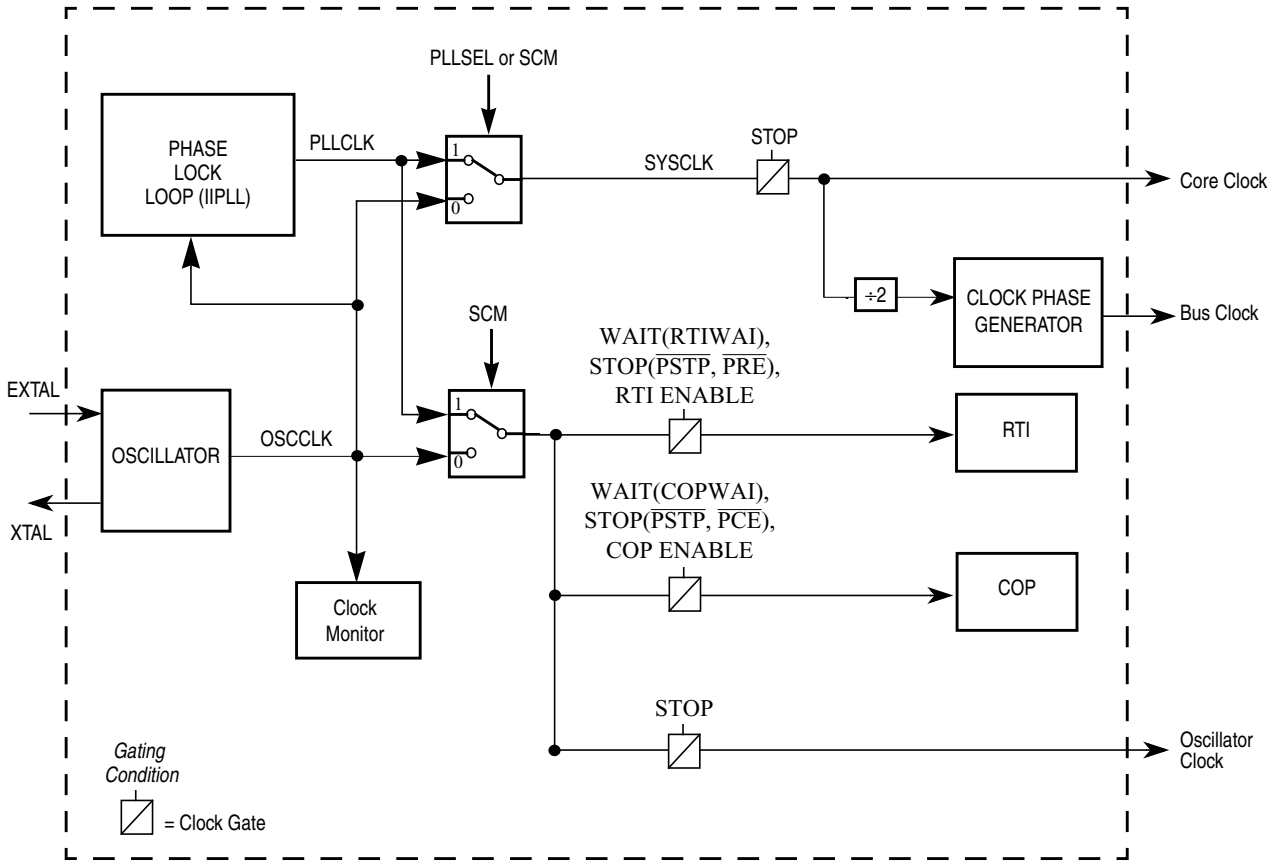


Figure 2-16. System Clocks Generator

The clock generator creates the clocks used in the MCU (see Figure 2-16). The gating condition placed on top of the individual clock gates indicates the dependencies of different modes (STOP, WAIT) and the setting of the respective configuration bits.

The peripheral modules use the Bus Clock. Some peripheral modules also use the Oscillator Clock. If the MCU enters Self Clock Mode (see Section 2.4.2.2, “Self Clock Mode”) Oscillator clock source is switched to PLLCLK running at its minimum frequency f_{SCM}. The Bus Clock is used to generate the clock visible at the ECLK pin. The Core Clock signal is the clock for the CPU. The Core Clock is twice the Bus Clock. But note that a CPU cycle corresponds to one Bus Clock.

IPLL clock mode is selected with PLLSEL bit in the CLKSEL register. When selected, the IPLL output clock drives SYSCLK for the main system including the CPU and peripherals. The IPLL cannot be turned off by clearing the PLLON bit, if the IPLL clock is selected. When PLLSEL is changed, it takes a maximum of 4 OSCCLK plus 4 PLLCLK cycles to make the transition. During the transition, all clocks freeze and CPU activity ceases.

2.4.1.3 Clock Monitor (CM)

If no OSCCLK edges are detected within a certain time, the clock monitor within the oscillator block generates a clock monitor fail event. The S12XECRG then asserts self clock mode or generates a system reset depending on the state of SCME bit. If the clock monitor is disabled or the presence of clocks is detected no failure is indicated by the oscillator block.The clock monitor function is enabled/disabled by the CME control bit.

2.4.1.4 Clock Quality Checker

The clock monitor performs a coarse check on the incoming clock signal. The clock quality checker provides a more accurate check in addition to the clock monitor.

A clock quality check is triggered by any of the following events:

- Power on reset (*POR*)
- Low voltage reset (*LVR*)
- Wake-up from Full Stop Mode (*exit full stop*)
- Clock Monitor fail indication (*CM fail*)

A time window of 50000 PLLCLK cycles[1] is called *check window*.

A number greater equal than 4096 rising OSCCLK edges within a *check window* is called *osc ok*. Note that *osc ok* immediately terminates the current *check window*. See Figure 2-17 as an example.

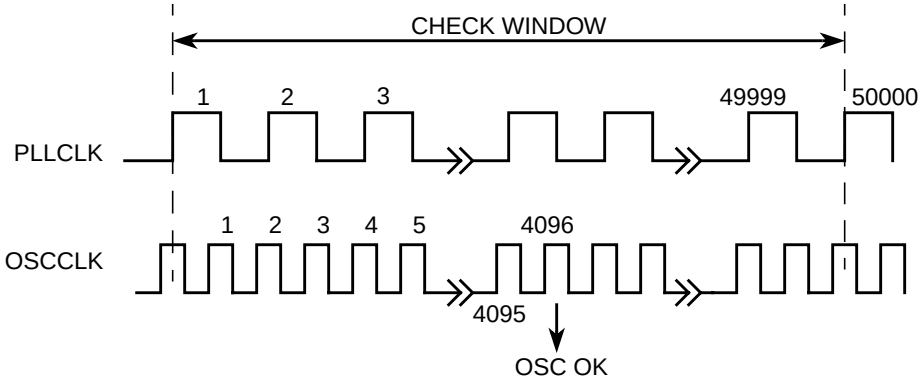


Figure 2-17. Check Window Example

1. IPLL is running at self clock mode frequency f_{SCM}.

The Sequence for clock quality check is shown in Figure 2-18.

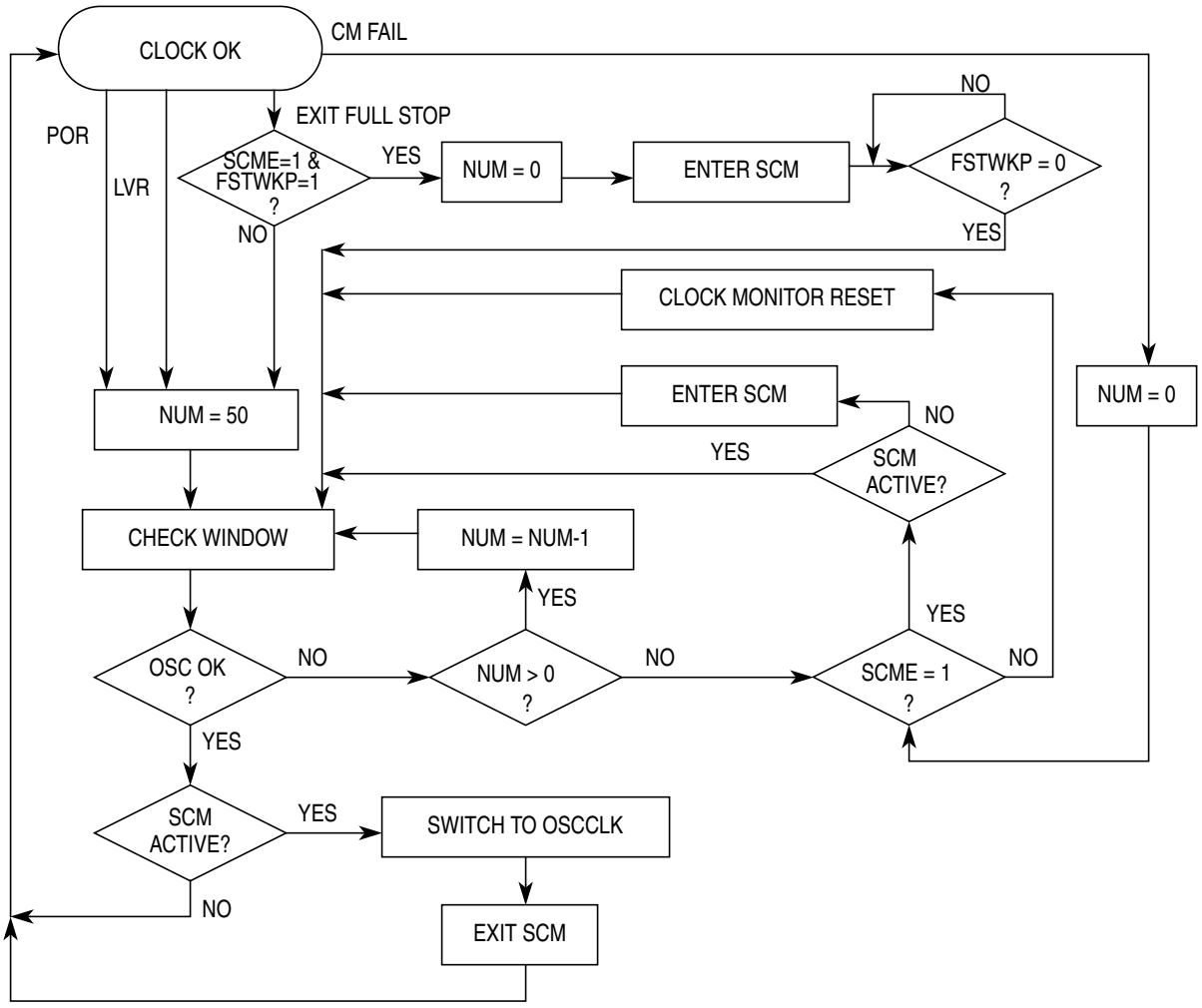


Figure 2-18. Sequence for Clock Quality Check

NOTE

Remember that in parallel to additional actions caused by Self Clock Mode or Clock Monitor Reset[1] handling the clock quality checker **continues** to check the OSCCLK signal.

NOTE

The Clock Quality Checker enables the IPLL and the voltage regulator (VREG) anytime a clock check has to be performed. An ongoing clock quality check could also cause a running IPLL (f_{SCM}) and an active VREG during Pseudo Stop Mode.

1. A Clock Monitor Reset will always set the SCME bit to logical'1'.

2.4.1.5 Computer Operating Properly Watchdog (COP)

The COP (free running watchdog timer) enables the user to check that a program is running and sequencing properly. When the COP is being used, software is responsible for keeping the COP from timing out. If the COP times out it is an indication that the software is no longer being executed in the intended sequence; thus a system reset is initiated (see Section 2.4.1.5, “Computer Operating Properly Watchdog (COP)”). The COP runs with a gated OSCCLK. Three control bits in the COPCTL register allow selection of seven COP time-out periods.

When COP is enabled, the program must write $55 and $AA (in this order) to the ARMCOP register during the selected time-out period. Once this is done, the COP time-out period is restarted. If the program fails to do this and the COP times out, the part will reset. Also, if any value other than $55 or $AA is written, the part is immediately reset.

Windowed COP operation is enabled by setting WCOP in the COPCTL register. In this mode, writes to the ARMCOP register to clear the COP timer must occur in the last 25% of the selected time-out period. A premature write will immediately reset the part.

If PCE bit is set, the COP will continue to run in Pseudo Stop Mode.

2.4.1.6 Real Time Interrupt (RTI)

The RTI can be used to generate a hardware interrupt at a fixed periodic rate. If enabled (by setting RTIE=1), this interrupt will occur at the rate selected by the RTICTL register. The RTI runs with a gated OSCCLK. At the end of the RTI time-out period the RTIF flag is set to one and a new RTI time-out period starts immediately.

A write to the RTICTL register restarts the RTI time-out period.

If the PRE bit is set, the RTI will continue to run in Pseudo Stop Mode.

2.4.2 Operation Modes

2.4.2.1 Normal Mode

The S12XECRG block behaves as described within this specification in all normal modes.

2.4.2.2 Self Clock Mode

If the external clock frequency is not available due to a failure or due to long crystal start-up time, the Bus Clock and the Core Clock are derived from the PLLCLK running at self clock mode frequency f_{SCM}; this mode of operation is called Self Clock Mode. This requires CME = 1 and SCME = 1, which is the default after reset. If the MCU was clocked by the PLLCLK prior to entering Self Clock Mode, the PLLSEL bit will be cleared. If the external clock signal has stabilized again, the S12XECRG will automatically select OSCCLK to be the system clock and return to normal mode. See Section 2.4.1.4, “Clock Quality Checker” for more information on entering and leaving Self Clock Mode.

NOTE

In order to detect a potential clock loss the CME bit should always be enabled (CME = 1).

If CME bit is disabled and the MCU is configured to run on PLLCLK, a loss of external clock (OSCCLK) will not be detected and will cause the system clock to drift towards lower frequencies. As soon as the external clock is available again the system clock ramps up to its IPLL target frequency. If the MCU is running on external clock any loss of clock will cause the system to go static.

2.4.3 Low Power Options

This section summarizes the low power options available in the S12XECRG.

2.4.3.1 Run Mode

This is the default mode after reset.

The RTI can be stopped by setting the associated rate select bits to zero.

The COP can be stopped by setting the associated rate select bits to zero.

2.4.3.2 Wait Mode

The WAI instruction puts the MCU in a low power consumption stand-by mode depending on setting of the individual bits in the CLKSEL register. All individual Wait Mode configuration bits can be superposed. This provides enhanced granularity in reducing the level of power consumption during Wait Mode. Table 2-15 lists the individual configuration bits and the parts of the MCU that are affected in Wait Mode.

Table 2-15. MCU Configuration During Wait Mode

	PLLWAI	RTIWAI	COPWAI
IPLL	Stopped	—	—
RTI	—	Stopped	—
COP	—	—	Stopped

After executing the WAI instruction the core requests the S12XECRG to switch MCU into Wait Mode. The S12XECRG then checks whether the PLLWAI bit is asserted. Depending on the configuration the S12XECRG switches the system and core clocks to OSCCLK by clearing the PLLSEL bit and disables the IPLL.

There are two ways to restart the MCU from Wait Mode:

1. Any reset
2. Any interrupt

2.4.3.3 Stop Mode

All clocks are stopped in STOP mode, dependent of the setting of the PCE, PRE and PSTP bit. The oscillator is disabled in STOP mode unless the PSTP bit is set. If the PRE or PCE bits are set, the RTI or COP continues to run in Pseudo Stop Mode. In addition to disabling system and core clocks the S12XECRG requests other functional units of the MCU (e.g. voltage-regulator) to enter their individual power saving modes (if available).

If the PLLSEL bit is still set when entering Stop Mode, the S12XECRG will switch the system and core clocks to OSCCLK by clearing the PLLSEL bit. Then the S12XECRG disables the IPLL, disables the core clock and finally disables the remaining system clocks.

If Pseudo Stop Mode is entered from Self-Clock Mode the S12XECRG will continue to check the clock quality until clock check is successful. In this case the IPLL and the voltage regulator (VREG) will remain enabled. If Full Stop Mode (PSTP = 0) is entered from Self-Clock Mode the ongoing clock quality check will be stopped. A complete timeout window check will be started when Stop Mode is left again.

There are two ways to restart the MCU from Stop Mode:

1. Any reset
2. Any interrupt

If the MCU is woken-up from Full Stop Mode by an interrupt and the fast wake-up feature is enabled (FSTWKP=1 and SCME=1), the system will immediately (no clock quality check) resume operation in Self-Clock Mode (see Section 2.4.1.4, “Clock Quality Checker”). The SCMIF flag will not be set for this special case. The system will remain in Self-Clock Mode with oscillator disabled until FSTWKP bit is cleared. The clearing of FSTWKP will start the oscillator and the clock quality check. If the clock quality check is successful, the S12XECRG will switch all system clocks to oscillator clock. The SCMIF flag will be set. See application examples in Figure 2-19 and Figure 2-20.

Because the IPLL has been powered-down during Stop Mode the PLLSEL bit is cleared and the MCU runs on OSCCLK after leaving Stop-Mode. The software must manually set the PLLSEL bit again, in order to switch system and core clocks to the PLLCLK.

NOTE

In Full Stop Mode or Self-Clock Mode caused by the fast wake-up feature the clock monitor and the oscillator are disabled.

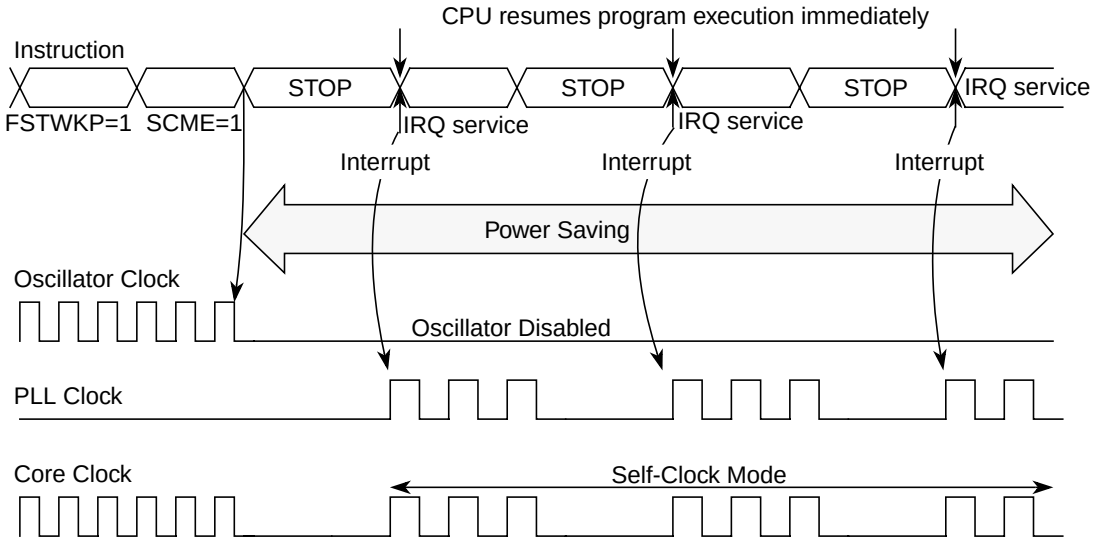


Figure 2-19. Fast Wake-up from Full Stop Mode: Example 1

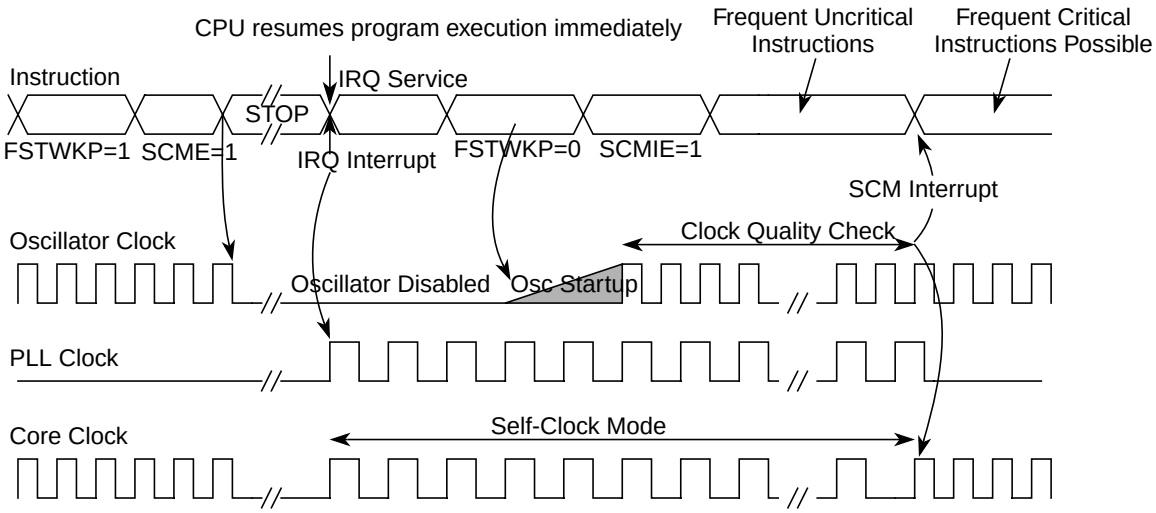


Figure 2-20. Fast Wake-up from Full Stop Mode: Example 2

2.5 Resets

All reset sources are listed in Table 2-16. Refer to MCU specification for related vector addresses and priorities.

Table 2-16. Reset Summary

Reset Source	Local Enable
Power on Reset	None
Low Voltage Reset	None
External Reset	None
Illegal Address Reset	None
Clock Monitor Reset	PLLCTL (CME=1, SCME=0)

Table 2-16. Reset Summary

Reset Source	Local Enable
COP Watchdog Reset	COPCTL (CR[2:0] nonzero)

2.5.1 Description of Reset Operation

The reset sequence is initiated by any of the following events:

- Low level is detected at the $\overline{\text{RESET}}$ pin (External Reset).
- Power on is detected.
- Low voltage is detected.
- Illegal Address Reset is detected (see S12XMMC Block Guide for details).
- COP watchdog times out.
- Clock monitor failure is detected and Self-Clock Mode was disabled (SCME=0).

Upon detection of any reset event, an internal circuit drives the $\overline{\text{RESET}}$ pin low for 128 SYSCLK cycles (see Figure 2-21). Since entry into reset is asynchronous it does not require a running SYSCLK. However, the internal reset circuit of the S12XECRG cannot sequence out of current reset condition without a running SYSCLK. The number of 128 SYSCLK cycles might be increased by n = 3 to 6 additional SYSCLK cycles depending on the internal synchronization latency. After 128+n SYSCLK cycles the $\overline{\text{RESET}}$ pin is released. The reset generator of the S12XECRG waits for additional 64 SYSCLK cycles and then samples the RESET pin to determine the originating source. Table 2-17 shows which vector will be fetched.

Table 2-17. Reset Vector Selection

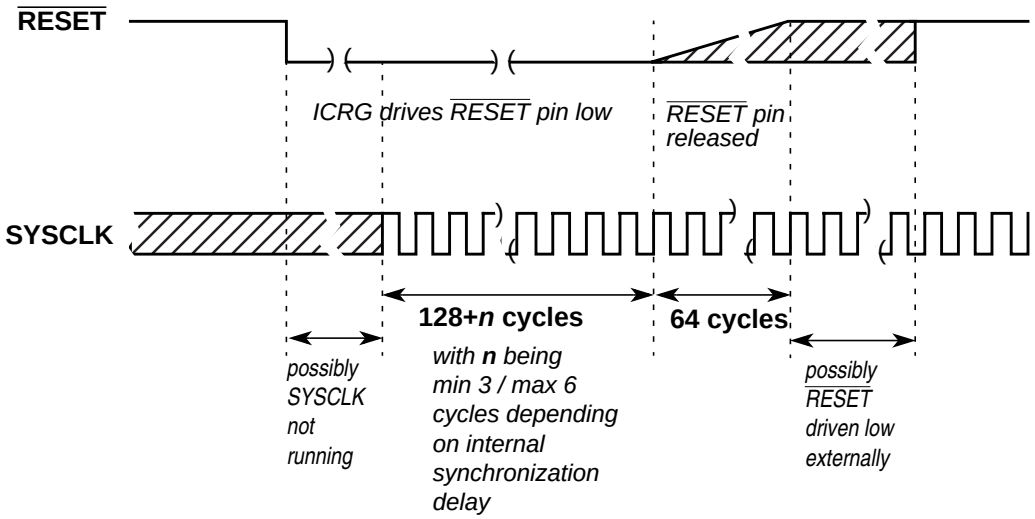
Sampled $\overline{\text{RESET}}$ Pin (64 cycles after release)	Clock Monitor Reset Pending	COP Reset Pending	Vector Fetch
1	0	0	POR / LVR / Illegal Address Reset/ External Reset
1	1	X	Clock Monitor Reset
1	0	1	COP Reset
0	X	X	POR / LVR / Illegal Address Reset/ External Reset with rise of $\overline{\text{RESET}}$ pin

NOTE

External circuitry connected to the RESET pin should be able to raise the signal to a valid logic one within 64 SYSCLK cycles after the low drive is released by the MCU. If this requirement is not adhered to the reset source will always be recognized as “External Reset” even if the reset was initially caused by an other reset source.

The internal reset of the MCU remains asserted while the reset generator completes the 192 SYSCLK long reset sequence. In case the RESET pin is externally driven low for more than these 192 SYSCLK cycles (External Reset), the internal reset remains asserted longer.

Figure 2-21. RESET Timing



2.5.1.1 Clock Monitor Reset

The S12XECRG generates a Clock Monitor Reset in case all of the following conditions are true:

- Clock monitor is enabled (CME = 1)
- Loss of clock is detected
- Self-Clock Mode is disabled (SCME = 0).

The reset event asynchronously forces the configuration registers to their default settings. In detail the CME and the SCME are reset to logical ‘1’ (which changes the state of the SCME bit. As a consequence the S12XECRG immediately enters Self Clock Mode and starts its internal reset sequence. In parallel the clock quality check starts. As soon as clock quality check indicates a valid Oscillator Clock the S12XECRG switches to OSCCLK and leaves Self Clock Mode. Since the clock quality checker is running in parallel to the reset generator, the S12XECRG may leave Self Clock Mode while still completing the internal reset sequence.

2.5.1.2 Computer Operating Properly Watchdog (COP) Reset

When COP is enabled, the S12XECRG expects sequential write of $55 and $AA (in this order) to the ARMCOP register during the selected time-out period. Once this is done, the COP time-out period restarts. If the program fails to do this the S12XECRG will generate a reset.

2.5.1.3 Power On Reset, Low Voltage Reset

The on-chip voltage regulator detects when V_{DD} to the MCU has reached a certain level and asserts power on reset or low voltage reset or both. As soon as a power on reset or low voltage reset is triggered the

S12XECRG performs a quality check on the incoming clock signal. As soon as clock quality check indicates a valid Oscillator Clock signal the reset sequence starts using the Oscillator clock. If after 50 check windows the clock quality check indicated a non-valid Oscillator Clock the reset sequence starts using Self-Clock Mode.

Figure 2-22 and Figure 2-23 show the power-up sequence for cases when the $\overline{\text{RESET}}$ pin is tied to V_{DD} and when the $\overline{\text{RESET}}$ pin is held low.

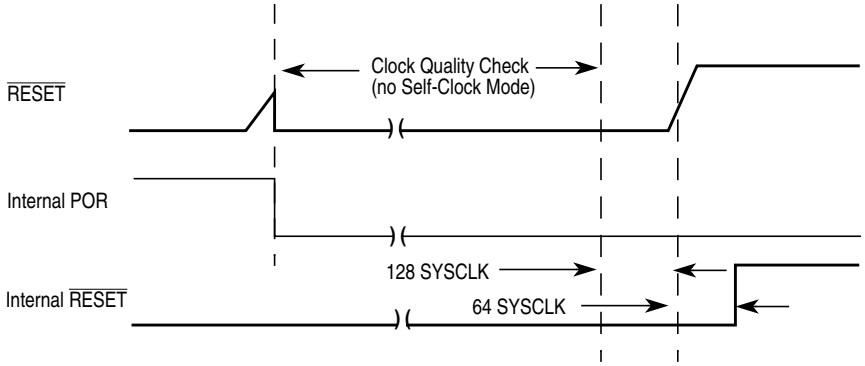


Figure 2-22. $\overline{\text{RESET}}$ Pin Tied to V_{DD} (by a Pull-up Resistor)

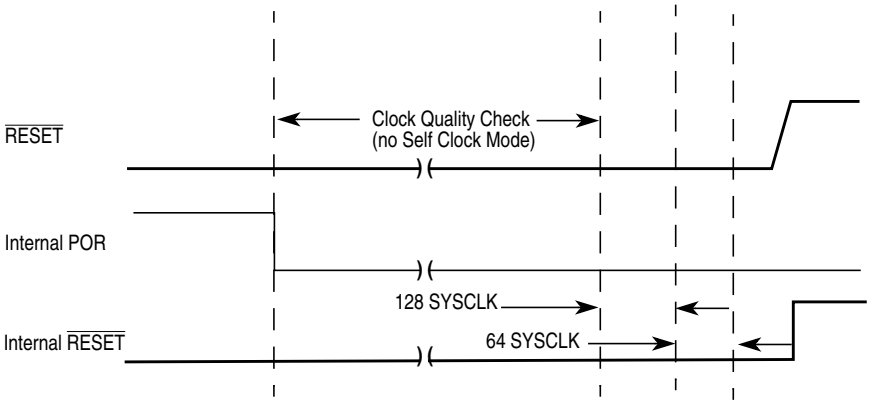


Figure 2-23. $\overline{\text{RESET}}$ Pin Held Low Externally

2.6 Interrupts

The interrupts/reset vectors requested by the S12XECRG are listed in Table 2-18. Refer to MCU specification for related vector addresses and priorities.

Table 2-18. S12XECRG Interrupt Vectors

Interrupt Source	CCR Mask	Local Enable
Real time interrupt	I bit	CRGINT (RTIE)
LOCK interrupt	I bit	CRGINT (LOCKIE)
SCM interrupt	I bit	CRGINT (SCMIE)

2.6.1 Description of Interrupt Operation

2.6.1.1 Real Time Interrupt

The S12XECRG generates a real time interrupt when the selected interrupt time period elapses. RTI interrupts are locally disabled by setting the RTIE bit to zero. The real time interrupt flag (RTIF) is set to1 when a timeout occurs, and is cleared to 0 by writing a 1 to the RTIF bit.

The RTI continues to run during Pseudo Stop Mode if the PRE bit is set to 1. This feature can be used for periodic wakeup from Pseudo Stop if the RTI interrupt is enabled.

2.6.1.2 IPLL Lock Interrupt

The S12XECRG generates a IPLL Lock interrupt when the LOCK condition of the IPLL has changed, either from a locked state to an unlocked state or vice versa. Lock interrupts are locally disabled by setting the LOCKIE bit to zero. The IPLL Lock interrupt flag (LOCKIF) is set to1 when the LOCK condition has changed, and is cleared to 0 by writing a 1 to the LOCKIF bit.

2.6.1.3 Self Clock Mode Interrupt

The S12XECRG generates a Self Clock Mode interrupt when the SCM condition of the system has changed, either entered or exited Self Clock Mode. SCM conditions are caused by a failing clock quality check after power on reset (POR) or low voltage reset (LVR) or recovery from Full Stop Mode (PSTP = 0) or Clock Monitor failure. For details on the clock quality check refer to Section 2.4.1.4, “Clock Quality Checker”. If the clock monitor is enabled (CME = 1) a loss of external clock will also cause a SCM condition (SCME = 1).

SCM interrupts are locally disabled by setting the SCMIE bit to zero. The SCM interrupt flag (SCMIF) is set to1 when the SCM condition has changed, and is cleared to 0 by writing a 1 to the SCMIF bit.

Chapter 3
Voltage Regulator (S12VREGL3V3V1)

Table 3-1. Revision History Table

Rev. No. (Item No.)	Date (Submitted By)	Sections Affected	Substantial Change(s)
V01.02	09 Sep 2005		Updates for API external access and LVR flags.
V01.03	23 Sep 2005		VAE reset value is 1.
V01.04	08 Jun 2007		Added temperature sensor to customer information

3.1 Introduction

Module VREG_3V3 is a tri output voltage regulator that provides two separate 1.84V (typical) supplies differing in the amount of current that can be sourced and a 2.82V (typical) supply. The regulator input voltage range is from 3.3V up to 5V (typical).

3.1.1 Features

Module VREG_3V3 includes these distinctive features:

- Three parallel, linear voltage regulators with bandgap reference
- Low-voltage detect (LVD) with low-voltage interrupt (LVI)
- Power-on reset (POR)
- Low-voltage reset (LVR)
- High Temperature Detect (HTD) with High Temperature Interrupt (HTI)
- Autonomous periodical interrupt (API)

3.1.2 Modes of Operation

There are three modes VREG_3V3 can operate in:

1. Full performance mode (FPM) (MCU is not in stop mode)

   The regulator is active, providing the nominal supply voltages with full current sourcing capability. Features LVD (low-voltage detect), LVR (low-voltage reset), and POR (power-on reset) and HTD (High Temperature Detect) are available. The API is available.

2. Reduced power mode (RPM) (MCU is in stop mode)

   The purpose is to reduce power consumption of the device. The output voltage may degrade to a lower value than in full performance mode, additionally the current sourcing capability is substantially reduced. Only the POR is available in this mode, LVD, LVR and HTD are disabled. The API is available.

3. Shutdown mode

   Controlled by VREGEN (see device level specification for connectivity of VREGEN).

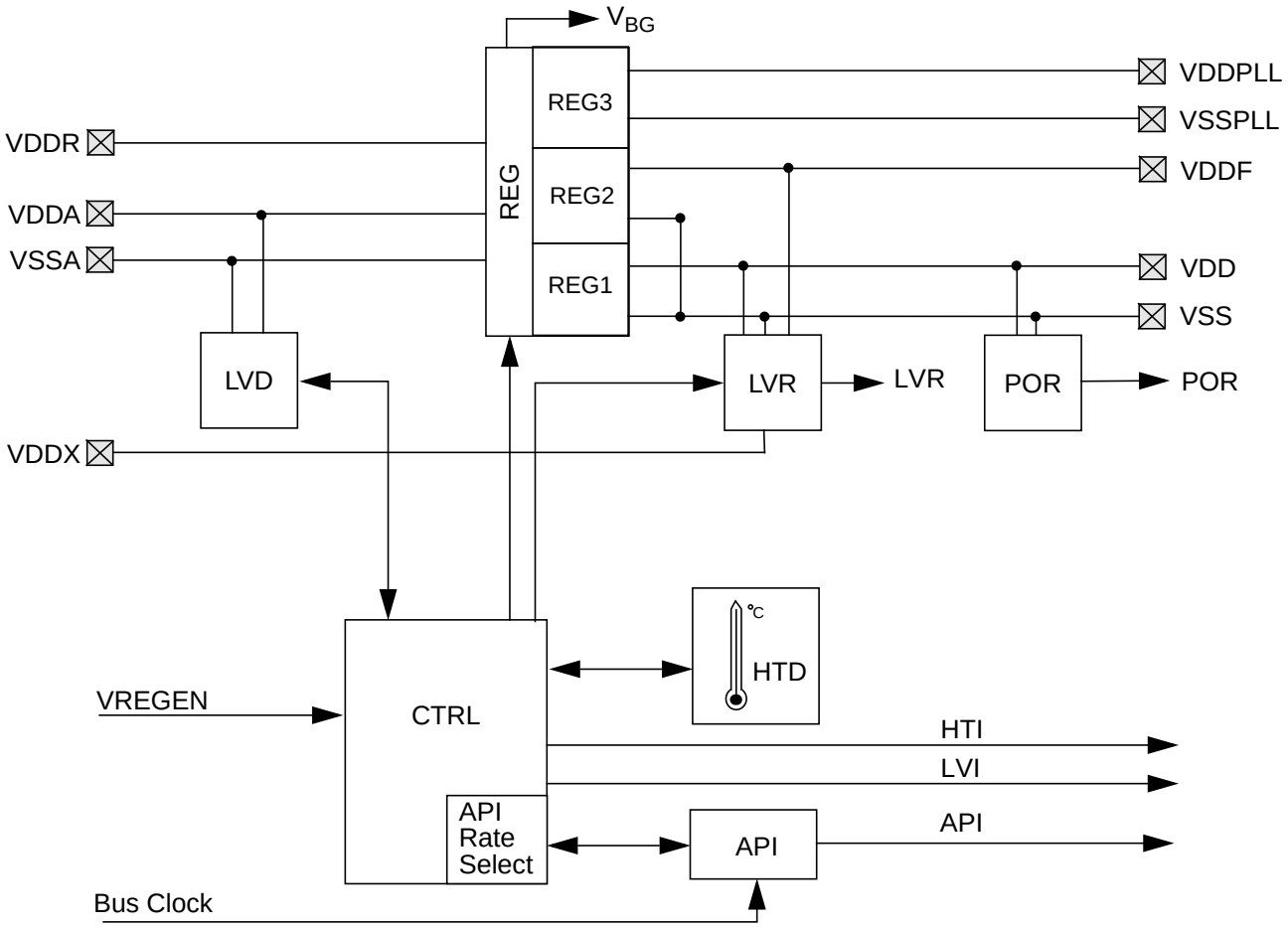
   This mode is characterized by minimum power consumption. The regulator outputs are in a high-impedance state, only the POR feature is available, LVD, LVR and HTD are disabled. The API internal RC oscillator clock is not available.

   This mode must be used to disable the chip internal regulator VREG_3V3, i.e., to bypass the VREG_3V3 to use external supplies.

3.1.3 Block Diagram

Figure 3-1 shows the function principle of VREG_3V3 by means of a block diagram. The regulator core REG consists of three parallel subblocks, REG1, REG2 and REG3, providing three independent output voltages.

Figure 3-1. VREG_3V3 Block Diagram



LVD: Low Voltage Detect
LVR: Low Voltage Reset
POR: Power-on Reset
HTD: High Temperature Detect
REG: Regulator Core
CTRL: Regulator Control
API: Auto. Periodical Interrupt
⊠ PIN

3.2 External Signal Description

Due to the nature of VREG_3V3 being a voltage regulator providing the chip internal power supply voltages, most signals are power supply signals connected to pads.

Table 3-2 shows all signals of VREG_3V3 associated with pins.

Table 3-2. Signal Properties

Name	Function	Reset State	Pull Up
VDDR	Power input (positive supply)	—	—
VDDA	Quiet input (positive supply)	—	—
VSSA	Quiet input (ground)	—	—
VDDX	Power input (positive supply)	—	—
VDD	Primary output (positive supply)	—	—
VSS	Primary output (ground)	—	—
VDDF	Secondary output (positive supply)	—	—
VDDPLL	Tertiary output (positive supply)	—	—
VSSPLL	Tertiary output (ground)	—	—
VREGEN (optional)	Optional Regulator Enable	—	—
VREG_API (optional)	VREG Autonomous Periodical Interrupt output	—	—

NOTE

Check device level specification for connectivity of the signals.

3.2.1 VDDR — Regulator Power Input Pins

Signal VDDR is the power input of VREG_3V3. All currents sourced into the regulator loads flow through this pin. A chip external decoupling capacitor (100 nF...220 nF, X7R ceramic) between VDDR and VSSR (if VSSR is not available VSS) can smooth ripple on VDDR.

For entering Shutdown Mode, pin VDDR should also be tied to ground on devices without VREGEN pin.

3.2.2 VDDA, VSSA — Regulator Reference Supply Pins

Signals VDDA/VSSA, which are supposed to be relatively quiet, are used to supply the analog parts of the regulator. Internal precision reference circuits are supplied from these signals. A chip external decoupling capacitor (100 nF...220 nF, X7R ceramic) between VDDA and VSSA can further improve the quality of this supply.

3.2.3 VDD, VSS — Regulator Output1 (Core Logic) Pins

Signals VDD/VSS are the primary outputs of VREG_3V3 that provide the power supply for the core logic. These signals are connected to device pins to allow external decoupling capacitors (220 nF, X7R ceramic).

In Shutdown Mode an external supply driving VDD/VSS can replace the voltage regulator.

3.2.4 VDDF — Regulator Output2 (NVM Logic) Pins

Signals VDDF/VSS are the secondary outputs of VREG_3V3 that provide the power supply for the NVM logic. These signals are connected to device pins to allow external decoupling capacitors (220 nF, X7R ceramic).

In Shutdown Mode an external supply driving VDDF/VSS can replace the voltage regulator.

3.2.5 VDDPLL, VSSPLL — Regulator Output3 (PLL) Pins

Signals VDDPLL/VSSPLL are the secondary outputs of VREG_3V3 that provide the power supply for the PLL and oscillator. These signals are connected to device pins to allow external decoupling capacitors (100 nF...220 nF, X7R ceramic).

In Shutdown Mode, an external supply driving VDDPLL/VSSPLL can replace the voltage regulator.

3.2.6 VDDX — Power Input Pin

Signals VDDX/VSS are monitored by VREG_3V3 with the LVR feature.

3.2.7 VREGEN — Optional Regulator Enable Pin

This optional signal is used to shutdown VREG_3V3. In that case, VDD/VSS and VDDPLL/VSSPLL must be provided externally. Shutdown mode is entered with VREGEN being low. If VREGEN is high, the VREG_3V3 is either in Full Performance Mode or in Reduced Power Mode.

For the connectivity of VREGEN, see device specification.

NOTE

Switching from FPM or RPM to shutdown of VREG_3V3 and vice versa is not supported while MCU is powered.

3.2.8 VREG_API — Optional Autonomous Periodical Interrupt Output Pin

This pin provides the signal selected via APIEA if system is set accordingly. See 3.3.2.3, “Autonomous Periodical Interrupt Control Register (VREGAPICL) and 3.4.8, “Autonomous Periodical Interrupt (API) for details.

For the connectivity of VREG_API, see device specification.

3.3 Memory Map and Register Definition

This section provides a detailed description of all registers accessible in VREG_3V3.

If enabled in the system, the VREG_3V3 will abort all read and write accesses to reserved registers within it’s memory slice. See device level specification for details.

3.3.1 Module Memory Map

A summary of the registers associated with the VREG_3V3 sub-block is shown in Table 3-3. Detailed descriptions of the registers and bits are given in the subsections that follow

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x02F0	VREGHTCL	R	0	0	VSEL	VAE	HTEN	HTDS	HTIE	HTIF
		W								
0x02F1	VREGCTRL	R	0	0	0	0	0	LVDS	LVIE	LVIF
		W								
0x02F2	VREGAPICL	R	APICLK	0	0	APIFES	APIEA	APIFE	APIE	APIF
		W								
0x02F3	VREGAPITR	R	APITR5	APITR4	APITR3	APITR2	APITR1	APITR0	0	0
		W								
0x02F4	VREGAPIRH	R	APIR15	APIR14	APIR13	APIR12	APIR11	APIR10	APIR9	APIR8
		W								
0x02F5	VREGAPIRL	R	APIR7	APIR6	APIR5	APIR4	APIR3	APIR2	APIR1	APIR0
		W								
0x02F6	Reserved 06	R	0	0	0	0	0	0	0	0
		W								
0x02F7	VREGHTTR	R	HTOEN	0	0	0	HTTR3	HTTR2	HTTR1	HTTR0
		W								

Table 3-3. Register Summary

3.3.2 Register Descriptions

This section describes all the VREG_3V3 registers and their individual bits.

3.3.2.1 HT Control Register (VREGHTCL)

0x02F0

	7	6	5	4	3	2	1	0
R	0	0	VSEL	VAE	HTEN	HTDS	HTIE	HTIF
W								
Reset	0	0	0	1	0	0	0	0

= Unimplemented or Reserved

3.3.2.2 Control Register (VREGCTRL)

Table 3-4. VREGHTCL Field Descriptions

Field	Description
7, 6 Reserved	These reserved bits are used for test purposes and writable only in special modes. They must remain clear for correct temperature sensor operation.
5 VSEL	**Voltage Access Select Bit** — If set, the bandgap reference voltage V_{BG} can be accessed internally (i.e. multiplexed to an internal Analog to Digital Converter channel). . The internal access must be enabled by bit VAE. See device level specification for connectivity. 0 An internal voltage can be accessed internally if VAE is set. 1 Bandgap reference voltage V_{BG} can be accessed internally if VAE is set.
4 VAE	**Voltage Access Enable Bit** — If set, the voltage selected by bit VSEL can be accessed internally (i.e. multiplexed to an internal Analog to Digital Converter channel). See device level specification for connectivity. 0 Voltage selected by VSEL can not be accessed internally (i.e. External analog input is connected to Analog to Digital Converter channel). 1 Voltage selected by VSEL can be accessed internally.
3 HTEN	**High Temperature Enable Bit** — If set the temperature sense is enabled. 0 The temperature sense is disabled. 1 The temperature sense is enabled.
2 HTDS	**High Temperature Detect Status Bit** — 0 Temperature T_{DIE} is below level T_{HTID} or RPM or Shutdown Mode. 1 Temperature T_{DIE} is above level T_{HTIA} and FPM.
1 HTIE	**High Temperature Interrupt Enable Bit** 0 Interrupt request is disabled. 1 Interrupt will be requested whenever HTIF is set.
0 HTIF	**High Temperature Interrupt Flag** — 0 No change in HTDS bit. 1 HTDS bit has changed. **Note:** On entering the reduced power mode the HTIF is not cleared by the VREG.

The VREGCTRL register allows the configuration of the VREG_3V3 low-voltage detect features.

0x02F1

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	LVDS	LVIE	LVIF
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 3-2. Control Register (VREGCTRL)

Table 3-5. VREGCTRL Field Descriptions

Field	Description
2 LVDS	**Low-Voltage Detect Status Bit** — This read-only status bit reflects the input voltage. Writes have no effect. 0 Input voltage V_{DDA} is above level V_{LVID} or RPM or shutdown mode. 1 Input voltage V_{DDA} is below level V_{LVIA} and FPM.
1 LVIE	**Low-Voltage Interrupt Enable Bit** 0 Interrupt request is disabled. 1 Interrupt will be requested whenever LVIF is set.
0 LVIF	**Low-Voltage Interrupt Flag** — LVIF is set to 1 when LVDS status bit changes. This flag can only be cleared by writing a 1. Writing a 0 has no effect. If enabled (LVIE = 1), LVIF causes an interrupt request. 0 No change in LVDS bit. 1 LVDS bit has changed. **Note:** On entering the Reduced Power Mode the LVIF is not cleared by the VREG_3V3.

3.3.2.3 Autonomous Periodical Interrupt Control Register (VREGAPICL)

The VREGAPICL register allows the configuration of the VREG_3V3 autonomous periodical interrupt features.

0x02F2

	7	6	5	4	3	2	1	0
R	APICLK	0	0	APIES	APIEA	APIFE	APIE	APIF
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 3-3. Autonomous Periodical Interrupt Control Register (VREGAPICL)

Table 3-6. VREGAPICL Field Descriptions

Field	Description
7 APICLK	**Autonomous Periodical Interrupt Clock Select Bit** — Selects the clock source for the API. Writable only if APIFE = 0; APICLK cannot be changed if APIFE is set by the same write operation. 0 Autonomous periodical interrupt clock used as source. 1 Bus clock used as source.
4 APIES	**Autonomous Periodical Interrupt External Select Bit** — Selects the waveform at the external pin.If set, at the external pin a clock is visible with 2 times the selected API Period (Table 3-10). If not set, at the external pin will be a high pulse at the end of every selected period with the size of half of the min period (Table 3-10). See device level specification for connectivity. 0 At the external periodic high pulses are visible, if APIEA and APIFE is set. 1 At the external pin a clock is visible, if APIEA and APIFE is set.
3 APIEA	**Autonomous Periodical Interrupt External Access Enable Bit** — If set, the waveform selected by bit APIES can be accessed externally. See device level specification for connectivity. 0 Waveform selected by APIES can not be accessed externally. 1 Waveform selected by APIES can be accessed externally, if APIFE is set.
2 APIFE	**Autonomous Periodical Interrupt Feature Enable Bit** — Enables the API feature and starts the API timer when set. 0 Autonomous periodical interrupt is disabled. 1 Autonomous periodical interrupt is enabled and timer starts running.
1 APIE	**Autonomous Periodical Interrupt Enable Bit** 0 API interrupt request is disabled. 1 API interrupt will be requested whenever APIF is set.
0 APIF	**Autonomous Periodical Interrupt Flag** — APIF is set to 1 when the in the API configured time has elapsed. This flag can only be cleared by writing a 1 to it. Clearing of the flag has precedence over setting. Writing a 0 has no effect. If enabled (APIE = 1), APIF causes an interrupt request. 0 API timeout has not yet occurred. 1 API timeout has occurred.

3.3.2.4 Autonomous Periodical Interrupt Trimming Register (VREGAPITR)

The VREGAPITR register allows to trim the API timeout period.

0x02F3

	7	6	5	4	3	2	1	0
R	APITR5	APITR4	APITR3	APITR2	APITR1	APITR0	0	0
W								
Reset	0[1]	0[1]	0[1]	0[1]	0[1]	0[1]	0	0

1. Reset value is either 0 or preset by factory. See Section 1 (Device Overview) for details.

= Unimplemented or Reserved

Figure 3-4. Autonomous Periodical Interrupt Trimming Register (VREGAPITR)

Table 3-7. VREGAPITR Field Descriptions

Field	Description
7–2 APITR[5:0]	**Autonomous Periodical Interrupt Period Trimming Bits** — See Table 3-8 for trimming effects.

Table 3-8. Trimming Effect of APIT

Bit	Trimming Effect
APITR[5]	Increases period
APITR[4]	Decreases period less than APITR[5] increased it
APITR[3]	Decreases period less than APITR[4]
APITR[2]	Decreases period less than APITR[3]
APITR[1]	Decreases period less than APITR[2]
APITR[0]	Decreases period less than APITR[1]

3.3.2.5 Autonomous Periodical Interrupt Rate High and Low Register (VREGAPIRH / VREGAPIRL)

The VREGAPIRH and VREGAPIRL register allows the configuration of the VREG_3V3 autonomous periodical interrupt rate.

0x02F4

	7	6	5	4	3	2	1	0
R W	APIR15	APIR14	APIR13	APIR12	APIR11	APIR10	APIR9	APIR8
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 3-5. Autonomous Periodical Interrupt Rate High Register (VREGAPIRH)

0x02F5

	7	6	5	4	3	2	1	0
R W	APIR7	APIR6	APIR5	APIR4	APIR3	APIR2	APIR1	APIR0
Reset	0	0	0	0	0	0	0	0

Figure 3-6. Autonomous Periodical Interrupt Rate Low Register (VREGAPIRL)

Table 3-9. VREGAPIRH / VREGAPIRL Field Descriptions

Field	Description
15-0 APIR[15:0]	**Autonomous Periodical Interrupt Rate Bits** — These bits define the timeout period of the API. See Table 3-10 for details of the effect of the autonomous periodical interrupt rate bits. Writable only if APIFE = 0 of VREGAPICL register.

Table 3-10. Selectable Autonomous Periodical Interrupt Periods

APICLK	APIR[15:0]	Selected Period
0	0000	0.2 ms(1)
0	0001	0.4 ms[1]
0	0002	0.6 ms[1]
0	0003	0.8 ms[1]
0	0004	1.0 ms[1]
0	0005	1.2 ms[1]
0	.....	.....
0	FFFD	13106.8 ms[1]
0	FFFE	13107.0 ms[1]
0	FFFF	13107.2 ms[1]
1	0000	2 * bus clock period
1	0001	4 * bus clock period
1	0002	6 * bus clock period
1	0003	8 * bus clock period
1	0004	10 * bus clock period
1	0005	12 * bus clock period
1	.....	.....
1	FFFD	131068 * bus clock period
1	FFFE	131070 * bus clock period
1	FFFF	131072 * bus clock period

1. When trimmed within specified accuracy. See electrical specifications for details.

The period can be calculated as follows depending of APICLK:

Period = 2*(APIR[15:0] + 1) * 0.1 ms or period = 2*(APIR[15:0] + 1) * bus clock period

3.3.2.6 Reserved 06

The Reserved 06 is reserved for test purposes.

0x02F6

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 3-7. Reserved 06

3.3.2.7 HTTrimming Register (VREGHTTR)

The VREGHTTR register allows to trim the VREGL3V3 temperature sense.

0x02F7

	7	6	5	4	3	2	1	0
R	HTOEN	0	0	0	HTTR3	HTTR2	HTTR1	HTTR0
W								
Reset	0	0	0	0	0[1]	0[1]	0[1]	0[1]

1. Reset value is either 0 or preset by factory. See Section 1 (Device Overview) for details.

= Unimplemented or Reserved

Figure 3-8. VREGHTTR

Table 3-11. VREGHTTR field descriptions

Field	Description
7 HTOEN	**High Temperature Offeset Enable Bit** — 01
3–0 HTTR[3:0]	**High Temperature Trimming Bits** — See Table 3-12 for trimming effects.

Table 3-12. Trimming Effect

Bit	Trimming Effect
HTTR[3]	Increases V_{HT} twice of HTTR[2]
HTTR[2]	Increases V_{HT} twice of HTTR[1]
HTTR[1]	Increases V_{HT} twice of HTTR[0]
HTTR[0]	Increases V_{HT} (to compensate Temperature Offset)

3.4 Functional Description

3.4.1 General

Module VREG_3V3 is a voltage regulator, as depicted in Figure 3-1. The regulator functional elements are the regulator core (REG), a low-voltage detect module (LVD), a control block (CTRL), a power-on reset module (POR), and a low-voltage reset module (LVR).

3.4.2 Regulator Core (REG)

Respectively its regulator core has three parallel, independent regulation loops (REG1,REG2 and REG3). REG1 and REG3 differ only in the amount of current that can be delivered.

The regulators are linear regulator with a bandgap reference when operated in Full Performance Mode. They act as a voltage clamp in Reduced Power Mode. All load currents flow from input VDDR to VSS or VSSPLL. The reference circuits are supplied by VDDA and VSSA.

3.4.2.1 Full Performance Mode

In Full Performance Mode, the output voltage is compared with a reference voltage by an operational amplifier. The amplified input voltage difference drives the gate of an output transistor.

3.4.2.2 Reduced Power Mode

In Reduced Power Mode, the gate of the output transistor is connected directly to a reference voltage to reduce power consumption. Mode switching from reduced power to full performance requires a transition time of t_{vup}, if the voltage regulator is enabled.

3.4.3 Low-Voltage Detect (LVD)

Subblock LVD is responsible for generating the low-voltage interrupt (LVI). LVD monitors the input voltage (V_{DDA}–V_{SSA}) and continuously updates the status flag LVDS. Interrupt flag LVIF is set whenever status flag LVDS changes its value. The LVD is available in FPM and is inactive in Reduced Power Mode or Shutdown Mode.

3.4.4 Power-On Reset (POR)

This functional block monitors VDD. If V_{DD} is below V_{PORD}, POR is asserted; if V_{DD} exceeds V_{PORD}, the POR is deasserted. POR asserted forces the MCU into Reset. POR Deasserted will trigger the power-on sequence.

3.4.5 Low-Voltage Reset (LVR)

Block LVR monitors the supplies VDD, VDDX and VDDF. If one (or more) drops below the corresponding assertion level, signal LVR asserts; if all VDD,VDDX and VDDF supplies are above their

corresponding deassertion levels, signal LVR deasserts. The LVR function is available only in Full Performance Mode.

3.4.6 HTD - High Temperature Detect

Subblock HTD is responsible for generating the high temperature interrupt (HTI). HTD monitors the die temperature T_{DIE} and continuously updates the status flag HTDS.
Interrupt flag HTIF is set whenever status flag HTDS changes its value.

The HTD is available in FPM and is inactive in Reduced Power Mode and Shutdown Mode.

The HT Trimming bits HTTR[3:0] can be set so that the temperature offset is zero, if accurate temperature measurement is desired.

See Table 22-12 for the trimming effect of APITR.

3.4.7 Regulator Control (CTRL)

This part contains the register block of VREG_3V3 and further digital functionality needed to control the operating modes. CTRL also represents the interface to the digital core logic.

3.4.8 Autonomous Periodical Interrupt (API)

Subblock API can generate periodical interrupts independent of the clock source of the MCU. To enable the timer, the bit APIFE needs to be set.

The API timer is either clocked by a trimmable internal RC oscillator or the bus clock. Timer operation will freeze when MCU clock source is selected and bus clock is turned off. See CRG specification for details. The clock source can be selected with bit APICLK. APICLK can only be written when APIFE is not set.

The APIR[15:0] bits determine the interrupt period. APIR[15:0] can only be written when APIFE is cleared. As soon as APIFE is set, the timer starts running for the period selected by APIR[15:0] bits. When the configured time has elapsed, the flag APIF is set. An interrupt, indicated by flag APIF = 1, is triggered if interrupt enable bit APIE = 1. The timer is started automatically again after it has set APIF.

The procedure to change APICLK or APIR[15:0] is first to clear APIFE, then write to APICLK or APIR[15:0], and afterwards set APIFE.

The API Trimming bits APITR[5:0] must be set so the minimum period equals 0.2 ms if stable frequency is desired.

See Table 3-8 for the trimming effect of APITR.

NOTE

The first period after enabling the counter by APIFE might be reduced by API start up delay t_{sdel}. The API internal RC oscillator clock is not available if VREG_3V3 is in Shutdown Mode.

It is possible to generate with the API a waveform at an external pin by enabling the API by setting APIFE and enabling the external access with setting APIEA. By setting APIES the waveform can be selected. If APIES is set, then at the external pin a clock is visible with 2 times the selected API Period (Table 3-10). If APIES is not set, then at the external pin will be a high pulse at the end of every selected period with the size of half of the min period (Table 3-10). See device level specification for connectivity.

3.4.9 Resets

This section describes how VREG_3V3 controls the reset of the MCU.The reset values of registers and signals are provided in Section 3.3, “Memory Map and Register Definition”. Possible reset sources are listed in Table 3-13.

Table 3-13. Reset Sources

Reset Source	Local Enable
Power-on reset	Always active
Low-voltage reset	Available only in Full Performance Mode

3.4.10 Description of Reset Operation

3.4.10.1 Power-On Reset (POR)

During chip power-up the digital core may not work if its supply voltage V_{DD} is below the POR deassertion level (V_{PORD}). Therefore, signal POR, which forces the other blocks of the device into reset, is kept high until V_{DD} exceeds V_{PORD}. The MCU will run the start-up sequence after POR deassertion. The power-on reset is active in all operation modes of VREG_3V3.

3.4.10.2 Low-Voltage Reset (LVR)

For details on low-voltage reset, see Section 3.4.5, “Low-Voltage Reset (LVR)”.

3.4.11 Interrupts

This section describes all interrupts originated by VREG_3V3.

The interrupt vectors requested by VREG_3V3 are listed in Table 3-14. Vector addresses and interrupt priorities are defined at MCU level.

Table 3-14. Interrupt Vectors

Interrupt Source	Local Enable
Low-voltage interrupt (LVI)	LVIE = 1; available only in Full Performance Mode
High Temperature Interrupt (HTI)	HTIE=1; available only in Full Performance Mode
Autonomous periodical interrupt (API)	APIE = 1

3.4.11.1 Low-Voltage Interrupt (LVI)

In FPM, VREG_3V3 monitors the input voltage V_{DDA}. Whenever V_{DDA} drops below level $V_{LVIA,}$ the status bit LVDS is set to 1. On the other hand, LVDS is reset to 0 when V_{DDA} rises above level V_{LVID}. An interrupt, indicated by flag LVIF = 1, is triggered by any change of the status bit LVDS if interrupt enable bit LVIE = 1.

NOTE

On entering the Reduced Power Mode, the LVIF is not cleared by the VREG_3V3.

3.4.11.2 HTI - High Temperature Interrupt

In FPM VREGL3V3 monitors the die temperature T_{DIE}. Whenever T_{DIE} exceeds level T_{HTIA} the status bit HTDS is set to 1. Vice versa, HTDS is reset to 0 when T_{DIE} get below level T_{HTID}. An interrupt, indicated by flag HTIF=1, is triggered by any change of the status bit HTDS if interrupt enable bit HTIE=1.

NOTE

On entering the Reduced Power Mode the HTIF is not cleared by the VREGL3V3.

3.4.11.3 Autonomous Periodical Interrupt (API)

As soon as the configured timeout period of the API has elapsed, the APIF bit is set. An interrupt, indicated by flag APIF = 1, is triggered if interrupt enable bit APIE = 1.

Chapter 4
384 KByte Flash Module (S12XFTM384K2V1)

Table 4-1. Revision History

Revision Number	Revision Date	Sections Affected	Description of Changes
V01.10	29 Nov 2007		- Cleanup
V01.11	19 Dec 2007	4.4.2/4-167 4.4.2/4-167 4.3.1/4-136	- Updated Command Error Handling tables based on parent-child relationship with FTM512K3 - Corrected Error Handling table for Full Partition D-Flash, Partition D-Flash, and EEPROM Emulation Query commands - Corrected P-Flash IFR Accessibility table
V01.12	25 Sep 2009	4.1/4-131 4.3.2.1/4-143 4.4.2.4/4-170 4.4.2.7/4-173 4.4.2.12/4-177 4.4.2.12/4-177 4.4.2.12/4-177 4.4.2.20/4-186 4.3.2/4-141 4.3.2.1/4-143 4.4.1.2/4-162 4.6/4-192	- Clarify single bit fault correction for P-Flash phrase - Expand FDIV vs OSCCLK Frequency table - Add statement concerning code runaway when executing Read Once command from Flash block containing associated fields - Add statement concerning code runaway when executing Program Once command from Flash block containing associated fields - Add statement concerning code runaway when executing Verify Backdoor Access Key command from Flash block containing associated fields - Relate Key 0 to associated Backdoor Comparison Key address - Change “power down reset” to “reset” - Add ACCERR condition for Disable EEPROM Emulation command The following changes were made to clarify module behavior related to Flash register access during reset sequence and while Flash commands are active: - Add caution concerning register writes while command is active - Writes to FCLKDIV are allowed during reset sequence while CCIF is clear - Add caution concerning register writes while command is active - Writes to FCCOBIX, FCCOBHI, FCCOBLO registers are ignored during reset sequence

4.1 Introduction

The FTM384K2 module implements the following:

- 384 Kbytes of P-Flash (Program Flash) memory, consisting of 2 physical Flash blocks, intended primarily for nonvolatile code storage
- 32 Kbytes of D-Flash (Data Flash) memory, consisting of 1 physical Flash block, that can be used as nonvolatile storage to support the built-in hardware scheme for emulated EEPROM, as basic Flash memory primarily intended for nonvolatile data storage, or as a combination of both
- 4 Kbytes of buffer RAM, consisting of 1 physical RAM block, that can be used as emulated EEPROM using a built-in hardware scheme, as basic RAM, or as a combination of both

The Flash memory is ideal for single-supply applications allowing for field reprogramming without requiring external high voltage sources for program or erase operations. The Flash module includes a memory controller that executes commands to modify Flash memory contents or configure module resources for emulated EEPROM operation. The user interface to the memory controller consists of the indexed Flash Common Command Object (FCCOB) register which is written to with the command, global address, data, and any required command parameters. The memory controller must complete the execution of a command before the FCCOB register can be written to with a new command.

CAUTION

A Flash word or phrase must be in the erased state before being programmed. Cumulative programming of bits within a Flash word or phrase is not allowed.

The RAM and Flash memory may be read as bytes, aligned words, or misaligned words. Read access time is one bus cycle for bytes and aligned words, and two bus cycles for misaligned words. For Flash memory, an erased bit reads 1 and a programmed bit reads 0.

It is not possible to read from a Flash block while any command is executing on that specific Flash block. It is possible to read from a Flash block while a command is executing on a different Flash block.

Both P-Flash and D-Flash memories are implemented with Error Correction Codes (ECC) that can resolve single bit faults and detect double bit faults. For P-Flash memory, the ECC implementation requires that programming be done on an aligned 8 byte basis (a Flash phrase). Since P-Flash memory is always read by phrase, only one single bit fault in the phrase containing the byte or word accessed will be corrected.

4.1.1 Glossary

Buffer RAM — The buffer RAM constitutes the volatile memory store required for EEE. Memory space in the buffer RAM not required for EEE can be partitioned to provide volatile memory space for applications.

Command Write Sequence — An MCU instruction sequence to execute built-in algorithms (including program and erase) on the Flash memory.

D-Flash Memory — The D-Flash memory constitutes the nonvolatile memory store required for EEE. Memory space in the D-Flash memory not required for EEE can be partitioned to provide nonvolatile memory space for applications.

D-Flash Sector — The D-Flash sector is the smallest portion of the D-Flash memory that can be erased. The D-Flash sector consists of four 64 byte rows for a total of 256 bytes.

EEE (Emulated EEPROM) — A method to emulate the small sector size features and endurance characteristics associated with an EEPROM.

EEE IFR — Nonvolatile information register located in the D-Flash block that contains data required to partition the D-Flash memory and buffer RAM for EEE. The EEE IFR is visible in the global memory map by setting the EEEIFRON bit in the MMCCTL1 register.

NVM Command Mode — An NVM mode using the CPU to setup the FCCOB register to pass parameters required for Flash command execution.

Phrase — An aligned group of four 16-bit words within the P-Flash memory. Each phrase includes eight ECC bits for single bit fault correction and double bit fault detection within the phrase.

P-Flash Memory — The P-Flash memory constitutes the main nonvolatile memory store for applications.

P-Flash Sector — The P-Flash sector is the smallest portion of the P-Flash memory that can be erased. Each P-Flash sector contains 1024 bytes.

Program IFR — Nonvolatile information register located in the P-Flash block that contains the Device ID, Version ID, and the Program Once field. The Program IFR is visible in the global memory map by setting the PGMIFRON bit in the MMCCTL1 register.

4.1.2 Features

4.1.2.1 P-Flash Features

- 384 Kbytes of P-Flash memory composed of one 256 Kbyte Flash block and one 128 Kbyte Flash block. The 256 Kbyte Flash block consists of two 128 Kbyte sections each divided into 128 sectors of 1024 bytes. The 128 Kbyte Flash block is divided into 128 sectors of 1024 bytes.
- Single bit fault correction and double bit fault detection within a 64-bit phrase during read operations
- Automated program and erase algorithm with verify and generation of ECC parity bits
- Fast sector erase and phrase program operation
- Ability to program up to one phrase in each P-Flash block simultaneously
- Flexible protection scheme to prevent accidental program or erase of P-Flash memory

4.1.2.2 D-Flash Features

- Up to 32 Kbytes of D-Flash memory with 256 byte sectors for user access
- Dedicated commands to control access to the D-Flash memory over EEE operation
- Single bit fault correction and double bit fault detection within a word during read operations
- Automated program and erase algorithm with verify and generation of ECC parity bits
- Fast sector erase and word program operation
- Ability to program up to four words in a burst sequence

4.1.2.3 Emulated EEPROM Features

- Up to 4 Kbytes of emulated EEPROM (EEE) accessible as 4 Kbytes of RAM
- Flexible protection scheme to prevent accidental program or erase of data

- Automatic EEE file handling using an internal Memory Controller
- Automatic transfer of valid EEE data from D-Flash memory to buffer RAM on reset
- Ability to monitor the number of outstanding EEE related buffer RAM words left to be programmed into D-Flash memory
- Ability to disable EEE operation and allow priority access to the D-Flash memory
- Ability to cancel all pending EEE operations and allow priority access to the D-Flash memory

4.1.2.4 User Buffer RAM Features

- Up to 4 Kbytes of RAM for user access

4.1.2.5 Other Flash Module Features

- No external high-voltage power supply required for Flash memory program and erase operations
- Interrupt generation on Flash command completion and Flash error detection
- Security mechanism to prevent unauthorized access to the Flash memory

4.1.3 Block Diagram

The block diagram of the Flash module is shown in Figure 4-1.

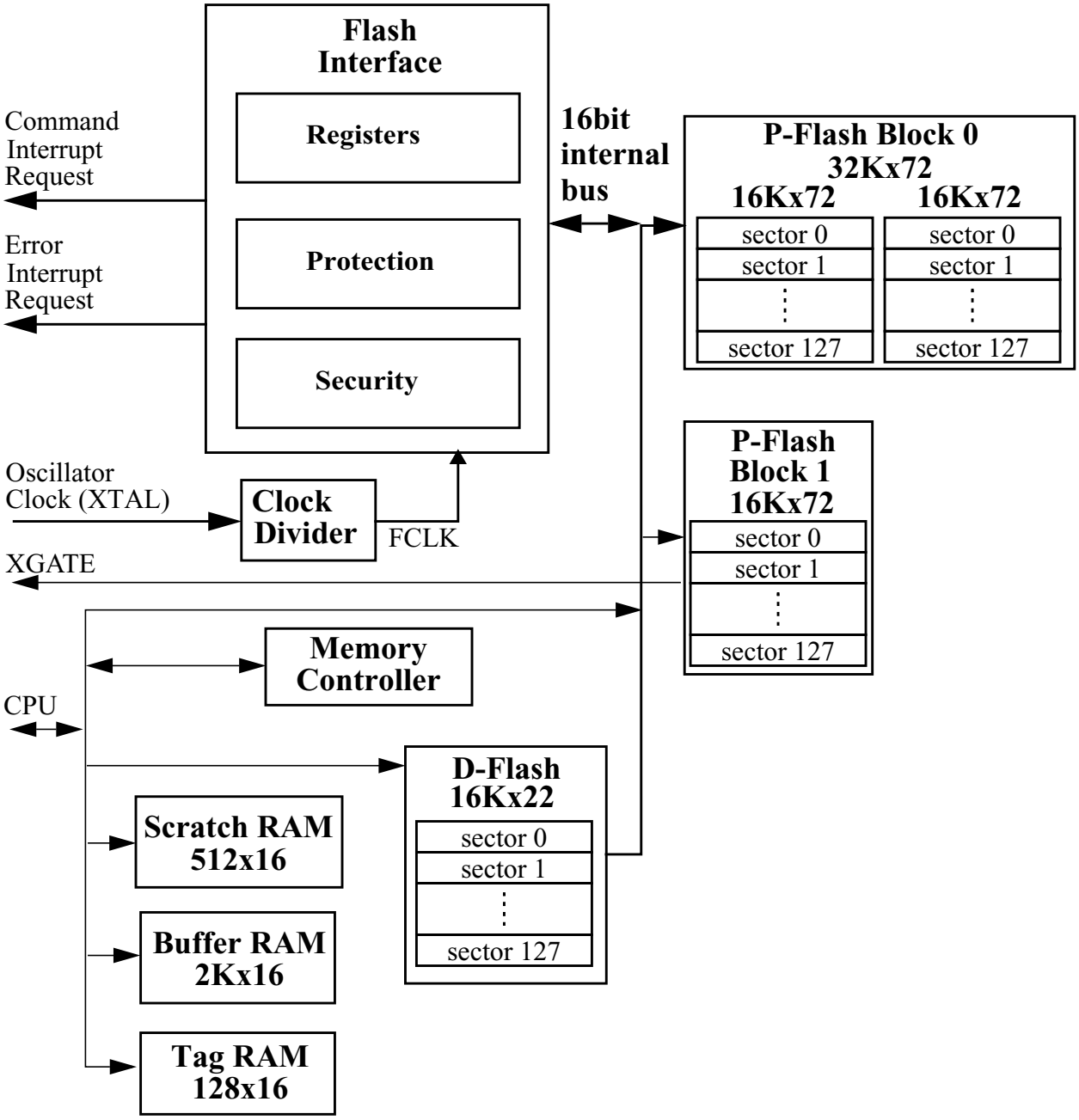


Figure 4-1. FTM384K2 Block Diagram

4.2 External Signal Description

The Flash module contains no signals that connect off-chip.

4.3 Memory Map and Registers

This section describes the memory map and registers for the Flash module. Read data from unimplemented memory space in the Flash module is undefined. Write access to unimplemented or reserved memory space in the Flash module will be ignored by the Flash module.

4.3.1 Module Memory Map

The S12X architecture places the P-Flash memory between global addresses 0x78_0000 and 0x7F_FFFF as shown in Table 4-2. The P-Flash memory map is shown in Figure 4-2.

Table 4-2. P-Flash Memory Addressing

Global Address	Size (Bytes)	Description
0x7C_0000 – 0x7F_FFFF	256 K	P-Flash Block 0 Contains Flash Configuration Field (see Table 4-3)
0x7A_0000 – 0x7B_FFFF	128 K	No P-Flash Memory
0x78_0000 – 0x79_FFFF	128 K	P-Flash Block 1

The FPROT register, described in Section 4.3.2.9, can be set to protect regions in the Flash memory from accidental program or erase. Three separate memory regions, one growing upward from global address 0x7F_8000 in the Flash memory (called the lower region), one growing downward from global address 0x7F_FFFF in the Flash memory (called the higher region), and the remaining addresses in the Flash memory, can be activated for protection. The Flash memory addresses covered by these protectable regions are shown in the P-Flash memory map. The higher address region is mainly targeted to hold the boot loader code since it covers the vector space. Default protection settings as well as security information that allows the MCU to restrict access to the Flash module are stored in the Flash configuration field as described in Table 4-3.

Table 4-3. Flash Configuration Field(1)

Global Address	Size (Bytes)	Description
0x7F_FF00 – 0x7F_FF07	8	Backdoor Comparison Key Refer to Section 4.4.2.12, “Verify Backdoor Access Key Command,” and Section 4.5.1, “Unsecuring the MCU using Backdoor Key Access”
0x7F_FF08 – 0x7F_FF0B(2)	4	Reserved
0x7F_FF0C[2]	1	P-Flash Protection byte. Refer to Section 4.3.2.9, “P-Flash Protection Register (FPROT)”
0x7F_FF0D[2]	1	EEE Protection byte Refer to Section 4.3.2.10, “EEE Protection Register (EPROT)”
0x7F_FF0E[2]	1	Flash Nonvolatile byte Refer to Section 4.3.2.14, “Flash Option Register (FOPT)”

Table 4-3. Flash Configuration Field[1]

Global Address	Size (Bytes)	Description
0x7F_FF0F[2]	1	Flash Security byte Refer to Section 4.3.2.2, “Flash Security Register (FSEC)”

1. Older versions may have swapped protection byte addresses
2. 0x7FF08 - 0x7F_FF0F form a Flash phrase and must be programmed in a single command write sequence. Each byte in the 0x7F_FF08 - 0x7F_FF0B reserved field should be programmed to 0xFF.

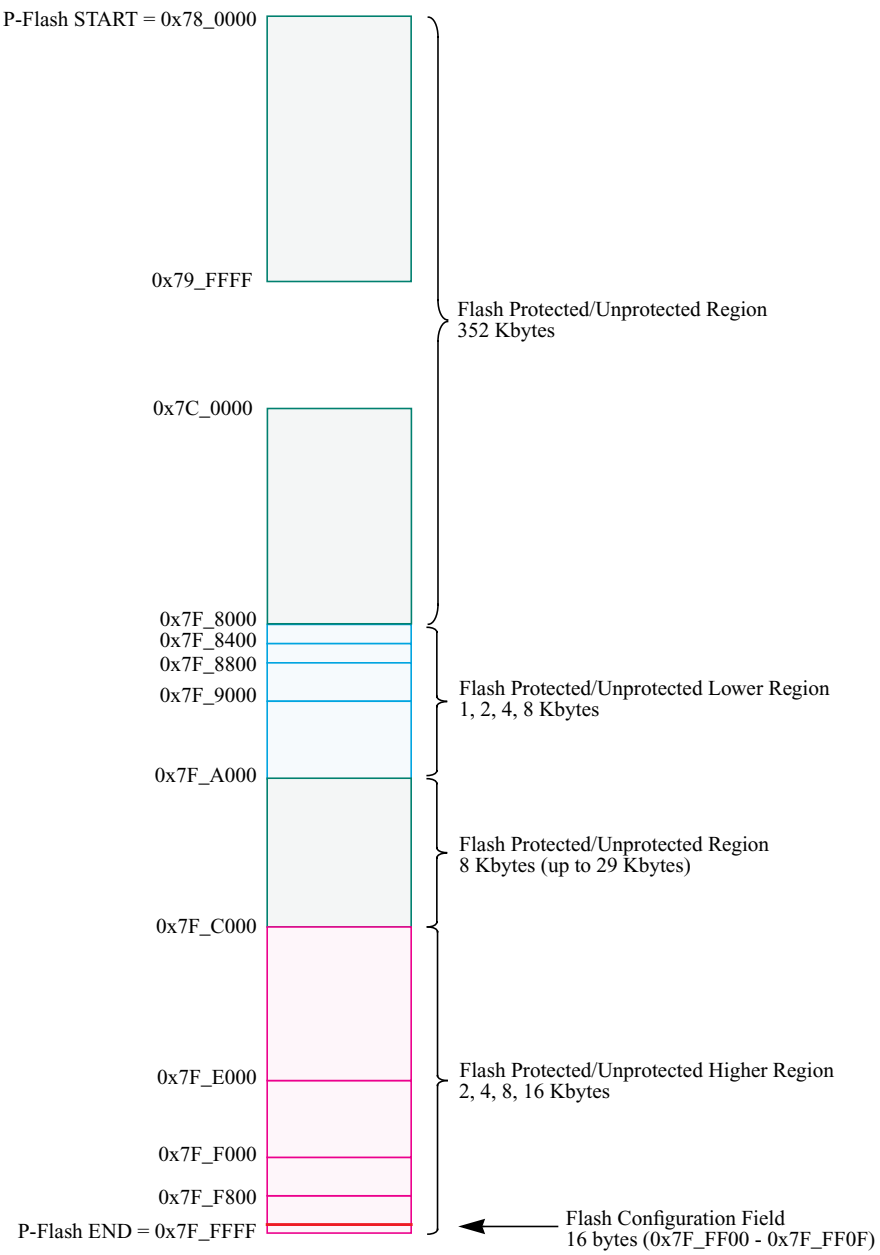


Figure 4-2. P-Flash Memory Map

Table 4-4. Program IFR Fields

Global Address (PGMIFRON)	Size (Bytes)	Field Description
0x40_0000 – 0x40_0007	8	Device ID
0x40_0008 – 0x40_00E7	224	Reserved
0x40_00E8 – 0x40_00E9	2	Version ID
0x40_00EA – 0x40_00FF	22	Reserved
0x40_0100 – 0x40_013F	64	Program Once Field Refer to Section 4.4.2.7, “Program Once Command”
0x40_0140 – 0x40_01FF	192	Reserved

Table 4-5. P-Flash IFR Accessibility

Global Address (PGMIFRON)	Size (Bytes)	Accessed From
0x40_0000 – 0x40_01FF	512	XBUS0 (PBLK0)(1)
0x40_0200 – 0x40_03FF	512	Unimplemented
0x40_0400 – 0x40_05FF	512	Unimplemented
0x40_0600 – 0x40_07FF	512	XBUS1 (PBLK1)

1. Refer to Table 4-4 for more details.

Table 4-6. EEE Resource Fields

Global Address	Size (Bytes)	Description
0x10_0000 – 0x10_7FFF	32,768	D-Flash Memory (User and EEE)
0x10_8000 – 0x11_FFFF	98,304	Reserved
0x12_0000 – 0x12_007F	128	EEE Nonvolatile Information Register (EEEIFRON(1) = 1)
0x12_0080 – 0x12_0FFF	3,968	Reserved
0x12_1000 – 0x12_1EFF	3,840	Reserved
0x12_1F00 – 0x12_1FFF	256	EEE Tag RAM (TMGRAMON[1] = 1)
0x12_2000 – 0x12_3BFF	7,168	Reserved
0x12_3C00 – 0x12_3FFF	1,024	Memory Controller Scratch RAM (TMGRAMON[1] = 1)
0x12_4000 – 0x12_DFFF	40,960	Reserved
0x12_E000 – 0x12_FFFF	8,192	Reserved
0x13_0000 – 0x13_EFFF	61,440	Reserved
0x13_F000 – 0x13_FFFF	4,096	Buffer RAM (User and EEE)

1. MMCCTL1 register bit

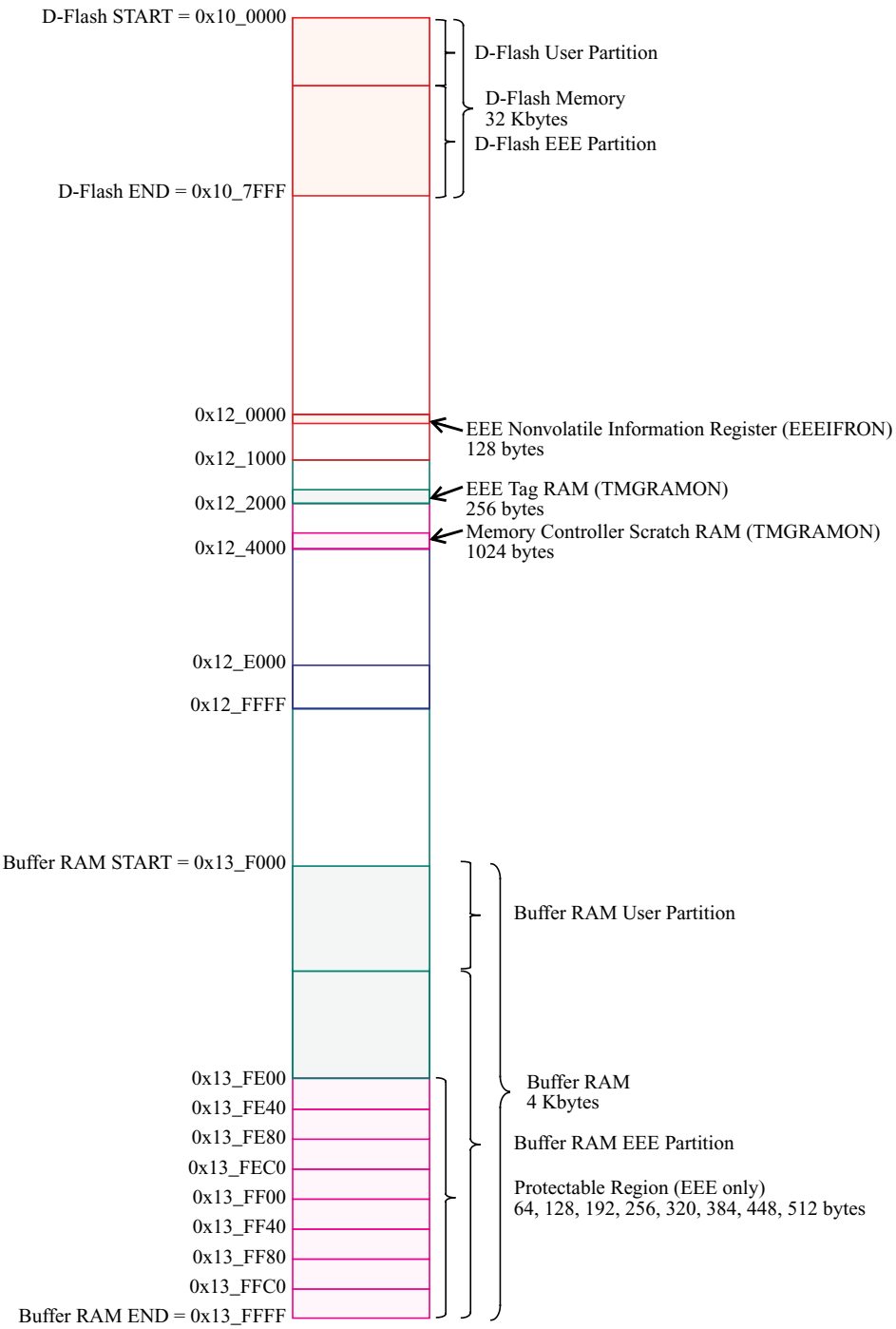


Figure 4-3. EEE Resource Memory Map

The Full Partition D-Flash command (see Section 4.4.2.15) is used to program the EEE nonvolatile information register fields where address 0x12_0000 defines the D-Flash partition for user access and address 0x12_0004 defines the buffer RAM partition for EEE operations.

Table 4-7. EEE Nonvolatile Information Register Fields

Global Address (EEEIFRON)	Size (Bytes)	Description
0x12_0000 – 0x12_0001	2	D-Flash User Partition (DFPART) Refer to Section 4.4.2.15, “Full Partition D-Flash Command”
0x12_0002 – 0x12_0003	2	D-Flash User Partition (duplicate(1))
0x12_0004 – 0x12_0005	2	Buffer RAM EEE Partition (ERPART) Refer to Section 4.4.2.15, “Full Partition D-Flash Command”
0x12_0006 – 0x12_0007	2	Buffer RAM EEE Partition (duplicate1)
0x12_0008 – 0x12_007F	120	Reserved

1. Duplicate value used if primary value generates a double bit fault when read during the reset sequence.

4.3.2 Register Descriptions

The Flash module contains a set of 20 control and status registers located between Flash module base + 0x0000 and 0x0013. A summary of the Flash module registers is given in Figure 4-4 with detailed descriptions in the following subsections.

CAUTION

Writes to any Flash register must be avoided while a Flash command is active (CCIF=0) to prevent corruption of Flash register contents and Memory Controller behavior.

Address & Name		7	6	5	4	3	2	1	0
0x0000 FCLKDIV	R	FDIVLD	FDIV6	FDIV5	FDIV4	FDIV3	FDIV2	FDIV1	FDIV0
	W								
0x0001 FSEC	R	KEYEN1	KEYEN0	RNV5	RNV4	RNV3	RNV2	SEC1	SEC0
	W								
0x0002 FCCOBIX	R	0	0	0	0	0	CCOBIX2	CCOBIX1	CCOBIX0
	W								
0x0003 FECCRIX	R	0	0	0	0	0	ECCRIX2	ECCRIX1	ECCRIX0
	W								
0x0004 FCNFG	R	CCIE	0	0	IGNSF	0	0	FDFD	FSFD
	W								
0x0005 FERCNFG	R	ERSERIE	PGMERIE	0	EPVIOLIE	ERSVIE1	ERSVIE0	DFDIE	SFDIE
	W								

Figure 4-4. FTM384K2 Register Summary

Address & Name		7	6	5	4	3	2	1	0
0x0006 FSTAT	R	CCIF	0	ACCERR	FPVIOL	MGBUSY	RSVD	MGSTAT1	MGSTAT0
	W								
0x0007 FERSTAT	R	ERSERIF	PGMERIF	0	EPVIOLIF	ERSVIF1	ERSVIF0	DFDIF	SFDIF
	W								
0x0008 FPROT	R	FPOPEN	RNV6	FPHDIS	FPHS1	FPHS0	FPLDIS	FPLS1	FPLS0
	W								
0x0009 EPROT	R	EPOPEN	RNV6	RNV5	RNV4	EPDIS	EPS2	EPS1	EPS0
	W								
0x000A FCCOBHI	R	CCOB15	CCOB14	CCOB13	CCOB12	CCOB11	CCOB10	CCOB9	CCOB8
	W								
0x000B FCCOBLO	R	CCOB7	CCOB6	CCOB5	CCOB4	CCOB3	CCOB2	CCOB1	CCOB0
	W								
0x000C ETAGHI	R	ETAG15	ETAG14	ETAG13	ETAG12	ETAG11	ETAG10	ETAG9	ETAG8
	W								
0x000D ETAGLO	R	ETAG7	ETAG6	ETAG5	ETAG4	ETAG3	ETAG2	ETAG1	ETAG0
	W								
0x000E FECCRHI	R	ECCR15	ECCR14	ECCR13	ECCR12	ECCR11	ECCR10	ECCR9	ECCR8
	W								
0x000F FECCRLO	R	ECCR7	ECCR6	ECCR5	ECCR4	ECCR3	ECCR2	ECCR1	ECCR0
	W								
0x0010 FOPT	R	NV7	NV6	NV5	NV4	NV3	NV2	NV1	NV0
	W								
0x0011 FRSV0	R	0	0	0	0	0	0	0	0
	W								
0x0012 FRSV1	R	0	0	0	0	0	0	0	0
	W								
0x0013 FRSV2	R	0	0	0	0	0	0	0	0
	W								

Figure 4-4. FTM384K2 Register Summary (continued)

Address & Name	7	6	5	4	3	2	1	0

= Unimplemented or Reserved

Figure 4-4. FTM384K2 Register Summary (continued)

4.3.2.1 Flash Clock Divider Register (FCLKDIV)

The FCLKDIV register is used to control timed events in program and erase algorithms.

Offset Module Base + 0x0000

	7	6	5	4	3	2	1	0
R	FDIVLD	FDIV[6:0]						
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-5. Flash Clock Divider Register (FCLKDIV)

All bits in the FCLKDIV register are readable, bits 6–0 are write once and bit 7 is not writable.

Table 4-8. FCLKDIV Field Descriptions

Field	Description
7 FDIVLD	**Clock Divider Loaded** 0 FCLKDIV register has not been written 1 FCLKDIV register has been written since the last reset
6–0 FDIV[6:0]	**Clock Divider Bits** — FDIV[6:0] must be set to effectively divide OSCCLK down to generate an internal Flash clock, FCLK, with a target frequency of 1 MHz for use by the Flash module to control timed events during program and erase algorithms. Table 4-9 shows recommended values for FDIV[6:0] based on OSCCLK frequency. Please refer to Section 4.4.1, “Flash Command Operations,” for more information.

CAUTION

The FCLKDIV register should never be written while a Flash command is executing (CCIF=0). The FCLKDIV register is writable during the Flash reset sequence even though CCIF is clear.

Table 4-9. FDIV vs OSCCLK Frequency

OSCCLK Frequency (MHz)		FDIV[6:0]	OSCCLK Frequency (MHz)		FDIV[6:0]	OSCCLK Frequency (MHz)		FDIV[6:0]
MIN(1)	MAX(2)		MIN1	MAX2		MIN1	MAX2	
			33.60	34.65	0x20	67.20	68.25	0x40
1.60	2.10	0x01	34.65	35.70	0x21	68.25	69.30	0x41
2.40	3.15	0x02	35.70	36.75	0x22	69.30	70.35	0x42
3.20	4.20	0x03	36.75	37.80	0x23	70.35	71.40	0x43
4.20	5.25	0x04	37.80	38.85	0x24	71.40	72.45	0x44
5.25	6.30	0x05	38.85	39.90	0x25	72.45	73.50	0x45
6.30	7.35	0x06	39.90	40.95	0x26	73.50	74.55	0x46
7.35	8.40	0x07	40.95	42.00	0x27	74.55	75.60	0x47
8.40	9.45	0x08	42.00	43.05	0x28	75.60	76.65	0x48
9.45	10.50	0x09	43.05	44.10	0x29	76.65	77.70	0x49
10.50	11.55	0x0A	44.10	45.15	0x2A	77.70	78.75	0x4A
11.55	12.60	0x0B	45.15	46.20	0x2B	78.75	79.80	0x4B
12.60	13.65	0x0C	46.20	47.25	0x2C	79.80	80.85	0x4C
13.65	14.70	0x0D	47.25	48.30	0x2D	80.85	81.90	0x4D
14.70	15.75	0x0E	48.30	49.35	0x2E	81.90	82.95	0x4E
15.75	16.80	0x0F	49.35	50.40	0x2F	82.95	84.00	0x4F
16.80	17.85	0x10	50.40	51.45	0x30	84.00	85.05	0x50
17.85	18.90	0x11	51.45	52.50	0x31	85.05	86.10	0x51
18.90	19.95	0x12	52.50	53.55	0x32	86.10	87.15	0x52
19.95	21.00	0x13	53.55	54.60	0x33	87.15	88.20	0x53
21.00	22.05	0x14	54.60	55.65	0x34	88.20	89.25	0x54
22.05	23.10	0x15	55.65	56.70	0x35	89.25	90.30	0x55
23.10	24.15	0x16	56.70	57.75	0x36	90.30	91.35	0x56
24.15	25.20	0x17	57.75	58.80	0x37	91.35	92.40	0x57
25.20	26.25	0x18	58.80	59.85	0x38	92.40	93.45	0x58
26.25	27.30	0x19	59.85	60.90	0x39	93.45	94.50	0x59
27.30	28.35	0x1A	60.90	61.95	0x3A	94.50	95.55	0x5A
28.35	29.40	0x1B	61.95	63.00	0x3B	95.55	96.60	0x5B
29.40	30.45	0x1C	63.00	64.05	0x3C	96.60	97.65	0x5C
30.45	31.50	0x1D	64.05	65.10	0x3D	97.65	98.70	0x5D
31.50	32.55	0x1E	65.10	66.15	0x3E	98.70	99.75	0x5E
32.55	33.60	0x1F	66.15	67.20	0x3F	99.75	100.80	0x5F

1. FDIV shown generates an FCLK frequency of >0.8 MHz

2. FDIV shown generates an FCLK frequency of 1.05 MHz

4.3.2.2 Flash Security Register (FSEC)

The FSEC register holds all bits associated with the security of the MCU and Flash module.

Offset Module Base + 0x0001

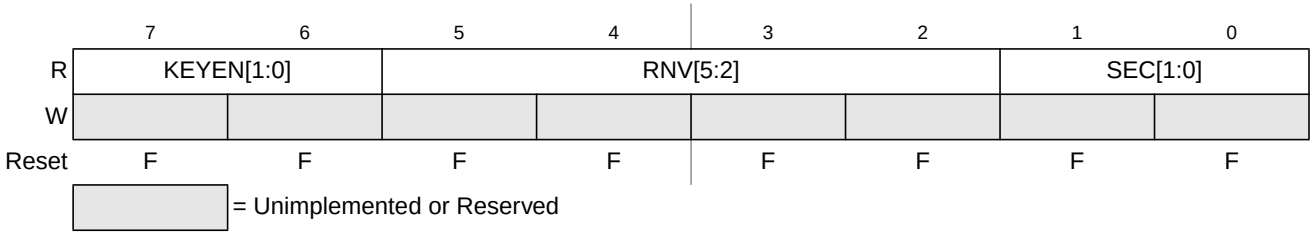


Figure 4-6. Flash Security Register (FSEC)

All bits in the FSEC register are readable but not writable.

During the reset sequence, the FSEC register is loaded with the contents of the Flash security byte in the Flash configuration field at global address 0x7F_FF0F located in P-Flash memory (see Table 4-3) as indicated by reset condition F in Figure 4-6. If a double bit fault is detected while reading the P-Flash phrase containing the Flash security byte during the reset sequence, all bits in the FSEC register will be set to leave the Flash module in a secured state with backdoor key access disabled.

Table 4-10. FSEC Field Descriptions

Field	Description
7–6 KEYEN[1:0]	**Backdoor Key Security Enable Bits** — The KEYEN[1:0] bits define the enabling of backdoor key access to the Flash module as shown in Table 4-11.
5–2 RNV[5:2}	**Reserved Nonvolatile Bits** — The RNV bits should remain in the erased state for future enhancements.
1–0 SEC[1:0]	**Flash Security Bits** — The SEC[1:0] bits define the security state of the MCU as shown in Table 4-12. If the Flash module is unsecured using backdoor key access, the SEC bits are forced to 10.

Table 4-11. Flash KEYEN States

KEYEN[1:0]	Status of Backdoor Key Access
00	DISABLED
01	DISABLED(1)
10	ENABLED
11	DISABLED

1. Preferred KEYEN state to disable backdoor key access.

Table 4-12. Flash Security States

SEC[1:0]	Status of Security
00	SECURED
01	SECURED(1)
10	UNSECURED
11	SECURED

1. Preferred SEC state to set MCU to secured state.

The security function in the Flash module is described in Section 4.5.

4.3.2.3 Flash CCOB Index Register (FCCOBIX)

The FCCOBIX register is used to index the FCCOB register for Flash memory operations.

Offset Module Base + 0x0002

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	CCOBIX[2:0]		
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-7. FCCOB Index Register (FCCOBIX)

CCOBIX bits are readable and writable while remaining bits read 0 and are not writable.

Table 4-13. FCCOBIX Field Descriptions

Field	Description
2–0 CCOBIX[1:0]	**Common Command Register Index**— The CCOBIX bits are used to select which word of the FCCOB register array is being read or written to. See Section 4.3.2.11, “Flash Common Command Object Register (FCCOB),” for more details.

4.3.2.4 Flash ECCR Index Register (FECCRIX)

The FECCRIX register is used to index the FECCR register for ECC fault reporting.

Offset Module Base + 0x0003

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	ECCRIX[2:0]		
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-8. FECCR Index Register (FECCRIX)

ECCRIX bits are readable and writable while remaining bits read 0 and are not writable.

Table 4-14. FECCRIX Field Descriptions

Field	Description
2-0 ECCRIX[2:0]	**ECC Error Register Index**— The ECCRIX bits are used to select which word of the FECCR register array is being read. See Section 4.3.2.13, “Flash ECC Error Results Register (FECCR),” for more details.

4.3.2.5 Flash Configuration Register (FCNFG)

The FCNFG register enables the Flash command complete interrupt and forces ECC faults on Flash array read access from the CPU or XGATE.

Offset Module Base + 0x0004

	7	6	5	4	3	2	1	0
R	CCIE	0	0	IGNSF	0	0	FDFD	FSFD
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-9. Flash Configuration Register (FCNFG)

CCIE, IGNSF, FDFD, and FSFD bits are readable and writable while remaining bits read 0 and are not writable.

Table 4-15. FCNFG Field Descriptions

Field	Description
7 CCIE	**Command Complete Interrupt Enable** — The CCIE bit controls interrupt generation when a Flash command has completed. 0 Command complete interrupt disabled 1 An interrupt will be requested whenever the CCIF flag in the FSTAT register is set (see Section 4.3.2.7)
4 IGNSF	**Ignore Single Bit Fault** — The IGNSF controls single bit fault reporting in the FERSTAT register (see Section 4.3.2.8). 0 All single bit faults detected during array reads are reported 1 Single bit faults detected during array reads are not reported and the single bit fault interrupt will not be generated

Table 4-15. FCNFG Field Descriptions (continued)

Field	Description
1 FDFD	**Force Double Bit Fault Detect** — The FDFD bit allows the user to simulate a double bit fault during Flash array read operations and check the associated interrupt routine. The FDFD bit is cleared by writing a 0 to FDFD. The FECCR registers will not be updated during the Flash array read operation with FDFD set unless an actual double bit fault is detected. 0 Flash array read operations will set the DFDIF flag in the FERSTAT register only if a double bit fault is detected 1 Any Flash array read operation will force the DFDIF flag in the FERSTAT register to be set (see Section 4.3.2.7) and an interrupt will be generated as long as the DFDIE interrupt enable in the FERCNFG register is set (see Section 4.3.2.6)
0 FSFD	**Force Single Bit Fault Detect** — The FSFD bit allows the user to simulate a single bit fault during Flash array read operations and check the associated interrupt routine. The FSFD bit is cleared by writing a 0 to FSFD. The FECCR registers will not be updated during the Flash array read operation with FSFD set unless an actual single bit fault is detected. 0 Flash array read operations will set the SFDIF flag in the FERSTAT register only if a single bit fault is detected 1 Flash array read operation will force the SFDIF flag in the FERSTAT register to be set (see Section 4.3.2.7) and an interrupt will be generated as long as the SFDIE interrupt enable in the FERCNFG register is set (see Section 4.3.2.6)

4.3.2.6 Flash Error Configuration Register (FERCNFG)

The FERCNFG register enables the Flash error interrupts for the FERSTAT flags.

Offset Module Base + 0x0005

	7	6	5	4	3	2	1	0
R	ERSERIE	PGMERIE	0	EPVIOLIE	ERSVIE1	ERSVIE0	DFDIE	SFDIE
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-10. Flash Error Configuration Register (FERCNFG)

All assigned bits in the FERCNFG register are readable and writable.

Table 4-16. FERCNFG Field Descriptions

Field	Description
7 ERSERIE	**EEE Erase Error Interrupt Enable** — The ERSERIE bit controls interrupt generation when a failure is detected during an EEE erase operation. 0 ERSERIF interrupt disabled 1 An interrupt will be requested whenever the ERSERIF flag is set (see Section 4.3.2.8)
6 PGMERIE	**EEE Program Error Interrupt Enable** — The PGMERIE bit controls interrupt generation when a failure is detected during an EEE program operation. 0 PGMERIF interrupt disabled 1 An interrupt will be requested whenever the PGMERIF flag is set (see Section 4.3.2.8)
4 EPVIOLIE	**EEE Protection Violation Interrupt Enable** — The EPVIOLIE bit controls interrupt generation when a protection violation is detected during a write to the buffer RAM EEE partition. 0 EPVIOLIF interrupt disabled 1 An interrupt will be requested whenever the EPVIOLIF flag is set (see Section 4.3.2.8)

Table 4-16. FERCNFG Field Descriptions (continued)

Field	Description
3 ERSVIE1	**EEE Error Type 1 Interrupt Enable** — The ERSVIE1 bit controls interrupt generation when a change state error is detected during an EEE operation. 0 ERSVIF1 interrupt disabled 1 An interrupt will be requested whenever the ERSVIF1 flag is set (see Section 4.3.2.8)
2 ERSVIE0	**EEE Error Type 0 Interrupt Enable** — The ERSVIE0 bit controls interrupt generation when a sector format error is detected during an EEE operation. 0 ERSVIF0 interrupt disabled 1 An interrupt will be requested whenever the ERSVIF0 flag is set (see Section 4.3.2.8)
1 DFDIE	**Double Bit Fault Detect Interrupt Enable** — The DFDIE bit controls interrupt generation when a double bit fault is detected during a Flash block read operation. 0 DFDIF interrupt disabled 1 An interrupt will be requested whenever the DFDIF flag is set (see Section 4.3.2.8)
0 SFDIE	**Single Bit Fault Detect Interrupt Enable** — The SFDIE bit controls interrupt generation when a single bit fault is detected during a Flash block read operation. 0 SFDIF interrupt disabled whenever the SFDIF flag is set (see Section 4.3.2.8) 1 An interrupt will be requested whenever the SFDIF flag is set (see Section 4.3.2.8)

4.3.2.7 Flash Status Register (FSTAT)

The FSTAT register reports the operational status of the Flash module.

Offset Module Base + 0x0006

	7	6	5	4	3	2	1	0
R	CCIF	0	ACCERR	FPVIOL	MGBUSY	RSVD	MGSTAT[1:0]	
W								
Reset	1	0	0	0	0	0	0(1)	01

= Unimplemented or Reserved

Figure 4-11. Flash Status Register (FSTAT)

1. Reset value can deviate from the value shown if a double bit fault is detected during the reset sequence (see Section 4.6).

CCIF, ACCERR, and FPVIOL bits are readable and writable, MGBUSY and MGSTAT bits are readable but not writable, while remaining bits read 0 and are not writable.

Table 4-17. FSTAT Field Descriptions

Field	Description
7 CCIF	**Command Complete Interrupt Flag** — The CCIF flag indicates that a Flash command has completed. The CCIF flag is cleared by writing a 1 to CCIF to launch a command and CCIF will stay low until command completion or command violation. 0 Flash command in progress 1 Flash command has completed
5 ACCERR	**Flash Access Error Flag** — The ACCERR bit indicates an illegal access has occurred to the Flash memory caused by either a violation of the command write sequence (see Section 4.4.1.2) or issuing an illegal Flash command or when errors are encountered while initializing the EEE buffer ram during the reset sequence. While ACCERR is set, the CCIF flag cannot be cleared to launch a command. The ACCERR bit is cleared by writing a 1 to ACCERR. Writing a 0 to the ACCERR bit has no effect on ACCERR. 0 No access error detected 1 Access error detected
4 FPVIOL	**Flash Protection Violation Flag** —The FPVIOL bit indicates an attempt was made to program or erase an address in a protected area of P-Flash memory during a command write sequence. The FPVIOL bit is cleared by writing a 1 to FPVIOL. Writing a 0 to the FPVIOL bit has no effect on FPVIOL. While FPVIOL is set, it is not possible to launch a command or start a command write sequence. 0 No protection violation detected 1 Protection violation detected
3 MGBUSY	**Memory Controller Busy Flag** — The MGBUSY flag reflects the active state of the Memory Controller. 0 Memory Controller is idle 1 Memory Controller is busy executing a Flash command (CCIF = 0) or is handling internal EEE operations
2 RSVD	**Reserved Bit** — This bit is reserved and always reads 0.
1–0 MGSTAT[1:0]	**Memory Controller Command Completion Status Flag** — One or more MGSTAT flag bits are set if an error is detected during execution of a Flash command or during the Flash reset sequence. See Section 4.4.2, “Flash Command Description,” and Section 4.6, “Initialization” for details.

4.3.2.8 Flash Error Status Register (FERSTAT)

The FERSTAT register reflects the error status of internal Flash operations.

Offset Module Base + 0x0007

	7	6	5	4	3	2	1	0
R	ERSERIF	PGMERIF	0	EPVIOLIF	ERSVIF1	ERSVIF0	DFDIF	SFDIF
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-12. Flash Error Status Register (FERSTAT)

All flags in the FERSTAT register are readable and only writable to clear the flag.

Table 4-18. FERSTAT Field Descriptions

Field	Description
7 ERSERIF	**EEE Erase Error Interrupt Flag** — The setting of the ERSERIF flag occurs due to an error in a Flash erase command that resulted in the erase operation not being successful during EEE operations. The ERSERIF flag is cleared by writing a 1 to ERSERIF. Writing a 0 to the ERSERIF flag has no effect on ERSERIF. While ERSERIF is set, it is possible to write to the buffer RAM EEE partition but the data written will not be transferred to the D-Flash EEE partition. 0 Erase command successfully completed on the D-Flash EEE partition 1 Erase command failed on the D-Flash EEE partition
6 PGMERIF	**EEE Program Error Interrupt Flag** — The setting of the PGMERIF flag occurs due to an error in a Flash program command that resulted in the program operation not being successful during EEE operations. The PGMERIF flag is cleared by writing a 1 to PGMERIF. Writing a 0 to the PGMERIF flag has no effect on PGMERIF. While PGMERIF is set, it is possible to write to the buffer RAM EEE partition but the data written will not be transferred to the D-Flash EEE partition. 0 Program command successfully completed on the D-Flash EEE partition 1 Program command failed on the D-Flash EEE partition
4 EPVIOLIF	**EEE Protection Violation Interrupt Flag** —The setting of the EPVIOLIF flag indicates an attempt was made to write to a protected area of the buffer RAM EEE partition. The EPVIOLIF flag is cleared by writing a 1 to EPVIOLIF. Writing a 0 to the EPVIOLIF flag has no effect on EPVIOLIF. While EPVIOLIF is set, it is possible to write to the buffer RAM EEE partition as long as the address written to is not in a protected area. 0 No EEE protection violation 1 EEE protection violation detected
3 ERSVIF1	**EEE Error Interrupt 1 Flag** —The setting of the ERSVIF1 flag indicates that the memory controller was unable to change the state of a D-Flash EEE sector. The ERSVIF1 flag is cleared by writing a 1 to ERSVIF1. Writing a 0 to the ERSVIF1 flag has no effect on ERSVIF1. While ERSVIF1 is set, it is possible to write to the buffer RAM EEE partition but the data written will not be transferred to the D-Flash EEE partition. 0 No EEE sector state change error detected 1 EEE sector state change error detected
2 ERSVIF0	**EEE Error Interrupt 0 Flag** —The setting of the ERSVIF0 flag indicates that the memory controller was unable to format a D-Flash EEE sector for EEE use. The ERSVIF0 flag is cleared by writing a 1 to ERSVIF0. Writing a 0 to the ERSVIF0 flag has no effect on ERSVIF0. While ERSVIF0 is set, it is possible to write to the buffer RAM EEE partition but the data written will not be transferred to the D-Flash EEE partition. 0 No EEE sector format error detected 1 EEE sector format error detected
1 DFDIF	**Double Bit Fault Detect Interrupt Flag** — The setting of the DFDIF flag indicates that a double bit fault was detected in the stored parity and data bits during a Flash array read operation or that a Flash array read operation was attempted on a Flash block that was under a Flash command operation. The DFDIF flag is cleared by writing a 1 to DFDIF. Writing a 0 to DFDIF has no effect on DFDIF. 0 No double bit fault detected 1 Double bit fault detected or an invalid Flash array read operation attempted
0 SFDIF	**Single Bit Fault Detect Interrupt Flag** — With the IGNSF bit in the FCNFG register clear, the SFDIF flag indicates that a single bit fault was detected in the stored parity and data bits during a Flash array read operation or that a Flash array read operation was attempted on a Flash block that was under a Flash command operation. The SFDIF flag is cleared by writing a 1 to SFDIF. Writing a 0 to SFDIF has no effect on SFDIF. 0 No single bit fault detected 1 Single bit fault detected and corrected or an invalid Flash array read operation attempted

4.3.2.9 P-Flash Protection Register (FPROT)

The FPROT register defines which P-Flash sectors are protected against program and erase operations.

Offset Module Base + 0x0008

	7	6	5	4	3	2	1	0
R	FPOPEN	RNV6	FPHDIS	FPHS[1:0]		FPLDIS	FPLS[1:0]	
W								
Reset	F	F	F	F	F	F	F	F

= Unimplemented or Reserved

Figure 4-13. Flash Protection Register (FPROT)

The (unreserved) bits of the FPROT register are writable with the restriction that the size of the protected region can only be increased (see Section 4.3.2.9.1, “P-Flash Protection Restrictions,” and Table 4-23).

During the reset sequence, the FPROT register is loaded with the contents of the P-Flash protection byte in the Flash configuration field at global address 0x7F_FF0C located in P-Flash memory (see Table 4-3) as indicated by reset condition ‘F’ in Figure 4-13. To change the P-Flash protection that will be loaded during the reset sequence, the upper sector of the P-Flash memory must be unprotected, then the P-Flash protection byte must be reprogrammed. If a double bit fault is detected while reading the P-Flash phrase containing the P-Flash protection byte during the reset sequence, the FPOPEN bit will be cleared and remaining bits in the FPROT register will be set to leave the P-Flash memory fully protected.

Trying to alter data in any protected area in the P-Flash memory will result in a protection violation error and the FPVIOL bit will be set in the FSTAT register. The block erase of a P-Flash block is not possible if any of the P-Flash sectors contained in the same P-Flash block are protected.

Table 4-19. FPROT Field Descriptions

Field	Description
7 FPOPEN	**Flash Protection Operation Enable** — The FPOPEN bit determines the protection function for program or erase operations as shown in Table 4-20 for the P-Flash block. 0 When FPOPEN is clear, the FPHDIS and FPLDIS bits define unprotected address ranges as specified by the corresponding FPHS and FPLS bits 1 When FPOPEN is set, the FPHDIS and FPLDIS bits enable protection for the address range specified by the corresponding FPHS and FPLS bits
6 RNV[6]	**Reserved Nonvolatile Bit** — The RNV bit should remain in the erased state for future enhancements.
5 FPHDIS	**Flash Protection Higher Address Range Disable** — The FPHDIS bit determines whether there is a protected/unprotected area in a specific region of the P-Flash memory ending with global address 0x7F_FFFF. 0 Protection/Unprotection enabled 1 Protection/Unprotection disabled
4–3 FPHS[1:0]	**Flash Protection Higher Address Size** — The FPHS bits determine the size of the protected/unprotected area in P-Flash memory as shown inTable 4-21. The FPHS bits can only be written to while the FPHDIS bit is set.
2 FPLDIS	**Flash Protection Lower Address Range Disable** — The FPLDIS bit determines whether there is a protected/unprotected area in a specific region of the P-Flash memory beginning with global address 0x7F_8000. 0 Protection/Unprotection enabled 1 Protection/Unprotection disabled
1–0 FPLS[1:0]	**Flash Protection Lower Address Size** — The FPLS bits determine the size of the protected/unprotected area in P-Flash memory as shown in Table 4-22. The FPLS bits can only be written to while the FPLDIS bit is set.

Table 4-20. P-Flash Protection Function

FPOPEN	FPHDIS	FPLDIS	Function[1]
1	1	1	No P-Flash Protection
1	1	0	Protected Low Range
1	0	1	Protected High Range
1	0	0	Protected High and Low Ranges
0	1	1	Full P-Flash Memory Protected
0	1	0	Unprotected Low Range
0	0	1	Unprotected High Range
0	0	0	Unprotected High and Low Ranges

1. For range sizes, refer to Table 4-21 and Table 4-22.

Table 4-21. P-Flash Protection Higher Address Range

FPHS[1:0]	Global Address Range	Protected Size
00	0x7F_F800–0x7F_FFFF	2 Kbytes
01	0x7F_F000–0x7F_FFFF	4 Kbytes
10	0x7F_E000–0x7F_FFFF	8 Kbytes
11	0x7F_C000–0x7F_FFFF	16 Kbytes

Table 4-22. P-Flash Protection Lower Address Range

FPLS[1:0]	Global Address Range	Protected Size
00	0x7F_8000–0x7F_83FF	1 Kbyte
01	0x7F_8000–0x7F_87FF	2 Kbytes
10	0x7F_8000–0x7F_8FFF	4 Kbytes
11	0x7F_8000–0x7F_9FFF	8 Kbytes

All possible P-Flash protection scenarios are shown in Figure 4-14. Although the protection scheme is loaded from the Flash memory at global address 0x7F_FF0C during the reset sequence, it can be changed by the user. The P-Flash protection scheme can be used by applications requiring reprogramming in single chip mode while providing as much protection as possible if reprogramming is not required.

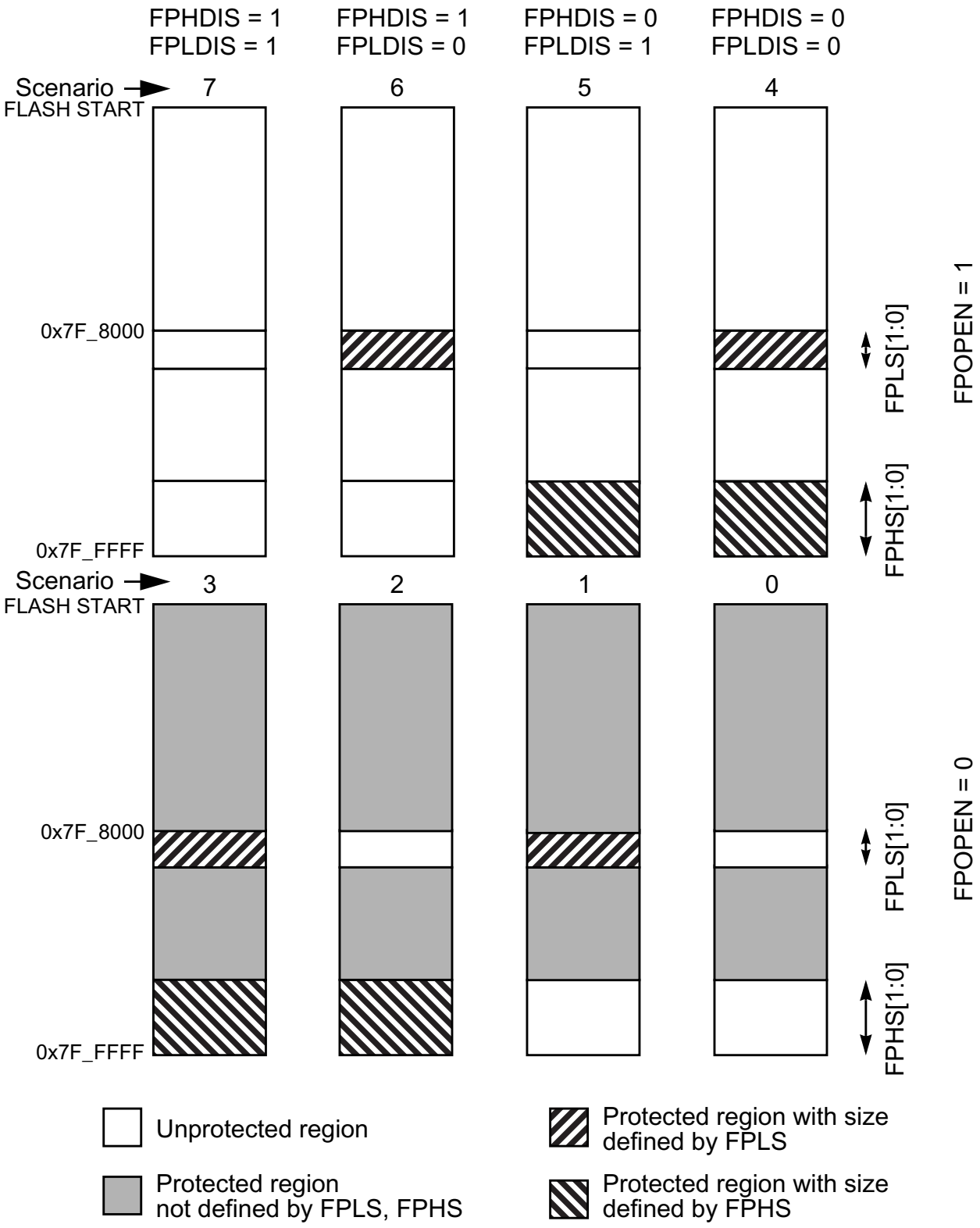


Figure 4-14. P-Flash Protection Scenarios

4.3.2.9.1 P-Flash Protection Restrictions

The general guideline is that P-Flash protection can only be added and not removed. Table 4-23 specifies all valid transitions between P-Flash protection scenarios. Any attempt to write an invalid scenario to the FPROT register will be ignored. The contents of the FPROT register reflect the active protection scenario. See the FPHS and FPLS bit descriptions for additional restrictions.

Table 4-23. P-Flash Protection Scenario Transitions

From Protection Scenario	To Protection Scenario[1]							
	0	1	2	3	4	5	6	7
0	X	X	X	X				
1		X		X				
2			X	X				
3				X				
4				X	X			
5			X	X	X	X		
6		X		X	X		X	
7	X	X	X	X	X	X	X	X

1. Allowed transitions marked with X, see Figure 4-14 for a definition of the scenarios.

4.3.2.10 EEE Protection Register (EPROT)

The EPROT register defines which buffer RAM EEE partition areas are protected against writes.

Offset Module Base + 0x0009

	7	6	5	4	3	2	1	0
R	EPOPEN	RNV[6:4]			EPDIS	EPS[2:0]		
W								
Reset	F	F	F	F	F	F	F	F

= Unimplemented or Reserved

Figure 4-15. EEE Protection Register (EPROT)

All bits in the EPROT register are readable and writable except for RNV[6:4] which are only readable. The EPOPEN and EPDIS bits can only be written to the protected state. The EPS bits can be written anytime until the EPDIS bit is cleared. If the EPOPEN bit is cleared, the state of the EPDIS and EPS bits is irrelevant.

During the reset sequence, the EPROT register is loaded from the EEE protection byte in the Flash configuration field at global address 0x7F_FF0D located in P-Flash memory (see Table 4-3) as indicated by reset condition F in Figure 4-15. To change the EEE protection that will be loaded during the reset sequence, the P-Flash sector containing the EEE protection byte must be unprotected, then the EEE protection byte must be programmed. If a double bit fault is detected while reading the P-Flash phrase

containing the EEE protection byte during the reset sequence, the EPOPEN bit will be cleared and remaining bits in the EPROT register will be set to leave the buffer RAM EEE partition fully protected.

Trying to write data to any protected area in the buffer RAM EEE partition will result in a protection violation error and the EPVIOLIF flag will be set in the FERSTAT register. Trying to write data to any protected area in the buffer RAM partitioned for user access will not be prevented and the EPVIOLIF flag in the FERSTAT register will not set.

Table 4-24. EPROT Field Descriptions

Field	Description
7 EPOPEN	**Enables writes to the Buffer RAM partitioned for EEE** 0 The entire buffer RAM EEE partition is protected from writes 1 Unprotected buffer RAM EEE partition areas are enabled for writes
6–4 RNV[6:4]	**Reserved Nonvolatile Bits** — The RNV bits should remain in the erased state for future enhancements
3 EPDIS	**Buffer RAM Protection Address Range Disable** — The EPDIS bit determines whether there is a protected area in a specific region of the buffer RAM EEE partition. 0 Protection enabled 1 Protection disabled
2–0 EPS[2:0]	**Buffer RAM Protection Size** — The EPS[2:0] bits determine the size of the protected area in the buffer RAM EEE partition as shown inTable 4-21. The EPS bits can only be written to while the EPDIS bit is set.

Table 4-25. Buffer RAM EEE Partition Protection Address Range

EPS[2:0]	Global Address Range	Protected Size
000	0x13_FFC0 – 0x13_FFFF	64 bytes
001	0x13_FF80 – 0x13_FFFF	128 bytes
010	0x13_FF40 – 0x13_FFFF	192 bytes
011	0x13_FF00 – 0x13_FFFF	256 bytes
100	0x13_FEC0 – 0x13_FFFF	320 bytes
101	0x13_FE80 – 0x13_FFFF	384 bytes
110	0x13_FE40 – 0x13_FFFF	448 bytes
111	0x13_FE00 – 0x13_FFFF	512 bytes

4.3.2.11 Flash Common Command Object Register (FCCOB)

The FCCOB is an array of six words addressed via the CCOBIX index found in the FCCOBIX register. Byte wide reads and writes are allowed to the FCCOB register.

Offset Module Base + 0x000A

	7	6	5	4	3	2	1	0
R W	CCOB[15:8]							
Reset	0	0	0	0	0	0	0	0

Figure 4-16. Flash Common Command Object High Register (FCCOBHI)

Offset Module Base + 0x000B

	7	6	5	4	3	2	1	0
R W	CCOB[7:0]							
Reset	0	0	0	0	0	0	0	0

Figure 4-17. Flash Common Command Object Low Register (FCCOBLO)

4.3.2.11.1 FCCOB - NVM Command Mode

NVM command mode uses the indexed FCCOB register to provide a command code and its relevant parameters to the Memory Controller. The user first sets up all required FCCOB fields and then initiates the command's execution by writing a 1 to the CCIF bit in the FSTAT register (a 1 written by the user clears the CCIF command completion flag to 0). When the user clears the CCIF bit in the FSTAT register all FCCOB parameter fields are locked and cannot be changed by the user until the command completes (as evidenced by the Memory Controller returning CCIF to 1). Some commands return information to the FCCOB register array.

The generic format for the FCCOB parameter fields in NVM command mode is shown in Table 4-26. The return values are available for reading after the CCIF flag in the FSTAT register has been returned to 1 by the Memory Controller. Writes to the unimplemented parameter fields (CCOBIX = 110 and CCOBIX = 111) are ignored with reads from these fields returning 0x0000.

Table 4-26 shows the generic Flash command format. The high byte of the first word in the CCOB array contains the command code, followed by the parameters for this specific Flash command. For details on the FCCOB settings required by each command, see the Flash command descriptions in Section 4.4.2.

Table 4-26. FCCOB - NVM Command Mode (Typical Usage)

CCOBIX[2:0]	Byte	FCCOB Parameter Fields (NVM Command Mode)
000	HI	FCMD[7:0] defining Flash command
	LO	0, Global address [22:16]
001	HI	Global address [15:8]
	LO	Global address [7:0]
010	HI	Data 0 [15:8]
	LO	Data 0 [7:0]

Table 4-26. FCCOB - NVM Command Mode (Typical Usage)

CCOBIX[2:0]	Byte	FCCOB Parameter Fields (NVM Command Mode)
011	HI	Data 1 [15:8]
	LO	Data 1 [7:0]
100	HI	Data 2 [15:8]
	LO	Data 2 [7:0]
101	HI	Data 3 [15:8]
	LO	Data 3 [7:0]

4.3.2.12 EEE Tag Counter Register (ETAG)

The ETAG register contains the number of outstanding words in the buffer RAM EEE partition that need to be programmed into the D-Flash EEE partition. The ETAG register is decremented prior to the related tagged word being programmed into the D-Flash EEE partition. All tagged words have been programmed into the D-Flash EEE partition once all bits in the ETAG register read 0 and the MGBUSY flag in the FSTAT register reads 0.

Offset Module Base + 0x000C

	7	6	5	4	3	2	1	0
R	ETAG[15:8]							
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-18. EEE Tag Counter High Register (ETAGHI)

Offset Module Base + 0x000D

	7	6	5	4	3	2	1	0
R	ETAG[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-19. EEE Tag Counter Low Register (ETAGLO)

All ETAG bits are readable but not writable and are cleared by the Memory Controller.

4.3.2.13 Flash ECC Error Results Register (FECCR)

The FECCR registers contain the result of a detected ECC fault for both single bit and double bit faults. The FECCR register provides access to several ECC related fields as defined by the ECCRIX index bits in the FECCRIX register (see Section 4.3.2.4). Once ECC fault information has been stored, no other fault

information will be recorded until the specific ECC fault flag has been cleared. In the event of simultaneous ECC faults, the priority for fault recording is:

1. Double bit fault over single bit fault
2. CPU over XGATE

Offset Module Base + 0x000E

	7	6	5	4	3	2	1	0
R	ECCR[15:8]							
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-20. Flash ECC Error Results High Register (FECCRHI)

Offset Module Base + 0x000F

	7	6	5	4	3	2	1	0
R	ECCR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-21. Flash ECC Error Results Low Register (FECCRLO)

All FECCR bits are readable but not writable.

Table 4-27. FECCR Index Settings

ECCRIX[2:0]	FECCR Register Content		
	Bits [15:8]	**Bit[7]**	**Bits[6:0]**
000	Parity bits read from Flash block	CPU or XGATE source identity	Global address [22:16]
001	Global address [15:0]		
010	Data 0 [15:0]		
011	Data 1 [15:0] (P-Flash only)		
100	Data 2 [15:0] (P-Flash only)		
101	Data 3 [15:0] (P-Flash only)		
110	Not used, returns 0x0000 when read		
111	Not used, returns 0x0000 when read		

Table 4-28. FECCR Index=000 Bit Descriptions

Field	Description
15:8 PAR[7:0]	**ECC Parity Bits** — Contains the 8 parity bits from the 72 bit wide P-Flash data word or the 6 parity bits, allocated to PAR[5:0], from the 22 bit wide D-Flash word with PAR[7:6]=00.
7 XBUS01	**Bus Source Identifier** — The XBUS01 bit determines whether the ECC error was caused by a read access from the CPU or XGATE. 0 ECC Error happened on the CPU access 1 ECC Error happened on the XGATE access
6–0 GADDR[22:16]	**Global Address** — The GADDR[22:16] field contains the upper seven bits of the global address having caused the error.

The P-Flash word addressed by ECCRIX = 001 contains the lower 16 bits of the global address. The following four words addressed by ECCRIX = 010 to 101 contain the 64-bit wide data phrase. The four data words and the parity byte are the uncorrected data read from the P-Flash block.

The D-Flash word addressed by ECCRIX = 001 contains the lower 16 bits of the global address. The uncorrected 16-bit data word is addressed by ECCRIX = 010.

4.3.2.14 Flash Option Register (FOPT)

The FOPT register is the Flash option register.

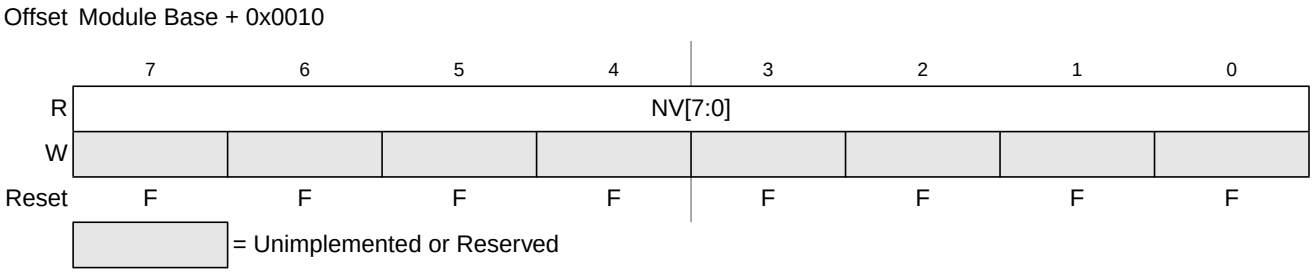


Figure 4-22. Flash Option Register (FOPT)

All bits in the FOPT register are readable but are not writable.

During the reset sequence, the FOPT register is loaded from the Flash nonvolatile byte in the Flash configuration field at global address 0x7F_FF0E located in P-Flash memory (see Table 4-3) as indicated by reset condition F in Figure 4-22. If a double bit fault is detected while reading the P-Flash phrase containing the Flash nonvolatile byte during the reset sequence, all bits in the FOPT register will be set.

Table 4-29. FOPT Field Descriptions

Field	Description
7–0 NV[7:0]	**Nonvolatile Bits** — The NV[7:0] bits are available as nonvolatile bits. Refer to the device user guide for proper use of the NV bits.

4.3.2.15 Flash Reserved0 Register (FRSV0)

This Flash register is reserved for factory testing.

Offset Module Base + 0x0011

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-23. Flash Reserved0 Register (FRSV0)

All bits in the FRSV0 register read 0 and are not writable.

4.3.2.16 Flash Reserved1 Register (FRSV1)

This Flash register is reserved for factory testing.

Offset Module Base + 0x0012

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-24. Flash Reserved1 Register (FRSV1)

All bits in the FRSV1 register read 0 and are not writable.

4.3.2.17 Flash Reserved2 Register (FRSV2)

This Flash register is reserved for factory testing.

Offset Module Base + 0x0013

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 4-25. Flash Reserved2 Register (FRSV2)

All bits in the FRSV2 register read 0 and are not writable.

4.4 Functional Description

4.4.1 Flash Command Operations

Flash command operations are used to modify Flash memory contents or configure module resources for EEE operation.

The next sections describe:

- How to write the FCLKDIV register that is used to generate a time base (FCLK) derived from OSCCLK for Flash program and erase command operations
- The command write sequence used to set Flash command parameters and launch execution
- Valid Flash commands available for execution

4.4.1.1 Writing the FCLKDIV Register

Prior to issuing any Flash program or erase command after a reset, the user is required to write the FCLKDIV register to divide OSCCLK down to a target FCLK of 1 MHz. Table 4-9 shows recommended values for the FDIV field based on OSCCLK frequency.

NOTE

Programming or erasing the Flash memory cannot be performed if the bus clock runs at less than 1 MHz. Setting FDIV too high can destroy the Flash memory due to overstress. Setting FDIV too low can result in incomplete programming or erasure of the Flash memory cells.

When the FCLKDIV register is written, the FDIVLD bit is set automatically. If the FDIVLD bit is 0, the FCLKDIV register has not been written since the last reset. If the FCLKDIV register has not been written, any Flash program or erase command loaded during a command write sequence will not execute and the ACCERR bit in the FSTAT register will set.

4.4.1.2 Command Write Sequence

The Memory Controller will launch all valid Flash commands entered using a command write sequence.

Before launching a command, the ACCERR and FPVIOL bits in the FSTAT register must be clear (see Section 4.3.2.7) and the CCIF flag should be tested to determine the status of the current command write sequence. If CCIF is 0, the previous command write sequence is still active, a new command write sequence cannot be started, and all writes to the FCCOB register are ignored.

CAUTION

Writes to any Flash register must be avoided while a Flash command is active (CCIF=0) to prevent corruption of Flash register contents and Memory Controller behavior.

4.4.1.2.1 Define FCCOB Contents

The FCCOB parameter fields must be loaded with all required parameters for the Flash command being executed. Access to the FCCOB parameter fields is controlled via the CCOBIX bits in the FCCOBIX register (see Section 4.3.2.3).

The contents of the FCCOB parameter fields are transferred to the Memory Controller when the user clears the CCIF command completion flag in the FSTAT register (writing 1 clears the CCIF to 0). The CCIF flag will remain clear until the Flash command has completed. Upon completion, the Memory Controller will return CCIF to 1 and the FCCOB register will be used to communicate any results. The flow for a generic command write sequence is shown in Figure 4-26.

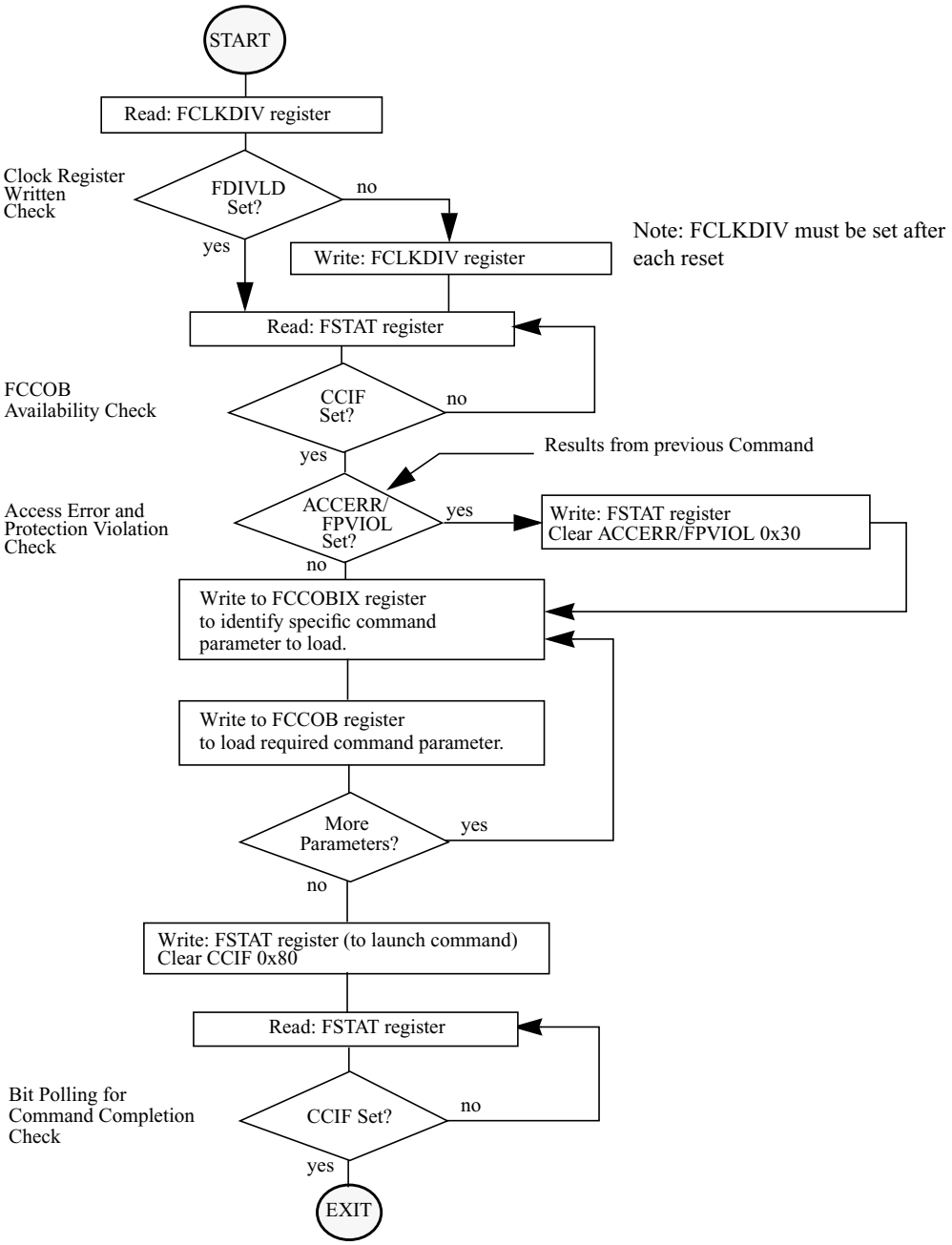


Figure 4-26. Generic Flash Command Write Sequence Flowchart

4.4.1.3 Valid Flash Module Commands

Table 4-30. Flash Commands by Mode

FCMD	Command	Unsecured				Secured			
		NS (1)	NX (2)	SS(3)	ST(4)	NS (5)	NX (6)	SS(7)	ST(8)
0x01	Erase Verify All Blocks	*	*	*	*	*	*	*	*
0x02	Erase Verify Block	*	*	*	*	*	*	*	*
0x03	Erase Verify P-Flash Section	*	*	*	*	*			
0x04	Read Once	*	*	*	*	*			
0x05	Load Data Field	*	*	*	*	*			
0x06	Program P-Flash	*	*	*	*	*			
0x07	Program Once	*	*	*	*	*			
0x08	Erase All Blocks			*	*			*	*
0x09	Erase P-Flash Block	*	*	*	*	*			
0x0A	Erase P-Flash Sector	*	*	*	*	*			
0x0B	Unsecure Flash			*	*			*	*
0x0C	Verify Backdoor Access Key	*				*			
0x0D	Set User Margin Level	*	*	*	*	*			
0x0E	Set Field Margin Level			*	*				
0x0F	Full Partition D-Flash			*	*				
0x10	Erase Verify D-Flash Section	*	*	*	*	*			
0x11	Program D-Flash	*	*	*	*	*			
0x12	Erase D-Flash Sector	*	*	*	*	*			
0x13	Enable EEPROM Emulation	*	*	*	*	*	*	*	*
0x14	Disable EEPROM Emulation	*	*	*	*	*	*	*	*
0x15	EEPROM Emulation Query	*	*	*	*	*	*	*	*
0x20	Partition D-Flash	*	*	*	*	*	*	*	*

1. Unsecured Normal Single Chip mode.
2. Unsecured Normal Expanded mode.
3. Unsecured Special Single Chip mode.
4. Unsecured Special Mode.
5. Secured Normal Single Chip mode.
6. Secured Normal Expanded mode.
7. Secured Special Single Chip mode.
8. Secured Special Mode.

4.4.1.4 P-Flash Commands

Table 4-31 summarizes the valid P-Flash commands along with the effects of the commands on the P-Flash block and other resources within the Flash module.

Table 4-31. P-Flash Commands

FCMD	Command	Function on P-Flash Memory
0x01	Erase Verify All Blocks	Verify that all P-Flash (and D-Flash) blocks are erased.
0x02	Erase Verify Block	Verify that a P-Flash block is erased.
0x03	Erase Verify P-Flash Section	Verify that a given number of words starting at the address provided are erased.
0x04	Read Once	Read a dedicated 64 byte field in the nonvolatile information register in P-Flash block 0 that was previously programmed using the Program Once command.
0x05	Load Data Field	Load data for simultaneous multiple P-Flash block operations.
0x06	Program P-Flash	Program a phrase in a P-Flash block and any previously loaded phrases for any other P-Flash block (see Load Data Field command).
0x07	Program Once	Program a dedicated 64 byte field in the nonvolatile information register in P-Flash block 0 that is allowed to be programmed only once.
0x08	Erase All Blocks	Erase all P-Flash (and D-Flash) blocks. An erase of all Flash blocks is only possible when the FPLDIS, FPHDIS, and FPOPEN bits in the FPROT register and the EPDIS and EPOPEN bits in the EPROT register are set prior to launching the command.
0x09	Erase P-Flash Block	Erase a single P-Flash block. An erase of the full P-Flash block is only possible when FPLDIS, FPHDIS and FPOPEN bits in the FPROT register are set prior to launching the command.
0x0A	Erase P-Flash Sector	Erase all bytes in a P-Flash sector.
0x0B	Unsecure Flash	Supports a method of releasing MCU security by erasing all P-Flash (and D-Flash) blocks and verifying that all P-Flash (and D-Flash) blocks are erased.
0x0C	Verify Backdoor Access Key	Supports a method of releasing MCU security by verifying a set of security keys.
0x0D	Set User Margin Level	Specifies a user margin read level for all P-Flash blocks.
0x0E	Set Field Margin Level	Specifies a field margin read level for all P-Flash blocks (special modes only).

4.4.1.5 D-Flash and EEE Commands

Table 4-32 summarizes the valid D-Flash and EEE commands along with the effects of the commands on the D-Flash block and EEE operation.

Table 4-32. D-Flash Commands

FCMD	Command	Function on D-Flash Memory
0x01	Erase Verify All Blocks	Verify that all D-Flash (and P-Flash) blocks are erased.
0x02	Erase Verify Block	Verify that the D-Flash block is erased.

Table 4-32. D-Flash Commands

FCMD	Command	Function on D-Flash Memory
0x08	Erase All Blocks	Erase all D-Flash (and P-Flash) blocks. An erase of all Flash blocks is only possible when the FPLDIS, FPHDIS, and FPOPEN bits in the FPROT register and the EPDIS and EPOPEN bits in the EPROT register are set prior to launching the command.
0x0B	Unsecure Flash	Supports a method of releasing MCU security by erasing all D-Flash (and P-Flash) blocks and verifying that all D-Flash (and P-Flash) blocks are erased.
0x0D	Set User Margin Level	Specifies a user margin read level for the D-Flash block.
0x0E	Set Field Margin Level	Specifies a field margin read level for the D-Flash block (special modes only).
0x0F	Full Partition D-Flash	Erase the D-Flash block and partition an area of the D-Flash block for user access.
0x10	Erase Verify D-Flash Section	Verify that a given number of words starting at the address provided are erased.
0x11	Program D-Flash	Program up to four words in the D-Flash block.
0x12	Erase D-Flash Sector	Erase all bytes in a sector of the D-Flash block.
0x13	Enable EEPROM Emulation	Enable EEPROM emulation where writes to the buffer RAM EEE partition will be copied to the D-Flash EEE partition.
0x14	Disable EEPROM Emulation	Suspend all current erase and program activity related to EEPROM emulation but leave current EEE tags set.
0x15	EEPROM Emulation Query	Returns EEE partition and status variables.
0x20	Partition D-Flash	Partition an area of the D-Flash block for user access.

4.4.2 Flash Command Description

This section provides details of all available Flash commands launched by a command write sequence. The ACCERR bit in the FSTAT register will be set during the command write sequence if any of the following illegal steps are performed, causing the command not to be processed by the Memory Controller:

- Starting any command write sequence that programs or erases Flash memory before initializing the FCLKDIV register
- Writing an invalid command as part of the command write sequence
- For additional possible errors, refer to the error handling table provided for each command

If a Flash block is read during execution of an algorithm (CCIF = 0) on that same block, the read operation will return invalid data. If the SFDIF or DFDIF flags were not previously set when the invalid read operation occurred, both the SFDIF and DFDIF flags will be set and the FECCR registers will be loaded with the global address used in the invalid read operation with the data and parity fields set to all 0.

If the ACCERR or FPVIOL bits are set in the FSTAT register, the user must clear these bits before starting any command write sequence (see Section 4.3.2.7).

CAUTION

A Flash word or phrase must be in the erased state before being programmed. Cumulative programming of bits within a Flash word or phrase is not allowed.

4.4.2.1 Erase Verify All Blocks Command

The Erase Verify All Blocks command will verify that all P-Flash and D-Flash blocks have been erased.

Table 4-33. Erase Verify All Blocks Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x01	Not required

Upon clearing CCIF to launch the Erase Verify All Blocks command, the Memory Controller will verify that the entire Flash memory space is erased. The CCIF flag will set after the Erase Verify All Blocks operation has completed.

Table 4-34. Erase Verify All Blocks Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if a Load Data Field command sequence is currently active
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read(1)
	MGSTAT0	Set if any non-correctable errors have been encountered during the read[1]
FERSTAT	EPVIOLIF	None

1. As found in the memory map for FTM512K3.

4.4.2.2 Erase Verify Block Command

The Erase Verify Block command allows the user to verify that an entire P-Flash or D-Flash block has been erased. The FCCOB upper global address bits determine which block must be verified.

Table 4-35. Erase Verify Block Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x02	Global address [22:16] of the Flash block to be verified.

Upon clearing CCIF to launch the Erase Verify Block command, the Memory Controller will verify that the selected P-Flash or D-Flash block is erased. The CCIF flag will set after the Erase Verify Block operation has completed.

Table 4-36. Erase Verify Block Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if an invalid global address [22:16] is supplied(1)
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read(2)
	MGSTAT0	Set if any non-correctable errors have been encountered during the read2
FERSTAT	EPVIOLIF	None

1. As defined by the memory map for FTM512K3.
2. As found in the memory map for FTM512K3.

4.4.2.3 Erase Verify P-Flash Section Command

The Erase Verify P-Flash Section command will verify that a section of code in the P-Flash memory is erased. The Erase Verify P-Flash Section command defines the starting point of the code to be verified and the number of phrases. The section to be verified cannot cross a 256 Kbyte boundary in the P-Flash memory space.

Table 4-37. Erase Verify P-Flash Section Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x03	Global address [22:16] of a P-Flash block
001	Global address [15:0] of the first phrase to be verified	
010	Number of phrases to be verified	

Upon clearing CCIF to launch the Erase Verify P-Flash Section command, the Memory Controller will verify the selected section of Flash memory is erased. The CCIF flag will set after the Erase Verify P-Flash Section operation has completed.

Table 4-38. Erase Verify P-Flash Section Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 010 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 4-30)
		Set if an invalid global address [22:0] is supplied(1)
		Set if a misaligned phrase address is supplied (global address [2:0] != 000)
		Set if the requested section crosses a 256 Kbyte boundary
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read(2)
	MGSTAT0	Set if any non-correctable errors have been encountered during the read2
FERSTAT	EPVIOLIF	None

1. As defined by the memory map for FTM512K3.
2. As found in the memory map for FTM512K3.

4.4.2.4 Read Once Command

The Read Once command provides read access to a reserved 64 byte field (8 phrases) located in the nonvolatile information register of P-Flash block 0. The Read Once field is programmed using the Program Once command described in Section 4.4.2.7. The Read Once command must not be executed from the Flash block containing the Program Once reserved field to avoid code runaway.

Table 4-39. Read Once Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x04	Not Required
001	Read Once phrase index (0x0000 - 0x0007)	
010	Read Once word 0 value	
011	Read Once word 1 value	
100	Read Once word 2 value	
101	Read Once word 3 value	

Upon clearing CCIF to launch the Read Once command, a Read Once phrase is fetched and stored in the FCCOB indexed register. The CCIF flag will set after the Read Once operation has completed. Valid phrase index values for the Read Once command range from 0x0000 to 0x0007. During execution of the Read Once command, any attempt to read addresses within P-Flash block 0 will return invalid data.

Table 4-40. Read Once Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 4-30)
		Set if an invalid phrase index is supplied
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read
	MGSTAT0	Set if any non-correctable errors have been encountered during the read
FERSTAT	EPVIOLIF	None

4.4.2.5 Load Data Field Command

The Load Data Field command is executed to provide FCCOB parameters for multiple P-Flash blocks for a future simultaneous program operation in the P-Flash memory space.

Table 4-41. Load Data Field Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x05	Global address [22:16] to identify P-Flash block
001	Global address [15:0] of phrase location to be programmed(1)	
010	Word 0	
011	Word 1	
100	Word 2	
101	Word 3	

1. Global address [2:0] must be 000

Upon clearing CCIF to launch the Load Data Field command, the FCCOB registers will be transferred to the Memory Controller and be programmed in the block specified at the global address given with a future Program P-Flash command launched on a P-Flash block. The CCIF flag will set after the Load Data Field operation has completed. Note that once a Load Data Field command sequence has been initiated, the Load Data Field command sequence will be cancelled if any command other than Load Data Field or the future Program P-Flash is launched. Similarly, if an error occurs after launching a Load Data Field or Program P-Flash command, the associated Load Data Field command sequence will be cancelled.

Table 4-42. Load Data Field Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 101 at command launch
		Set if command not available in current mode (see Table 4-30)
		Set if an invalid global address [22:0] is supplied(1)
		Set if a misaligned phrase address is supplied (global address [2:0] != 000)
		Set if a Load Data Field command sequence is currently active and the selected block has previously been selected in the same command sequence
		Set if a Load Data Field command sequence is currently active and global address [17:0] does not match that previously supplied in the same command sequence
	FPVIOL	Set if the global address [22:0] points to a protected area
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

1. As defined by the memory map for FTM512K3.

4.4.2.6 Program P-Flash Command

The Program P-Flash operation will program a previously erased phrase in the P-Flash memory using an embedded algorithm.

CAUTION

A P-Flash phrase must be in the erased state before being programmed. Cumulative programming of bits within a Flash phrase is not allowed.

Table 4-43. Program P-Flash Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x06	Global address [22:16] to identify P-Flash block
001	Global address [15:0] of phrase location to be programmed(1)	
010	Word 0 program value	
011	Word 1 program value	
100	Word 2 program value	
101	Word 3 program value	

1. Global address [2:0] must be 000

Upon clearing CCIF to launch the Program P-Flash command, the Memory Controller will program the data words to the supplied global address and will then proceed to verify the data words read back as expected. The CCIF flag will set after the Program P-Flash operation has completed.

Table 4-44. Program P-Flash Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 101 at command launch
		Set if command not available in current mode (see Table 4-30)
		Set if an invalid global address [22:0] is supplied(1)
		Set if a misaligned phrase address is supplied (global address [2:0] != 000)
		Set if a Load Data Field command sequence is currently active and the selected block has previously been selected in the same command sequence
		Set if a Load Data Field command sequence is currently active and global address [17:0] does not match that previously supplied in the same command sequence
	FPVIOL	Set if the global address [22:0] points to a protected area
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation
FERSTAT	EPVIOLIF	None

1. As defined by the memory map for FTM512K3.

4.4.2.7 Program Once Command

The Program Once command restricts programming to a reserved 64 byte field (8 phrases) in the nonvolatile information register located in P-Flash block 0. The Program Once reserved field can be read using the Read Once command as described in Section 4.4.2.4. The Program Once command must only be issued once since the nonvolatile information register in P-Flash block 0 cannot be erased. The Program Once command must not be executed from the Flash block containing the Program Once reserved field to avoid code runaway.

Table 4-45. Program Once Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x07	Not Required
001	Program Once phrase index (0x0000 - 0x0007)	
010	Program Once word 0 value	
011	Program Once word 1 value	
100	Program Once word 2 value	
101	Program Once word 3 value	

Upon clearing CCIF to launch the Program Once command, the Memory Controller first verifies that the selected phrase is erased. If erased, then the selected phrase will be programmed and then verified with read back. The CCIF flag will remain clear, setting only after the Program Once operation has completed.

The reserved nonvolatile information register accessed by the Program Once command cannot be erased and any attempt to program one of these phrases a second time will not be allowed. Valid phrase index

values for the Program Once command range from 0x0000 to 0x0007. During execution of the Program Once command, any attempt to read addresses within P-Flash block 0 will return invalid data.

Table 4-46. Program Once Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 101 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 4-30)
		Set if an invalid phrase index is supplied
		Set if the requested phrase has already been programmed(1)
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation
FERSTAT	EPVIOLIF	None

1. If a Program Once phrase is initially programmed to 0xFFFF_FFFF_FFFF_FFFF, the Program Once command will be allowed to execute again on that same phrase.

4.4.2.8 Erase All Blocks Command

The Erase All Blocks operation will erase the entire P-Flash and D-Flash memory space including the EEE nonvolatile information register.

Table 4-47. Erase All Blocks Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x08	Not required

Upon clearing CCIF to launch the Erase All Blocks command, the Memory Controller will erase the entire Flash memory space and verify that it is erased. If the Memory Controller verifies that the entire Flash memory space was properly erased, security will be released. During the execution of this command (CCIF=0) the user must not write to any Flash module register. The CCIF flag will set after the Erase All Blocks operation has completed.

Table 4-48. Erase All Blocks Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 4-30)
	FPVIOL	Set if any area of the P-Flash memory is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation(1)
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation[1]
FERSTAT	EPVIOLIF	Set if any area of the buffer RAM EEE partition is protected

1. As found in the memory map for FTM512K3.

4.4.2.9 Erase P-Flash Block Command

The Erase P-Flash Block operation will erase all addresses in a P-Flash block.

Table 4-49. Erase P-Flash Block Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x09	Global address [22:16] to identify P-Flash block
001	Global address [15:0] in P-Flash block to be erased	

Upon clearing CCIF to launch the Erase P-Flash Block command, the Memory Controller will erase the selected P-Flash block and verify that it is erased. The CCIF flag will set after the Erase P-Flash Block operation has completed.

Table 4-50. Erase P-Flash Block Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 4-30)
		Set if an invalid global address [22:16] is supplied(1)
	FPVIOL	Set if an area of the selected P-Flash block is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation(2)
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation[2]
FERSTAT	EPVIOLIF	None

1. As defined by the memory map for FTM512K3.
2. As found in the memory map for FTM512K3.

4.4.2.10 Erase P-Flash Sector Command

The Erase P-Flash Sector operation will erase all addresses in a P-Flash sector.

Table 4-51. Erase P-Flash Sector Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x0A	Global address [22:16] to identify P-Flash block to be erased
001	Global address [15:0] anywhere within the sector to be erased. Refer to Section 4.1.2.1 for the P-Flash sector size.	

Upon clearing CCIF to launch the Erase P-Flash Sector command, the Memory Controller will erase the selected Flash sector and then verify that it is erased. The CCIF flag will be set after the Erase P-Flash Sector operation has completed.

Table 4-52. Erase P-Flash Sector Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 4-30)
		Set if an invalid global address [22:16] is supplied(1)
		Set if a misaligned phrase address is supplied (global address [2:0] != 000)
	FPVIOL	Set if the selected P-Flash sector is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation
FERSTAT	EPVIOLIF	None

1. As defined by the memory map for FTM512K3.

4.4.2.11 Unsecure Flash Command

The Unsecure Flash command will erase the entire P-Flash and D-Flash memory space and, if the erase is successful, will release security.

Table 4-53. Unsecure Flash Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x0B	Not required

Upon clearing CCIF to launch the Unsecure Flash command, the Memory Controller will erase the entire P-Flash and D-Flash memory space and verify that it is erased. If the Memory Controller verifies that the entire Flash memory space was properly erased, security will be released. If the erase verify is not successful, the Unsecure Flash operation sets MGSTAT1 and terminates without changing the security

state. During the execution of this command (CCIF=0) the user must not write to any Flash module register. The CCIF flag is set after the Unsecure Flash operation has completed.

Table 4-54. Unsecure Flash Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 4-30)
	FPVIOL	Set if any area of the P-Flash memory is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation(1)
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation[1]
FERSTAT	EPVIOLIF	Set if any area of the buffer RAM EEE partition is protected

1. As found in the memory map for FTM512K3.

4.4.2.12 Verify Backdoor Access Key Command

The Verify Backdoor Access Key command will only execute if it is enabled by the KEYEN bits in the FSEC register (see Table 4-11). The Verify Backdoor Access Key command releases security if user-supplied keys match those stored in the Flash security bytes of the Flash configuration field (see Table 4-3). The Verify Backdoor Access Key command must not be executed from the Flash block containing the backdoor comparison key to avoid code runaway.

Table 4-55. Verify Backdoor Access Key Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x0C	Not required
001	Key 0	
010	Key 1	
011	Key 2	
100	Key 3	

Upon clearing CCIF to launch the Verify Backdoor Access Key command, the Memory Controller will check the FSEC KEYEN bits to verify that this command is enabled. If not enabled, the Memory Controller sets the ACCERR bit in the FSTAT register and terminates. If the command is enabled, the Memory Controller compares the key provided in FCCOB to the backdoor comparison key in the Flash configuration field with Key 0 compared to 0x7F_FF00, etc. If the backdoor keys match, security will be released. If the backdoor keys do not match, security is not released and all future attempts to execute the Verify Backdoor Access Key command are aborted (set ACCERR) until a reset occurs. The CCIF flag is set after the Verify Backdoor Access Key operation has completed.

Table 4-56. Verify Backdoor Access Key Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 100 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if an incorrect backdoor key is supplied
		Set if backdoor key access has not been enabled (KEYEN[1:0] != 10, see Section 4.3.2.2)
		Set if the backdoor key has mismatched since the last reset
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

4.4.2.13 Set User Margin Level Command

The Set User Margin Level command causes the Memory Controller to set the margin level for future read operations of a specific P-Flash or D-Flash block.

Table 4-57. Set User Margin Level Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x0D	Global address [22:16] to identify the Flash block
001	Margin level setting	

Upon clearing CCIF to launch the Set User Margin Level command, the Memory Controller will set the user margin level for the targeted block and then set the CCIF flag.

Valid margin level settings for the Set User Margin Level command are defined in Table 4-58.

Table 4-58. Valid Set User Margin Level Settings

CCOB (CCOBIX=001)	Level Description
0x0000	Return to Normal Level
0x0001	User Margin-1 Level(1)
0x0002	User Margin-0 Level(2)

1. Read margin to the erased state
2. Read margin to the programmed state

Table 4-59. Set User Margin Level Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 4-30)
		Set if an invalid global address [22:16] is supplied(1)
		Set if an invalid margin level setting is supplied
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

1. As defined by the memory map for FTM512K3.

NOTE

User margin levels can be used to check that Flash memory contents have adequate margin for normal level read operations. If unexpected results are encountered when checking Flash memory contents at user margin levels, a potential loss of information has been detected.

4.4.2.14 Set Field Margin Level Command

The Set Field Margin Level command, valid in special modes only, causes the Memory Controller to set the margin level specified for future read operations of a specific P-Flash or D-Flash block.

Table 4-60. Set Field Margin Level Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x0E	Global address [22:16] to identify the Flash block
001	Margin level setting	

Upon clearing CCIF to launch the Set Field Margin Level command, the Memory Controller will set the field margin level for the targeted block and then set the CCIF flag.

Valid margin level settings for the Set Field Margin Level command are defined in Table 4-61.

Table 4-61. Valid Set Field Margin Level Settings

CCOB (CCOBIX=001)	Level Description
0x0000	Return to Normal Level
0x0001	User Margin-1 Level(1)

Table 4-61. Valid Set Field Margin Level Settings

CCOB (CCOBIX=001)	Level Description
0x0002	User Margin-0 Level(2)
0x0003	Field Margin-1 Level1
0x0004	Field Margin-0 Level2

1. Read margin to the erased state
2. Read margin to the programmed state

Table 4-62. Set Field Margin Level Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 4-30)
		Set if an invalid global address [22:16] is supplied(1)
		Set if an invalid margin level setting is supplied
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

1. As defined by the memory map for FTM512K3.

CAUTION

Field margin levels must only be used during verify of the initial factory programming.

NOTE

Field margin levels can be used to check that Flash memory contents have adequate margin for data retention at the normal level setting. If unexpected results are encountered when checking Flash memory contents at field margin levels, the Flash memory contents should be erased and reprogrammed.

4.4.2.15 Full Partition D-Flash Command

The Full Partition D-Flash command allows the user to allocate sectors within the D-Flash block for applications and a partition within the buffer RAM for EEPROM access. The D-Flash block consists of 128 sectors with 256 bytes per sector.

Table 4-63. Full Partition D-Flash Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x0F	Not required
001	Number of 256 byte sectors for the D-Flash user partition (DFPART)	
010	Number of 256 byte sectors for buffer RAM EEE partition (ERPART)	

Upon clearing CCIF to launch the Full Partition D-Flash command, the following actions are taken to define a partition within the D-Flash block for direct access (DFPART) and a partition within the buffer RAM for EEE use (ERPART):

- Validate the DFPART and ERPART values provided:
  - DFPART <= 128 (maximum number of 256 byte sectors in D-Flash block)
  - ERPART <= 16 (maximum number of 256 byte sectors in buffer RAM)
  - If ERPART > 0, 128 - DFPART >= 12 (minimum number of 256 byte sectors in the D-Flash block required to support EEE)
  - If ERPART > 0, ((128-DFPART)/ERPART) >= 8 (minimum ratio of D-Flash EEE space to buffer RAM EEE space to support EEE)
- Erase the D-Flash block and the EEE nonvolatile information register
- Program DFPART to the EEE nonvolatile information register at global address 0x12_0000 (see Table 4-7)
- Program a duplicate DFPART to the EEE nonvolatile information register at global address 0x12_0002 (see Table 4-7)
- Program ERPART to the EEE nonvolatile information register at global address 0x12_0004 (see Table 4-7)
- Program a duplicate ERPART to the EEE nonvolatile information register at global address 0x12_0006 (see Table 4-7)

The D-Flash user partition will start at global address 0x10_0000. The buffer RAM EEE partition will end at global address 0x13_FFFF. After the Full Partition D-Flash operation has completed, the CCIF flag will set.

Running the Full Partition D-Flash command a second time will result in the previous partition values and the entire D-Flash memory being erased. The data value written corresponds to the number of 256 byte sectors allocated for either direct D-Flash access (DFPART) or buffer RAM EEE access (ERPART).

Table 4-64. Full Partition D-Flash Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 010 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 4-30)
		Set if an invalid DFPART or ERPART selection is supplied
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read
	MGSTAT0	Set if any non-correctable errors have been encountered during the read
FERSTAT	EPVIOLIF	None

4.4.2.16 Erase Verify D-Flash Section Command

The Erase Verify D-Flash Section command will verify that a section of code in the D-Flash user partition is erased. The Erase Verify D-Flash Section command defines the starting point of the data to be verified and the number of words.

Table 4-65. Erase Verify D-Flash Section Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x10	Global address [22:16] to identify the D-Flash block
001	Global address [15:0] of the first word to be verified	
010	Number of words to be verified	

Upon clearing CCIF to launch the Erase Verify D-Flash Section command, the Memory Controller will verify the selected section of D-Flash memory is erased. The CCIF flag will set after the Erase Verify D-Flash Section operation has completed.

Table 4-66. Erase Verify D-Flash Section Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 010 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 4-30)
		Set if an invalid global address [22:0] is supplied
		Set if a misaligned word address is supplied (global address [0] != 0)
		Set if the global address [22:0] points to an area of the D-Flash EEE partition
		Set if the requested section breaches the end of the D-Flash block or goes into the D-Flash EEE partition
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read
	MGSTAT0	Set if any non-correctable errors have been encountered during the read
FERSTAT	EPVIOLIF	None

4.4.2.17 Program D-Flash Command

The Program D-Flash operation programs one to four previously erased words in the D-Flash user partition. The Program D-Flash operation will confirm that the targeted location(s) were successfully programmed upon completion.

CAUTION

A Flash word must be in the erased state before being programmed. Cumulative programming of bits within a Flash word is not allowed.

Table 4-67. Program D-Flash Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x11	Global address [22:16] to identify the D-Flash block
001	Global address [15:0] of word to be programmed	
010	Word 0 program value	
011	Word 1 program value, if desired	
100	Word 2 program value, if desired	
101	Word 3 program value, if desired	

Upon clearing CCIF to launch the Program D-Flash command, the user-supplied words will be transferred to the Memory Controller and be programmed. The CCOBIX index value at Program D-Flash command launch determines how many words will be programmed in the D-Flash block. No protection checks are made in the Program D-Flash operation on the D-Flash block, only access error checks. The CCIF flag is set when the operation has completed.

Table 4-68. Program D-Flash Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] < 010 at command launch
		Set if CCOBIX[2:0] > 101 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 4-30)
		Set if an invalid global address [22:0] is supplied
		Set if a misaligned word address is supplied (global address [0] != 0)
		Set if the global address [22:0] points to an area in the D-Flash EEE partition
		Set if the requested group of words breaches the end of the D-Flash block or goes into the D-Flash EEE partition
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation
FERSTAT	EPVIOLIF	None

4.4.2.18 Erase D-Flash Sector Command

The Erase D-Flash Sector operation will erase all addresses in a sector of the D-Flash user partition.

Table 4-69. Erase D-Flash Sector Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x12	Global address [22:16] to identify D-Flash block
001	Global address [15:0] anywhere within the sector to be erased. See Section 4.1.2.2 for D-Flash sector size.	

Upon clearing CCIF to launch the Erase D-Flash Sector command, the Memory Controller will erase the selected Flash sector and verify that it is erased. The CCIF flag will set after the Erase D-Flash Sector operation has completed.

Table 4-70. Erase D-Flash Sector Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 4-30)
		Set if an invalid global address [22:0] is supplied
		Set if a misaligned word address is supplied (global address [0] != 0)
		Set if the global address [22:0] points to the D-Flash EEE partition
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation
FERSTAT	EPVIOLIF	None

4.4.2.19 Enable EEPROM Emulation Command

The Enable EEPROM Emulation command causes the Memory Controller to enable EEE activity. EEE activity is disabled after any reset.

Table 4-71. Enable EEPROM Emulation Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x13	Not required

Upon clearing CCIF to launch the Enable EEPROM Emulation command, the CCIF flag will set after the Memory Controller enables EEE operations using the contents of the EEE tag RAM and tag counter. The Full Partition D-Flash or the Partition D-Flash command must be run prior to launching the Enable EEPROM Emulation command.

Table 4-72. Enable EEPROM Emulation Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if Full Partition D-Flash or Partition D-Flash command not previously run
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

4.4.2.20 Disable EEPROM Emulation Command

The Disable EEPROM Emulation command causes the Memory Controller to suspend current EEE activity.

Table 4-73. Disable EEPROM Emulation Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x14	Not required

Upon clearing CCIF to launch the Disable EEPROM Emulation command, the Memory Controller will halt EEE operations at the next convenient point without clearing the EEE tag RAM or tag counter before setting the CCIF flag.

Table 4-74. Disable EEPROM Emulation Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if Full Partition D-Flash or Partition D-Flash command not previously run
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

4.4.2.21 EEPROM Emulation Query Command

The EEPROM Emulation Query command returns EEE partition and status variables.

Table 4-75. EEPROM Emulation Query Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x15	Not required
001	Return DFPART	
010	Return ERPART	
011	Return ECOUNT(1)	
100	Return Dead Sector Count	Return Ready Sector Count

1. Indicates sector erase count

Upon clearing CCIF to launch the EEPROM Emulation Query command, the CCIF flag will set after the EEE partition and status variables are stored in the FCCOBIX register.If the Emulation Query command is executed prior to partitioning (Partition D-Flash Command Section 4.4.2.15), the following reset values are returned: DFPART = 0x_FFFF, ERPART = 0x_FFFF, ECOUNT = 0x_FFFF, Dead Sector Count = 0x_00, Ready Sector Count = 0x_00.

Table 4-76. EEPROM Emulation Query Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 4-30)
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

4.4.2.22 Partition D-Flash Command

The Partition D-Flash command allows the user to allocate sectors within the D-Flash block for applications and a partition within the buffer RAM for EEPROM access. The D-Flash block consists of 128 sectors with 256 bytes per sector. The Erase All Blocks command must be run prior to launching the Partition D-Flash command.

Table 4-77. Partition D-Flash Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x20	Not required
001	Number of 256 byte sectors for the D-Flash user partition (DFPART)	
010	Number of 256 byte sectors for buffer RAM EEE partition (ERPART)	

Upon clearing CCIF to launch the Partition D-Flash command, the following actions are taken to define a partition within the D-Flash block for direct access (DFPART) and a partition within the buffer RAM for EEE use (ERPART):

- Validate the DFPART and ERPART values provided:
  - DFPART <= 128 (maximum number of 256 byte sectors in D-Flash block)
  - ERPART <= 16 (maximum number of 256 byte sectors in buffer RAM)
  - If ERPART > 0, 128 - DFPART >= 12 (minimum number of 256 byte sectors in the D-Flash block required to support EEE)
  - If ERPART > 0, ((128-DFPART)/ERPART) >= 8 (minimum ratio of D-Flash EEE space to buffer RAM EEE space to support EEE)
- Erase verify the D-Flash block and the EEE nonvolatile information register
- Program DFPART to the EEE nonvolatile information register at global address 0x12_0000 (see Table 4-7)

- Program a duplicate DFPART to the EEE nonvolatile information register at global address 0x12_0002 (see Table 4-7)
- Program ERPART to the EEE nonvolatile information register at global address 0x12_0004 (see Table 4-7)
- Program a duplicate ERPART to the EEE nonvolatile information register at global address 0x12_0006 (see Table 4-7)

The D-Flash user partition will start at global address 0x10_0000. The buffer RAM EEE partition will end at global address 0x13_FFFF. After the Partition D-Flash operation has completed, the CCIF flag will set.

Running the Partition D-Flash command a second time will result in the ACCERR bit within the FSTAT register being set. The data value written corresponds to the number of 256 byte sectors allocated for either direct D-Flash access (DFPART) or buffer RAM EEE access (ERPART).

Table 4-78. Partition D-Flash Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 010 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 4-30)
		Set if partitions have already been defined
		Set if an invalid DFPART or ERPART selection is supplied
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read
	MGSTAT0	Set if any non-correctable errors have been encountered during the read
FERSTAT	EPVIOLIF	None

4.4.3 Interrupts

The Flash module can generate an interrupt when a Flash command operation has completed or when a Flash command operation has detected an EEE error or an ECC fault.

Table 4-79. Flash Interrupt Sources

Interrupt Source	Interrupt Flag	Local Enable	Global (CCR) Mask
Flash Command Complete	CCIF (FSTAT register)	CCIE (FCNFG register)	I Bit
Flash EEE Erase Error	ERSERIF (FERSTAT register)	ERSERIE (FERCNFG register)	I Bit
Flash EEE Program Error	PGMERIF (FERSTAT register)	PGMERIE (FERCNFG register)	I Bit
Flash EEE Protection Violation	EPVIOLIF (FERSTAT register)	EPVIOLIE (FERCNFG register)	I Bit
Flash EEE Error Type 1 Violation	ERSVIF1 (FERSTAT register)	ERSVIE1 (FERCNFG register)	I Bit
Flash EEE Error Type 0 Violation	ERSVIF0 (FERSTAT register)	ERSVIE0 (FERCNFG register)	I Bit
ECC Double Bit Fault on Flash Read	DFDIF (FERSTAT register)	DFDIE (FERCNFG register)	I Bit
ECC Single Bit Fault on Flash Read	SFDIF (FERSTAT register)	SFDIE (FERCNFG register)	I Bit

NOTE

Vector addresses and their relative interrupt priority are determined at the MCU level.

4.4.3.1 Description of Flash Interrupt Operation

The Flash module uses the CCIF flag in combination with the CCIE interrupt enable bit to generate the Flash command interrupt request. The Flash module uses the ERSEIF, PGMEIF, EPVIOLIF, ERSVIF1, ERSVIF0, DFDIF and SFDIF flags in combination with the ERSEIE, PGMEIE, EPVIOLIE, ERSVIE1, ERSVIE0, DFDIE and SFDIE interrupt enable bits to generate the Flash error interrupt request. For a detailed description of the register bits involved, refer to Section 4.3.2.5, “Flash Configuration Register (FCNFG)”, Section 4.3.2.6, “Flash Error Configuration Register (FERCNFG)”, Section 4.3.2.7, “Flash Status Register (FSTAT)”, and Section 4.3.2.8, “Flash Error Status Register (FERSTAT)”.

The logic used for generating the Flash module interrupts is shown in Figure 4-27.

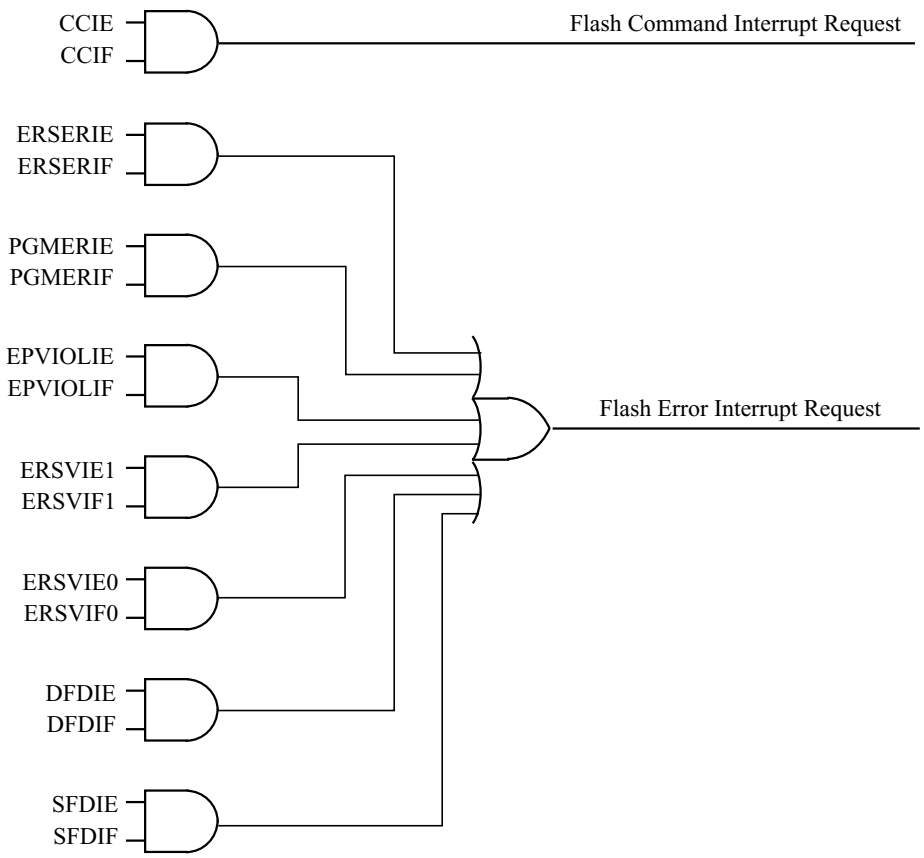


Figure 4-27. Flash Module Interrupts Implementation

4.4.4 Wait Mode

The Flash module is not affected if the MCU enters wait mode. The Flash module can recover the MCU from wait via the CCIF interrupt (see Section 4.4.3, “Interrupts”).

4.4.5 Stop Mode

If a Flash command is active (CCIF = 0) or an EE-Emulation operation is pending when the MCU requests stop mode, the current Flash operation will be completed before the CPU is allowed to enter stop mode.

4.5 Security

The Flash module provides security information to the MCU. The Flash security state is defined by the SEC bits of the FSEC register (see Table 4-12). During reset, the Flash module initializes the FSEC register using data read from the security byte of the Flash configuration field at global address 0x7F_FF0F.

The security state out of reset can be permanently changed by programming the security byte of the Flash configuration field. This assumes that you are starting from a mode where the necessary P-Flash erase and program commands are available and that the upper region of the P-Flash is unprotected. If the Flash security byte is successfully programmed, its new value will take affect after the next MCU reset.

The following subsections describe these security-related subjects:

- Unsecuring the MCU using Backdoor Key Access
- Unsecuring the MCU in Special Single Chip Mode using BDM
- Mode and Security Effects on Flash Command Availability

4.5.1 Unsecuring the MCU using Backdoor Key Access

The MCU may be unsecured by using the backdoor key access feature which requires knowledge of the contents of the backdoor keys (four 16-bit words programmed at addresses 0x7F_FF00–0x7F_FF07). If the KEYEN[1:0] bits are in the enabled state (see Section 4.3.2.2), the Verify Backdoor Access Key command (see Section 4.4.2.12) allows the user to present four prospective keys for comparison to the keys stored in the Flash memory via the Memory Controller. If the keys presented in the Verify Backdoor Access Key command match the backdoor keys stored in the Flash memory, the SEC bits in the FSEC register (see Table 4-12) will be changed to unsecure the MCU. Key values of 0x0000 and 0xFFFF are not permitted as backdoor keys. While the Verify Backdoor Access Key command is active, P-Flash block 0 will not be available for read access and will return invalid data.

The user code stored in the P-Flash memory must have a method of receiving the backdoor keys from an external stimulus. This external stimulus would typically be through one of the on-chip serial ports.

If the KEYEN[1:0] bits are in the enabled state (see Section 4.3.2.2), the MCU can be unsecured by the backdoor key access sequence described below:

1. Follow the command sequence for the Verify Backdoor Access Key command as explained in Section 4.4.2.12
2. If the Verify Backdoor Access Key command is successful, the MCU is unsecured and the SEC[1:0] bits in the FSEC register are forced to the unsecure state of 10

The Verify Backdoor Access Key command is monitored by the Memory Controller and an illegal key will prohibit future use of the Verify Backdoor Access Key command. A reset of the MCU is the only method to re-enable the Verify Backdoor Access Key command.

After the backdoor keys have been correctly matched, the MCU will be unsecured. After the MCU is unsecured, the sector containing the Flash security byte can be erased and the Flash security byte can be reprogrammed to the unsecure state, if desired.

In the unsecure state, the user has full control of the contents of the backdoor keys by programming addresses 0x7F_FF00–0x7F_FF07 in the Flash configuration field.

The security as defined in the Flash security byte (0x7F_FF0F) is not changed by using the Verify Backdoor Access Key command sequence. The backdoor keys stored in addresses 0x7F_FF00–0x7F_FF07 are unaffected by the Verify Backdoor Access Key command sequence. After the next reset of the MCU, the security state of the Flash module is determined by the Flash security byte

(0x7F_FF0F). The Verify Backdoor Access Key command sequence has no effect on the program and erase protections defined in the Flash protection register, FPROT.

4.5.2 Unsecuring the MCU in Special Single Chip Mode using BDM

The MCU can be unsecured in special single chip mode by erasing the P-Flash and D-Flash memory by one of the following methods:

- Reset the MCU into special single chip mode, delay while the erase test is performed by the BDM, send BDM commands to disable protection in the P-Flash and D-Flash memory, and execute the Erase All Blocks command write sequence to erase the P-Flash and D-Flash memory.
- Reset the MCU into special expanded wide mode, disable protection in the P-Flash and D-Flash memory and run code from external memory to execute the Erase All Blocks command write sequence to erase the P-Flash and D-Flash memory.

After the CCIF flag sets to indicate that the Erase All Blocks operation has completed, reset the MCU into special single chip mode. The BDM will execute the Erase Verify All Blocks command write sequence to verify that the P-Flash and D-Flash memory is erased. If the P-Flash and D-Flash memory are verified as erased the MCU will be unsecured. All BDM commands will be enabled and the Flash security byte may be programmed to the unsecure state by the following method:

- Send BDM commands to execute a ‘Program P-Flash’ command sequence to program the Flash security byte to the unsecured state and reset the MCU.

4.5.3 Mode and Security Effects on Flash Command Availability

The availability of Flash module commands depends on the MCU operating mode and security state as shown in Table 4-30.

4.6 Initialization

On each system reset the Flash module executes a reset sequence which establishes initial values for the Flash Block Configuration Parameters, the FPROT and DFPROT protection registers, and the FOPT and FSEC registers. The Flash module reverts to built-in default values that leave the module in a fully protected and secured state if errors are encountered during execution of the reset sequence. If a double bit fault is detected during the reset sequence, both MGSTAT bits in the FSTAT register will be set. The ACCERR bit in the FSTAT register is set if errors are encountered while initializing the EEE buffer ram during the reset sequence.

CCIF remains clear throughout the reset sequence. The Flash module holds off all CPU access for the initial portion of the reset sequence. While Flash reads are possible when the hold is removed, writes to the FCCOBIX, FCCOBHI, and FCCOBLO registers are ignored to prevent command activity while the Memory Controller remains busy. Completion of the reset sequence is marked by setting CCIF high which enables writes to the FCCOBIX, FCCOBHI, and FCCOBLO registers to launch any available Flash command.

If a reset occurs while any Flash command is in progress, that command will be immediately aborted. The state of the word being programmed or the sector/block being erased is not guaranteed.

Chapter 5
Pierce Oscillator (S12OSCLCPV2)

Revision History

Revision Number	Revision Date	Author	Description of Changes
01.05	19-Jul-06		All xclks info was removed
02.00	04-Aug-06		incremented revision to match the design system spec revision

5.1 Introduction

The Pierce oscillator (XOSC) module provides a robust, low-noise and low-power clock source. The module will be operated from the V_{DDPLL} supply rail (1.8 V nominal) and require the minimum number of external components. It is designed for optimal start-up margin with typical crystal oscillators.

5.1.1 Features

The XOSC will contain circuitry to dynamically control current gain in the output amplitude. This ensures a signal with low harmonic distortion, low power and good noise immunity.

- High noise immunity due to input hysteresis
- Low RF emissions with peak-to-peak swing limited dynamically
- Transconductance (gm) sized for optimum start-up margin for typical oscillators
- Dynamic gain control eliminates the need for external current limiting resistor
- Integrated resistor eliminates the need for external bias resistor in loop controlled Pierce mode.
- Low power consumption:
  - Operates from 1.8 V (nominal) supply
  - Amplitude control limits power
- Clock monitor

5.1.2 Modes of Operation

Two modes of operation exist:

1. Loop controlled Pierce (LCP) oscillator
2. External square wave mode featuring also full swing Pierce (FSP) without internal bias resistor

The oscillator mode selection is described in the Device Overview section, subsection Oscillator Configuration.

5.1.3 Block Diagram

Figure 5-1 shows a block diagram of the XOSC.

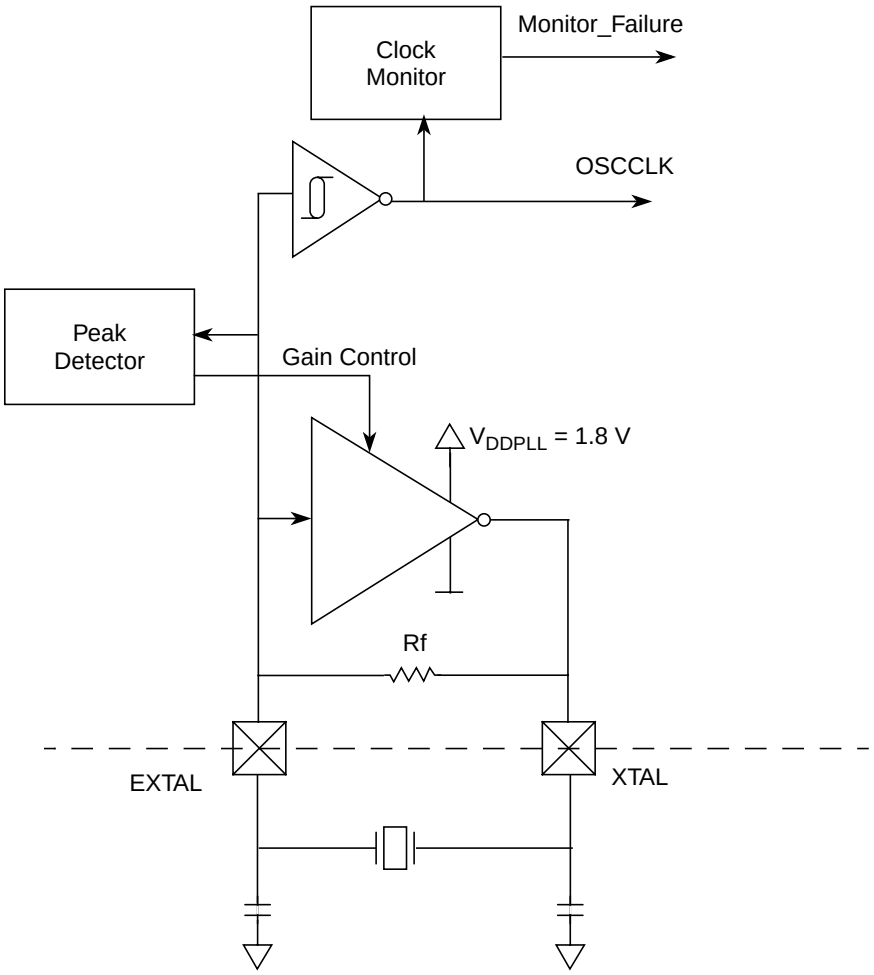


Figure 5-1. XOSC Block Diagram

5.2 External Signal Description

This section lists and describes the signals that connect off chip

5.2.1 VDDPLL and VSSPLL — Operating and Ground Voltage Pins

Theses pins provides operating voltage (V_{DDPLL}) and ground (V_{SSPLL}) for the XOSC circuitry. This allows the supply voltage to the XOSC to use an independent bypass capacitor.

5.2.2 EXTAL and XTAL — Input and Output Pins

These pins provide the interface for either a crystal or a 1.8V CMOS compatible clock to control the internal clock generator circuitry. EXTAL is the external clock input or the input to the crystal oscillator amplifier. XTAL is the output of the crystal oscillator amplifier. The MCU internal system clock is derived

from the EXTAL input frequency. In full stop mode (PSTP = 0), the EXTAL pin is pulled down by an internal resistor of typical 200 kΩ.

NOTE

Freescale recommends an evaluation of the application board and chosen resonator or crystal by the resonator or crystal supplier.

Loop controlled circuit is not suited for overtone resonators and crystals.

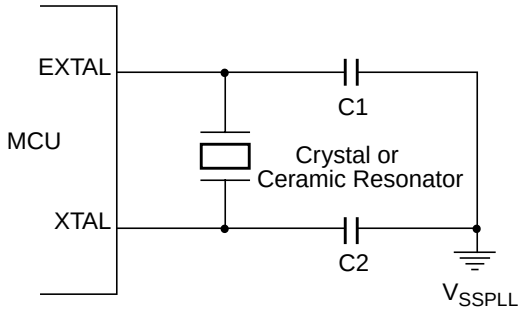
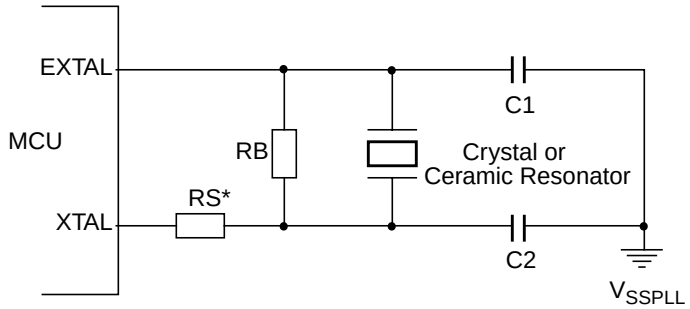


Figure 5-2. Loop Controlled Pierce Oscillator Connections (LCP mode selected)

NOTE

Full swing Pierce circuit is not suited for overtone resonators and crystals without a careful component selection.



* R_S can be zero (shorted) when use with higher frequency crystals. Refer to manufacturer's data.

Figure 5-3. Full Swing Pierce Oscillator Connections (FSP mode selected)

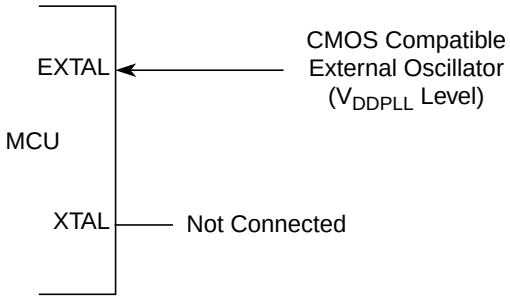


Figure 5-4. External Clock Connections (FSP mode selected)

5.3 Memory Map and Register Definition

The CRG contains the registers and associated bits for controlling and monitoring the oscillator module.

5.4 Functional Description

The XOSC module has control circuitry to maintain the crystal oscillator circuit voltage level to an optimal level which is determined by the amount of hysteresis being used and the maximum oscillation range.

The oscillator block has two external pins, EXTAL and XTAL. The oscillator input pin, EXTAL, is intended to be connected to either a crystal or an external clock source. The XTAL pin is an output signal that provides crystal circuit feedback.

A buffered EXTAL signal becomes the internal clock. To improve noise immunity, the oscillator is powered by the VDDPLL and VSSPLL power supply pins.

5.4.1 Gain Control

In LCP mode a closed loop control system will be utilized whereby the amplifier is modulated to keep the output waveform sinusoidal and to limit the oscillation amplitude. The output peak to peak voltage will be kept above twice the maximum hysteresis level of the input buffer. Electrical specification details are provided in the Electrical Characteristics appendix.

5.4.2 Clock Monitor

The clock monitor circuit is based on an internal RC time delay so that it can operate without any MCU clocks. If no OSCCLK edges are detected within this RC time delay, the clock monitor indicates failure which asserts self-clock mode or generates a system reset depending on the state of SCME bit. If the clock monitor is disabled or the presence of clocks is detected no failure is indicated.The clock monitor function is enabled/disabled by the CME control bit, described in the CRG block description chapter.

5.4.3 Wait Mode Operation

During wait mode, XOSC is not impacted.

5.4.4 Stop Mode Operation

XOSC is placed in a static state when the part is in stop mode except when pseudo-stop mode is enabled. During pseudo-stop mode, XOSC is not impacted.

Chapter 6
Security (S12XFSECV2)

Revision History

Version Number	Revision Date	Effective Date	Author	Description of Changes
01.00	26 Sep 2003	26 Sep 2003		Initial Release
02.00	27 Aug 2004	08 Sep 2004		reviewed and updated for S12XD architecture
02.01	21 Feb 2007	21 Feb 2007		added S12XE, S12XF and S12XS architectures

6.1 Introduction

This specification describes the function of the security mechanism in the S12XF chip family (SEC).

6.1.1 Features

The user must be reminded that part of the security must lie with the application code. An extreme example would be application code that dumps the contents of the internal memory. This would defeat the purpose of security. At the same time, the user may also wish to put a backdoor in the application program. An example of this is the user downloads a security key through the SCI, which allows access to a programming routine that updates parameters stored in another section of the Flash memory.

The security features of the S12XDF chip family (in secure mode) are:

- Protect the content of non-volatile memories (Flash, EEPROM)
- Execution of NVM commands is restricted
- Disable access to internal memory via background debug module (BDM)
- Disable access to internal Flash/EEPROM in expanded modes
- Disable debugging features for the CPU and XGATE

Table 6-1 gives an overview over availability of security relevant features in unsecure and secure modes.

Table 6-1. Feature Availability in Unsecure and Secure Modes on S12XF

	Unsecure Mode						Secure Mode					
	NS	SS	NX	ES	EX	ST	NS	SS	NX	ES	EX	ST
Flash Array Access	✔	✔	✔(1)	✔1	✔1	✔1	✔	✔	—	—	—	—
EEPROM Array Access	✔	✔	✔	✔	✔	✔	✔	✔	—	—	—	—
NVM Commands	✔(2)	✔	✔2	✔2	✔2	✔	✔2	✔2	✔2	✔2	✔2	✔2
BDM	✔	✔	✔	✔	✔	✔	—	✔(3)	—	—	—	—
DBG Module Trace	✔	✔	✔	✔	✔	✔	—	—	—	—	—	—
XGATE Debugging	✔	✔	✔	✔	✔	✔	—	—	—	—	—	—
External Bus Interface	—	—	✔	✔	✔	✔	—	—	✔	✔	✔	✔
Internal status visible multiplexed on external bus	—	—	—	✔	✔	—	—	—	—	✔	✔	—
Internal accesses visible on external bus	—	—	—	—	—	✔	—	—	—	—	—	✔

1. Availability of Flash arrays in the memory map depends on ROMCTL/EROMCTL pins and/or the state of the ROMON/EROMON bits in the MMCCTL1 register. Please refer to the S12X_MMC block guide for detailed information.
2. Restricted NVM command set only. Please refer to the NVM wrapper block guides for detailed information.
3. BDM hardware commands restricted to peripheral registers only.

6.1.2 Modes of Operation

6.1.3 Securing the Microcontroller

Once the user has programmed the Flash and EEPROM, the chip can be secured by programming the security bits located in the options/security byte in the Flash memory array. These non-volatile bits will keep the device secured through reset and power-down.

The options/security byte is located at address 0xFF0F (= global address 0x7F_FF0F) in the Flash memory array. This byte can be erased and programmed like any other Flash location. Two bits of this byte are used for security (SEC[1:0]). On devices which have a memory page window, the Flash options/security byte is also available at address 0xBF0F by selecting page 0x3F with the PPAGE register. The contents of this byte are copied into the Flash security register (FSEC) during a reset sequence.

	7	6	5	4	3	2	1	0
0xFF0F	KEYEN1	KEYEN0	NV5	NV4	NV3	NV2	SEC1	SEC0

Figure 6-1. Flash Options/Security Byte

The meaning of the bits KEYEN[1:0] is shown in Table 6-2. Please refer to Section 6.1.5.1, “Unsecuring the MCU Using the Backdoor Key Access” for more information.

Table 6-2. Backdoor Key Access Enable Bits

KEYEN[1:0]	Backdoor Key Access Enabled
00	0 (disabled)
01	0 (disabled)
10	1 (enabled)
11	0 (disabled)

The meaning of the security bits SEC[1:0] is shown in Table 6-3. For security reasons, the state of device security is controlled by two bits. To put the device in unsecured mode, these bits must be programmed to SEC[1:0] = '10'. All other combinations put the device in a secured mode. The recommended value to put the device in secured state is the inverse of the unsecured state, i.e. SEC[1:0] = '01'.

Table 6-3. Security Bits

SEC[1:0]	Security State
00	1 (secured)
01	1 (secured)
10	***0 (unsecured)***
11	1 (secured)

NOTE

Please refer to the Flash block guide for actual security configuration (in section "Flash Module Security").

6.1.4 Operation of the Secured Microcontroller

By securing the device, unauthorized access to the EEPROM and Flash memory contents can be prevented. However, it must be understood that the security of the EEPROM and Flash memory contents also depends on the design of the application program. For example, if the application has the capability of downloading code through a serial port and then executing that code (e.g. an application containing bootloader code), then this capability could potentially be used to read the EEPROM and Flash memory contents even when the microcontroller is in the secure state. In this example, the security of the application could be enhanced by requiring a challenge/response authentication before any code can be downloaded.

Secured operation has the following effects on the microcontroller:

6.1.4.1 Normal Single Chip Mode (NS)

- Background debug module (BDM) operation is completely disabled.
- Execution of Flash and EEPROM commands is restricted. Please refer to the NVM block guide for details.
- Tracing code execution using the DBG module is disabled.
- Debugging XGATE code (breakpoints, single-stepping) is disabled.

6.1.4.2 Special Single Chip Mode (SS)

- BDM firmware commands are disabled.
- BDM hardware commands are restricted to the register space.
- Execution of Flash and EEPROM commands is restricted. Please refer to the NVM block guide for details.
- Tracing code execution using the DBG module is disabled.
- Debugging XGATE code (breakpoints, single-stepping) is disabled.

Special single chip mode means BDM is active after reset. The availability of BDM firmware commands depends on the security state of the device. The BDM secure firmware first performs a blank check of both the Flash memory and the EEPROM. If the blank check succeeds, security will be temporarily turned off and the state of the security bits in the appropriate Flash memory location can be changed If the blank check fails, security will remain active, only the BDM hardware commands will be enabled, and the accessible memory space is restricted to the peripheral register area. This will allow the BDM to be used to erase the EEPROM and Flash memory without giving access to their contents. After erasing both Flash memory and EEPROM, another reset into special single chip mode will cause the blank check to succeed and the options/security byte can be programmed to “unsecured” state via BDM.

While the BDM is executing the blank check, the BDM interface is completely blocked, which means that all BDM commands are temporarily blocked.

6.1.4.3 Expanded Modes (NX, ES, EX, and ST)

- BDM operation is completely disabled.
- Internal Flash memory and EEPROM are disabled.
- Execution of Flash and EEPROM commands is restricted. Please refer to the NVM block guide for details.
- Tracing code execution using the DBG module is disabled.
- Debugging XGATE code (breakpoints, single-stepping) is disabled

6.1.5 Unsecuring the Microcontroller

Unsecuring the microcontroller can be done by three different methods:

1. Backdoor key access
2. Reprogramming the security bits
3. Complete memory erase (special modes)

6.1.5.1 Unsecuring the MCU Using the Backdoor Key Access

In normal modes (single chip and expanded), security can be temporarily disabled using the backdoor key access method. This method requires that:

- The backdoor key at 0xFF00–0xFF07 (= global addresses 0x7F_FF00–0x7F_FF07) has been programmed to a valid value.
- The KEYEN[1:0] bits within the Flash options/security byte select ‘enabled’.
- In single chip mode, the application program programmed into the microcontroller must be designed to have the capability to write to the backdoor key locations.

The backdoor key values themselves would not normally be stored within the application data, which means the application program would have to be designed to receive the backdoor key values from an external source (e.g. through a serial port). It is not possible to download the backdoor keys using background debug mode.

The backdoor key access method allows debugging of a secured microcontroller without having to erase the Flash. This is particularly useful for failure analysis.

NOTE

No word of the backdoor key is allowed to have the value 0x0000 or 0xFFFF.

6.1.6 Reprogramming the Security Bits

In normal single chip mode (NS), security can also be disabled by erasing and reprogramming the security bits within Flash options/security byte to the unsecured value. Because the erase operation will erase the entire sector from 0xFE00–0xFFFF (0x7F_FE00–0x7F_FFFF), the backdoor key and the interrupt vectors will also be erased; this method is not recommended for normal single chip mode. The application software can only erase and program the Flash options/security byte if the Flash sector containing the Flash options/security byte is not protected (see Flash protection). Thus Flash protection is a useful means of preventing this method. The microcontroller will enter the unsecured state after the next reset following the programming of the security bits to the unsecured value.

This method requires that:

- The application software previously programmed into the microcontroller has been designed to have the capability to erase and program the Flash options/security byte, or security is first disabled using the backdoor key method, allowing BDM to be used to issue commands to erase and program the Flash options/security byte.
- The Flash sector containing the Flash options/security byte is not protected.

6.1.7 Complete Memory Erase (Special Modes)

The microcontroller can be unsecured in special modes by erasing the entire EEPROM and Flash memory contents.

When a secure microcontroller is reset into special single chip mode (SS), the BDM firmware verifies whether the EEPROM and Flash memory are erased. If any EEPROM or Flash memory address is not erased, only BDM hardware commands are enabled. BDM hardware commands can then be used to write to the EEPROM and Flash registers to mass erase the EEPROM and all Flash memory blocks.

When next reset into special single chip mode, the BDM firmware will again verify whether all EEPROM and Flash memory are erased, and this being the case, will enable all BDM commands, allowing the Flash options/security byte to be programmed to the unsecured value. The security bits SEC[1:0] in the Flash security register will indicate the unsecure state following the next reset.

Chapter 7 Interrupt (S12XINTV2)

Table 7-1. Revision History

Version Number	Revision Date	Effective Date	Author	Description of Changes
02.00	01 JUL 2005	01 JUL 2005		initial V2 release, added new features: - XGATE threads can be interrupted - SYS instruction vector - access violation interrupt vectors
02.04	11 JAN 2007	11 JAN 2007		- added Notes for devices without XGATE module
02.05	20 MAR 2007	23 MAR 2007		- fixed priority definition for software exceptions in “1.4.6 Exception Priority”
02.06	07 JAN 2008	07 JAN 2008		- added clarification of “Wake-up from STOP or WAIT by XIRQ with X bit set” feature

7.1 Introduction

The INT module decodes the priority of all system exception requests and provides the applicable vector for processing the exception to either the CPU or the XGATE module. The INT module supports:

- I bit and X bit maskable interrupt requests
- One non-maskable unimplemented op-code trap
- One non-maskable software interrupt (SWI) or background debug mode request
- One non-maskable system call interrupt (SYS)
- Three non-maskable access violation interrupt
- One spurious interrupt vector request
- Three system reset vector requests

Each of the I bit maskable interrupt requests can be assigned to one of seven priority levels supporting a flexible priority scheme. For interrupt requests that are configured to be handled by the CPU, the priority scheme can be used to implement nested interrupt capability where interrupts from a lower level are automatically blocked if a higher level interrupt is being processed. Interrupt requests configured to be handled by the XGATE module can be nested one level deep.

NOTE

The HPRIO register and functionality of the original S12 interrupt module is no longer supported, since it is superseded by the 7-level interrupt request priority scheme.

7.1.1 Glossary

The following terms and abbreviations are used in the document.

Table 7-2. Terminology

Term	Meaning
CCR	Condition Code Register (in the S12X CPU)
DMA	Direct Memory Access
INT	Interrupt
IPL	Interrupt Processing Level
ISR	Interrupt Service Routine
MCU	Micro-Controller Unit
XGATE	refers to the XGATE co-processor; XGATE is an optional feature
$\overline{\text{IRQ}}$	refers to the interrupt request associated with the $\overline{\text{IRQ}}$ pin
$\overline{\text{XIRQ}}$	refers to the interrupt request associated with the $\overline{\text{XIRQ}}$ pin

7.1.2 Features

- Interrupt vector base register (IVBR)
- One spurious interrupt vector (at address vector base[1] + 0x0010).
- One non-maskable system call interrupt vector request (at address vector base + 0x0012).
- Three non-maskable access violation interrupt vector requests (at address vector base + 0x0014–0x0018).
- 2–109 I bit maskable interrupt vector requests (at addresses vector base + 0x001A–0x00F2).
- Each I bit maskable interrupt request has a configurable priority level and can be configured to be handled by either the CPU or the XGATE module[2].
- I bit maskable interrupts can be nested, depending on their priority levels.
- One X bit maskable interrupt vector request (at address vector base + 0x00F4).
- One non-maskable software interrupt request (SWI) or background debug mode vector request (at address vector base + 0x00F6).
- One non-maskable unimplemented op-code trap (TRAP) vector (at address vector base + 0x00F8).
- Three system reset vectors (at addresses 0xFFFA–0xFFFE).
- Determines the highest priority XGATE and interrupt vector requests, drives the vector to the XGATE module or to the bus on CPU request, respectively.

1. The vector base is a 16-bit address which is accumulated from the contents of the interrupt vector base register (IVBR, used as upper byte) and 0x00 (used as lower byte).
2. The $\overline{\text{IRQ}}$ interrupt can only be handled by the CPU

- Wakes up the system from stop or wait mode when an appropriate interrupt request occurs or whenever $\overline{\text{XIRQ}}$ is asserted, even if X interrupt is masked.
- XGATE can wake up and execute code, even with the CPU remaining in stop or wait mode.

7.1.3 Modes of Operation

- Run mode

  This is the basic mode of operation.
- Wait mode

  In wait mode, the INT module is frozen. It is however capable of either waking up the CPU if an interrupt occurs or waking up the XGATE if an XGATE request occurs. Please refer to Section 7.5.3, “Wake Up from Stop or Wait Mode” for details.
- Stop Mode

  In stop mode, the INT module is frozen. It is however capable of either waking up the CPU if an interrupt occurs or waking up the XGATE if an XGATE request occurs. Please refer to Section 7.5.3, “Wake Up from Stop or Wait Mode” for details.
- Freeze mode (BDM active)

  In freeze mode (BDM active), the interrupt vector base register is overridden internally. Please refer to Section 7.3.2.1, “Interrupt Vector Base Register (IVBR)” for details.

7.1.4 Block Diagram

Figure 7-1 shows a block diagram of the INT module.

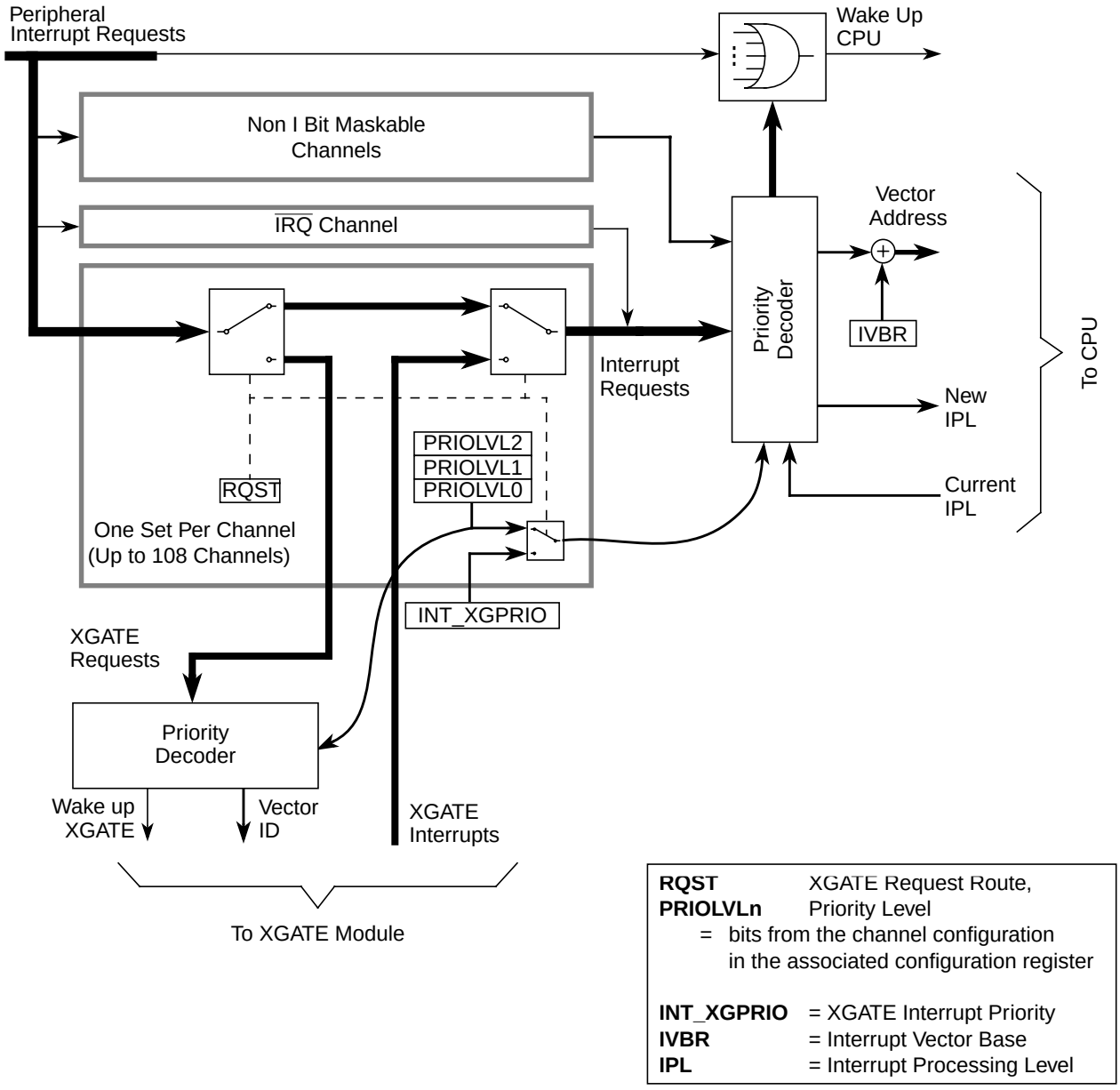


Figure 7-1. INT Block Diagram

7.2 External Signal Description

The INT module has no external signals.

7.3 Memory Map and Register Definition

This section provides a detailed description of all registers accessible in the INT module.

7.3.1 Module Memory Map

Table 7-3 gives an overview over all INT module registers.

Table 7-3. INT Memory Map

Address	Use	Access
0x0120	RESERVED	—
0x0121	Interrupt Vector Base Register (IVBR)	R/W
0x0122–0x0125	RESERVED	—
0x0126	XGATE Interrupt Priority Configuration Register (INT_XGPRIO)	R/W
0x0127	Interrupt Request Configuration Address Register (INT_CFADDR)	R/W
0x0128	Interrupt Request Configuration Data Register 0 (INT_CFDATA0)	R/W
0x0129	Interrupt Request Configuration Data Register 1 (INT_CFDATA1)	R/W
0x012A	Interrupt Request Configuration Data Register 2 (INT_CFDATA2	R/W
0x012B	Interrupt Request Configuration Data Register 3 (INT_CFDATA3)	R/W
0x012C	Interrupt Request Configuration Data Register 4 (INT_CFDATA4)	R/W
0x012D	Interrupt Request Configuration Data Register 5 (INT_CFDATA5)	R/W
0x012E	Interrupt Request Configuration Data Register 6 (INT_CFDATA6)	R/W
0x012F	Interrupt Request Configuration Data Register 7 (INT_CFDATA7)	R/W

7.3.2 Register Descriptions

This section describes in address order all the INT module registers and their individual bits.

Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0121	IVBR	R	IVB_ADDR[7:0]7							
		W								
0x0126	INT_XGPRIO	R	0	0	0	0	0	XILVL[2:0]		
		W								
0x0127	INT_CFADDR	R	INT_CFADDR[7:4]				0	0	0	0
		W								
0x0128	INT_CFDATA0	R	RQST	0	0	0	0	PRIOLVL[2:0]		
		W								
0x0129	INT_CFDATA1	R	RQST	0	0	0	0	PRIOLVL[2:0]		
		W								
0x012A	INT_CFDATA2	R	RQST	0	0	0	0	PRIOLVL[2:0]		
		W								
0x012B	INT_CFDATA3	R	RQST	0	0	0	0	PRIOLVL[2:0]		
		W								
0x012C	INT_CFDATA4	R	RQST	0	0	0	0	PRIOLVL[2:0]		
		W								
0x012D	INT_CFDATA5	R	RQST	0	0	0	0	PRIOLVL[2:0]		
		W								
0x012E	INT_CFDATA6	R	RQST	0	0	0	0	PRIOLVL[2:0]		
		W								
0x012F	INT_CFDATA7	R	RQST	0	0	0	0	PRIOLVL[2:0]		
		W								

= Unimplemented or Reserved

Figure 7-2. INT Register Summary

7.3.2.1 Interrupt Vector Base Register (IVBR)

Address: 0x0121

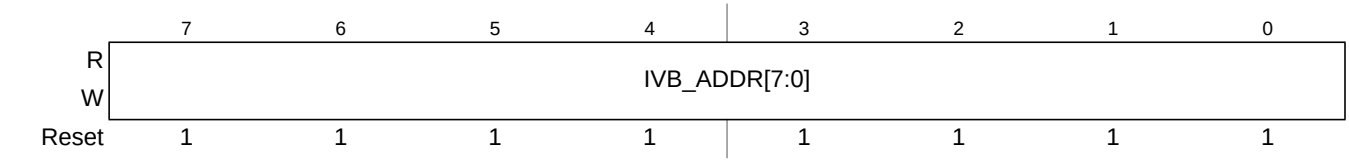


Figure 7-3. Interrupt Vector Base Register (IVBR)

Read: Anytime

Write: Anytime

Table 7-4. IVBR Field Descriptions

Field	Description
7–0 IVB_ADDR[7:0]	**Interrupt Vector Base Address Bits** — These bits represent the upper byte of all vector addresses. Out of reset these bits are set to 0xFF (i.e., vectors are located at 0xFF10–0xFFFE) to ensure compatibility to previous S12 microcontrollers. **Note:** A system reset will initialize the interrupt vector base register with “0xFF” before it is used to determine the reset vector address. Therefore, changing the IVBR has no effect on the location of the three reset vectors (0xFFFA–0xFFFE). **Note:** If the BDM is active (i.e., the CPU is in the process of executing BDM firmware code), the contents of IVBR are ignored and the upper byte of the vector address is fixed as “0xFF”.

7.3.2.2 XGATE Interrupt Priority Configuration Register (INT_XGPRIO)

Address: 0x0126

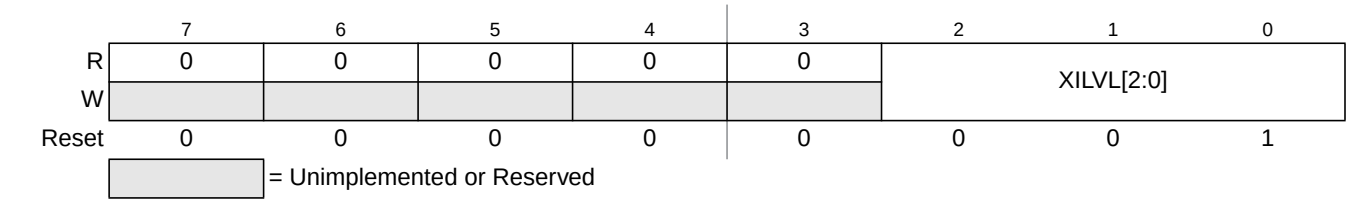


Figure 7-4. XGATE Interrupt Priority Configuration Register (INT_XGPRIO)

Read: Anytime

Write: Anytime

Table 7-5. INT_XGPRIO Field Descriptions

Field	Description
2–0 XILVL[2:0]	**XGATE Interrupt Priority Level** — The XILVL[2:0] bits configure the shared interrupt level of the XGATE interrupts coming from the XGATE module. Out of reset the priority is set to the lowest active level (“1”). **Note:** If the XGATE module is not available on the device, write accesses to this register are ignored and read accesses to this register will return all 0.

Table 7-6. XGATE Interrupt Priority Levels

Priority	XILVL2	XILVL1	XILVL0	Meaning
	0	0	0	Interrupt request is disabled
low	0	0	1	Priority level 1
	0	1	0	Priority level 2
	0	1	1	Priority level 3
	1	0	0	Priority level 4
	1	0	1	Priority level 5
	1	1	0	Priority level 6
high	1	1	1	Priority level 7

7.3.2.3 Interrupt Request Configuration Address Register (INT_CFADDR)

Address: 0x0127

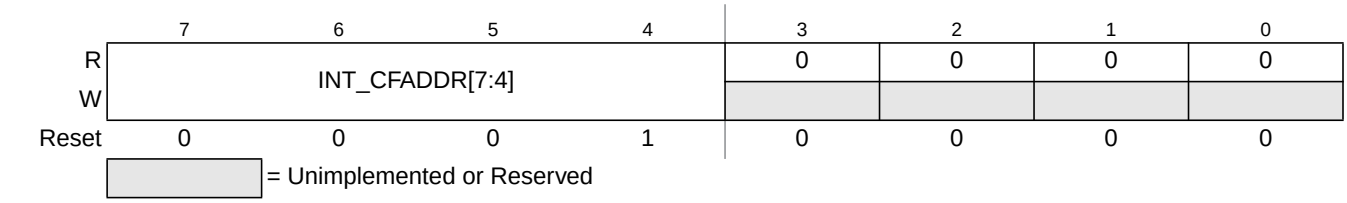


Figure 7-5. Interrupt Configuration Address Register (INT_CFADDR)

Read: Anytime

Write: Anytime

Table 7-7. INT_CFADDR Field Descriptions

Field	Description
7–4 INT_CFADDR[7:4]	**Interrupt Request Configuration Data Register Select Bits** — These bits determine which of the 128 configuration data registers are accessible in the 8 register window at INT_CFDATA0–7. The hexadecimal value written to this register corresponds to the upper nibble of the lower byte of the address of the interrupt vector, i.e., writing 0xE0 to this register selects the configuration data register block for the 8 interrupt vector requests starting with vector at address (vector base + 0x00E0) to be accessible as INT_CFDATA0–7. **Note:** Writing all 0s selects non-existing configuration registers. In this case write accesses to INT_CFDATA0–7 will be ignored and read accesses will return all 0.

7.3.2.4 Interrupt Request Configuration Data Registers (INT_CFDATA0–7)

The eight register window visible at addresses INT_CFDATA0–7 contains the configuration data for the block of eight interrupt requests (out of 128) selected by the interrupt configuration address register (INT_CFADDR) in ascending order. INT_CFDATA0 represents the interrupt configuration data register of the vector with the lowest address in this block, while INT_CFDATA7 represents the interrupt configuration data register of the vector with the highest address, respectively.

Address: 0x0128

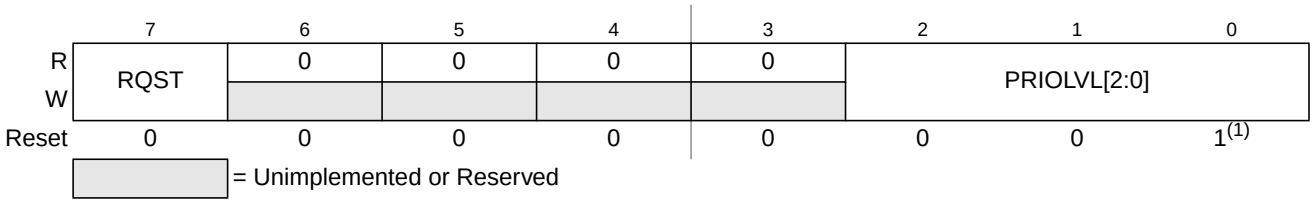


Figure 7-6. Interrupt Request Configuration Data Register 0 (INT_CFDATA0)

1. Please refer to the notes following the PRIOLVL[2:0] description below.

Address: 0x0129

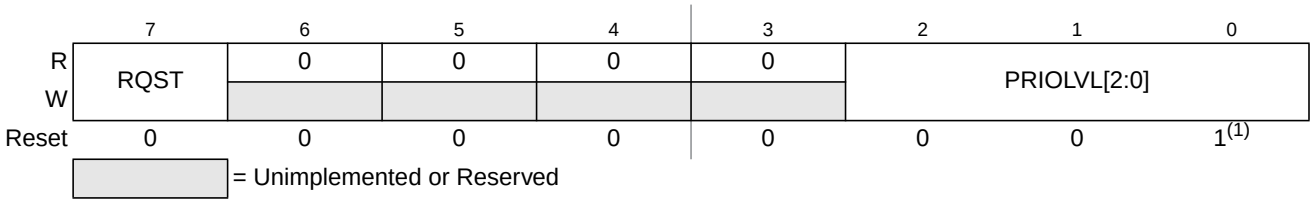


Figure 7-7. Interrupt Request Configuration Data Register 1 (INT_CFDATA1)

1. Please refer to the notes following the PRIOLVL[2:0] description below.

Address: 0x012A

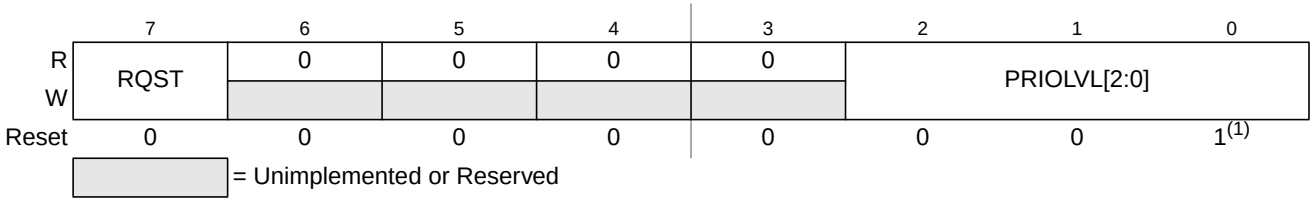


Figure 7-8. Interrupt Request Configuration Data Register 2 (INT_CFDATA2)

1. Please refer to the notes following the PRIOLVL[2:0] description below.

Address: 0x012B

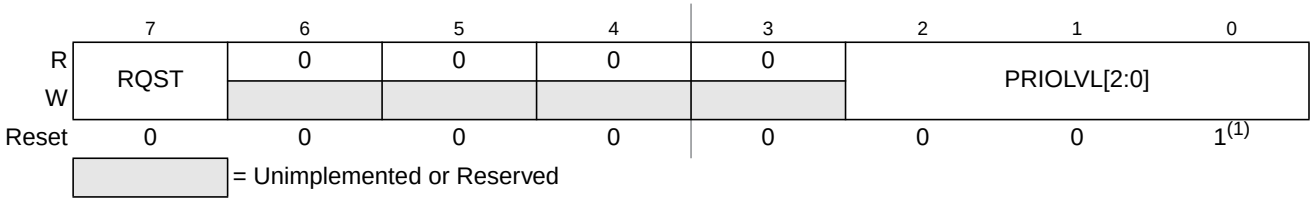


Figure 7-9. Interrupt Request Configuration Data Register 3 (INT_CFDATA3)

1. Please refer to the notes following the PRIOLVL[2:0] description below.

Address: 0x012C

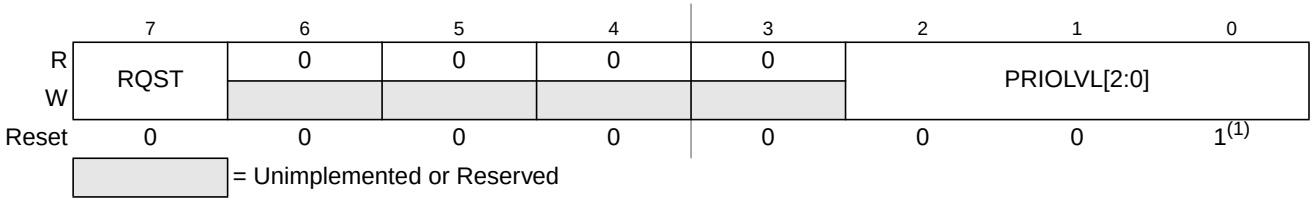


Figure 7-10. Interrupt Request Configuration Data Register 4 (INT_CFDATA4)

1. Please refer to the notes following the PRIOLVL[2:0] description below.

Address: 0x012D

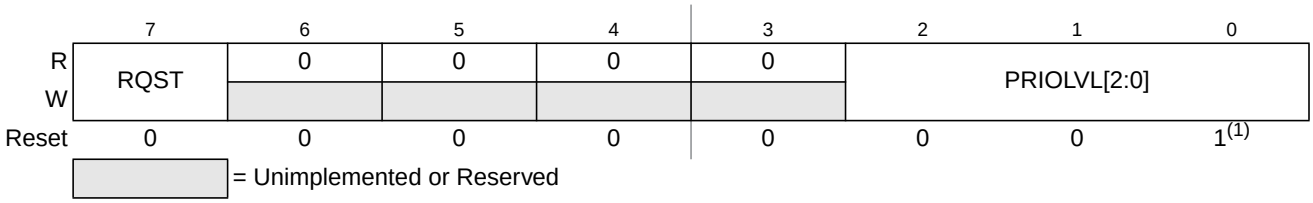


Figure 7-11. Interrupt Request Configuration Data Register 5 (INT_CFDATA5)

1. Please refer to the notes following the PRIOLVL[2:0] description below.

Address: 0x012E

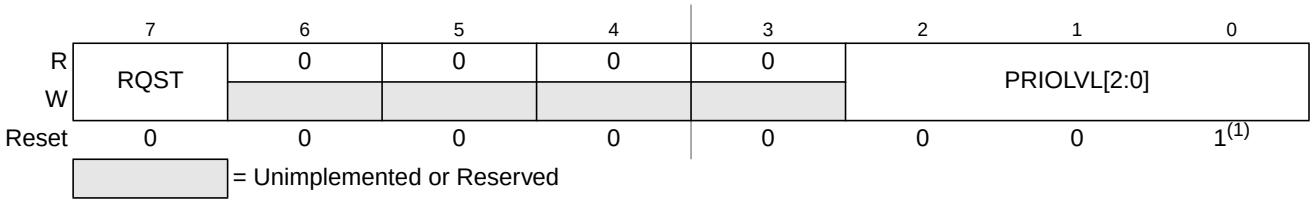


Figure 7-12. Interrupt Request Configuration Data Register 6 (INT_CFDATA6)

1. Please refer to the notes following the PRIOLVL[2:0] description below.

Address: 0x012F

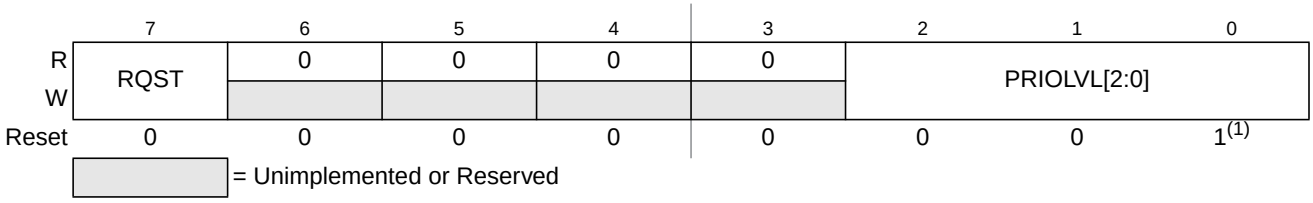


Figure 7-13. Interrupt Request Configuration Data Register 7 (INT_CFDATA7)

1. Please refer to the notes following the PRIOLVL[2:0] description below.

Read: Anytime

Write: Anytime

Table 7-8. INT_CFDATA0–7 Field Descriptions

Field	Description
7 RQST	**XGATE Request Enable** — This bit determines if the associated interrupt request is handled by the CPU or by the XGATE module. 0 Interrupt request is handled by the CPU 1 Interrupt request is handled by the XGATE module **Note:** The $\overline{\text{IRQ}}$ interrupt cannot be handled by the XGATE module. For this reason, the configuration register for vector (vector base + 0x00F2) = $\overline{\text{IRQ}}$ vector address) does not contain a RQST bit. Writing a 1 to the location of the RQST bit in this register will be ignored and a read access will return 0. **Note:** If the XGATE module is not available on the device, writing a 1 to the location of the RQST bit in this register will be ignored and a read access will return 0.
2–0 PRIOLVL[2:0]	**Interrupt Request Priority Level Bits** — The PRIOLVL[2:0] bits configure the interrupt request priority level of the associated interrupt request. Out of reset all interrupt requests are enabled at the lowest active level (“1”) to provide backwards compatibility with previous S12 interrupt controllers. Please also refer to Table 7-9 for available interrupt request priority levels. **Note:** Write accesses to configuration data registers of unused interrupt channels will be ignored and read accesses will return all 0. For information about what interrupt channels are used in a specific MCU, please refer to the Device Reference Manual of that MCU. **Note:** When vectors (vector base + 0x00F0–0x00FE) are selected by writing 0xF0 to INT_CFADDR, writes to INT_CFDATA2–7 (0x00F4–0x00FE) will be ignored and read accesses will return all 0s. The corresponding vectors do not have configuration data registers associated with them. **Note:** When vectors (vector base + 0x0010–0x001E) are selected by writing 0x10 to INT_CFADDR, writes to INT_CFDATA1–INT_CFDATA4 (0x0012–0x0018) will be ignored and read accesses will return all 0s. The corresponding vectors do not have configuration data registers associated with them. **Note:** Write accesses to the configuration register for the spurious interrupt vector request (vector base + 0x0010) will be ignored and read accesses will return 0x07 (request is handled by the CPU, PRIOLVL = 7).

Table 7-9. Interrupt Priority Levels

Priority	PRIOLVL2	PRIOLVL1	PRIOLVL0	Meaning
	0	0	0	Interrupt request is disabled
low	0	0	1	Priority level 1
	0	1	0	Priority level 2
	0	1	1	Priority level 3
	1	0	0	Priority level 4
	1	0	1	Priority level 5
	1	1	0	Priority level 6
high	1	1	1	Priority level 7

7.4 Functional Description

The INT module processes all exception requests to be serviced by the CPU module. These exceptions include interrupt vector requests and reset vector requests. Each of these exception types and their overall priority level is discussed in the subsections below.

7.4.1 S12X Exception Requests

The CPU handles both reset requests and interrupt requests. The INT module contains registers to configure the priority level of each I bit maskable interrupt request which can be used to implement an interrupt priority scheme. This also includes the possibility to nest interrupt requests. A priority decoder is used to evaluate the priority of a pending interrupt request.

7.4.2 Interrupt Prioritization

After system reset all interrupt requests with a vector address lower than or equal to (vector base + 0x00F2) are enabled, are set up to be handled by the CPU and have a pre-configured priority level of 1. Exceptions to this rule are the non-maskable interrupt requests and the spurious interrupt vector request at (vector base + 0x0010) which cannot be disabled, are always handled by the CPU and have a fixed priority levels. A priority level of 0 effectively disables the associated I bit maskable interrupt request.

If more than one interrupt request is configured to the same interrupt priority level the interrupt request with the higher vector address wins the prioritization.

The following conditions must be met for an I bit maskable interrupt request to be processed.

1. The local interrupt enabled bit in the peripheral module must be set.
2. The setup in the configuration register associated with the interrupt request channel must meet the following conditions:
   a) The XGATE request enable bit must be 0 to have the CPU handle the interrupt request.
   b) The priority level must be set to non zero.
   c) The priority level must be greater than the current interrupt processing level in the condition code register (CCR) of the CPU (PRIOLVL[2:0] > IPL[2:0]).
3. The I bit in the condition code register (CCR) of the CPU must be cleared.
4. There is no access violation interrupt request pending.
5. There is no SYS, SWI, BDM, TRAP, or $\overline{\text{XIRQ}}$ request pending.

NOTE

All non I bit maskable interrupt requests always have higher priority than I bit maskable interrupt requests. If an I bit maskable interrupt request is interrupted by a non I bit maskable interrupt request, the currently active interrupt processing level (IPL) remains unaffected. It is possible to nest non I bit maskable interrupt requests, e.g., by nesting SWI or TRAP calls.

7.4.2.1 Interrupt Priority Stack

The current interrupt processing level (IPL) is stored in the condition code register (CCR) of the CPU. This way the current IPL is automatically pushed to the stack by the standard interrupt stacking procedure. The new IPL is copied to the CCR from the priority level of the highest priority active interrupt request channel which is configured to be handled by the CPU. The copying takes place when the interrupt vector is fetched. The previous IPL is automatically restored by executing the RTI instruction.

7.4.3 XGATE Requests

If the XGATE module is implemented on the device, the INT module is also used to process all exception requests to be serviced by the XGATE module. The overall priority level of those exceptions is discussed in the subsections below.

7.4.3.1 XGATE Request Prioritization

An interrupt request channel is configured to be handled by the XGATE module, if the RQST bit of the associated configuration register is set to 1 (please refer to Section 7.3.2.4, “Interrupt Request Configuration Data Registers (INT_CFDATA0–7)”). The priority level configuration (PRIOLVL) for this channel becomes the XGATE priority which will be used to determine the highest priority XGATE request to be serviced next by the XGATE module. Additionally, XGATE interrupts may be raised by the XGATE module by setting one or more of the XGATE channel interrupt flags (by using the SIF instruction). This will result in an CPU interrupt with vector address vector base + (2 * channel ID number), where the channel ID number corresponds to the highest set channel interrupt flag, if the XGIE and channel RQST bits are set.

The shared interrupt priority for the XGATE interrupt requests is taken from the XGATE interrupt priority configuration register (please refer to Section 7.3.2.2, “XGATE Interrupt Priority Configuration Register (INT_XGPRIO)”). If more than one XGATE interrupt request channel becomes active at the same time, the channel with the highest vector address wins the prioritization.

7.4.4 Priority Decoders

The INT module contains priority decoders to determine the priority for all interrupt requests pending for the respective target.

There are two priority decoders, one for each interrupt request target, CPU or XGATE. The function of both priority decoders is basically the same with one exception: the priority decoder for the XGATE module does not take the current XGATE thread processing level into account. Instead, XGATE requests are handed to the XGATE module including a 1-bit priority identifier. The XGATE module uses this additional information to decide if the new request can interrupt a currently running thread. The 1-bit priority identifier corresponds to the most significant bit of the priority level configuration of the requesting channel. This means that XGATE requests with priority levels 4, 5, 6 or 7 can interrupt running XGATE threads with priority levels 1, 2 and 3.

A CPU interrupt vector is not supplied until the CPU requests it. Therefore, it is possible that a higher priority interrupt request could override the original exception which caused the CPU to request the vector. In this case, the CPU will receive the highest priority vector and the system will process this exception instead of the original request.

If the interrupt source is unknown (for example, in the case where an interrupt request becomes inactive after the interrupt has been recognized, but prior to the vector request), the vector address supplied to the CPU will default to that of the spurious interrupt vector.

NOTE

Care must be taken to ensure that all exception requests remain active until the system begins execution of the applicable service routine; otherwise, the exception request may not get processed at all or the result may be a spurious interrupt request (vector at address (vector base + 0x0010)).

7.4.5 Reset Exception Requests

The INT module supports three system reset exception request types (for details please refer to the Clock and Reset Generator module (CRG)):

1. Pin reset, power-on reset, low-voltage reset, or illegal address reset
2. Clock monitor reset request
3. COP watchdog reset request

7.4.6 Exception Priority

The priority (from highest to lowest) and address of all exception vectors issued by the INT module upon request by the CPU is shown in Table 7-10. Generally, all non-maskable interrupts have higher priorities than maskable interrupts. Please note that between the three software interrupts (Unimplemented op-code trap request, SWI/BGND request, SYS request) there is no real priority defined because they cannot occur simultaneously (the S12XCPU executes one instruction at a time).

Table 7-10. Exception Vector Map and Priority

Vector Address[1]	Source
0xFFFE	Pin reset, power-on reset, low-voltage reset, illegal address reset
0xFFFC	Clock monitor reset
0xFFFA	COP watchdog reset
(Vector base + 0x00F8)	Unimplemented op-code trap
(Vector base + 0x00F6)	Software interrupt instruction (SWI) or BDM vector request
(Vector base + 0x0012)	System call interrupt instruction (SYS)
(Vector base + 0x0018)	(reserved for future use)
(Vector base + 0x0016)	XGATE Access violation interrupt request[2]
(Vector base + 0x0014)	CPU Access violation interrupt request[3]
(Vector base + 0x00F4)	$\overline{\text{XIRQ}}$ interrupt request
(Vector base + 0x00F2)	$\overline{\text{IRQ}}$ interrupt request
(Vector base + 0x00F0–0x001A)	Device specific I bit maskable interrupt sources (priority determined by the associated configuration registers, in descending order)
(Vector base + 0x0010)	Spurious interrupt

1. 16 bits vector address based
2. only implemented if device features both a Memory Protection Unit (MPU) and an XGATE co-processor
3. only implemented if device features a Memory Protection Unit (MPU)

7.5 Initialization/Application Information

7.5.1 Initialization

After system reset, software should:

- Initialize the interrupt vector base register if the interrupt vector table is not located at the default location (0xFF10–0xFFF9).
- Initialize the interrupt processing level configuration data registers (INT_CFADDR, INT_CFDATA0–7) for all interrupt vector requests with the desired priority levels and the request target (CPU or XGATE module). It might be a good idea to disable unused interrupt requests.
- If the XGATE module is used, setup the XGATE interrupt priority register (INT_XGPRIO) and configure the XGATE module (please refer the XGATE Block Guide for details).
- Enable I maskable interrupts by clearing the I bit in the CCR.
- Enable the X maskable interrupt by clearing the X bit in the CCR (if required).

7.5.2 Interrupt Nesting

The interrupt request priority level scheme makes it possible to implement priority based interrupt request nesting for the I bit maskable interrupt requests handled by the CPU.

- I bit maskable interrupt requests can be interrupted by an interrupt request with a higher priority, so that there can be up to seven nested I bit maskable interrupt requests at a time (refer to Figure 7-14 for an example using up to three nested interrupt requests).

I bit maskable interrupt requests cannot be interrupted by other I bit maskable interrupt requests per default. In order to make an interrupt service routine (ISR) interruptible, the ISR must explicitly clear the I bit in the CCR (CLI). After clearing the I bit, I bit maskable interrupt requests with higher priority can interrupt the current ISR.

An ISR of an interruptible I bit maskable interrupt request could basically look like this:

- Service interrupt, e.g., clear interrupt flags, copy data, etc.
- Clear I bit in the CCR by executing the instruction CLI (thus allowing interrupt requests with higher priority)
- Process data
- Return from interrupt by executing the instruction RTI

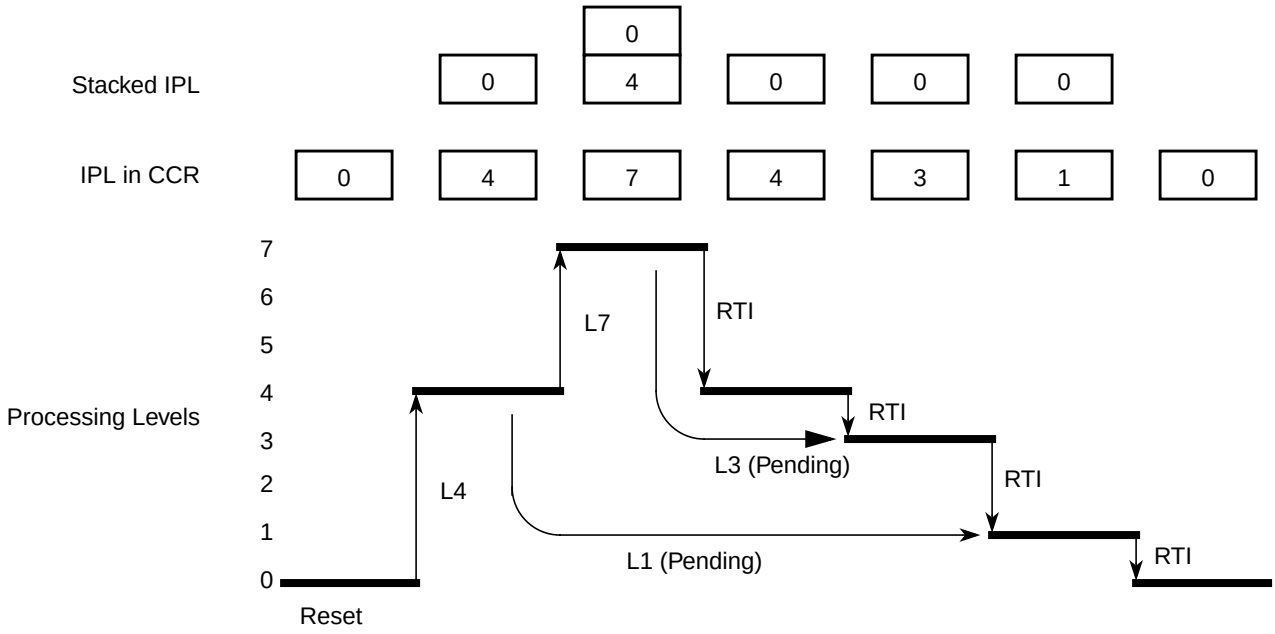


Figure 7-14. Interrupt Processing Example

7.5.3 Wake Up from Stop or Wait Mode

7.5.3.1 CPU Wake Up from Stop or Wait Mode

Every I bit maskable interrupt request which is configured to be handled by the CPU is capable of waking the MCU from stop or wait mode. To determine whether an I bit maskable interrupts is qualified to wake up the CPU or not, the same settings as in normal run mode are applied during stop or wait mode:

- If the I bit in the CCR is set, all I bit maskable interrupts are masked from waking up the MCU.
- An I bit maskable interrupt is ignored if it is configured to a priority level below or equal to the current IPL in CCR.
- I bit maskable interrupt requests which are configured to be handled by the XGATE module are not capable of waking up the CPU.

The X bit maskable interrupt request can wake up the MCU from stop or wait mode at anytime, even if the X bit in CCR is set.

If the X bit maskable interrupt request is used to wake-up the MCU with the X bit in the CCR set, the associated ISR is not called. The CPU then resumes program execution with the instruction following the WAI or STOP instruction. This features works following the same rules like any interrupt request, i.e. care must be taken that the X interrupt request used for wake-up remains active at least until the system begins execution of the instruction following the WAI or STOP instruction; otherwise, wake-up may not occur.

7.5.3.2 XGATE Wake Up from Stop or Wait Mode

Interrupt request channels which are configured to be handled by the XGATE module are capable of waking up the XGATE module. Interrupt request channels handled by the XGATE module do not affect the state of the CPU.

Chapter 8
256 KByte Flash Module (S12XFTM256K2XFV1)

Table 8-1. Revision History

Revision Number	Revision Date	Sections Affected	Description of Changes
V01.00	12 Dec 2007		- Initial version
V01.01	19 Dec 2007	8.4.2/8-257 8.4.2/8-257 8.4.2/8-257 8.4.2/8-257 8.3.1/8-226 8.1.3/8-224 8.1.2.4/8-224 8.3.1/8-226	- Removed Load Data Field command 0x05 - Updated Command Error Handling tables based on parent-child relationship with FTM512K3 - Corrected Error Handling table for Full Partition D-Flash, Partition D-Flash, and EEPROM Emulation Query commands - Corrected maximum allowed ERPART for Full Partition D-Flash and Partition D-Flash commands - Corrected P-Flash IFR Accessibility table - Corrected Tag RAM size in Block Diagram - Corrected Buffer RAM size in Feature List - Added EEE Resource Field table and Memory Map
V01.02	25 Sep 2009	8.1/8-221 8.3.2.1/8-233 8.4.2.4/8-260 8.4.2.6/8-261 8.4.2.11/8-265 8.4.2.11/8-265 8.4.2.11/8-265 8.4.2.19/8-274 8.3.2/8-231 8.3.2.1/8-233 8.4.1.2/8-252 8.6/8-280	- Clarify single bit fault correction for P-Flash phrase - Expand FDIV vs OSCCLK Frequency table - Add statement concerning code runaway when executing Read Once command from Flash block containing associated fields - Add statement concerning code runaway when executing Program Once command from Flash block containing associated fields - Add statement concerning code runaway when executing Verify Backdoor Access Key command from Flash block containing associated fields - Relate Key 0 to associated Backdoor Comparison Key address - Change “power down reset” to “reset” - Add ACCERR condition for Disable EEPROM Emulation command The following changes were made to clarify module behavior related to Flash register access during reset sequence and while Flash commands are active: - Add caution concerning register writes while command is active - Writes to FCLKDIV are allowed during reset sequence while CCIF is clear - Add caution concerning register writes while command is active - Writes to FCCOBIX, FCCOBHI, FCCOBLO registers are ignored during reset sequence

8.1 Introduction

The FTM256K2XF module implements the following:

- 256 Kbytes of P-Flash (Program Flash) memory, consisting of 2 physical Flash blocks, intended primarily for nonvolatile code storage

- 32 Kbytes of D-Flash (Data Flash) memory, consisting of 1 physical Flash block, that can be used as nonvolatile storage to support the built-in hardware scheme for emulated EEPROM, as basic Flash memory primarily intended for nonvolatile data storage, or as a combination of both
- 2 Kbytes of buffer RAM, consisting of 1 physical RAM block, that can be used as emulated EEPROM using a built-in hardware scheme, as basic RAM, or as a combination of both

The Flash memory is ideal for single-supply applications allowing for field reprogramming without requiring external high voltage sources for program or erase operations. The Flash module includes a memory controller that executes commands to modify Flash memory contents or configure module resources for emulated EEPROM operation. The user interface to the memory controller consists of the indexed Flash Common Command Object (FCCOB) register which is written to with the command, global address, data, and any required command parameters. The memory controller must complete the execution of a command before the FCCOB register can be written to with a new command.

CAUTION

A Flash word or phrase must be in the erased state before being programmed. Cumulative programming of bits within a Flash word or phrase is not allowed.

The RAM and Flash memory may be read as bytes, aligned words, or misaligned words. Read access time is one bus cycle for bytes and aligned words, and two bus cycles for misaligned words. For Flash memory, an erased bit reads 1 and a programmed bit reads 0.

It is not possible to read from a Flash block while any command is executing on that specific Flash block. It is possible to read from a Flash block while a command is executing on a different Flash block.

Both P-Flash and D-Flash memories are implemented with Error Correction Codes (ECC) that can resolve single bit faults and detect double bit faults. For P-Flash memory, the ECC implementation requires that programming be done on an aligned 8 byte basis (a Flash phrase). Since P-Flash memory is always read by phrase, only one single bit fault in the phrase containing the byte or word accessed will be corrected.

8.1.1 Glossary

Buffer RAM — The buffer RAM constitutes the volatile memory store required for EEE. Memory space in the buffer RAM not required for EEE can be partitioned to provide volatile memory space for applications.

Command Write Sequence — An MCU instruction sequence to execute built-in algorithms (including program and erase) on the Flash memory.

D-Flash Memory — The D-Flash memory constitutes the nonvolatile memory store required for EEE. Memory space in the D-Flash memory not required for EEE can be partitioned to provide nonvolatile memory space for applications.

D-Flash Sector — The D-Flash sector is the smallest portion of the D-Flash memory that can be erased. The D-Flash sector consists of four 64 byte rows for a total of 256 bytes.

EEE (Emulated EEPROM) — A method to emulate the small sector size features and endurance characteristics associated with an EEPROM.

EEE IFR — Nonvolatile information register located in the D-Flash block that contains data required to partition the D-Flash memory and buffer RAM for EEE. The EEE IFR is visible in the global memory map by setting the EEEIFRON bit in the MMCCTL1 register.

NVM Command Mode — An NVM mode using the CPU to setup the FCCOB register to pass parameters required for Flash command execution.

Phrase — An aligned group of four 16-bit words within the P-Flash memory. Each phrase includes eight ECC bits for single bit fault correction and double bit fault detection within the phrase.

P-Flash Memory — The P-Flash memory constitutes the main nonvolatile memory store for applications.

P-Flash Sector — The P-Flash sector is the smallest portion of the P-Flash memory that can be erased. Each P-Flash sector contains 1024 bytes.

Program IFR — Nonvolatile information register located in the P-Flash block that contains the Device ID, Version ID, and the Program Once field. The Program IFR is visible in the global memory map by setting the PGMIFRON bit in the MMCCTL1 register.

8.1.2 Features

8.1.2.1 P-Flash Features

- 256 Kbytes of P-Flash memory composed of two 128 Kbyte Flash blocks. The 128 Kbyte Flash blocks are each divided into 128 sectors of 1024 bytes.
- Single bit fault correction and double bit fault detection within a 64-bit phrase during read operations
- Automated program and erase algorithm with verify and generation of ECC parity bits
- Fast sector erase and phrase program operation
- Flexible protection scheme to prevent accidental program or erase of P-Flash memory

8.1.2.2 D-Flash Features

- Up to 32 Kbytes of D-Flash memory with 256 byte sectors for user access
- Dedicated commands to control access to the D-Flash memory over EEE operation
- Single bit fault correction and double bit fault detection within a word during read operations
- Automated program and erase algorithm with verify and generation of ECC parity bits
- Fast sector erase and word program operation
- Ability to program up to four words in a burst sequence

8.1.2.3 Emulated EEPROM Features

- Up to 2Kbytes of emulated EEPROM (EEE) accessible as 2 Kbytes of RAM
- Flexible protection scheme to prevent accidental program or erase of data
- Automatic EEE file handling using an internal Memory Controller
- Automatic transfer of valid EEE data from D-Flash memory to buffer RAM on reset
- Ability to monitor the number of outstanding EEE related buffer RAM words left to be programmed into D-Flash memory
- Ability to disable EEE operation and allow priority access to the D-Flash memory
- Ability to cancel all pending EEE operations and allow priority access to the D-Flash memory

8.1.2.4 User Buffer RAM Features

- Up to 2 Kbytes of RAM for user access

8.1.2.5 Other Flash Module Features

- No external high-voltage power supply required for Flash memory program and erase operations
- Interrupt generation on Flash command completion and Flash error detection
- Security mechanism to prevent unauthorized access to the Flash memory

8.1.3 Block Diagram

The block diagram of the Flash module is shown in Figure 8-1.

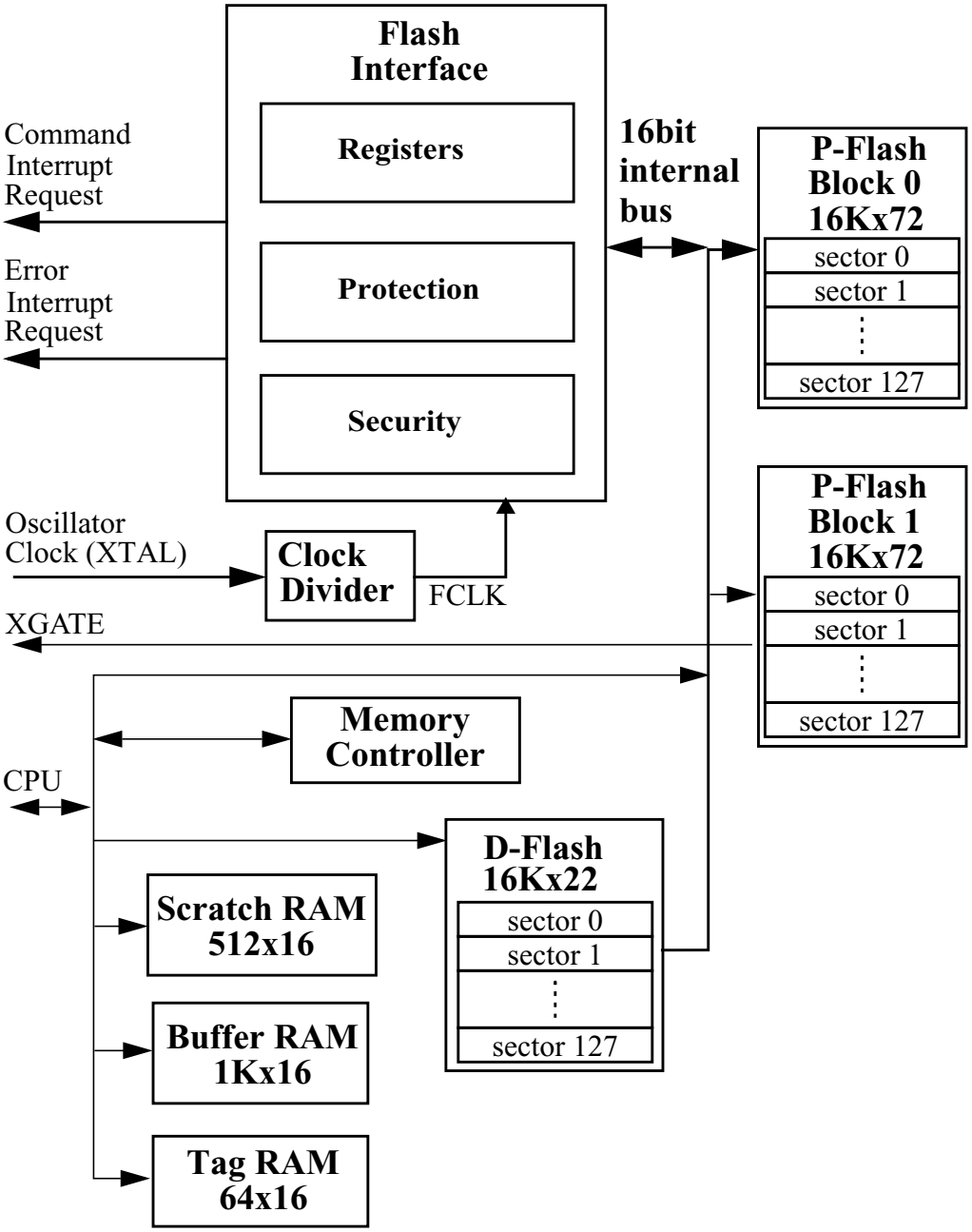


Figure 8-1. FTM256K2 Block Diagram

8.2 External Signal Description

The Flash module contains no signals that connect off-chip.

8.3 Memory Map and Registers

This section describes the memory map and registers for the Flash module. Read data from unimplemented memory space in the Flash module is undefined. Write access to unimplemented or reserved memory space in the Flash module will be ignored by the Flash module.

8.3.1 Module Memory Map

The S12X architecture places the P-Flash memory between global addresses 0x78_0000 and 0x7F_FFFF as shown in Table 8-2. The P-Flash memory map is shown in Figure 8-2.

Table 8-2. P-Flash Memory Addressing

Global Address	Size (Bytes)	Description
0x7E_0000 – 0x7F_FFFF	128 K	P-Flash Block 0 Contains Flash Configuration Field (see Table 8-3)
0x7A_0000 – 0x7D_FFFF	256 K	No P-Flash Memory
0x78_0000 – 0x79_FFFF	128 K	P-Flash Block 1

The FPROT register, described in Section 8.3.2.9, can be set to protect regions in the Flash memory from accidental program or erase. Three separate memory regions, one growing upward from global address 0x7F_8000 in the Flash memory (called the lower region), one growing downward from global address 0x7F_FFFF in the Flash memory (called the higher region), and the remaining addresses in the Flash memory, can be activated for protection. The Flash memory addresses covered by these protectable regions are shown in the P-Flash memory map. The higher address region is mainly targeted to hold the boot loader code since it covers the vector space. Default protection settings as well as security information that allows the MCU to restrict access to the Flash module are stored in the Flash configuration field as described in Table 8-3.

Table 8-3. Flash Configuration Field(1)

Global Address	Size (Bytes)	Description
0x7F_FF00 – 0x7F_FF07	8	Backdoor Comparison Key Refer to Section 8.4.2.11, “Verify Backdoor Access Key Command,” and Section 8.5.1, “Unsecuring the MCU using Backdoor Key Access”
0x7F_FF08 – 0x7F_FF0B(2)	4	Reserved
0x7F_FF0C[2]	1	P-Flash Protection byte. Refer to Section 8.3.2.9, “P-Flash Protection Register (FPROT)”
0x7F_FF0D[2]	1	EEE Protection byte Refer to Section 8.3.2.10, “EEE Protection Register (EPROT)”
0x7F_FF0E[2]	1	Flash Nonvolatile byte Refer to Section 8.3.2.14, “Flash Option Register (FOPT)”

Table 8-3. Flash Configuration Field(1)

Global Address	Size (Bytes)	Description
0x7F_FF0F[2]	1	Flash Security byte Refer to Section 8.3.2.2, “Flash Security Register (FSEC)”

1. Older versions may have swapped protection byte addresses
2. 0x7FF08 - 0x7F_FF0F form a Flash phrase and must be programmed in a single command write sequence. Each byte in the 0x7F_FF08 - 0x7F_FF0B reserved field should be programmed to 0xFF.

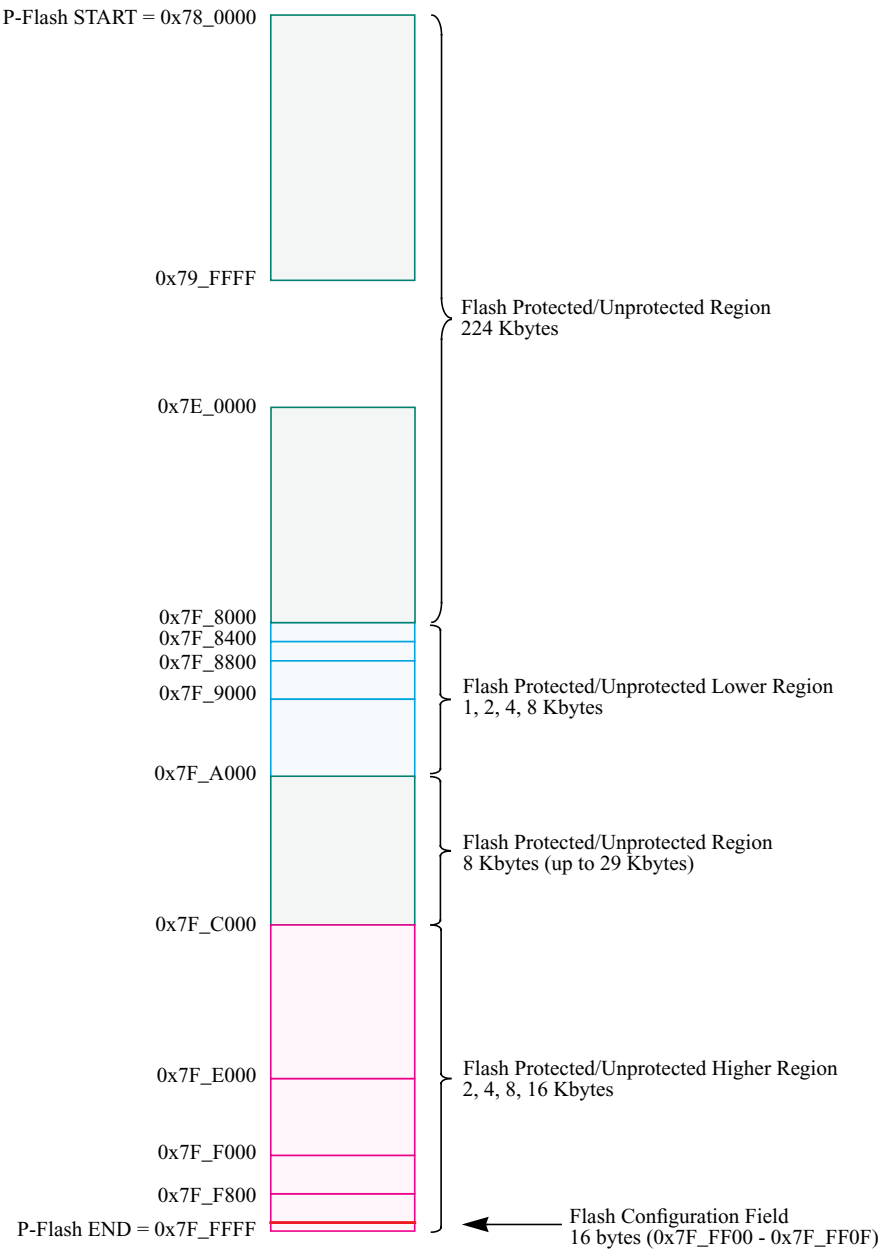


Figure 8-2. P-Flash Memory Map

Table 8-4. Program IFR Fields

Global Address (PGMIFRON)	Size (Bytes)	Field Description
0x40_0000 – 0x40_0007	8	Device ID
0x40_0008 – 0x40_00E7	224	Reserved
0x40_00E8 – 0x40_00E9	2	Version ID
0x40_00EA – 0x40_00FF	22	Reserved
0x40_0100 – 0x40_013F	64	Program Once Field Refer to Section 8.4.2.6, “Program Once Command”
0x40_0140 – 0x40_01FF	192	Reserved

Table 8-5. P-Flash IFR Accessibility

Global Address (PGMIFRON)	Size (Bytes)	Accessed From
0x40_0000 – 0x40_01FF	512	XBUS0 (PBLK0)(1)
0x40_0200 – 0x40_03FF	512	Unimplemented
0x40_0400 – 0x40_05FF	512	Unimplemented
0x40_0600 – 0x40_07FF	512	XBUS1 (PBLK1)

1. Refer to Table 8-4 for more details.

Table 8-6. EEE Resource Fields

Global Address	Size (Bytes)	Description
0x10_0000 – 0x10_7FFF	32,768	D-Flash Memory (User and EEE)
0x10_8000 – 0x11_FFFF	98,304	Reserved
0x12_0000 – 0x12_007F	128	EEE Nonvolatile Information Register (EEEIFRON(1) = 1)
0x12_0080 – 0x12_0FFF	3,968	Reserved
0x12_1000 – 0x12_1F7F	3,968	Reserved
0x12_1F80 – 0x12_1FFF	128	EEE Tag RAM (TMGRAMON1 = 1)
0x12_2000 – 0x12_3BFF	7,168	Reserved
0x12_3C00 – 0x12_3FFF	1,024	Memory Controller Scratch RAM (TMGRAMON1 = 1)
0x12_4000 – 0x12_DFFF	40,960	Reserved
0x12_E000 – 0x12_FFFF	8,192	Reserved
0x13_0000 – 0x13_F7FF	63,488	Reserved
0x13_F800 – 0x13_FFFF	2,048	Buffer RAM (User and EEE)

1. MMCCTL1 register bit

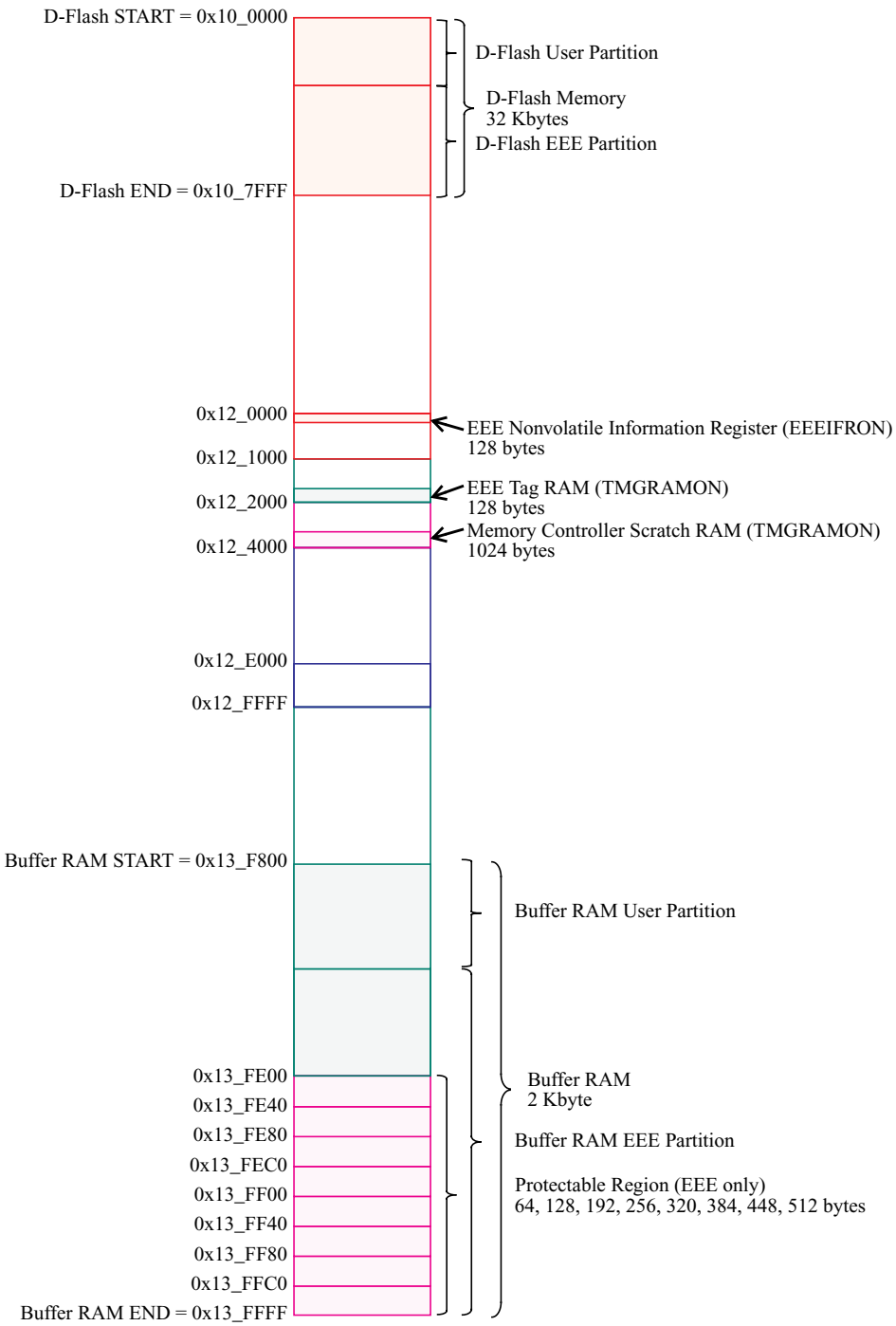


Figure 8-3. EEE Resource Memory Map

The Full Partition D-Flash command (see Section 8.4.2.14) is used to program the EEE nonvolatile information register fields where address 0x12_0000 defines the D-Flash partition for user access and address 0x12_0004 defines the buffer RAM partition for EEE operations.

Table 8-7. EEE Nonvolatile Information Register Fields

Global Address (EEEIFRON)	Size (Bytes)	Description
0x12_0000 – 0x12_0001	2	D-Flash User Partition (DFPART) Refer to Section 8.4.2.14, “Full Partition D-Flash Command”
0x12_0002 – 0x12_0003	2	D-Flash User Partition (duplicate(1))
0x12_0004 – 0x12_0005	2	Buffer RAM EEE Partition (ERPART) Refer to Section 8.4.2.14, “Full Partition D-Flash Command”
0x12_0006 – 0x12_0007	2	Buffer RAM EEE Partition (duplicate[1])
0x12_0008 – 0x12_007F	120	Reserved

1. Duplicate value used if primary value generates a double bit fault when read during the reset sequence.

8.3.2 Register Descriptions

The Flash module contains a set of 20 control and status registers located between Flash module base + 0x0000 and 0x0013. A summary of the Flash module registers is given in Figure 8-4 with detailed descriptions in the following subsections.

CAUTION

Writes to any Flash register must be avoided while a Flash command is active (CCIF=0) to prevent corruption of Flash register contents and Memory Controller behavior.

Address & Name		7	6	5	4	3	2	1	0
0x0000 FCLKDIV	R	FDIVLD	FDIV6	FDIV5	FDIV4	FDIV3	FDIV2	FDIV1	FDIV0
	W								
0x0001 FSEC	R	KEYEN1	KEYEN0	RNV5	RNV4	RNV3	RNV2	SEC1	SEC0
	W								
0x0002 FCCOBIX	R	0	0	0	0	0	CCOBIX2	CCOBIX1	CCOBIX0
	W								
0x0003 FECCRIX	R	0	0	0	0	0	ECCRIX2	ECCRIX1	ECCRIX0
	W								
0x0004 FCNFG	R	CCIE	0	0	IGNSF	0	0	FDFD	FSFD
	W								

Figure 8-4. FTM256K2XF Register Summary

Address & Name		7	6	5	4	3	2	1	0
0x0005 FERCNFG	R	ERSERIE	PGMERIE	0	EPVIOLIE	ERSVIE1	ERSVIE0	DFDIE	SFDIE
	W								
0x0006 FSTAT	R	CCIF	0	ACCERR	FPVIOL	MGBUSY	RSVD	MGSTAT1	MGSTAT0
	W								
0x0007 FERSTAT	R	ERSERIF	PGMERIF	0	EPVIOLIF	ERSVIF1	ERSVIF0	DFDIF	SFDIF
	W								
0x0008 FPROT	R	FPOPEN	RNV6	FPHDIS	FPHS1	FPHS0	FPLDIS	FPLS1	FPLS0
	W								
0x0009 EPROT	R	EPOPEN	RNV6	RNV5	RNV4	EPDIS	EPS2	EPS1	EPS0
	W								
0x000A FCCOBHI	R	CCOB15	CCOB14	CCOB13	CCOB12	CCOB11	CCOB10	CCOB9	CCOB8
	W								
0x000B FCCOBLO	R	CCOB7	CCOB6	CCOB5	CCOB4	CCOB3	CCOB2	CCOB1	CCOB0
	W								
0x000C ETAGHI	R	ETAG15	ETAG14	ETAG13	ETAG12	ETAG11	ETAG10	ETAG9	ETAG8
	W								
0x000D ETAGLO	R	ETAG7	ETAG6	ETAG5	ETAG4	ETAG3	ETAG2	ETAG1	ETAG0
	W								
0x000E FECCRHI	R	ECCR15	ECCR14	ECCR13	ECCR12	ECCR11	ECCR10	ECCR9	ECCR8
	W								
0x000F FECCRLO	R	ECCR7	ECCR6	ECCR5	ECCR4	ECCR3	ECCR2	ECCR1	ECCR0
	W								
0x0010 FOPT	R	NV7	NV6	NV5	NV4	NV3	NV2	NV1	NV0
	W								
0x0011 FRSV0	R	0	0	0	0	0	0	0	0
	W								
0x0012 FRSV1	R	0	0	0	0	0	0	0	0
	W								

Figure 8-4. FTM256K2XF Register Summary (continued)

Address & Name		7	6	5	4	3	2	1	0
0x0013 FRSV2	R	0	0	0	0	0	0	0	0
	W								

= Unimplemented or Reserved

Figure 8-4. FTM256K2XF Register Summary (continued)

8.3.2.1 Flash Clock Divider Register (FCLKDIV)

The FCLKDIV register is used to control timed events in program and erase algorithms.

Offset Module Base + 0x0000

	7	6	5	4	3	2	1	0
R	FDIVLD	FDIV[6:0]						
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 8-5. Flash Clock Divider Register (FCLKDIV)

All bits in the FCLKDIV register are readable, bits 6–0 are write once and bit 7 is not writable.

Table 8-8. FCLKDIV Field Descriptions

Field	Description
7 FDIVLD	**Clock Divider Loaded** 0 FCLKDIV register has not been written 1 FCLKDIV register has been written since the last reset
6–0 FDIV[6:0]	**Clock Divider Bits** — FDIV[6:0] must be set to effectively divide OSCCLK down to generate an internal Flash clock, FCLK, with a target frequency of 1 MHz for use by the Flash module to control timed events during program and erase algorithms. Table 8-9 shows recommended values for FDIV[6:0] based on OSCCLK frequency. Please refer to Section 8.4.1, “Flash Command Operations,” for more information.

CAUTION

The FCLKDIV register should never be written while a Flash command is executing (CCIF=0). The FCLKDIV register is writable during the Flash reset sequence even though CCIF is clear.

Table 8-9. FDIV vs OSCCLK Frequency

OSCCLK Frequency (MHz)		FDIV[6:0]	OSCCLK Frequency (MHz)		FDIV[6:0]	OSCCLK Frequency (MHz)		FDIV[6:0]
MIN(1)	MAX(2)		MIN1	MAX2		MIN1	MAX2	
			33.60	34.65	0x20	67.20	68.25	0x40
1.60	2.10	0x01	34.65	35.70	0x21	68.25	69.30	0x41
2.40	3.15	0x02	35.70	36.75	0x22	69.30	70.35	0x42
3.20	4.20	0x03	36.75	37.80	0x23	70.35	71.40	0x43
4.20	5.25	0x04	37.80	38.85	0x24	71.40	72.45	0x44
5.25	6.30	0x05	38.85	39.90	0x25	72.45	73.50	0x45
6.30	7.35	0x06	39.90	40.95	0x26	73.50	74.55	0x46
7.35	8.40	0x07	40.95	42.00	0x27	74.55	75.60	0x47
8.40	9.45	0x08	42.00	43.05	0x28	75.60	76.65	0x48
9.45	10.50	0x09	43.05	44.10	0x29	76.65	77.70	0x49
10.50	11.55	0x0A	44.10	45.15	0x2A	77.70	78.75	0x4A
11.55	12.60	0x0B	45.15	46.20	0x2B	78.75	79.80	0x4B
12.60	13.65	0x0C	46.20	47.25	0x2C	79.80	80.85	0x4C
13.65	14.70	0x0D	47.25	48.30	0x2D	80.85	81.90	0x4D
14.70	15.75	0x0E	48.30	49.35	0x2E	81.90	82.95	0x4E
15.75	16.80	0x0F	49.35	50.40	0x2F	82.95	84.00	0x4F
16.80	17.85	0x10	50.40	51.45	0x30	84.00	85.05	0x50
17.85	18.90	0x11	51.45	52.50	0x31	85.05	86.10	0x51
18.90	19.95	0x12	52.50	53.55	0x32	86.10	87.15	0x52
19.95	21.00	0x13	53.55	54.60	0x33	87.15	88.20	0x53
21.00	22.05	0x14	54.60	55.65	0x34	88.20	89.25	0x54
22.05	23.10	0x15	55.65	56.70	0x35	89.25	90.30	0x55
23.10	24.15	0x16	56.70	57.75	0x36	90.30	91.35	0x56
24.15	25.20	0x17	57.75	58.80	0x37	91.35	92.40	0x57
25.20	26.25	0x18	58.80	59.85	0x38	92.40	93.45	0x58
26.25	27.30	0x19	59.85	60.90	0x39	93.45	94.50	0x59
27.30	28.35	0x1A	60.90	61.95	0x3A	94.50	95.55	0x5A
28.35	29.40	0x1B	61.95	63.00	0x3B	95.55	96.60	0x5B
29.40	30.45	0x1C	63.00	64.05	0x3C	96.60	97.65	0x5C
30.45	31.50	0x1D	64.05	65.10	0x3D	97.65	98.70	0x5D
31.50	32.55	0x1E	65.10	66.15	0x3E	98.70	99.75	0x5E
32.55	33.60	0x1F	66.15	67.20	0x3F	99.75	100.80	0x5F

1. FDIV shown generates an FCLK frequency of >0.8 MHz

2. FDIV shown generates an FCLK frequency of 1.05 MHz

8.3.2.2 Flash Security Register (FSEC)

The FSEC register holds all bits associated with the security of the MCU and Flash module.

Offset Module Base + 0x0001

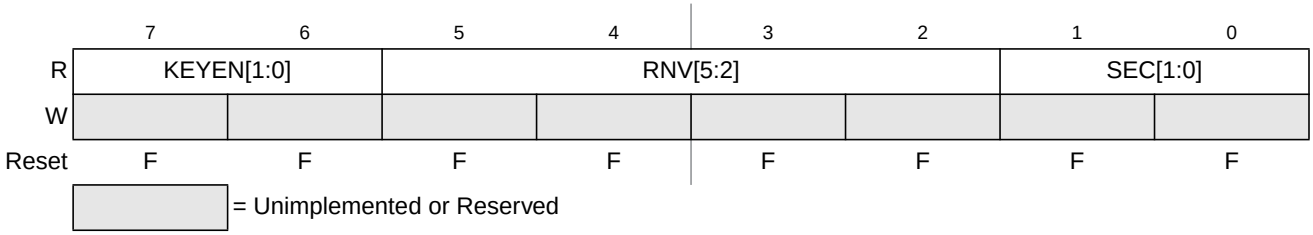


Figure 8-6. Flash Security Register (FSEC)

All bits in the FSEC register are readable but not writable.

During the reset sequence, the FSEC register is loaded with the contents of the Flash security byte in the Flash configuration field at global address 0x7F_FF0F located in P-Flash memory (see Table 8-3) as indicated by reset condition F in Figure 8-6. If a double bit fault is detected while reading the P-Flash phrase containing the Flash security byte during the reset sequence, all bits in the FSEC register will be set to leave the Flash module in a secured state with backdoor key access disabled.

Table 8-10. FSEC Field Descriptions

Field	Description
7–6 KEYEN[1:0]	**Backdoor Key Security Enable Bits** — The KEYEN[1:0] bits define the enabling of backdoor key access to the Flash module as shown in Table 8-11.
5–2 RNV[5:2}	**Reserved Nonvolatile Bits** — The RNV bits should remain in the erased state for future enhancements.
1–0 SEC[1:0]	**Flash Security Bits** — The SEC[1:0] bits define the security state of the MCU as shown in Table 8-12. If the Flash module is unsecured using backdoor key access, the SEC bits are forced to 10.

Table 8-11. Flash KEYEN States

KEYEN[1:0]	Status of Backdoor Key Access
00	DISABLED
01	DISABLED(1)
10	ENABLED
11	DISABLED

1. Preferred KEYEN state to disable backdoor key access.

Table 8-12. Flash Security States

SEC[1:0]	Status of Security
00	SECURED
01	SECURED(1)
10	UNSECURED
11	SECURED

1. Preferred SEC state to set MCU to secured state.

The security function in the Flash module is described in Section 8.5.

8.3.2.3 Flash CCOB Index Register (FCCOBIX)

The FCCOBIX register is used to index the FCCOB register for Flash memory operations.

Offset Module Base + 0x0002

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	CCOBIX[2:0]		
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 8-7. FCCOB Index Register (FCCOBIX)

CCOBIX bits are readable and writable while remaining bits read 0 and are not writable.

Table 8-13. FCCOBIX Field Descriptions

Field	Description
2–0 CCOBIX[1:0]	**Common Command Register Index**— The CCOBIX bits are used to select which word of the FCCOB register array is being read or written to. See Section 8.3.2.11, “Flash Common Command Object Register (FCCOB),” for more details.

8.3.2.4 Flash ECCR Index Register (FECCRIX)

The FECCRIX register is used to index the FECCR register for ECC fault reporting.

Offset Module Base + 0x0003

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	ECCRIX[2:0]		
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 8-8. FECCR Index Register (FECCRIX)

ECCRIX bits are readable and writable while remaining bits read 0 and are not writable.

Table 8-14. FECCRIX Field Descriptions

Field	Description
2-0 ECCRIX[2:0]	**ECC Error Register Index**— The ECCRIX bits are used to select which word of the FECCR register array is being read. See Section 8.3.2.13, “Flash ECC Error Results Register (FECCR),” for more details.

8.3.2.5 Flash Configuration Register (FCNFG)

The FCNFG register enables the Flash command complete interrupt and forces ECC faults on Flash array read access from the CPU or XGATE.

Offset Module Base + 0x0004

	7	6	5	4	3	2	1	0
R	CCIE	0	0	IGNSF	0	0	FDFD	FSFD
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 8-9. Flash Configuration Register (FCNFG)

CCIE, IGNSF, FDFD, and FSFD bits are readable and writable while remaining bits read 0 and are not writable.

Table 8-15. FCNFG Field Descriptions

Field	Description
7 CCIE	**Command Complete Interrupt Enable** — The CCIE bit controls interrupt generation when a Flash command has completed. 0 Command complete interrupt disabled 1 An interrupt will be requested whenever the CCIF flag in the FSTAT register is set (see Section 8.3.2.7)
4 IGNSF	**Ignore Single Bit Fault** — The IGNSF controls single bit fault reporting in the FERSTAT register (see Section 8.3.2.8). 0 All single bit faults detected during array reads are reported 1 Single bit faults detected during array reads are not reported and the single bit fault interrupt will not be generated

Table 8-15. FCNFG Field Descriptions (continued)

Field	Description
1 FDFD	**Force Double Bit Fault Detect** — The FDFD bit allows the user to simulate a double bit fault during Flash array read operations and check the associated interrupt routine. The FDFD bit is cleared by writing a 0 to FDFD. The FECCR registers will not be updated during the Flash array read operation with FDFD set unless an actual double bit fault is detected. 0 Flash array read operations will set the DFDIF flag in the FERSTAT register only if a double bit fault is detected 1 Any Flash array read operation will force the DFDIF flag in the FERSTAT register to be set (see Section 8.3.2.7) and an interrupt will be generated as long as the DFDIE interrupt enable in the FERCNFG register is set (see Section 8.3.2.6)
0 FSFD	**Force Single Bit Fault Detect** — The FSFD bit allows the user to simulate a single bit fault during Flash array read operations and check the associated interrupt routine. The FSFD bit is cleared by writing a 0 to FSFD. The FECCR registers will not be updated during the Flash array read operation with FSFD set unless an actual single bit fault is detected. 0 Flash array read operations will set the SFDIF flag in the FERSTAT register only if a single bit fault is detected 1 Flash array read operation will force the SFDIF flag in the FERSTAT register to be set (see Section 8.3.2.7) and an interrupt will be generated as long as the SFDIE interrupt enable in the FERCNFG register is set (see Section 8.3.2.6)

8.3.2.6 Flash Error Configuration Register (FERCNFG)

The FERCNFG register enables the Flash error interrupts for the FERSTAT flags.

Offset Module Base + 0x0005

	7	6	5	4	3	2	1	0
R	ERSERIE	PGMERIE	0	EPVIOLIE	ERSVIE1	ERSVIE0	DFDIE	SFDIE
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 8-10. Flash Error Configuration Register (FERCNFG)

All assigned bits in the FERCNFG register are readable and writable.

Table 8-16. FERCNFG Field Descriptions

Field	Description
7 ERSERIE	**EEE Erase Error Interrupt Enable** — The ERSERIE bit controls interrupt generation when a failure is detected during an EEE erase operation. 0 ERSERIF interrupt disabled 1 An interrupt will be requested whenever the ERSERIF flag is set (see Section 8.3.2.8)
6 PGMERIE	**EEE Program Error Interrupt Enable** — The PGMERIE bit controls interrupt generation when a failure is detected during an EEE program operation. 0 PGMERIF interrupt disabled 1 An interrupt will be requested whenever the PGMERIF flag is set (see Section 8.3.2.8)
4 EPVIOLIE	**EEE Protection Violation Interrupt Enable** — The EPVIOLIE bit controls interrupt generation when a protection violation is detected during a write to the buffer RAM EEE partition. 0 EPVIOLIF interrupt disabled 1 An interrupt will be requested whenever the EPVIOLIF flag is set (see Section 8.3.2.8)

Table 8-16. FERCNFG Field Descriptions (continued)

Field	Description
3 ERSVIE1	**EEE Error Type 1 Interrupt Enable** — The ERSVIE1 bit controls interrupt generation when a change state error is detected during an EEE operation. 0 ERSVIF1 interrupt disabled 1 An interrupt will be requested whenever the ERSVIF1 flag is set (see Section 8.3.2.8)
2 ERSVIE0	**EEE Error Type 0 Interrupt Enable** — The ERSVIE0 bit controls interrupt generation when a sector format error is detected during an EEE operation. 0 ERSVIF0 interrupt disabled 1 An interrupt will be requested whenever the ERSVIF0 flag is set (see Section 8.3.2.8)
1 DFDIE	**Double Bit Fault Detect Interrupt Enable** — The DFDIE bit controls interrupt generation when a double bit fault is detected during a Flash block read operation. 0 DFDIF interrupt disabled 1 An interrupt will be requested whenever the DFDIF flag is set (see Section 8.3.2.8)
0 SFDIE	**Single Bit Fault Detect Interrupt Enable** — The SFDIE bit controls interrupt generation when a single bit fault is detected during a Flash block read operation. 0 SFDIF interrupt disabled whenever the SFDIF flag is set (see Section 8.3.2.8) 1 An interrupt will be requested whenever the SFDIF flag is set (see Section 8.3.2.8)

8.3.2.7 Flash Status Register (FSTAT)

The FSTAT register reports the operational status of the Flash module.

Offset Module Base + 0x0006

	7	6	5	4	3	2	1	0
R	CCIF	0	ACCERR	FPVIOL	MGBUSY	RSVD	MGSTAT[1:0]	
W								
Reset	1	0	0	0	0	0	0(1)	0[1]

= Unimplemented or Reserved

Figure 8-11. Flash Status Register (FSTAT)

1. Reset value can deviate from the value shown if a double bit fault is detected during the reset sequence (see Section 8.6).

CCIF, ACCERR, and FPVIOL bits are readable and writable, MGBUSY and MGSTAT bits are readable but not writable, while remaining bits read 0 and are not writable.

Table 8-17. FSTAT Field Descriptions

Field	Description
7 CCIF	**Command Complete Interrupt Flag** — The CCIF flag indicates that a Flash command has completed. The CCIF flag is cleared by writing a 1 to CCIF to launch a command and CCIF will stay low until command completion or command violation. 0 Flash command in progress 1 Flash command has completed
5 ACCERR	**Flash Access Error Flag** — The ACCERR bit indicates an illegal access has occurred to the Flash memory caused by either a violation of the command write sequence (see Section 8.4.1.2) or issuing an illegal Flash command or when errors are encountered while initializing the EEE buffer ram during the reset sequence. While ACCERR is set, the CCIF flag cannot be cleared to launch a command. The ACCERR bit is cleared by writing a 1 to ACCERR. Writing a 0 to the ACCERR bit has no effect on ACCERR. 0 No access error detected 1 Access error detected
4 FPVIOL	**Flash Protection Violation Flag** —The FPVIOL bit indicates an attempt was made to program or erase an address in a protected area of P-Flash memory during a command write sequence. The FPVIOL bit is cleared by writing a 1 to FPVIOL. Writing a 0 to the FPVIOL bit has no effect on FPVIOL. While FPVIOL is set, it is not possible to launch a command or start a command write sequence. 0 No protection violation detected 1 Protection violation detected
3 MGBUSY	**Memory Controller Busy Flag** — The MGBUSY flag reflects the active state of the Memory Controller. 0 Memory Controller is idle 1 Memory Controller is busy executing a Flash command (CCIF = 0) or is handling internal EEE operations
2 RSVD	**Reserved Bit** — This bit is reserved and always reads 0.
1–0 MGSTAT[1:0]	**Memory Controller Command Completion Status Flag** — One or more MGSTAT flag bits are set if an error is detected during execution of a Flash command or during the Flash reset sequence. See Section 8.4.2, “Flash Command Description,” and Section 8.6, “Initialization” for details.

8.3.2.8 Flash Error Status Register (FERSTAT)

The FERSTAT register reflects the error status of internal Flash operations.

Offset Module Base + 0x0007

	7	6	5	4	3	2	1	0
R	ERSERIF	PGMERIF	0	EPVIOLIF	ERSVIF1	ERSVIF0	DFDIF	SFDIF
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 8-12. Flash Error Status Register (FERSTAT)

All flags in the FERSTAT register are readable and only writable to clear the flag.

Table 8-18. FERSTAT Field Descriptions

Field	Description
7 ERSERIF	**EEE Erase Error Interrupt Flag** — The setting of the ERSERIF flag occurs due to an error in a Flash erase command that resulted in the erase operation not being successful during EEE operations. The ERSERIF flag is cleared by writing a 1 to ERSERIF. Writing a 0 to the ERSERIF flag has no effect on ERSERIF. While ERSERIF is set, it is possible to write to the buffer RAM EEE partition but the data written will not be transferred to the D-Flash EEE partition. 0 Erase command successfully completed on the D-Flash EEE partition 1 Erase command failed on the D-Flash EEE partition
6 PGMERIF	**EEE Program Error Interrupt Flag** — The setting of the PGMERIF flag occurs due to an error in a Flash program command that resulted in the program operation not being successful during EEE operations. The PGMERIF flag is cleared by writing a 1 to PGMERIF. Writing a 0 to the PGMERIF flag has no effect on PGMERIF. While PGMERIF is set, it is possible to write to the buffer RAM EEE partition but the data written will not be transferred to the D-Flash EEE partition. 0 Program command successfully completed on the D-Flash EEE partition 1 Program command failed on the D-Flash EEE partition
4 EPVIOLIF	**EEE Protection Violation Interrupt Flag** —The setting of the EPVIOLIF flag indicates an attempt was made to write to a protected area of the buffer RAM EEE partition. The EPVIOLIF flag is cleared by writing a 1 to EPVIOLIF. Writing a 0 to the EPVIOLIF flag has no effect on EPVIOLIF. While EPVIOLIF is set, it is possible to write to the buffer RAM EEE partition as long as the address written to is not in a protected area. 0 No EEE protection violation 1 EEE protection violation detected
3 ERSVIF1	**EEE Error Interrupt 1 Flag** —The setting of the ERSVIF1 flag indicates that the memory controller was unable to change the state of a D-Flash EEE sector. The ERSVIF1 flag is cleared by writing a 1 to ERSVIF1. Writing a 0 to the ERSVIF1 flag has no effect on ERSVIF1. While ERSVIF1 is set, it is possible to write to the buffer RAM EEE partition but the data written will not be transferred to the D-Flash EEE partition. 0 No EEE sector state change error detected 1 EEE sector state change error detected
2 ERSVIF0	**EEE Error Interrupt 0 Flag** —The setting of the ERSVIF0 flag indicates that the memory controller was unable to format a D-Flash EEE sector for EEE use. The ERSVIF0 flag is cleared by writing a 1 to ERSVIF0. Writing a 0 to the ERSVIF0 flag has no effect on ERSVIF0. While ERSVIF0 is set, it is possible to write to the buffer RAM EEE partition but the data written will not be transferred to the D-Flash EEE partition. 0 No EEE sector format error detected 1 EEE sector format error detected
1 DFDIF	**Double Bit Fault Detect Interrupt Flag** — The setting of the DFDIF flag indicates that a double bit fault was detected in the stored parity and data bits during a Flash array read operation or that a Flash array read operation was attempted on a Flash block that was under a Flash command operation. The DFDIF flag is cleared by writing a 1 to DFDIF. Writing a 0 to DFDIF has no effect on DFDIF. 0 No double bit fault detected 1 Double bit fault detected or an invalid Flash array read operation attempted
0 SFDIF	**Single Bit Fault Detect Interrupt Flag** — With the IGNSF bit in the FCNFG register clear, the SFDIF flag indicates that a single bit fault was detected in the stored parity and data bits during a Flash array read operation or that a Flash array read operation was attempted on a Flash block that was under a Flash command operation. The SFDIF flag is cleared by writing a 1 to SFDIF. Writing a 0 to SFDIF has no effect on SFDIF. 0 No single bit fault detected 1 Single bit fault detected and corrected or an invalid Flash array read operation attempted

8.3.2.9 P-Flash Protection Register (FPROT)

The FPROT register defines which P-Flash sectors are protected against program and erase operations.

Offset Module Base + 0x0008

	7	6	5	4	3	2	1	0
R	FPOPEN	RNV6	FPHDIS	FPHS[1:0]		FPLDIS	FPLS[1:0]	
W								
Reset	F	F	F	F	F	F	F	F

= Unimplemented or Reserved

Figure 8-13. Flash Protection Register (FPROT)

The (unreserved) bits of the FPROT register are writable with the restriction that the size of the protected region can only be increased (see Section 8.3.2.9.1, "P-Flash Protection Restrictions," and Table 8-23).

During the reset sequence, the FPROT register is loaded with the contents of the P-Flash protection byte in the Flash configuration field at global address 0x7F_FF0C located in P-Flash memory (see Table 8-3) as indicated by reset condition 'F' in Figure 8-13. To change the P-Flash protection that will be loaded during the reset sequence, the upper sector of the P-Flash memory must be unprotected, then the P-Flash protection byte must be reprogrammed. If a double bit fault is detected while reading the P-Flash phrase containing the P-Flash protection byte during the reset sequence, the FPOPEN bit will be cleared and remaining bits in the FPROT register will be set to leave the P-Flash memory fully protected.

Trying to alter data in any protected area in the P-Flash memory will result in a protection violation error and the FPVIOL bit will be set in the FSTAT register. The block erase of a P-Flash block is not possible if any of the P-Flash sectors contained in the same P-Flash block are protected.

Table 8-19. FPROT Field Descriptions

Field	Description
7 FPOPEN	**Flash Protection Operation Enable** — The FPOPEN bit determines the protection function for program or erase operations as shown in Table 8-20 for the P-Flash block. 0 When FPOPEN is clear, the FPHDIS and FPLDIS bits define unprotected address ranges as specified by the corresponding FPHS and FPLS bits 1 When FPOPEN is set, the FPHDIS and FPLDIS bits enable protection for the address range specified by the corresponding FPHS and FPLS bits
6 RNV[6]	**Reserved Nonvolatile Bit** — The RNV bit should remain in the erased state for future enhancements.
5 FPHDIS	**Flash Protection Higher Address Range Disable** — The FPHDIS bit determines whether there is a protected/unprotected area in a specific region of the P-Flash memory ending with global address 0x7F_FFFF. 0 Protection/Unprotection enabled 1 Protection/Unprotection disabled
4–3 FPHS[1:0]	**Flash Protection Higher Address Size** — The FPHS bits determine the size of the protected/unprotected area in P-Flash memory as shown inTable 8-21. The FPHS bits can only be written to while the FPHDIS bit is set.
2 FPLDIS	**Flash Protection Lower Address Range Disable** — The FPLDIS bit determines whether there is a protected/unprotected area in a specific region of the P-Flash memory beginning with global address 0x7F_8000. 0 Protection/Unprotection enabled 1 Protection/Unprotection disabled
1–0 FPLS[1:0]	**Flash Protection Lower Address Size** — The FPLS bits determine the size of the protected/unprotected area in P-Flash memory as shown in Table 8-22. The FPLS bits can only be written to while the FPLDIS bit is set.

Table 8-20. P-Flash Protection Function

FPOPEN	FPHDIS	FPLDIS	Function[1]
1	1	1	No P-Flash Protection
1	1	0	Protected Low Range
1	0	1	Protected High Range
1	0	0	Protected High and Low Ranges
0	1	1	Full P-Flash Memory Protected
0	1	0	Unprotected Low Range
0	0	1	Unprotected High Range
0	0	0	Unprotected High and Low Ranges

1. For range sizes, refer to Table 8-21 and Table 8-22.

Table 8-21. P-Flash Protection Higher Address Range

FPHS[1:0]	Global Address Range	Protected Size
00	0x7F_F800–0x7F_FFFF	2 Kbytes
01	0x7F_F000–0x7F_FFFF	4 Kbytes
10	0x7F_E000–0x7F_FFFF	8 Kbytes
11	0x7F_C000–0x7F_FFFF	16 Kbytes

Table 8-22. P-Flash Protection Lower Address Range

FPLS[1:0]	Global Address Range	Protected Size
00	0x7F_8000–0x7F_83FF	1 Kbyte
01	0x7F_8000–0x7F_87FF	2 Kbytes
10	0x7F_8000–0x7F_8FFF	4 Kbytes
11	0x7F_8000–0x7F_9FFF	8 Kbytes

All possible P-Flash protection scenarios are shown in Figure 8-14. Although the protection scheme is loaded from the Flash memory at global address 0x7F_FF0C during the reset sequence, it can be changed by the user. The P-Flash protection scheme can be used by applications requiring reprogramming in single chip mode while providing as much protection as possible if reprogramming is not required.

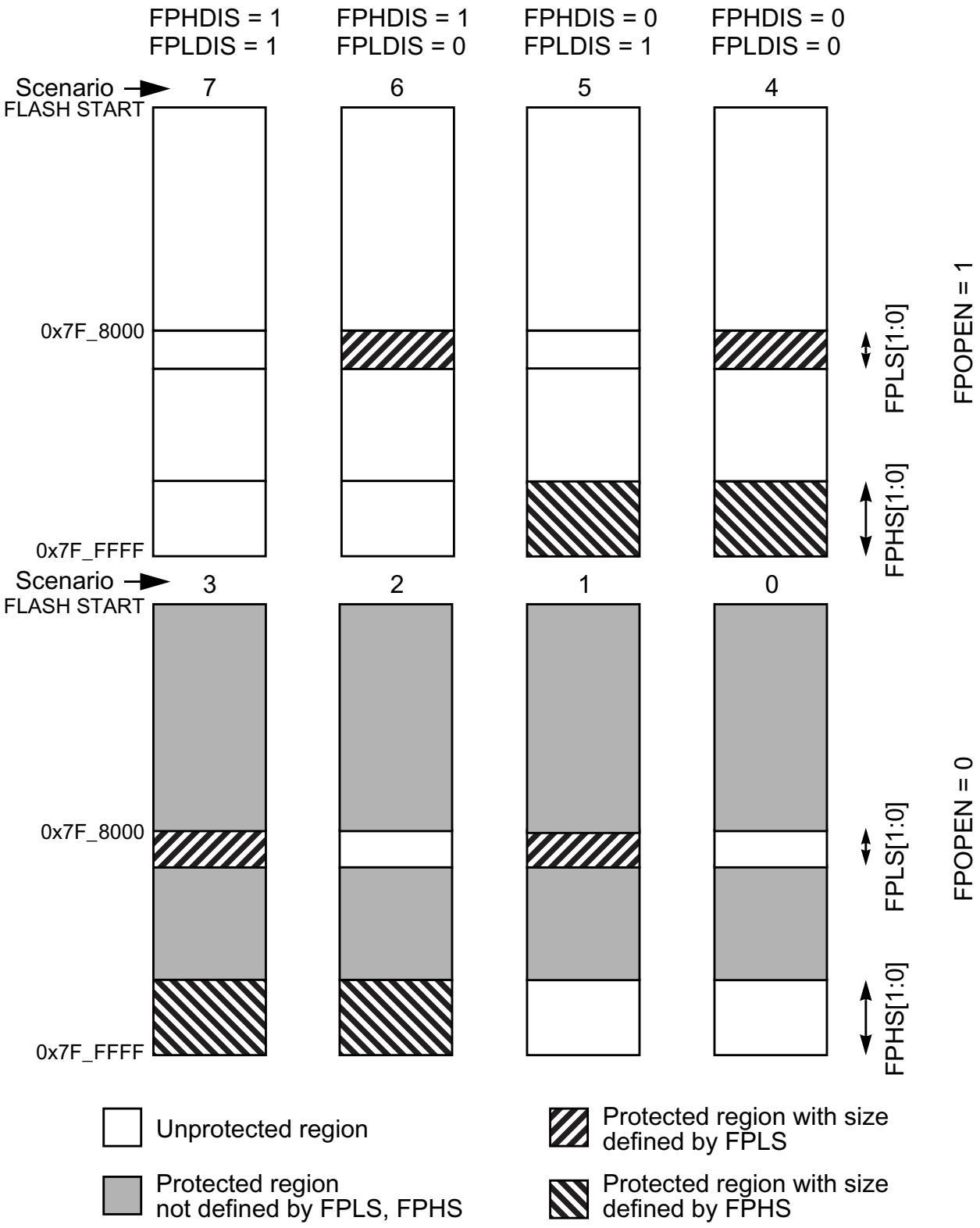


Figure 8-14. P-Flash Protection Scenarios

8.3.2.9.1 P-Flash Protection Restrictions

The general guideline is that P-Flash protection can only be added and not removed. Table 8-23 specifies all valid transitions between P-Flash protection scenarios. Any attempt to write an invalid scenario to the FPROT register will be ignored. The contents of the FPROT register reflect the active protection scenario. See the FPHS and FPLS bit descriptions for additional restrictions.

Table 8-23. P-Flash Protection Scenario Transitions

From Protection Scenario	To Protection Scenario[1]							
	0	1	2	3	4	5	6	7
0	X	X	X	X				
1		X		X				
2			X	X				
3				X				
4				X	X			
5			X	X	X	X		
6		X		X	X		X	
7	X	X	X	X	X	X	X	X

1. Allowed transitions marked with X, see Figure 8-14 for a definition of the scenarios.

8.3.2.10 EEE Protection Register (EPROT)

The EPROT register defines which buffer RAM EEE partition areas are protected against writes.

Offset Module Base + 0x0009

	7	6	5	4	3	2	1	0
R	EPOPEN	RNV[6:4]			EPDIS	EPS[2:0]		
W								
Reset	F	F	F	F	F	F	F	F

= Unimplemented or Reserved

Figure 8-15. EEE Protection Register (EPROT)

All bits in the EPROT register are readable and writable except for RNV[6:4] which are only readable. The EPOPEN and EPDIS bits can only be written to the protected state. The EPS bits can be written anytime until the EPDIS bit is cleared. If the EPOPEN bit is cleared, the state of the EPDIS and EPS bits is irrelevant.

During the reset sequence, the EPROT register is loaded from the EEE protection byte in the Flash configuration field at global address 0x7F_FF0D located in P-Flash memory (see Table 8-3) as indicated by reset condition F in Figure 8-15. To change the EEE protection that will be loaded during the reset sequence, the P-Flash sector containing the EEE protection byte must be unprotected, then the EEE protection byte must be programmed. If a double bit fault is detected while reading the P-Flash phrase

containing the EEE protection byte during the reset sequence, the EPOPEN bit will be cleared and remaining bits in the EPROT register will be set to leave the buffer RAM EEE partition fully protected.

Trying to write data to any protected area in the buffer RAM EEE partition will result in a protection violation error and the EPVIOLIF flag will be set in the FERSTAT register. Trying to write data to any protected area in the buffer RAM partitioned for user access will not be prevented and the EPVIOLIF flag in the FERSTAT register will not set.

Table 8-24. EPROT Field Descriptions

Field	Description
7 EPOPEN	**Enables writes to the Buffer RAM partitioned for EEE** 0 The entire buffer RAM EEE partition is protected from writes 1 Unprotected buffer RAM EEE partition areas are enabled for writes
6–4 RNV[6:4]	**Reserved Nonvolatile Bits** — The RNV bits should remain in the erased state for future enhancements
3 EPDIS	**Buffer RAM Protection Address Range Disable** — The EPDIS bit determines whether there is a protected area in a specific region of the buffer RAM EEE partition. 0 Protection enabled 1 Protection disabled
2–0 EPS[2:0]	**Buffer RAM Protection Size** — The EPS[2:0] bits determine the size of the protected area in the buffer RAM EEE partition as shown inTable 8-21. The EPS bits can only be written to while the EPDIS bit is set.

Table 8-25. Buffer RAM EEE Partition Protection Address Range

EPS[2:0]	Global Address Range	Protected Size
000	0x13_FFC0 – 0x13_FFFF	64 bytes
001	0x13_FF80 – 0x13_FFFF	128 bytes
010	0x13_FF40 – 0x13_FFFF	192 bytes
011	0x13_FF00 – 0x13_FFFF	256 bytes
100	0x13_FEC0 – 0x13_FFFF	320 bytes
101	0x13_FE80 – 0x13_FFFF	384 bytes
110	0x13_FE40 – 0x13_FFFF	448 bytes
111	0x13_FE00 – 0x13_FFFF	512 bytes

8.3.2.11 Flash Common Command Object Register (FCCOB)

The FCCOB is an array of six words addressed via the CCOBIX index found in the FCCOBIX register. Byte wide reads and writes are allowed to the FCCOB register.

Offset Module Base + 0x000A

	7	6	5	4	3	2	1	0
R W	CCOB[15:8]							
Reset	0	0	0	0	0	0	0	0

Figure 8-16. Flash Common Command Object High Register (FCCOBHI)

Offset Module Base + 0x000B

	7	6	5	4	3	2	1	0
R W	CCOB[7:0]							
Reset	0	0	0	0	0	0	0	0

Figure 8-17. Flash Common Command Object Low Register (FCCOBLO)

8.3.2.11.1 FCCOB - NVM Command Mode

NVM command mode uses the indexed FCCOB register to provide a command code and its relevant parameters to the Memory Controller. The user first sets up all required FCCOB fields and then initiates the command's execution by writing a 1 to the CCIF bit in the FSTAT register (a 1 written by the user clears the CCIF command completion flag to 0). When the user clears the CCIF bit in the FSTAT register all FCCOB parameter fields are locked and cannot be changed by the user until the command completes (as evidenced by the Memory Controller returning CCIF to 1). Some commands return information to the FCCOB register array.

The generic format for the FCCOB parameter fields in NVM command mode is shown in Table 8-26. The return values are available for reading after the CCIF flag in the FSTAT register has been returned to 1 by the Memory Controller. Writes to the unimplemented parameter fields (CCOBIX = 110 and CCOBIX = 111) are ignored with reads from these fields returning 0x0000.

Table 8-26 shows the generic Flash command format. The high byte of the first word in the CCOB array contains the command code, followed by the parameters for this specific Flash command. For details on the FCCOB settings required by each command, see the Flash command descriptions in Section 8.4.2.

Table 8-26. FCCOB - NVM Command Mode (Typical Usage)

CCOBIX[2:0]	Byte	FCCOB Parameter Fields (NVM Command Mode)
000	HI	FCMD[7:0] defining Flash command
	LO	0, Global address [22:16]
001	HI	Global address [15:8]
	LO	Global address [7:0]
010	HI	Data 0 [15:8]
	LO	Data 0 [7:0]

Table 8-26. FCCOB - NVM Command Mode (Typical Usage)

CCOBIX[2:0]	Byte	FCCOB Parameter Fields (NVM Command Mode)
011	HI	Data 1 [15:8]
	LO	Data 1 [7:0]
100	HI	Data 2 [15:8]
	LO	Data 2 [7:0]
101	HI	Data 3 [15:8]
	LO	Data 3 [7:0]

8.3.2.12 EEE Tag Counter Register (ETAG)

The ETAG register contains the number of outstanding words in the buffer RAM EEE partition that need to be programmed into the D-Flash EEE partition. The ETAG register is decremented prior to the related tagged word being programmed into the D-Flash EEE partition. All tagged words have been programmed into the D-Flash EEE partition once all bits in the ETAG register read 0 and the MGBUSY flag in the FSTAT register reads 0.

Offset Module Base + 0x000C

	7	6	5	4	3	2	1	0
R	ETAG[15:8]							
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 8-18. EEE Tag Counter High Register (ETAGHI)

Offset Module Base + 0x000D

	7	6	5	4	3	2	1	0
R	ETAG[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 8-19. EEE Tag Counter Low Register (ETAGLO)

All ETAG bits are readable but not writable and are cleared by the Memory Controller.

8.3.2.13 Flash ECC Error Results Register (FECCR)

The FECCR registers contain the result of a detected ECC fault for both single bit and double bit faults. The FECCR register provides access to several ECC related fields as defined by the ECCRIX index bits in the FECCRIX register (see Section 8.3.2.4). Once ECC fault information has been stored, no other fault

information will be recorded until the specific ECC fault flag has been cleared. In the event of simultaneous ECC faults, the priority for fault recording is:

1. Double bit fault over single bit fault
2. CPU over XGATE

Offset Module Base + 0x000E

	7	6	5	4	3	2	1	0
R	ECCR[15:8]							
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 8-20. Flash ECC Error Results High Register (FECCRHI)

Offset Module Base + 0x000F

	7	6	5	4	3	2	1	0
R	ECCR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 8-21. Flash ECC Error Results Low Register (FECCRLO)

All FECCR bits are readable but not writable.

Table 8-27. FECCR Index Settings

ECCRIX[2:0]	FECCR Register Content		
	Bits [15:8]	**Bit[7]**	**Bits[6:0]**
000	Parity bits read from Flash block	CPU or XGATE source identity	Global address [22:16]
001	Global address [15:0]		
010	Data 0 [15:0]		
011	Data 1 [15:0] (P-Flash only)		
100	Data 2 [15:0] (P-Flash only)		
101	Data 3 [15:0] (P-Flash only)		
110	Not used, returns 0x0000 when read		
111	Not used, returns 0x0000 when read		

Table 8-28. FECCR Index=000 Bit Descriptions

Field	Description
15:8 PAR[7:0]	**ECC Parity Bits** — Contains the 8 parity bits from the 72 bit wide P-Flash data word or the 6 parity bits, allocated to PAR[5:0], from the 22 bit wide D-Flash word with PAR[7:6]=00.
7 XBUS01	**Bus Source Identifier** — The XBUS01 bit determines whether the ECC error was caused by a read access from the CPU or XGATE. 0 ECC Error happened on the CPU access 1 ECC Error happened on the XGATE access
6–0 GADDR[22:16]	**Global Address** — The GADDR[22:16] field contains the upper seven bits of the global address having caused the error.

The P-Flash word addressed by ECCRIX = 001 contains the lower 16 bits of the global address. The following four words addressed by ECCRIX = 010 to 101 contain the 64-bit wide data phrase. The four data words and the parity byte are the uncorrected data read from the P-Flash block.

The D-Flash word addressed by ECCRIX = 001 contains the lower 16 bits of the global address. The uncorrected 16-bit data word is addressed by ECCRIX = 010.

8.3.2.14 Flash Option Register (FOPT)

The FOPT register is the Flash option register.

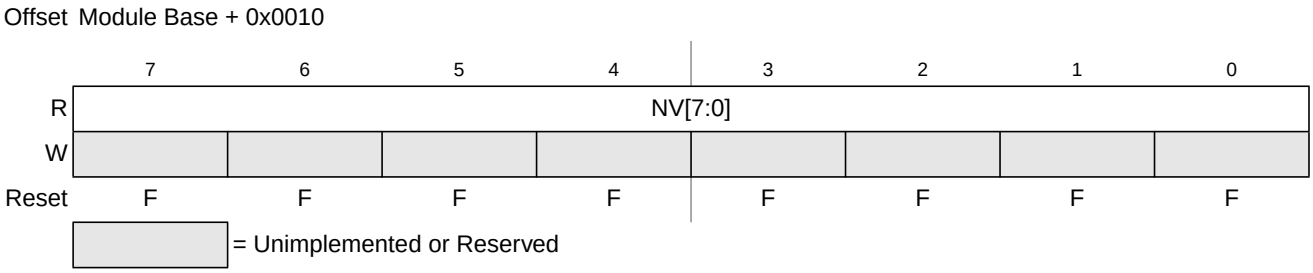


Figure 8-22. Flash Option Register (FOPT)

All bits in the FOPT register are readable but are not writable.

During the reset sequence, the FOPT register is loaded from the Flash nonvolatile byte in the Flash configuration field at global address 0x7F_FF0E located in P-Flash memory (see Table 8-3) as indicated by reset condition F in Figure 8-22. If a double bit fault is detected while reading the P-Flash phrase containing the Flash nonvolatile byte during the reset sequence, all bits in the FOPT register will be set.

Table 8-29. FOPT Field Descriptions

Field	Description
7–0 NV[7:0]	**Nonvolatile Bits** — The NV[7:0] bits are available as nonvolatile bits. Refer to the device user guide for proper use of the NV bits.

8.3.2.15 Flash Reserved0 Register (FRSV0)

This Flash register is reserved for factory testing.

Offset Module Base + 0x0011

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 8-23. Flash Reserved0 Register (FRSV0)

All bits in the FRSV0 register read 0 and are not writable.

8.3.2.16 Flash Reserved1 Register (FRSV1)

This Flash register is reserved for factory testing.

Offset Module Base + 0x0012

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 8-24. Flash Reserved1 Register (FRSV1)

All bits in the FRSV1 register read 0 and are not writable.

8.3.2.17 Flash Reserved2 Register (FRSV2)

This Flash register is reserved for factory testing.

Offset Module Base + 0x0013

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 8-25. Flash Reserved2 Register (FRSV2)

All bits in the FRSV2 register read 0 and are not writable.

8.4 Functional Description

8.4.1 Flash Command Operations

Flash command operations are used to modify Flash memory contents or configure module resources for EEE operation.

The next sections describe:

- How to write the FCLKDIV register that is used to generate a time base (FCLK) derived from OSCCLK for Flash program and erase command operations
- The command write sequence used to set Flash command parameters and launch execution
- Valid Flash commands available for execution

8.4.1.1 Writing the FCLKDIV Register

Prior to issuing any Flash program or erase command after a reset, the user is required to write the FCLKDIV register to divide OSCCLK down to a target FCLK of 1 MHz. Table 8-9 shows recommended values for the FDIV field based on OSCCLK frequency.

NOTE

Programming or erasing the Flash memory cannot be performed if the bus clock runs at less than 1 MHz. Setting FDIV too high can destroy the Flash memory due to overstress. Setting FDIV too low can result in incomplete programming or erasure of the Flash memory cells.

When the FCLKDIV register is written, the FDIVLD bit is set automatically. If the FDIVLD bit is 0, the FCLKDIV register has not been written since the last reset. If the FCLKDIV register has not been written, any Flash program or erase command loaded during a command write sequence will not execute and the ACCERR bit in the FSTAT register will set.

8.4.1.2 Command Write Sequence

The Memory Controller will launch all valid Flash commands entered using a command write sequence.

Before launching a command, the ACCERR and FPVIOL bits in the FSTAT register must be clear (see Section 8.3.2.7) and the CCIF flag should be tested to determine the status of the current command write sequence. If CCIF is 0, the previous command write sequence is still active, a new command write sequence cannot be started, and all writes to the FCCOB register are ignored.

CAUTION

Writes to any Flash register must be avoided while a Flash command is active (CCIF=0) to prevent corruption of Flash register contents and Memory Controller behavior.

8.4.1.2.1 Define FCCOB Contents

The FCCOB parameter fields must be loaded with all required parameters for the Flash command being executed. Access to the FCCOB parameter fields is controlled via the CCOBIX bits in the FCCOBIX register (see Section 8.3.2.3).

The contents of the FCCOB parameter fields are transferred to the Memory Controller when the user clears the CCIF command completion flag in the FSTAT register (writing 1 clears the CCIF to 0). The CCIF flag will remain clear until the Flash command has completed. Upon completion, the Memory Controller will return CCIF to 1 and the FCCOB register will be used to communicate any results. The flow for a generic command write sequence is shown in Figure 8-26.

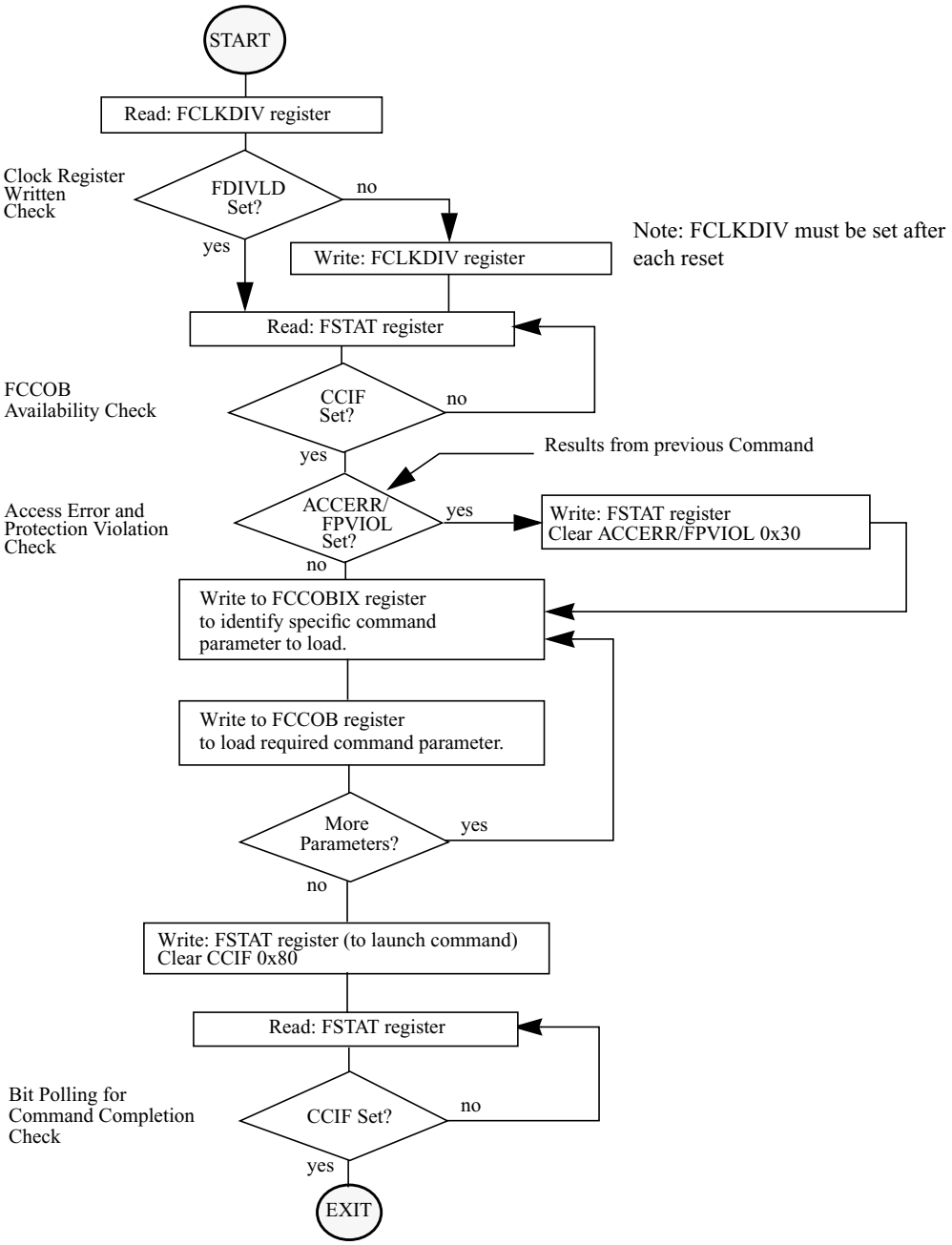


Figure 8-26. Generic Flash Command Write Sequence Flowchart

8.4.1.3 Valid Flash Module Commands

Table 8-30. Flash Commands by Mode

FCMD	Command	Unsecured				Secured			
		NS (1)	NX (2)	SS(3)	ST(4)	NS (5)	NX (6)	SS(7)	ST(8)
0x01	Erase Verify All Blocks	*	*	*	*	*	*	*	*
0x02	Erase Verify Block	*	*	*	*	*	*	*	*
0x03	Erase Verify P-Flash Section	*	*	*	*	*			
0x04	Read Once	*	*	*	*	*			
0x05	Reserved	*	*	*	*	*			
0x06	Program P-Flash	*	*	*	*	*			
0x07	Program Once	*	*	*	*	*			
0x08	Erase All Blocks			*	*			*	*
0x09	Erase P-Flash Block	*	*	*	*	*			
0x0A	Erase P-Flash Sector	*	*	*	*	*			
0x0B	Unsecure Flash			*	*			*	*
0x0C	Verify Backdoor Access Key	*				*			
0x0D	Set User Margin Level	*	*	*	*	*			
0x0E	Set Field Margin Level			*	*				
0x0F	Full Partition D-Flash			*	*				
0x10	Erase Verify D-Flash Section	*	*	*	*	*			
0x11	Program D-Flash	*	*	*	*	*			
0x12	Erase D-Flash Sector	*	*	*	*	*			
0x13	Enable EEPROM Emulation	*	*	*	*	*	*	*	*
0x14	Disable EEPROM Emulation	*	*	*	*	*	*	*	*
0x15	EEPROM Emulation Query	*	*	*	*	*	*	*	*
0x20	Partition D-Flash	*	*	*	*	*	*	*	*

1. Unsecured Normal Single Chip mode.
2. Unsecured Normal Expanded mode.
3. Unsecured Special Single Chip mode.
4. Unsecured Special Mode.
5. Secured Normal Single Chip mode.
6. Secured Normal Expanded mode.
7. Secured Special Single Chip mode.
8. Secured Special Mode.

8.4.1.4 P-Flash Commands

Table 8-31 summarizes the valid P-Flash commands along with the effects of the commands on the P-Flash block and other resources within the Flash module.

Table 8-31. P-Flash Commands

FCMD	Command	Function on P-Flash Memory
0x01	Erase Verify All Blocks	Verify that all P-Flash (and D-Flash) blocks are erased.
0x02	Erase Verify Block	Verify that a P-Flash block is erased.
0x03	Erase Verify P-Flash Section	Verify that a given number of words starting at the address provided are erased.
0x04	Read Once	Read a dedicated 64 byte field in the nonvolatile information register in P-Flash block 0 that was previously programmed using the Program Once command.
0x06	Program P-Flash	Program a phrase in a P-Flash block.
0x07	Program Once	Program a dedicated 64 byte field in the nonvolatile information register in P-Flash block 0 that is allowed to be programmed only once.
0x08	Erase All Blocks	Erase all P-Flash (and D-Flash) blocks. An erase of all Flash blocks is only possible when the FPLDIS, FPHDIS, and FPOPEN bits in the FPROT register and the EPDIS and EPOPEN bits in the EPROT register are set prior to launching the command.
0x09	Erase P-Flash Block	Erase a single P-Flash block. An erase of the full P-Flash block is only possible when FPLDIS, FPHDIS and FPOPEN bits in the FPROT register are set prior to launching the command.
0x0A	Erase P-Flash Sector	Erase all bytes in a P-Flash sector.
0x0B	Unsecure Flash	Supports a method of releasing MCU security by erasing all P-Flash (and D-Flash) blocks and verifying that all P-Flash (and D-Flash) blocks are erased.
0x0C	Verify Backdoor Access Key	Supports a method of releasing MCU security by verifying a set of security keys.
0x0D	Set User Margin Level	Specifies a user margin read level for all P-Flash blocks.
0x0E	Set Field Margin Level	Specifies a field margin read level for all P-Flash blocks (special modes only).

8.4.1.5 D-Flash and EEE Commands

Table 8-32 summarizes the valid D-Flash and EEE commands along with the effects of the commands on the D-Flash block and EEE operation.

Table 8-32. D-Flash Commands

FCMD	Command	Function on D-Flash Memory
0x01	Erase Verify All Blocks	Verify that all D-Flash (and P-Flash) blocks are erased.
0x02	Erase Verify Block	Verify that the D-Flash block is erased.
0x08	Erase All Blocks	Erase all D-Flash (and P-Flash) blocks. An erase of all Flash blocks is only possible when the FPLDIS, FPHDIS, and FPOPEN bits in the FPROT register and the EPDIS and EPOPEN bits in the EPROT register are set prior to launching the command.

Table 8-32. D-Flash Commands

FCMD	Command	Function on D-Flash Memory
0x0B	Unsecure Flash	Supports a method of releasing MCU security by erasing all D-Flash (and P-Flash) blocks and verifying that all D-Flash (and P-Flash) blocks are erased.
0x0D	Set User Margin Level	Specifies a user margin read level for the D-Flash block.
0x0E	Set Field Margin Level	Specifies a field margin read level for the D-Flash block (special modes only).
0x0F	Full Partition D-Flash	Erase the D-Flash block and partition an area of the D-Flash block for user access.
0x10	Erase Verify D-Flash Section	Verify that a given number of words starting at the address provided are erased.
0x11	Program D-Flash	Program up to four words in the D-Flash block.
0x12	Erase D-Flash Sector	Erase all bytes in a sector of the D-Flash block.
0x13	Enable EEPROM Emulation	Enable EEPROM emulation where writes to the buffer RAM EEE partition will be copied to the D-Flash EEE partition.
0x14	Disable EEPROM Emulation	Suspend all current erase and program activity related to EEPROM emulation but leave current EEE tags set.
0x15	EEPROM Emulation Query	Returns EEE partition and status variables.
0x20	Partition D-Flash	Partition an area of the D-Flash block for user access.

8.4.2 Flash Command Description

This section provides details of all available Flash commands launched by a command write sequence. The ACCERR bit in the FSTAT register will be set during the command write sequence if any of the following illegal steps are performed, causing the command not to be processed by the Memory Controller:

- Starting any command write sequence that programs or erases Flash memory before initializing the FCLKDIV register
- Writing an invalid command as part of the command write sequence
- For additional possible errors, refer to the error handling table provided for each command

If a Flash block is read during execution of an algorithm (CCIF = 0) on that same block, the read operation will return invalid data. If the SFDIF or DFDIF flags were not previously set when the invalid read operation occurred, both the SFDIF and DFDIF flags will be set and the FECCR registers will be loaded with the global address used in the invalid read operation with the data and parity fields set to all 0.

If the ACCERR or FPVIOL bits are set in the FSTAT register, the user must clear these bits before starting any command write sequence (see Section 8.3.2.7).

CAUTION

A Flash word or phrase must be in the erased state before being programmed. Cumulative programming of bits within a Flash word or phrase is not allowed.

8.4.2.1 Erase Verify All Blocks Command

The Erase Verify All Blocks command will verify that all P-Flash and D-Flash blocks have been erased.

Table 8-33. Erase Verify All Blocks Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x01	Not required

Upon clearing CCIF to launch the Erase Verify All Blocks command, the Memory Controller will verify that the entire Flash memory space is erased. The CCIF flag will set after the Erase Verify All Blocks operation has completed.

Table 8-34. Erase Verify All Blocks Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read(1)
	MGSTAT0	Set if any non-correctable errors have been encountered during the read[1]
FERSTAT	EPVIOLIF	None

1. As found in the memory map for FTM512K3.

8.4.2.2 Erase Verify Block Command

The Erase Verify Block command allows the user to verify that an entire P-Flash or D-Flash block has been erased. The FCCOB upper global address bits determine which block must be verified.

Table 8-35. Erase Verify Block Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x02	Global address [22:16] of the Flash block to be verified.

Upon clearing CCIF to launch the Erase Verify Block command, the Memory Controller will verify that the selected P-Flash or D-Flash block is erased. The CCIF flag will set after the Erase Verify Block operation has completed.

Table 8-36. Erase Verify Block Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if an invalid global address [22:16] is supplied(1)
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read(2)
	MGSTAT0	Set if any non-correctable errors have been encountered during the read2
FERSTAT	EPVIOLIF	None

1. As defined by the memory map for FTM512K3.
2. As found in the memory map for FTM512K3.

8.4.2.3 Erase Verify P-Flash Section Command

The Erase Verify P-Flash Section command will verify that a section of code in the P-Flash memory is erased. The Erase Verify P-Flash Section command defines the starting point of the code to be verified and the number of phrases.

Table 8-37. Erase Verify P-Flash Section Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x03	Global address [22:16] of a P-Flash block
001	Global address [15:0] of the first phrase to be verified	
010	Number of phrases to be verified	

Upon clearing CCIF to launch the Erase Verify P-Flash Section command, the Memory Controller will verify the selected section of Flash memory is erased. The CCIF flag will set after the Erase Verify P-Flash Section operation has completed.

Table 8-38. Erase Verify P-Flash Section Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 010 at command launch
		Set if command not available in current mode (see Table 8-30)
		Set if an invalid global address [22:0] is supplied(1)
		Set if a misaligned phrase address is supplied (global address [2:0] != 000)
		Set if the requested section crosses a 256 Kbyte boundary
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read(2)
	MGSTAT0	Set if any non-correctable errors have been encountered during the read2
FERSTAT	EPVIOLIF	None

1. As defined by the memory map for FTM512K3.
2. As found in the memory map for FTM512K3.

8.4.2.4 Read Once Command

The Read Once command provides read access to a reserved 64 byte field (8 phrases) located in the nonvolatile information register of P-Flash block 0. The Read Once field is programmed using the Program Once command described in Section 8.4.2.6. The Read Once command must not be executed from the Flash block containing the Program Once reserved field to avoid code runaway.

Table 8-39. Read Once Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x04	Not Required
001	Read Once phrase index (0x0000 - 0x0007)	
010	Read Once word 0 value	
011	Read Once word 1 value	
100	Read Once word 2 value	
101	Read Once word 3 value	

Upon clearing CCIF to launch the Read Once command, a Read Once phrase is fetched and stored in the FCCOB indexed register. The CCIF flag will set after the Read Once operation has completed. Valid phrase index values for the Read Once command range from 0x0000 to 0x0007. During execution of the Read Once command, any attempt to read addresses within P-Flash block 0 will return invalid data.

Table 8-40. Read Once Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if command not available in current mode (see Table 8-30)
		Set if an invalid phrase index is supplied
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read
	MGSTAT0	Set if any non-correctable errors have been encountered during the read
FERSTAT	EPVIOLIF	None

8.4.2.5 Program P-Flash Command

The Program P-Flash operation will program a previously erased phrase in the P-Flash memory using an embedded algorithm.

CAUTION

A P-Flash phrase must be in the erased state before being programmed. Cumulative programming of bits within a Flash phrase is not allowed.

Table 8-41. Program P-Flash Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x06	Global address [22:16] to identify P-Flash block
001	Global address [15:0] of phrase location to be programmed(1)	
010	Word 0 program value	
011	Word 1 program value	
100	Word 2 program value	
101	Word 3 program value	

1. Global address [2:0] must be 000

Upon clearing CCIF to launch the Program P-Flash command, the Memory Controller will program the data words to the supplied global address and will then proceed to verify the data words read back as expected. The CCIF flag will set after the Program P-Flash operation has completed.

Table 8-42. Program P-Flash Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 101 at command launch
		Set if command not available in current mode (see Table 8-30)
		Set if an invalid global address [22:0] is supplied(1)
		Set if a misaligned phrase address is supplied (global address [2:0] != 000)
	FPVIOL	Set if the global address [22:0] points to a protected area
	MGSTAT1	Set if any errors have been encountered during the verify operation(2)
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation2
FERSTAT	EPVIOLIF	None

1. As defined by the memory map for FTM512K3.
2. As found in the memory map for FTM512K3.

8.4.2.6 Program Once Command

The Program Once command restricts programming to a reserved 64 byte field (8 phrases) in the nonvolatile information register located in P-Flash block 0. The Program Once reserved field can be read using the Read Once command as described in Section 8.4.2.4. The Program Once command must only be issued once since the nonvolatile information register in P-Flash block 0 cannot be erased. The Program

Once command must not be executed from the Flash block containing the Program Once reserved field to avoid code runaway.

Table 8-43. Program Once Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x07	Not Required
001	Program Once phrase index (0x0000 - 0x0007)	
010	Program Once word 0 value	
011	Program Once word 1 value	
100	Program Once word 2 value	
101	Program Once word 3 value	

Upon clearing CCIF to launch the Program Once command, the Memory Controller first verifies that the selected phrase is erased. If erased, then the selected phrase will be programmed and then verified with read back. The CCIF flag will remain clear, setting only after the Program Once operation has completed.

The reserved nonvolatile information register accessed by the Program Once command cannot be erased and any attempt to program one of these phrases a second time will not be allowed. Valid phrase index values for the Program Once command range from 0x0000 to 0x0007. During execution of the Program Once command, any attempt to read addresses within P-Flash block 0 will return invalid data.

Table 8-44. Program Once Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 101 at command launch
		Set if command not available in current mode (see Table 8-30)
		Set if an invalid phrase index is supplied
		Set if the requested phrase has already been programmed(1)
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation
FERSTAT	EPVIOLIF	None

1. If a Program Once phrase is initially programmed to 0xFFFF_FFFF_FFFF_FFFF, the Program Once command will be allowed to execute again on that same phrase.

8.4.2.7 Erase All Blocks Command

The Erase All Blocks operation will erase the entire P-Flash and D-Flash memory space including the EEE nonvolatile information register.

Table 8-45. Erase All Blocks Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x08	Not required

Upon clearing CCIF to launch the Erase All Blocks command, the Memory Controller will erase the entire Flash memory space and verify that it is erased. If the Memory Controller verifies that the entire Flash memory space was properly erased, security will be released. During the execution of this command (CCIF=0) the user must not write to any Flash module register. The CCIF flag will set after the Erase All Blocks operation has completed.

Table 8-46. Erase All Blocks Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if command not available in current mode (see Table 8-30)
	FPVIOL	Set if any area of the P-Flash memory is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation(1)
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation[1]
FERSTAT	EPVIOLIF	Set if any area of the buffer RAM EEE partition is protected

1. As found in the memory map for FTM512K3.

8.4.2.8 Erase P-Flash Block Command

The Erase P-Flash Block operation will erase all addresses in a P-Flash block.

Table 8-47. Erase P-Flash Block Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x09	Global address [22:16] to identify P-Flash block
001	Global address [15:0] in P-Flash block to be erased	

Upon clearing CCIF to launch the Erase P-Flash Block command, the Memory Controller will erase the selected P-Flash block and verify that it is erased. The CCIF flag will set after the Erase P-Flash Block operation has completed.

Table 8-48. Erase P-Flash Block Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if command not available in current mode (see Table 8-30)
		Set if an invalid global address [22:16] is supplied(1)
	FPVIOL	Set if an area of the selected P-Flash block is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation(2)
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation2
FERSTAT	EPVIOLIF	None

1. As defined by the memory map for FTM512K3.
2. As found in the memory map for FTM512K3.

8.4.2.9 Erase P-Flash Sector Command

The Erase P-Flash Sector operation will erase all addresses in a P-Flash sector.

Table 8-49. Erase P-Flash Sector Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x0A	Global address [22:16] to identify P-Flash block to be erased
001	Global address [15:0] anywhere within the sector to be erased. Refer to Section 8.1.2.1 for the P-Flash sector size.	

Upon clearing CCIF to launch the Erase P-Flash Sector command, the Memory Controller will erase the selected Flash sector and then verify that it is erased. The CCIF flag will be set after the Erase P-Flash Sector operation has completed.

Table 8-50. Erase P-Flash Sector Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if command not available in current mode (see Table 8-30)
		Set if an invalid global address [22:16] is supplied(1)
		Set if a misaligned phrase address is supplied (global address [2:0] != 000)
	FPVIOL	Set if the selected P-Flash sector is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation
FERSTAT	EPVIOLIF	None

1. As defined by the memory map for FTM512K3.

8.4.2.10 Unsecure Flash Command

The Unsecure Flash command will erase the entire P-Flash and D-Flash memory space and, if the erase is successful, will release security.

Table 8-51. Unsecure Flash Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x0B	Not required

Upon clearing CCIF to launch the Unsecure Flash command, the Memory Controller will erase the entire P-Flash and D-Flash memory space and verify that it is erased. If the Memory Controller verifies that the entire Flash memory space was properly erased, security will be released. If the erase verify is not successful, the Unsecure Flash operation sets MGSTAT1 and terminates without changing the security state. During the execution of this command (CCIF=0) the user must not write to any Flash module register. The CCIF flag is set after the Unsecure Flash operation has completed.

Table 8-52. Unsecure Flash Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if command not available in current mode (see Table 8-30)
	FPVIOL	Set if any area of the P-Flash memory is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation(1)
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation[1]
FERSTAT	EPVIOLIF	Set if any area of the buffer RAM EEE partition is protected

1. As found in the memory map for FTM512K3.

8.4.2.11 Verify Backdoor Access Key Command

The Verify Backdoor Access Key command will only execute if it is enabled by the KEYEN bits in the FSEC register (see Table 8-11). The Verify Backdoor Access Key command releases security if user-supplied keys match those stored in the Flash security bytes of the Flash configuration field (see Table 8-3). The Verify Backdoor Access Key command must not be executed from the Flash block containing the backdoor comparison key to avoid code runaway.

Table 8-53. Verify Backdoor Access Key Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x0C	Not required
001	Key 0	
010	Key 1	

Table 8-53. Verify Backdoor Access Key Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters
011	Key 2
100	Key 3

Upon clearing CCIF to launch the Verify Backdoor Access Key command, the Memory Controller will check the FSEC KEYEN bits to verify that this command is enabled. If not enabled, the Memory Controller sets the ACCERR bit in the FSTAT register and terminates. If the command is enabled, the Memory Controller compares the key provided in FCCOB to the backdoor comparison key in the Flash configuration field with Key 0 compared to 0x7F_FF00, etc. If the backdoor keys match, security will be released. If the backdoor keys do not match, security is not released and all future attempts to execute the Verify Backdoor Access Key command are aborted (set ACCERR) until a reset occurs. The CCIF flag is set after the Verify Backdoor Access Key operation has completed.

Table 8-54. Verify Backdoor Access Key Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 100 at command launch
		Set if an incorrect backdoor key is supplied
		Set if backdoor key access has not been enabled (KEYEN[1:0] != 10, see Section 8.3.2.2)
		Set if the backdoor key has mismatched since the last reset
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

8.4.2.12 Set User Margin Level Command

The Set User Margin Level command causes the Memory Controller to set the margin level for future read operations of a specific P-Flash or D-Flash block.

Table 8-55. Set User Margin Level Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x0D	Global address [22:16] to identify the Flash block
001	Margin level setting	

Upon clearing CCIF to launch the Set User Margin Level command, the Memory Controller will set the user margin level for the targeted block and then set the CCIF flag.

Valid margin level settings for the Set User Margin Level command are defined in Table 8-56.

Table 8-56. Valid Set User Margin Level Settings

CCOB (CCOBIX=001)	Level Description
0x0000	Return to Normal Level
0x0001	User Margin-1 Level(1)
0x0002	User Margin-0 Level(2)

1. Read margin to the erased state
2. Read margin to the programmed state

Table 8-57. Set User Margin Level Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if command not available in current mode (see Table 8-30)
		Set if an invalid global address [22:16] is supplied(1)
		Set if an invalid margin level setting is supplied
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

1. As defined by the memory map for FTM512K3.

NOTE

User margin levels can be used to check that Flash memory contents have adequate margin for normal level read operations. If unexpected results are encountered when checking Flash memory contents at user margin levels, a potential loss of information has been detected.

8.4.2.13 Set Field Margin Level Command

The Set Field Margin Level command, valid in special modes only, causes the Memory Controller to set the margin level specified for future read operations of a specific P-Flash or D-Flash block.

Table 8-58. Set Field Margin Level Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x0E	Global address [22:16] to identify the Flash block
001	Margin level setting	

Upon clearing CCIF to launch the Set Field Margin Level command, the Memory Controller will set the field margin level for the targeted block and then set the CCIF flag.

Valid margin level settings for the Set Field Margin Level command are defined in Table 8-59.

Table 8-59. Valid Set Field Margin Level Settings

CCOB (CCOBIX=001)	Level Description
0x0000	Return to Normal Level
0x0001	User Margin-1 Level(1)
0x0002	User Margin-0 Level(2)
0x0003	Field Margin-1 Level[1]
0x0004	Field Margin-0 Level[2]

1. Read margin to the erased state
2. Read margin to the programmed state

Table 8-60. Set Field Margin Level Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if command not available in current mode (see Table 8-30)
		Set if an invalid global address [22:16] is supplied(1)
		Set if an invalid margin level setting is supplied
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

1. As defined by the memory map for FTM512K3.

CAUTION

Field margin levels must only be used during verify of the initial factory programming.

NOTE

Field margin levels can be used to check that Flash memory contents have adequate margin for data retention at the normal level setting. If unexpected results are encountered when checking Flash memory contents at field margin levels, the Flash memory contents should be erased and reprogrammed.

8.4.2.14 Full Partition D-Flash Command

The Full Partition D-Flash command allows the user to allocate sectors within the D-Flash block for applications and a partition within the buffer RAM for EEPROM access. The D-Flash block consists of 128 sectors with 256 bytes per sector.

Table 8-61. Full Partition D-Flash Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x0F	Not required
001	Number of 256 byte sectors for the D-Flash user partition (DFPART)	
010	Number of 256 byte sectors for buffer RAM EEE partition (ERPART)	

Upon clearing CCIF to launch the Full Partition D-Flash command, the following actions are taken to define a partition within the D-Flash block for direct access (DFPART) and a partition within the buffer RAM for EEE use (ERPART):

- Validate the DFPART and ERPART values provided:
  - DFPART <= 128 (maximum number of 256 byte sectors in D-Flash block)
  - ERPART <= 8 (maximum number of 256 byte sectors in buffer RAM)
  - If ERPART > 0, 128 - DFPART >= 12 (minimum number of 256 byte sectors in the D-Flash block required to support EEE)
  - If ERPART > 0, ((128-DFPART)/ERPART) >= 8 (minimum ratio of D-Flash EEE space to buffer RAM EEE space to support EEE)
- Erase the D-Flash block and the EEE nonvolatile information register
- Program DFPART to the EEE nonvolatile information register at global address 0x12_0000 (see Table 8-7)
- Program a duplicate DFPART to the EEE nonvolatile information register at global address 0x12_0002 (see Table 8-7)
- Program ERPART to the EEE nonvolatile information register at global address 0x12_0004 (see Table 8-7)
- Program a duplicate ERPART to the EEE nonvolatile information register at global address 0x12_0006 (see Table 8-7)

The D-Flash user partition will start at global address 0x10_0000. The buffer RAM EEE partition will end at global address 0x13_FFFF. After the Full Partition D-Flash operation has completed, the CCIF flag will set.

Running the Full Partition D-Flash command a second time will result in the previous partition values and the entire D-Flash memory being erased. The data value written corresponds to the number of 256 byte sectors allocated for either direct D-Flash access (DFPART) or buffer RAM EEE access (ERPART).

Table 8-62. Full Partition D-Flash Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 010 at command launch
		Set if command not available in current mode (see Table 8-30)
		Set if an invalid DFPART or ERPART selection is supplied(1)
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read
	MGSTAT0	Set if any non-correctable errors have been encountered during the read
FERSTAT	EPVIOLIF	None

1. As defined by the maximum ERPART for FTM512K3.

8.4.2.15 Erase Verify D-Flash Section Command

The Erase Verify D-Flash Section command will verify that a section of code in the D-Flash user partition is erased. The Erase Verify D-Flash Section command defines the starting point of the data to be verified and the number of words.

Table 8-63. Erase Verify D-Flash Section Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x10	Global address [22:16] to identify the D-Flash block
001	Global address [15:0] of the first word to be verified	
010	Number of words to be verified	

Upon clearing CCIF to launch the Erase Verify D-Flash Section command, the Memory Controller will verify the selected section of D-Flash memory is erased. The CCIF flag will set after the Erase Verify D-Flash Section operation has completed.

Table 8-64. Erase Verify D-Flash Section Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 010 at command launch
		Set if command not available in current mode (see Table 8-30)
		Set if an invalid global address [22:0] is supplied
		Set if a misaligned word address is supplied (global address [0] != 0)
		Set if the global address [22:0] points to an area of the D-Flash EEE partition
		Set if the requested section breaches the end of the D-Flash block or goes into the D-Flash EEE partition
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read
	MGSTAT0	Set if any non-correctable errors have been encountered during the read
FERSTAT	EPVIOLIF	None

8.4.2.16 Program D-Flash Command

The Program D-Flash operation programs one to four previously erased words in the D-Flash user partition. The Program D-Flash operation will confirm that the targeted location(s) were successfully programmed upon completion.

CAUTION

A Flash word must be in the erased state before being programmed. Cumulative programming of bits within a Flash word is not allowed.

Table 8-65. Program D-Flash Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x11	Global address [22:16] to identify the D-Flash block
001	Global address [15:0] of word to be programmed	
010	Word 0 program value	
011	Word 1 program value, if desired	
100	Word 2 program value, if desired	
101	Word 3 program value, if desired	

Upon clearing CCIF to launch the Program D-Flash command, the user-supplied words will be transferred to the Memory Controller and be programmed. The CCOBIX index value at Program D-Flash command launch determines how many words will be programmed in the D-Flash block. No protection checks are made in the Program D-Flash operation on the D-Flash block, only access error checks. The CCIF flag is set when the operation has completed.

Table 8-66. Program D-Flash Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] < 010 at command launch
		Set if CCOBIX[2:0] > 101 at command launch
		Set if command not available in current mode (see Table 8-30)
		Set if an invalid global address [22:0] is supplied
		Set if a misaligned word address is supplied (global address [0] != 0)
		Set if the global address [22:0] points to an area in the D-Flash EEE partition
		Set if the requested group of words breaches the end of the D-Flash block or goes into the D-Flash EEE partition
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation
FERSTAT	EPVIOLIF	None

8.4.2.17 Erase D-Flash Sector Command

The Erase D-Flash Sector operation will erase all addresses in a sector of the D-Flash user partition.

Table 8-67. Erase D-Flash Sector Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x12	Global address [22:16] to identify D-Flash block
001	Global address [15:0] anywhere within the sector to be erased. See Section 8.1.2.2 for D-Flash sector size.	

Upon clearing CCIF to launch the Erase D-Flash Sector command, the Memory Controller will erase the selected Flash sector and verify that it is erased. The CCIF flag will set after the Erase D-Flash Sector operation has completed.

Table 8-68. Erase D-Flash Sector Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if command not available in current mode (see Table 8-30)
		Set if an invalid global address [22:0] is supplied
		Set if a misaligned word address is supplied (global address [0] != 0)
		Set if the global address [22:0] points to the D-Flash EEE partition
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation
FERSTAT	EPVIOLIF	None

8.4.2.18 Enable EEPROM Emulation Command

The Enable EEPROM Emulation command causes the Memory Controller to enable EEE activity. EEE activity is disabled after any reset.

Table 8-69. Enable EEPROM Emulation Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x13	Not required

Upon clearing CCIF to launch the Enable EEPROM Emulation command, the CCIF flag will set after the Memory Controller enables EEE operations using the contents of the EEE tag RAM and tag counter. The Full Partition D-Flash or the Partition D-Flash command must be run prior to launching the Enable EEPROM Emulation command.

Table 8-70. Enable EEPROM Emulation Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if Full Partition D-Flash or Partition D-Flash command not previously run
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

8.4.2.19 Disable EEPROM Emulation Command

The Disable EEPROM Emulation command causes the Memory Controller to suspend current EEE activity.

Table 8-71. Disable EEPROM Emulation Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x14	Not required

Upon clearing CCIF to launch the Disable EEPROM Emulation command, the Memory Controller will halt EEE operations at the next convenient point without clearing the EEE tag RAM or tag counter before setting the CCIF flag.

Table 8-72. Disable EEPROM Emulation Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if Full Partition D-Flash or Partition D-Flash command not previously run
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

8.4.2.20 EEPROM Emulation Query Command

The EEPROM Emulation Query command returns EEE partition and status variables.

Table 8-73. EEPROM Emulation Query Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x15	Not required
001	Return DFPART	
010	Return ERPART	
011	Return ECOUNT(1)	
100	Return Dead Sector Count	Return Ready Sector Count

1. Indicates sector erase count

Upon clearing CCIF to launch the EEPROM Emulation Query command, the CCIF flag will set after the EEE partition and status variables are stored in the FCCOBIX register.If the Emulation Query command is executed prior to partitioning (Partition D-Flash Command Section 8.4.2.14), the following reset values are returned: DFPART = 0x_FFFF, ERPART = 0x_FFFF, ECOUNT = 0x_FFFF, Dead Sector Count = 0x_00, Ready Sector Count = 0x_00.

Table 8-74. EEPROM Emulation Query Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if command not available in current mode (see Table 8-30)
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

8.4.2.21 Partition D-Flash Command

The Partition D-Flash command allows the user to allocate sectors within the D-Flash block for applications and a partition within the buffer RAM for EEPROM access. The D-Flash block consists of 64 sectors with 256 bytes per sector. The Erase All Blocks command must be run prior to launching the Partition D-Flash command.

Table 8-75. Partition D-Flash Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x20	Not required
001	Number of 256 byte sectors for the D-Flash user partition (DFPART)	
010	Number of 256 byte sectors for buffer RAM EEE partition (ERPART)	

Upon clearing CCIF to launch the Partition D-Flash command, the following actions are taken to define a partition within the D-Flash block for direct access (DFPART) and a partition within the buffer RAM for EEE use (ERPART):

- Validate the DFPART and ERPART values provided:
  - DFPART <= 128 (maximum number of 256 byte sectors in D-Flash block)
  - ERPART <= 8 (maximum number of 256 byte sectors in buffer RAM)
  - If ERPART > 0, 128 - DFPART >= 12 (minimum number of 256 byte sectors in the D-Flash block required to support EEE)
  - If ERPART > 0, ((128-DFPART)/ERPART) >= 8 (minimum ratio of D-Flash EEE space to buffer RAM EEE space to support EEE)
- Erase verify the D-Flash block and the EEE nonvolatile information register
- Program DFPART to the EEE nonvolatile information register at global address 0x12_0000 (see Table 8-7)
- Program a duplicate DFPART to the EEE nonvolatile information register at global address 0x12_0002 (see Table 8-7)

- Program ERPART to the EEE nonvolatile information register at global address 0x12_0004 (see Table 8-7)
- Program a duplicate ERPART to the EEE nonvolatile information register at global address 0x12_0006 (see Table 8-7)

The D-Flash user partition will start at global address 0x10_0000. The buffer RAM EEE partition will end at global address 0x13_FFFF. After the Partition D-Flash operation has completed, the CCIF flag will set.

Running the Partition D-Flash command a second time will result in the ACCERR bit within the FSTAT register being set. The data value written corresponds to the number of 256 byte sectors allocated for either direct D-Flash access (DFPART) or buffer RAM EEE access (ERPART).

Table 8-76. Partition D-Flash Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 010 at command launch
		Set if command not available in current mode (see Table 8-30)
		Set if partitions have already been defined
		Set if an invalid DFPART or ERPART selection is supplied(1)
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read
	MGSTAT0	Set if any non-correctable errors have been encountered during the read
FERSTAT	EPVIOLIF	None

1. As defined by the maximum ERPART for FTM512K3.

8.4.3 Interrupts

The Flash module can generate an interrupt when a Flash command operation has completed or when a Flash command operation has detected an EEE error or an ECC fault.

Table 8-77. Flash Interrupt Sources

Interrupt Source	Interrupt Flag	Local Enable	Global (CCR) Mask
Flash Command Complete	CCIF (FSTAT register)	CCIE (FCNFG register)	I Bit
Flash EEE Erase Error	ERSERIF (FERSTAT register)	ERSERIE (FERCNFG register)	I Bit
Flash EEE Program Error	PGMERIF (FERSTAT register)	PGMERIE (FERCNFG register)	I Bit
Flash EEE Protection Violation	EPVIOLIF (FERSTAT register)	EPVIOLIE (FERCNFG register)	I Bit
Flash EEE Error Type 1 Violation	ERSVIF1 (FERSTAT register)	ERSVIE1 (FERCNFG register)	I Bit
Flash EEE Error Type 0 Violation	ERSVIF0 (FERSTAT register)	ERSVIE0 (FERCNFG register)	I Bit
ECC Double Bit Fault on Flash Read	DFDIF (FERSTAT register)	DFDIE (FERCNFG register)	I Bit
ECC Single Bit Fault on Flash Read	SFDIF (FERSTAT register)	SFDIE (FERCNFG register)	I Bit

NOTE

Vector addresses and their relative interrupt priority are determined at the MCU level.

8.4.3.1 Description of Flash Interrupt Operation

The Flash module uses the CCIF flag in combination with the CCIE interrupt enable bit to generate the Flash command interrupt request. The Flash module uses the ERSEIF, PGMEIF, EPVIOLIF, ERSVIF1, ERSVIF0, DFDIF and SFDIF flags in combination with the ERSEIE, PGMEIE, EPVIOLIE, ERSVIE1, ERSVIE0, DFDIE and SFDIE interrupt enable bits to generate the Flash error interrupt request. For a detailed description of the register bits involved, refer to Section 8.3.2.5, “Flash Configuration Register (FCNFG)”, Section 8.3.2.6, “Flash Error Configuration Register (FERCNFG)”, Section 8.3.2.7, “Flash Status Register (FSTAT)”, and Section 8.3.2.8, “Flash Error Status Register (FERSTAT)”.

The logic used for generating the Flash module interrupts is shown in Figure 8-27.

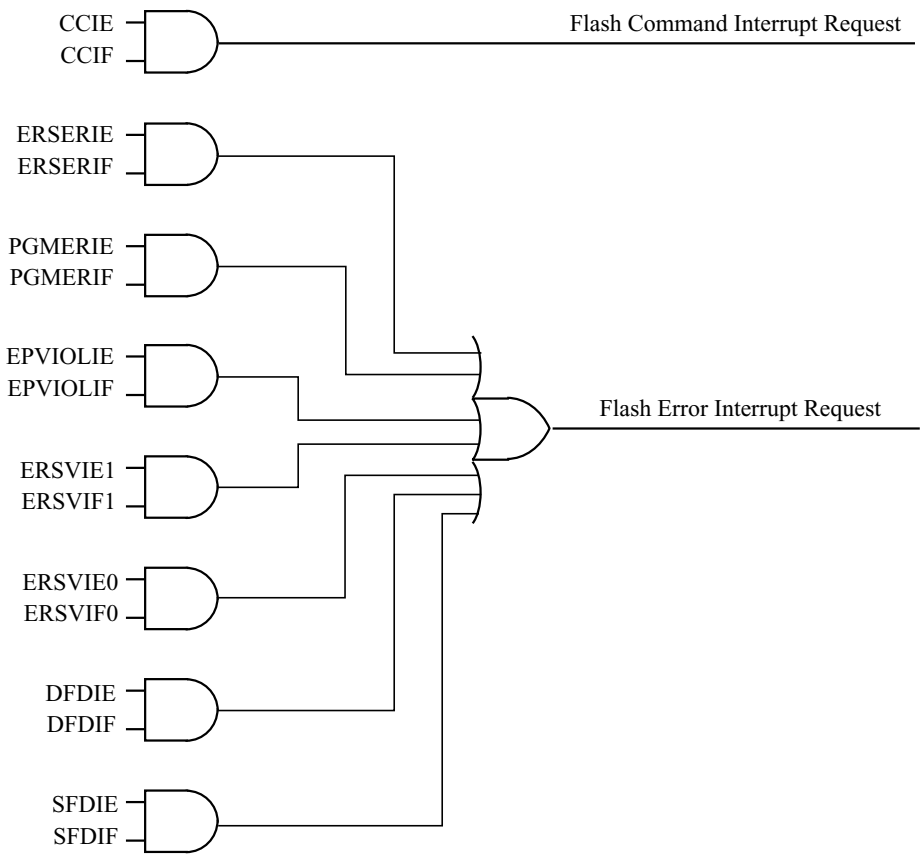


Figure 8-27. Flash Module Interrupts Implementation

8.4.4 Wait Mode

The Flash module is not affected if the MCU enters wait mode. The Flash module can recover the MCU from wait via the CCIF interrupt (see Section 8.4.3, “Interrupts”).

8.4.5 Stop Mode

If a Flash command is active (CCIF = 0) or an EE-Emulation operation is pending when the MCU requests stop mode, the current Flash operation will be completed before the CPU is allowed to enter stop mode.

8.5 Security

The Flash module provides security information to the MCU. The Flash security state is defined by the SEC bits of the FSEC register (see Table 8-12). During reset, the Flash module initializes the FSEC register using data read from the security byte of the Flash configuration field at global address 0x7F_FF0F.

The security state out of reset can be permanently changed by programming the security byte of the Flash configuration field. This assumes that you are starting from a mode where the necessary P-Flash erase and program commands are available and that the upper region of the P-Flash is unprotected. If the Flash security byte is successfully programmed, its new value will take affect after the next MCU reset.

The following subsections describe these security-related subjects:

- Unsecuring the MCU using Backdoor Key Access
- Unsecuring the MCU in Special Single Chip Mode using BDM
- Mode and Security Effects on Flash Command Availability

8.5.1 Unsecuring the MCU using Backdoor Key Access

The MCU may be unsecured by using the backdoor key access feature which requires knowledge of the contents of the backdoor keys (four 16-bit words programmed at addresses 0x7F_FF00–0x7F_FF07). If the KEYEN[1:0] bits are in the enabled state (see Section 8.3.2.2), the Verify Backdoor Access Key command (see Section 8.4.2.11) allows the user to present four prospective keys for comparison to the keys stored in the Flash memory via the Memory Controller. If the keys presented in the Verify Backdoor Access Key command match the backdoor keys stored in the Flash memory, the SEC bits in the FSEC register (see Table 8-12) will be changed to unsecure the MCU. Key values of 0x0000 and 0xFFFF are not permitted as backdoor keys. While the Verify Backdoor Access Key command is active, P-Flash block 0 will not be available for read access and will return invalid data.

The user code stored in the P-Flash memory must have a method of receiving the backdoor keys from an external stimulus. This external stimulus would typically be through one of the on-chip serial ports.

If the KEYEN[1:0] bits are in the enabled state (see Section 8.3.2.2), the MCU can be unsecured by the backdoor key access sequence described below:

1. Follow the command sequence for the Verify Backdoor Access Key command as explained in Section 8.4.2.11
2. If the Verify Backdoor Access Key command is successful, the MCU is unsecured and the SEC[1:0] bits in the FSEC register are forced to the unsecure state of 10

The Verify Backdoor Access Key command is monitored by the Memory Controller and an illegal key will prohibit future use of the Verify Backdoor Access Key command. A reset of the MCU is the only method to re-enable the Verify Backdoor Access Key command.

After the backdoor keys have been correctly matched, the MCU will be unsecured. After the MCU is unsecured, the sector containing the Flash security byte can be erased and the Flash security byte can be reprogrammed to the unsecure state, if desired.

In the unsecure state, the user has full control of the contents of the backdoor keys by programming addresses 0x7F_FF00–0x7F_FF07 in the Flash configuration field.

The security as defined in the Flash security byte (0x7F_FF0F) is not changed by using the Verify Backdoor Access Key command sequence. The backdoor keys stored in addresses 0x7F_FF00–0x7F_FF07 are unaffected by the Verify Backdoor Access Key command sequence. After the next reset of the MCU, the security state of the Flash module is determined by the Flash security byte

(0x7F_FF0F). The Verify Backdoor Access Key command sequence has no effect on the program and erase protections defined in the Flash protection register, FPROT.

8.5.2 Unsecuring the MCU in Special Single Chip Mode using BDM

The MCU can be unsecured in special single chip mode by erasing the P-Flash and D-Flash memory by one of the following methods:

- Reset the MCU into special single chip mode, delay while the erase test is performed by the BDM, send BDM commands to disable protection in the P-Flash and D-Flash memory, and execute the Erase All Blocks command write sequence to erase the P-Flash and D-Flash memory.
- Reset the MCU into special expanded wide mode, disable protection in the P-Flash and D-Flash memory and run code from external memory to execute the Erase All Blocks command write sequence to erase the P-Flash and D-Flash memory.

After the CCIF flag sets to indicate that the Erase All Blocks operation has completed, reset the MCU into special single chip mode. The BDM will execute the Erase Verify All Blocks command write sequence to verify that the P-Flash and D-Flash memory is erased. If the P-Flash and D-Flash memory are verified as erased the MCU will be unsecured. All BDM commands will be enabled and the Flash security byte may be programmed to the unsecure state by the following method:

- Send BDM commands to execute a ‘Program P-Flash’ command sequence to program the Flash security byte to the unsecured state and reset the MCU.

8.5.3 Mode and Security Effects on Flash Command Availability

The availability of Flash module commands depends on the MCU operating mode and security state as shown in Table 8-30.

8.6 Initialization

On each system reset the Flash module executes a reset sequence which establishes initial values for the Flash Block Configuration Parameters, the FPROT and DFPROT protection registers, and the FOPT and FSEC registers. The Flash module reverts to built-in default values that leave the module in a fully protected and secured state if errors are encountered during execution of the reset sequence. If a double bit fault is detected during the reset sequence, both MGSTAT bits in the FSTAT register will be set. The ACCERR bit in the FSTAT register is set if errors are encountered while initializing the EEE buffer ram during the reset sequence.

CCIF remains clear throughout the reset sequence. The Flash module holds off all CPU access for the initial portion of the reset sequence. While Flash reads are possible when the hold is removed, writes to the FCCOBIX, FCCOBHI, and FCCOBLO registers are ignored to prevent command activity while the Memory Controller remains busy. Completion of the reset sequence is marked by setting CCIF high which enables writes to the FCCOBIX, FCCOBHI, and FCCOBLO registers to launch any available Flash command.

If a reset occurs while any Flash command is in progress, that command will be immediately aborted. The state of the word being programmed or the sector/block being erased is not guaranteed.

Chapter 9
512 KByte Flash Module (S12XFTM512K3V1)

Table 9-1. Revision History

Revision Number	Revision Date	Sections Affected	Description of Changes
V01.09	14 Nov 2007	9.5.2/9-341	- Changed terminology from ‘word program’ to “Program P-Flash’ in the BDM unsecuring description, Section 9.5.2
		9.4.2/9-317	- Added requirement that user not write any Flash module register during execution of commands ‘Erase All Blocks’, Section 9.4.2.8, and ‘Unsecure Flash’, Section 9.4.2.11
		9.4.2.8/9-323	- Added statement that security is released upon successful completion of command ‘Erase All Blocks’, Section 9.4.2.8
V01.10	19 Dec 2007	9.4.2/9-317	- Corrected Error Handling table for Full Partition D-Flash, Partition D-Flash, and EEPROM Emulation Query commands
V01.11	25 Sep 2009	9.1/9-281	- Clarify single bit fault correction for P-Flash phrase
		9.3.2.1/9-293	- Expand FDIV vs OSCCLK Frequency table
		9.4.2.4/9-320	- Add statement concerning code runaway when executing Read Once command from Flash block containing associated fields
		9.4.2.7/9-322	- Add statement concerning code runaway when executing Program Once command from Flash block containing associated fields
		9.4.2.12/9-326	- Add statement concerning code runaway when executing Verify Backdoor Access Key command from Flash block containing associated fields
		9.4.2.12/9-326	- Relate Key 0 to associated Backdoor Comparison Key address
		9.4.2.12/9-326	- Change “power down reset” to “reset”
		9.4.2.20/9-335	- Add ACCERR condition for Disable EEPROM Emulation command
			The following changes were made to clarify module behavior related to Flash register access during reset sequence and while Flash commands are active:
		9.3.2/9-291	- Add caution concerning register writes while command is active
		9.3.2.1/9-293	- Writes to FCLKDIV are allowed during reset sequence while CCIF is clear
		9.4.1.2/9-312	- Add caution concerning register writes while command is active
		9.6/9-341	- Writes to FCCOBIX, FCCOBHI, FCCOBLO registers are ignored during reset sequence

9.1 Introduction

The FTM512K3 module implements the following:

- 512 Kbytes of P-Flash (Program Flash) memory, consisting of 3 physical Flash blocks, intended primarily for nonvolatile code storage
- 32 Kbytes of D-Flash (Data Flash) memory, consisting of 1 physical Flash block, that can be used as nonvolatile storage to support the built-in hardware scheme for emulated EEPROM, as basic Flash memory primarily intended for nonvolatile data storage, or as a combination of both
- 4 Kbytes of buffer RAM, consisting of 1 physical RAM block, that can be used as emulated EEPROM using a built-in hardware scheme, as basic RAM, or as a combination of both

The Flash memory is ideal for single-supply applications allowing for field reprogramming without requiring external high voltage sources for program or erase operations. The Flash module includes a memory controller that executes commands to modify Flash memory contents or configure module resources for emulated EEPROM operation. The user interface to the memory controller consists of the indexed Flash Common Command Object (FCCOB) register which is written to with the command, global address, data, and any required command parameters. The memory controller must complete the execution of a command before the FCCOB register can be written to with a new command.

CAUTION

A Flash word or phrase must be in the erased state before being programmed. Cumulative programming of bits within a Flash word or phrase is not allowed.

The RAM and Flash memory may be read as bytes, aligned words, or misaligned words. Read access time is one bus cycle for bytes and aligned words, and two bus cycles for misaligned words. For Flash memory, an erased bit reads 1 and a programmed bit reads 0.

It is not possible to read from a Flash block while any command is executing on that specific Flash block. It is possible to read from a Flash block while a command is executing on a different Flash block.

Both P-Flash and D-Flash memories are implemented with Error Correction Codes (ECC) that can resolve single bit faults and detect double bit faults. For P-Flash memory, the ECC implementation requires that programming be done on an aligned 8 byte basis (a Flash phrase). Since P-Flash memory is always read by phrase, only one single bit fault in the phrase containing the byte or word accessed will be corrected.

9.1.1 Glossary

Buffer RAM — The buffer RAM constitutes the volatile memory store required for EEE. Memory space in the buffer RAM not required for EEE can be partitioned to provide volatile memory space for applications.

Command Write Sequence — An MCU instruction sequence to execute built-in algorithms (including program and erase) on the Flash memory.

D-Flash Memory — The D-Flash memory constitutes the nonvolatile memory store required for EEE. Memory space in the D-Flash memory not required for EEE can be partitioned to provide nonvolatile memory space for applications.

D-Flash Sector — The D-Flash sector is the smallest portion of the D-Flash memory that can be erased. The D-Flash sector consists of four 64 byte rows for a total of 256 bytes.

EEE (Emulated EEPROM) — A method to emulate the small sector size features and endurance characteristics associated with an EEPROM.

EEE IFR — Nonvolatile information register located in the D-Flash block that contains data required to partition the D-Flash memory and buffer RAM for EEE. The EEE IFR is visible in the global memory map by setting the EEEIFRON bit in the MMCCTL1 register.

NVM Command Mode — An NVM mode using the CPU to setup the FCCOB register to pass parameters required for Flash command execution.

Phrase — An aligned group of four 16-bit words within the P-Flash memory. Each phrase includes eight ECC bits for single bit fault correction and double bit fault detection within the phrase.

P-Flash Memory — The P-Flash memory constitutes the main nonvolatile memory store for applications.

P-Flash Sector — The P-Flash sector is the smallest portion of the P-Flash memory that can be erased. Each P-Flash sector contains 1024 bytes.

Program IFR — Nonvolatile information register located in the P-Flash block that contains the Device ID, Version ID, and the Program Once field. The Program IFR is visible in the global memory map by setting the PGMIFRON bit in the MMCCTL1 register.

9.1.2 Features

9.1.2.1 P-Flash Features

- 512 Kbytes of P-Flash memory composed of one 256 Kbyte Flash block and two 128 Kbyte Flash blocks. The 256 Kbyte Flash block consists of two 128 Kbyte sections each divided into 128 sectors of 1024 bytes. The 128 Kbyte Flash blocks are each divided into 128 sectors of 1024 bytes.
- Single bit fault correction and double bit fault detection within a 64-bit phrase during read operations
- Automated program and erase algorithm with verify and generation of ECC parity bits
- Fast sector erase and phrase program operation
- Ability to program up to one phrase in each P-Flash block simultaneously
- Flexible protection scheme to prevent accidental program or erase of P-Flash memory

9.1.2.2 D-Flash Features

- Up to 32 Kbytes of D-Flash memory with 256 byte sectors for user access
- Dedicated commands to control access to the D-Flash memory over EEE operation
- Single bit fault correction and double bit fault detection within a word during read operations
- Automated program and erase algorithm with verify and generation of ECC parity bits
- Fast sector erase and word program operation
- Ability to program up to four words in a burst sequence

9.1.2.3 Emulated EEPROM Features

- Up to 4 Kbytes of emulated EEPROM (EEE) accessible as 4 Kbytes of RAM

- Flexible protection scheme to prevent accidental program or erase of data
- Automatic EEE file handling using an internal Memory Controller
- Automatic transfer of valid EEE data from D-Flash memory to buffer RAM on reset
- Ability to monitor the number of outstanding EEE related buffer RAM words left to be programmed into D-Flash memory
- Ability to disable EEE operation and allow priority access to the D-Flash memory
- Ability to cancel all pending EEE operations and allow priority access to the D-Flash memory

9.1.2.4 User Buffer RAM Features

- Up to 4 Kbytes of RAM for user access

9.1.2.5 Other Flash Module Features

- No external high-voltage power supply required for Flash memory program and erase operations
- Interrupt generation on Flash command completion and Flash error detection
- Security mechanism to prevent unauthorized access to the Flash memory

9.1.3 Block Diagram

The block diagram of the Flash module is shown in Figure 9-1.

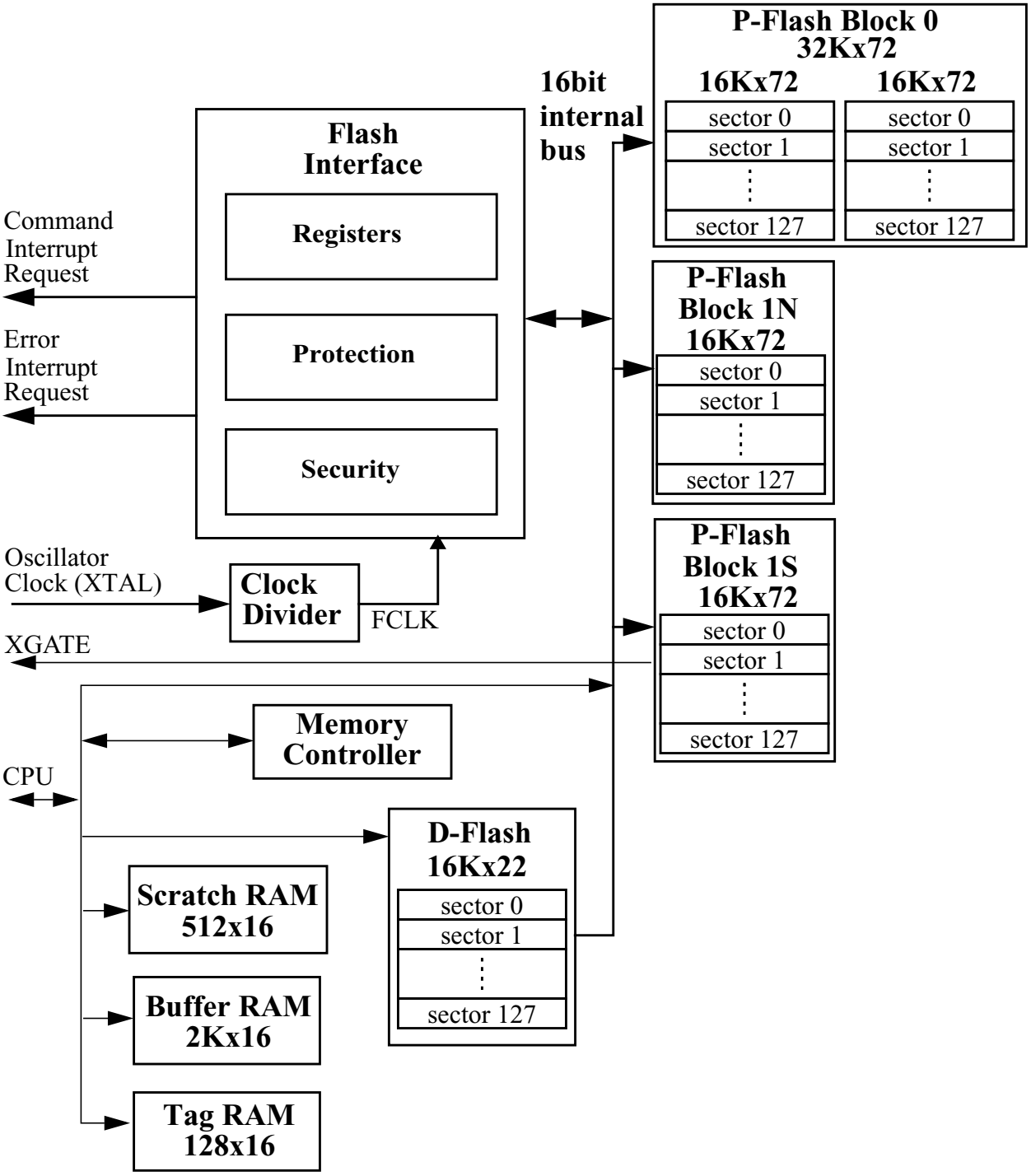


Figure 9-1. FTM512K3 Block Diagram

9.2 External Signal Description

The Flash module contains no signals that connect off-chip.

9.3 Memory Map and Registers

This section describes the memory map and registers for the Flash module. Read data from unimplemented memory space in the Flash module is undefined. Write access to unimplemented or reserved memory space in the Flash module will be ignored by the Flash module.

9.3.1 Module Memory Map

The S12X architecture places the P-Flash memory between global addresses 0x78_0000 and 0x7F_FFFF as shown in Table 9-2. The P-Flash memory map is shown in Figure 9-2.

Table 9-2. P-Flash Memory Addressing

Global Address	Size (Bytes)	Description
0x7C_0000 – 0x7F_FFFF	256 K	P-Flash Block 0 Contains Flash Configuration Field (see Table 9-3)
0x7A_0000 – 0x7B_FFFF	128 K	P-Flash Block 1N
0x78_0000 – 0x79_FFFF	128 K	P-Flash Block 1S

The FPROT register, described in Section 9.3.2.9, can be set to protect regions in the Flash memory from accidental program or erase. Three separate memory regions, one growing upward from global address 0x7F_8000 in the Flash memory (called the lower region), one growing downward from global address 0x7F_FFFF in the Flash memory (called the higher region), and the remaining addresses in the Flash memory, can be activated for protection. The Flash memory addresses covered by these protectable regions are shown in the P-Flash memory map. The higher address region is mainly targeted to hold the boot loader code since it covers the vector space. Default protection settings as well as security information that allows the MCU to restrict access to the Flash module are stored in the Flash configuration field as described in Table 9-3.

Table 9-3. Flash Configuration Field(1)

Global Address	Size (Bytes)	Description
0x7F_FF00 – 0x7F_FF07	8	Backdoor Comparison Key Refer to Section 9.4.2.12, “Verify Backdoor Access Key Command,” and Section 9.5.1, “Unsecuring the MCU using Backdoor Key Access”
0x7F_FF08 – 0x7F_FF0B(2)	4	Reserved
0x7F_FF0C[2]	1	P-Flash Protection byte. Refer to Section 9.3.2.9, “P-Flash Protection Register (FPROT)”
0x7F_FF0D[2]	1	EEE Protection byte Refer to Section 9.3.2.10, “EEE Protection Register (EPROT)”
0x7F_FF0E[2]	1	Flash Nonvolatile byte Refer to Section 9.3.2.14, “Flash Option Register (FOPT)”

Table 9-3. Flash Configuration Field[1]

Global Address	Size (Bytes)	Description
0x7F_FF0F[2]	1	Flash Security byte Refer to Section 9.3.2.2, “Flash Security Register (FSEC)”

1. Older versions may have swapped protection byte addresses
2. 0x7FF08 - 0x7F_FF0F form a Flash phrase and must be programmed in a single command write sequence. Each byte in the 0x7F_FF08 - 0x7F_FF0B reserved field should be programmed to 0xFF.

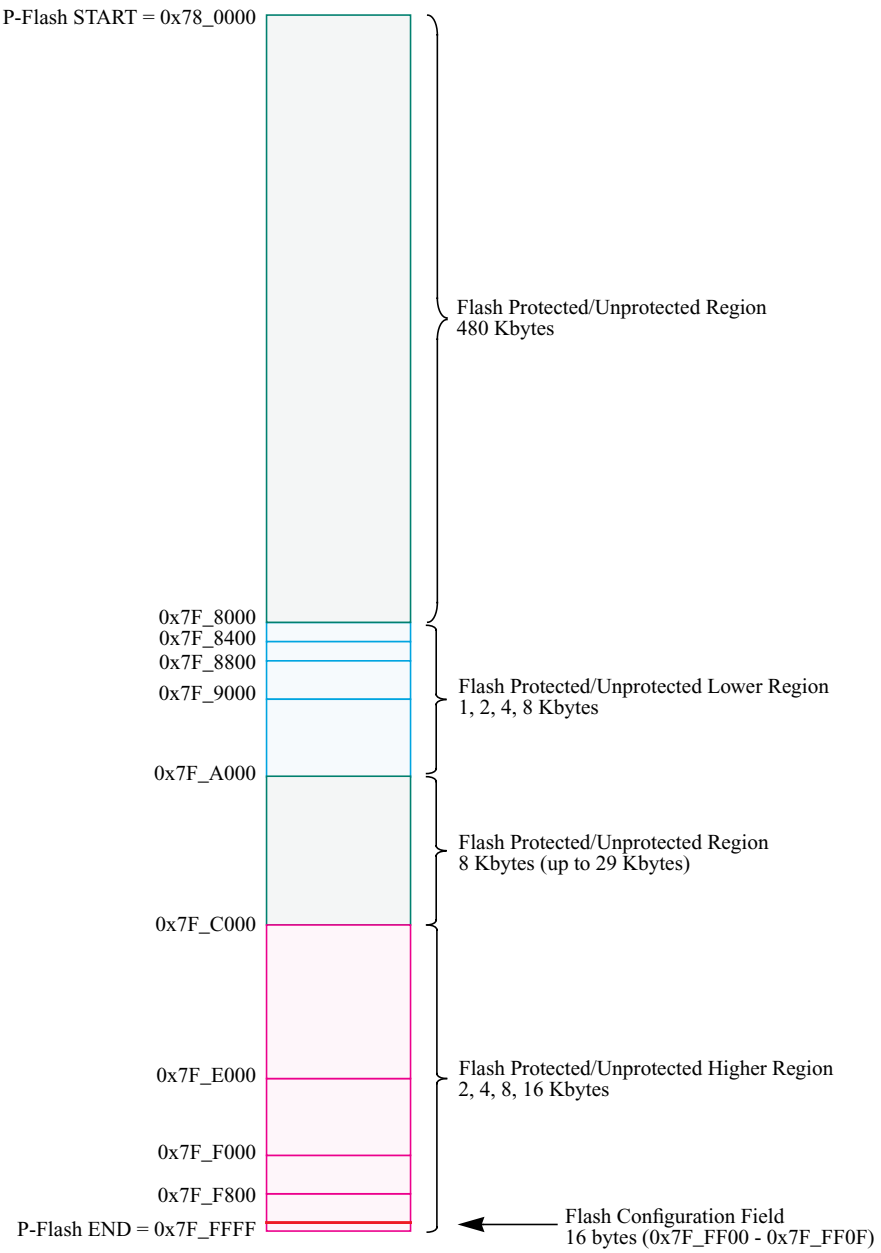


Figure 9-2. P-Flash Memory Map

Table 9-4. Program IFR Fields

Global Address (PGMIFRON)	Size (Bytes)	Field Description
0x40_0000 – 0x40_0007	8	Device ID
0x40_0008 – 0x40_00E7	224	Reserved
0x40_00E8 – 0x40_00E9	2	Version ID
0x40_00EA – 0x40_00FF	22	Reserved
0x40_0100 – 0x40_013F	64	Program Once Field Refer to Section 9.4.2.7, “Program Once Command”
0x40_0140 – 0x40_01FF	192	Reserved

Table 9-5. P-Flash IFR Accessibility

Global Address (PGMIFRON)	Size (Bytes)	Accessed From
0x40_0000 – 0x40_01FF	512	XBUS0 (PBLK0S)(1)
0x40_0200 – 0x40_03FF	512	Unimplemented
0x40_0400 – 0x40_05FF	512	XBUS0 (PBLK1N)
0x40_0600 – 0x40_07FF	512	XBUS1 (PBLK1S)

1. Refer to Table 9-4 for more details.

Table 9-6. EEE Resource Fields

Global Address	Size (Bytes)	Description
0x10_0000 – 0x10_7FFF	32,768	D-Flash Memory (User and EEE)
0x10_8000 – 0x11_FFFF	98,304	Reserved
0x12_0000 – 0x12_007F	128	EEE Nonvolatile Information Register (EEEIFRON(1) = 1)
0x12_0080 – 0x12_0FFF	3,968	Reserved
0x12_1000 – 0x12_1EFF	3,840	Reserved
0x12_1F00 – 0x12_1FFF	256	EEE Tag RAM (TMGRAMON[1] = 1)
0x12_2000 – 0x12_3BFF	7,168	Reserved
0x12_3C00 – 0x12_3FFF	1,024	Memory Controller Scratch RAM (TMGRAMON[1] = 1)
0x12_4000 – 0x12_DFFF	40,960	Reserved
0x12_E000 – 0x12_FFFF	8,192	Reserved
0x13_0000 – 0x13_EFFF	61,440	Reserved
0x13_F000 – 0x13_FFFF	4,096	Buffer RAM (User and EEE)

1. MMCCTL1 register bit

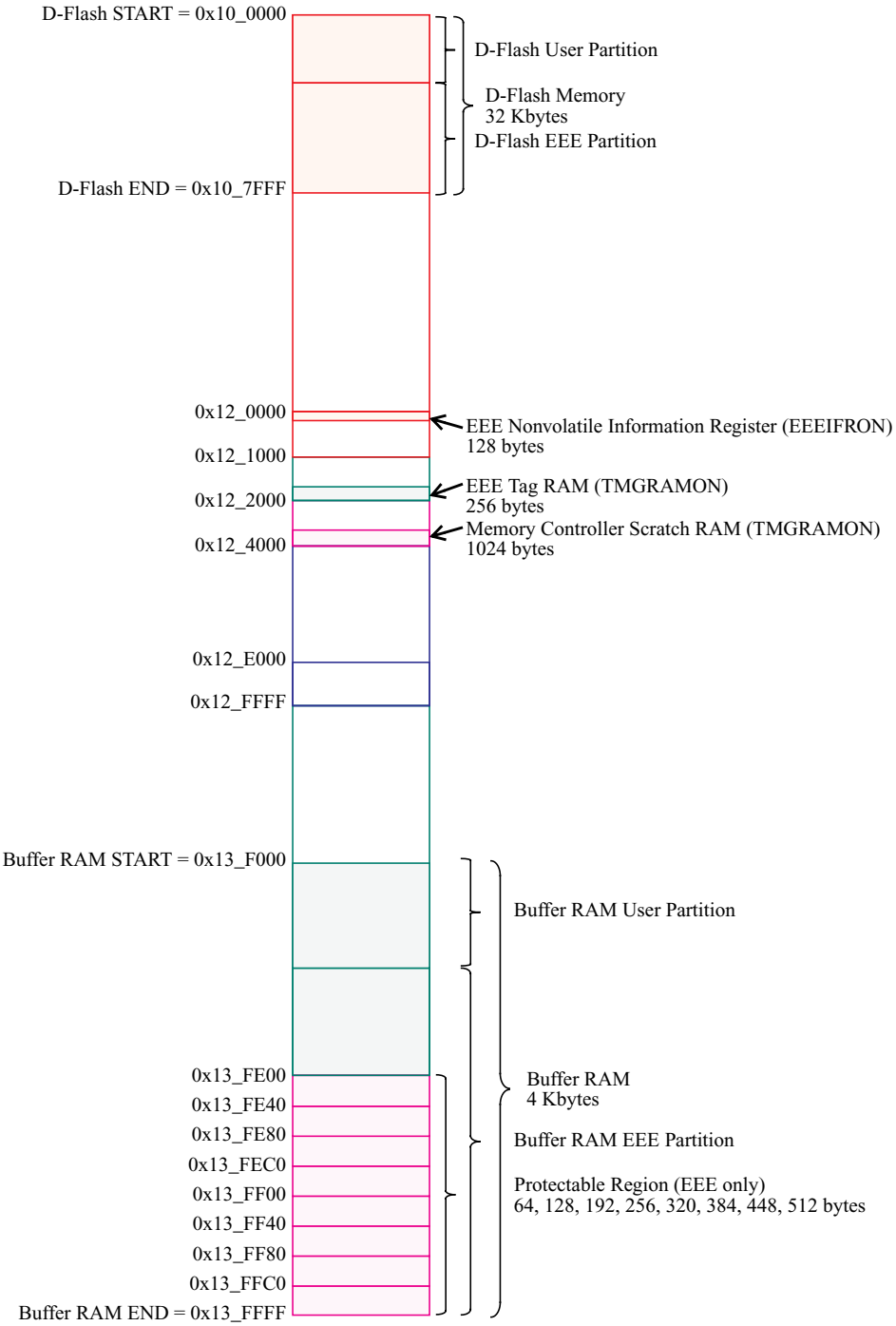


Figure 9-3. EEE Resource Memory Map

The Full Partition D-Flash command (see Section 9.4.2.15) is used to program the EEE nonvolatile information register fields where address 0x12_0000 defines the D-Flash partition for user access and address 0x12_0004 defines the buffer RAM partition for EEE operations.

Table 9-7. EEE Nonvolatile Information Register Fields

Global Address (EEEIFRON)	Size (Bytes)	Description
0x12_0000 – 0x12_0001	2	D-Flash User Partition (DFPART) Refer to Section 9.4.2.15, “Full Partition D-Flash Command”
0x12_0002 – 0x12_0003	2	D-Flash User Partition (duplicate(1))
0x12_0004 – 0x12_0005	2	Buffer RAM EEE Partition (ERPART) Refer to Section 9.4.2.15, “Full Partition D-Flash Command”
0x12_0006 – 0x12_0007	2	Buffer RAM EEE Partition (duplicate1)
0x12_0008 – 0x12_007F	120	Reserved

1. Duplicate value used if primary value generates a double bit fault when read during the reset sequence.

9.3.2 Register Descriptions

The Flash module contains a set of 20 control and status registers located between Flash module base + 0x0000 and 0x0013. A summary of the Flash module registers is given in Figure 9-4 with detailed descriptions in the following subsections.

CAUTION

Writes to any Flash register must be avoided while a Flash command is active (CCIF=0) to prevent corruption of Flash register contents and Memory Controller behavior.

Address & Name		7	6	5	4	3	2	1	0
0x0000 FCLKDIV	R	FDIVLD	FDIV6	FDIV5	FDIV4	FDIV3	FDIV2	FDIV1	FDIV0
	W								
0x0001 FSEC	R	KEYEN1	KEYEN0	RNV5	RNV4	RNV3	RNV2	SEC1	SEC0
	W								
0x0002 FCCOBIX	R	0	0	0	0	0	CCOBIX2	CCOBIX1	CCOBIX0
	W								
0x0003 FECCRIX	R	0	0	0	0	0	ECCRIX2	ECCRIX1	ECCRIX0
	W								
0x0004 FCNFG	R	CCIE	0	0	IGNSF	0	0	FDFD	FSFD
	W								
0x0005 FERCNFG	R	ERSERIE	PGMERIE	0	EPVIOLIE	ERSVIE1	ERSVIE0	DFDIE	SFDIE
	W								

Figure 9-4. FTM512K3 Register Summary

Address & Name		7	6	5	4	3	2	1	0
0x0006 FSTAT	R	CCIF	0	ACCERR	FPVIOL	MGBUSY	RSVD	MGSTAT1	MGSTAT0
	W								
0x0007 FERSTAT	R	ERSERIF	PGMERIF	0	EPVIOLIF	ERSVIF1	ERSVIF0	DFDIF	SFDIF
	W								
0x0008 FPROT	R	FPOPEN	RNV6	FPHDIS	FPHS1	FPHS0	FPLDIS	FPLS1	FPLS0
	W								
0x0009 EPROT	R	EPOPEN	RNV6	RNV5	RNV4	EPDIS	EPS2	EPS1	EPS0
	W								
0x000A FCCOBHI	R	CCOB15	CCOB14	CCOB13	CCOB12	CCOB11	CCOB10	CCOB9	CCOB8
	W								
0x000B FCCOBLO	R	CCOB7	CCOB6	CCOB5	CCOB4	CCOB3	CCOB2	CCOB1	CCOB0
	W								
0x000C ETAGHI	R	ETAG15	ETAG14	ETAG13	ETAG12	ETAG11	ETAG10	ETAG9	ETAG8
	W								
0x000D ETAGLO	R	ETAG7	ETAG6	ETAG5	ETAG4	ETAG3	ETAG2	ETAG1	ETAG0
	W								
0x000E FECCRHI	R	ECCR15	ECCR14	ECCR13	ECCR12	ECCR11	ECCR10	ECCR9	ECCR8
	W								
0x000F FECCRLO	R	ECCR7	ECCR6	ECCR5	ECCR4	ECCR3	ECCR2	ECCR1	ECCR0
	W								
0x0010 FOPT	R	NV7	NV6	NV5	NV4	NV3	NV2	NV1	NV0
	W								
0x0011 FRSV0	R	0	0	0	0	0	0	0	0
	W								
0x0012 FRSV1	R	0	0	0	0	0	0	0	0
	W								
0x0013 FRSV2	R	0	0	0	0	0	0	0	0
	W								

Figure 9-4. FTM512K3 Register Summary (continued)

Address & Name	7	6	5	4	3	2	1	0

= Unimplemented or Reserved

Figure 9-4. FTM512K3 Register Summary (continued)

9.3.2.1 Flash Clock Divider Register (FCLKDIV)

The FCLKDIV register is used to control timed events in program and erase algorithms.

Offset Module Base + 0x0000

	7	6	5	4	3	2	1	0
R	FDIVLD	FDIV[6:0]						
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 9-5. Flash Clock Divider Register (FCLKDIV)

All bits in the FCLKDIV register are readable, bits 6–0 are write once and bit 7 is not writable.

Table 9-8. FCLKDIV Field Descriptions

Field	Description
7 FDIVLD	**Clock Divider Loaded** 0 FCLKDIV register has not been written 1 FCLKDIV register has been written since the last reset
6–0 FDIV[6:0]	**Clock Divider Bits** — FDIV[6:0] must be set to effectively divide OSCCLK down to generate an internal Flash clock, FCLK, with a target frequency of 1 MHz for use by the Flash module to control timed events during program and erase algorithms. Table 9-9 shows recommended values for FDIV[6:0] based on OSCCLK frequency. Please refer to Section 9.4.1, “Flash Command Operations,” for more information.

CAUTION

The FCLKDIV register should never be written while a Flash command is executing (CCIF=0). The FCLKDIV register is writable during the Flash reset sequence even though CCIF is clear.

Table 9-9. FDIV vs OSCCLK Frequency

OSCCLK Frequency (MHz)		FDIV[6:0]	OSCCLK Frequency (MHz)		FDIV[6:0]	OSCCLK Frequency (MHz)		FDIV[6:0]
MIN(1)	MAX(2)		MIN1	MAX2		MIN1	MAX2	
			33.60	34.65	0x20	67.20	68.25	0x40
1.60	2.10	0x01	34.65	35.70	0x21	68.25	69.30	0x41
2.40	3.15	0x02	35.70	36.75	0x22	69.30	70.35	0x42
3.20	4.20	0x03	36.75	37.80	0x23	70.35	71.40	0x43
4.20	5.25	0x04	37.80	38.85	0x24	71.40	72.45	0x44
5.25	6.30	0x05	38.85	39.90	0x25	72.45	73.50	0x45
6.30	7.35	0x06	39.90	40.95	0x26	73.50	74.55	0x46
7.35	8.40	0x07	40.95	42.00	0x27	74.55	75.60	0x47
8.40	9.45	0x08	42.00	43.05	0x28	75.60	76.65	0x48
9.45	10.50	0x09	43.05	44.10	0x29	76.65	77.70	0x49
10.50	11.55	0x0A	44.10	45.15	0x2A	77.70	78.75	0x4A
11.55	12.60	0x0B	45.15	46.20	0x2B	78.75	79.80	0x4B
12.60	13.65	0x0C	46.20	47.25	0x2C	79.80	80.85	0x4C
13.65	14.70	0x0D	47.25	48.30	0x2D	80.85	81.90	0x4D
14.70	15.75	0x0E	48.30	49.35	0x2E	81.90	82.95	0x4E
15.75	16.80	0x0F	49.35	50.40	0x2F	82.95	84.00	0x4F
16.80	17.85	0x10	50.40	51.45	0x30	84.00	85.05	0x50
17.85	18.90	0x11	51.45	52.50	0x31	85.05	86.10	0x51
18.90	19.95	0x12	52.50	53.55	0x32	86.10	87.15	0x52
19.95	21.00	0x13	53.55	54.60	0x33	87.15	88.20	0x53
21.00	22.05	0x14	54.60	55.65	0x34	88.20	89.25	0x54
22.05	23.10	0x15	55.65	56.70	0x35	89.25	90.30	0x55
23.10	24.15	0x16	56.70	57.75	0x36	90.30	91.35	0x56
24.15	25.20	0x17	57.75	58.80	0x37	91.35	92.40	0x57
25.20	26.25	0x18	58.80	59.85	0x38	92.40	93.45	0x58
26.25	27.30	0x19	59.85	60.90	0x39	93.45	94.50	0x59
27.30	28.35	0x1A	60.90	61.95	0x3A	94.50	95.55	0x5A
28.35	29.40	0x1B	61.95	63.00	0x3B	95.55	96.60	0x5B
29.40	30.45	0x1C	63.00	64.05	0x3C	96.60	97.65	0x5C
30.45	31.50	0x1D	64.05	65.10	0x3D	97.65	98.70	0x5D
31.50	32.55	0x1E	65.10	66.15	0x3E	98.70	99.75	0x5E
32.55	33.60	0x1F	66.15	67.20	0x3F	99.75	100.80	0x5F

1. FDIV shown generates an FCLK frequency of >0.8 MHz

2. FDIV shown generates an FCLK frequency of 1.05 MHz

9.3.2.2 Flash Security Register (FSEC)

The FSEC register holds all bits associated with the security of the MCU and Flash module.

Offset Module Base + 0x0001

	7	6	5	4	3	2	1	0
R	KEYEN[1:0]		RNV[5:2]				SEC[1:0]	
W								
Reset	F	F	F	F	F	F	F	F

= Unimplemented or Reserved

Figure 9-6. Flash Security Register (FSEC)

All bits in the FSEC register are readable but not writable.

During the reset sequence, the FSEC register is loaded with the contents of the Flash security byte in the Flash configuration field at global address 0x7F_FF0F located in P-Flash memory (see Table 9-3) as indicated by reset condition F in Figure 9-6. If a double bit fault is detected while reading the P-Flash phrase containing the Flash security byte during the reset sequence, all bits in the FSEC register will be set to leave the Flash module in a secured state with backdoor key access disabled.

Table 9-10. FSEC Field Descriptions

Field	Description
7–6 KEYEN[1:0]	**Backdoor Key Security Enable Bits** — The KEYEN[1:0] bits define the enabling of backdoor key access to the Flash module as shown in Table 9-11.
5–2 RNV[5:2}	**Reserved Nonvolatile Bits** — The RNV bits should remain in the erased state for future enhancements.
1–0 SEC[1:0]	**Flash Security Bits** — The SEC[1:0] bits define the security state of the MCU as shown in Table 9-12. If the Flash module is unsecured using backdoor key access, the SEC bits are forced to 10.

Table 9-11. Flash KEYEN States

KEYEN[1:0]	Status of Backdoor Key Access
00	DISABLED
01	DISABLED(1)
10	ENABLED
11	DISABLED

1. Preferred KEYEN state to disable backdoor key access.

Table 9-12. Flash Security States

SEC[1:0]	Status of Security
00	SECURED
01	SECURED(1)
10	UNSECURED
11	SECURED

1. Preferred SEC state to set MCU to secured state.

The security function in the Flash module is described in Section 9.5.

9.3.2.3 Flash CCOB Index Register (FCCOBIX)

The FCCOBIX register is used to index the FCCOB register for Flash memory operations.

Offset Module Base + 0x0002

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	CCOBIX[2:0]		
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 9-7. FCCOB Index Register (FCCOBIX)

CCOBIX bits are readable and writable while remaining bits read 0 and are not writable.

Table 9-13. FCCOBIX Field Descriptions

Field	Description
2–0 CCOBIX[1:0]	**Common Command Register Index**— The CCOBIX bits are used to select which word of the FCCOB register array is being read or written to. See Section 9.3.2.11, “Flash Common Command Object Register (FCCOB),” for more details.

9.3.2.4 Flash ECCR Index Register (FECCRIX)

The FECCRIX register is used to index the FECCR register for ECC fault reporting.

Offset Module Base + 0x0003

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	ECCRIX[2:0]		
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 9-8. FECCR Index Register (FECCRIX)

ECCRIX bits are readable and writable while remaining bits read 0 and are not writable.

Table 9-14. FECCRIX Field Descriptions

Field	Description
2-0 ECCRIX[2:0]	**ECC Error Register Index**— The ECCRIX bits are used to select which word of the FECCR register array is being read. See Section 9.3.2.13, “Flash ECC Error Results Register (FECCR),” for more details.

9.3.2.5 Flash Configuration Register (FCNFG)

The FCNFG register enables the Flash command complete interrupt and forces ECC faults on Flash array read access from the CPU or XGATE.

Offset Module Base + 0x0004

	7	6	5	4	3	2	1	0
R	CCIE	0	0	IGNSF	0	0	FDFD	FSFD
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 9-9. Flash Configuration Register (FCNFG)

CCIE, IGNSF, FDFD, and FSFD bits are readable and writable while remaining bits read 0 and are not writable.

Table 9-15. FCNFG Field Descriptions

Field	Description
7 CCIE	**Command Complete Interrupt Enable** — The CCIE bit controls interrupt generation when a Flash command has completed. 0 Command complete interrupt disabled 1 An interrupt will be requested whenever the CCIF flag in the FSTAT register is set (see Section 9.3.2.7)
4 IGNSF	**Ignore Single Bit Fault** — The IGNSF controls single bit fault reporting in the FERSTAT register (see Section 9.3.2.8). 0 All single bit faults detected during array reads are reported 1 Single bit faults detected during array reads are not reported and the single bit fault interrupt will not be generated

Table 9-15. FCNFG Field Descriptions (continued)

Field	Description
1 FDFD	**Force Double Bit Fault Detect** — The FDFD bit allows the user to simulate a double bit fault during Flash array read operations and check the associated interrupt routine. The FDFD bit is cleared by writing a 0 to FDFD. The FECCR registers will not be updated during the Flash array read operation with FDFD set unless an actual double bit fault is detected. 0 Flash array read operations will set the DFDIF flag in the FERSTAT register only if a double bit fault is detected 1 Any Flash array read operation will force the DFDIF flag in the FERSTAT register to be set (see Section 9.3.2.7) and an interrupt will be generated as long as the DFDIE interrupt enable in the FERCNFG register is set (see Section 9.3.2.6)
0 FSFD	**Force Single Bit Fault Detect** — The FSFD bit allows the user to simulate a single bit fault during Flash array read operations and check the associated interrupt routine. The FSFD bit is cleared by writing a 0 to FSFD. The FECCR registers will not be updated during the Flash array read operation with FSFD set unless an actual single bit fault is detected. 0 Flash array read operations will set the SFDIF flag in the FERSTAT register only if a single bit fault is detected 1 Flash array read operation will force the SFDIF flag in the FERSTAT register to be set (see Section 9.3.2.7) and an interrupt will be generated as long as the SFDIE interrupt enable in the FERCNFG register is set (see Section 9.3.2.6)

9.3.2.6 Flash Error Configuration Register (FERCNFG)

The FERCNFG register enables the Flash error interrupts for the FERSTAT flags.

Offset Module Base + 0x0005

	7	6	5	4	3	2	1	0
R	ERSERIE	PGMERIE	0	EPVIOLIE	ERSVIE1	ERSVIE0	DFDIE	SFDIE
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 9-10. Flash Error Configuration Register (FERCNFG)

All assigned bits in the FERCNFG register are readable and writable.

Table 9-16. FERCNFG Field Descriptions

Field	Description
7 ERSERIE	**EEE Erase Error Interrupt Enable** — The ERSERIE bit controls interrupt generation when a failure is detected during an EEE erase operation. 0 ERSERIF interrupt disabled 1 An interrupt will be requested whenever the ERSERIF flag is set (see Section 9.3.2.8)
6 PGMERIE	**EEE Program Error Interrupt Enable** — The PGMERIE bit controls interrupt generation when a failure is detected during an EEE program operation. 0 PGMERIF interrupt disabled 1 An interrupt will be requested whenever the PGMERIF flag is set (see Section 9.3.2.8)
4 EPVIOLIE	**EEE Protection Violation Interrupt Enable** — The EPVIOLIE bit controls interrupt generation when a protection violation is detected during a write to the buffer RAM EEE partition. 0 EPVIOLIF interrupt disabled 1 An interrupt will be requested whenever the EPVIOLIF flag is set (see Section 9.3.2.8)

Table 9-16. FERCNFG Field Descriptions (continued)

Field	Description
3 ERSVIE1	**EEE Error Type 1 Interrupt Enable** — The ERSVIE1 bit controls interrupt generation when a change state error is detected during an EEE operation. 0 ERSVIF1 interrupt disabled 1 An interrupt will be requested whenever the ERSVIF1 flag is set (see Section 9.3.2.8)
2 ERSVIE0	**EEE Error Type 0 Interrupt Enable** — The ERSVIE0 bit controls interrupt generation when a sector format error is detected during an EEE operation. 0 ERSVIF0 interrupt disabled 1 An interrupt will be requested whenever the ERSVIF0 flag is set (see Section 9.3.2.8)
1 DFDIE	**Double Bit Fault Detect Interrupt Enable** — The DFDIE bit controls interrupt generation when a double bit fault is detected during a Flash block read operation. 0 DFDIF interrupt disabled 1 An interrupt will be requested whenever the DFDIF flag is set (see Section 9.3.2.8)
0 SFDIE	**Single Bit Fault Detect Interrupt Enable** — The SFDIE bit controls interrupt generation when a single bit fault is detected during a Flash block read operation. 0 SFDIF interrupt disabled whenever the SFDIF flag is set (see Section 9.3.2.8) 1 An interrupt will be requested whenever the SFDIF flag is set (see Section 9.3.2.8)

9.3.2.7 Flash Status Register (FSTAT)

The FSTAT register reports the operational status of the Flash module.

Offset Module Base + 0x0006

	7	6	5	4	3	2	1	0
R	CCIF	0	ACCERR	FPVIOL	MGBUSY	RSVD	MGSTAT[1:0]	
W								
Reset	1	0	0	0	0	0	0(1)	01

= Unimplemented or Reserved

Figure 9-11. Flash Status Register (FSTAT)

1. Reset value can deviate from the value shown if a double bit fault is detected during the reset sequence (see Section 9.6).

CCIF, ACCERR, and FPVIOL bits are readable and writable, MGBUSY and MGSTAT bits are readable but not writable, while remaining bits read 0 and are not writable.

Table 9-17. FSTAT Field Descriptions

Field	Description
7 CCIF	**Command Complete Interrupt Flag** — The CCIF flag indicates that a Flash command has completed. The CCIF flag is cleared by writing a 1 to CCIF to launch a command and CCIF will stay low until command completion or command violation. 0 Flash command in progress 1 Flash command has completed
5 ACCERR	**Flash Access Error Flag** — The ACCERR bit indicates an illegal access has occurred to the Flash memory caused by either a violation of the command write sequence (see Section 9.4.1.2) or issuing an illegal Flash command or when errors are encountered while initializing the EEE buffer ram during the reset sequence. While ACCERR is set, the CCIF flag cannot be cleared to launch a command. The ACCERR bit is cleared by writing a 1 to ACCERR. Writing a 0 to the ACCERR bit has no effect on ACCERR. 0 No access error detected 1 Access error detected
4 FPVIOL	**Flash Protection Violation Flag** —The FPVIOL bit indicates an attempt was made to program or erase an address in a protected area of P-Flash memory during a command write sequence. The FPVIOL bit is cleared by writing a 1 to FPVIOL. Writing a 0 to the FPVIOL bit has no effect on FPVIOL. While FPVIOL is set, it is not possible to launch a command or start a command write sequence. 0 No protection violation detected 1 Protection violation detected
3 MGBUSY	**Memory Controller Busy Flag** — The MGBUSY flag reflects the active state of the Memory Controller. 0 Memory Controller is idle 1 Memory Controller is busy executing a Flash command (CCIF = 0) or is handling internal EEE operations
2 RSVD	**Reserved Bit** — This bit is reserved and always reads 0.
1–0 MGSTAT[1:0]	**Memory Controller Command Completion Status Flag** — One or more MGSTAT flag bits are set if an error is detected during execution of a Flash command or during the Flash reset sequence. See Section 9.4.2, “Flash Command Description,” and Section 9.6, “Initialization” for details.

9.3.2.8 Flash Error Status Register (FERSTAT)

The FERSTAT register reflects the error status of internal Flash operations.

Offset Module Base + 0x0007

	7	6	5	4	3	2	1	0
R	ERSERIF	PGMERIF	0	EPVIOLIF	ERSVIF1	ERSVIF0	DFDIF	SFDIF
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 9-12. Flash Error Status Register (FERSTAT)

All flags in the FERSTAT register are readable and only writable to clear the flag.

Table 9-18. FERSTAT Field Descriptions

Field	Description
7 ERSERIF	**EEE Erase Error Interrupt Flag** — The setting of the ERSERIF flag occurs due to an error in a Flash erase command that resulted in the erase operation not being successful during EEE operations. The ERSERIF flag is cleared by writing a 1 to ERSERIF. Writing a 0 to the ERSERIF flag has no effect on ERSERIF. While ERSERIF is set, it is possible to write to the buffer RAM EEE partition but the data written will not be transferred to the D-Flash EEE partition. 0 Erase command successfully completed on the D-Flash EEE partition 1 Erase command failed on the D-Flash EEE partition
6 PGMERIF	**EEE Program Error Interrupt Flag** — The setting of the PGMERIF flag occurs due to an error in a Flash program command that resulted in the program operation not being successful during EEE operations. The PGMERIF flag is cleared by writing a 1 to PGMERIF. Writing a 0 to the PGMERIF flag has no effect on PGMERIF. While PGMERIF is set, it is possible to write to the buffer RAM EEE partition but the data written will not be transferred to the D-Flash EEE partition. 0 Program command successfully completed on the D-Flash EEE partition 1 Program command failed on the D-Flash EEE partition
4 EPVIOLIF	**EEE Protection Violation Interrupt Flag** —The setting of the EPVIOLIF flag indicates an attempt was made to write to a protected area of the buffer RAM EEE partition. The EPVIOLIF flag is cleared by writing a 1 to EPVIOLIF. Writing a 0 to the EPVIOLIF flag has no effect on EPVIOLIF. While EPVIOLIF is set, it is possible to write to the buffer RAM EEE partition as long as the address written to is not in a protected area. 0 No EEE protection violation 1 EEE protection violation detected
3 ERSVIF1	**EEE Error Interrupt 1 Flag** —The setting of the ERSVIF1 flag indicates that the memory controller was unable to change the state of a D-Flash EEE sector. The ERSVIF1 flag is cleared by writing a 1 to ERSVIF1. Writing a 0 to the ERSVIF1 flag has no effect on ERSVIF1. While ERSVIF1 is set, it is possible to write to the buffer RAM EEE partition but the data written will not be transferred to the D-Flash EEE partition. 0 No EEE sector state change error detected 1 EEE sector state change error detected
2 ERSVIF0	**EEE Error Interrupt 0 Flag** —The setting of the ERSVIF0 flag indicates that the memory controller was unable to format a D-Flash EEE sector for EEE use. The ERSVIF0 flag is cleared by writing a 1 to ERSVIF0. Writing a 0 to the ERSVIF0 flag has no effect on ERSVIF0. While ERSVIF0 is set, it is possible to write to the buffer RAM EEE partition but the data written will not be transferred to the D-Flash EEE partition. 0 No EEE sector format error detected 1 EEE sector format error detected
1 DFDIF	**Double Bit Fault Detect Interrupt Flag** — The setting of the DFDIF flag indicates that a double bit fault was detected in the stored parity and data bits during a Flash array read operation or that a Flash array read operation was attempted on a Flash block that was under a Flash command operation. The DFDIF flag is cleared by writing a 1 to DFDIF. Writing a 0 to DFDIF has no effect on DFDIF. 0 No double bit fault detected 1 Double bit fault detected or an invalid Flash array read operation attempted
0 SFDIF	**Single Bit Fault Detect Interrupt Flag** — With the IGNSF bit in the FCNFG register clear, the SFDIF flag indicates that a single bit fault was detected in the stored parity and data bits during a Flash array read operation or that a Flash array read operation was attempted on a Flash block that was under a Flash command operation. The SFDIF flag is cleared by writing a 1 to SFDIF. Writing a 0 to SFDIF has no effect on SFDIF. 0 No single bit fault detected 1 Single bit fault detected and corrected or an invalid Flash array read operation attempted

9.3.2.9 P-Flash Protection Register (FPROT)

The FPROT register defines which P-Flash sectors are protected against program and erase operations.

Offset Module Base + 0x0008

	7	6	5	4	3	2	1	0
R	FPOPEN	RNV6	FPHDIS	FPHS[1:0]		FPLDIS	FPLS[1:0]	
W								
Reset	F	F	F	F	F	F	F	F

= Unimplemented or Reserved

Figure 9-13. Flash Protection Register (FPROT)

The (unreserved) bits of the FPROT register are writable with the restriction that the size of the protected region can only be increased (see Section 9.3.2.9.1, “P-Flash Protection Restrictions,” and Table 9-23).

During the reset sequence, the FPROT register is loaded with the contents of the P-Flash protection byte in the Flash configuration field at global address 0x7F_FF0C located in P-Flash memory (see Table 9-3) as indicated by reset condition ‘F’ in Figure 9-13. To change the P-Flash protection that will be loaded during the reset sequence, the upper sector of the P-Flash memory must be unprotected, then the P-Flash protection byte must be reprogrammed. If a double bit fault is detected while reading the P-Flash phrase containing the P-Flash protection byte during the reset sequence, the FPOPEN bit will be cleared and remaining bits in the FPROT register will be set to leave the P-Flash memory fully protected.

Trying to alter data in any protected area in the P-Flash memory will result in a protection violation error and the FPVIOL bit will be set in the FSTAT register. The block erase of a P-Flash block is not possible if any of the P-Flash sectors contained in the same P-Flash block are protected.

Table 9-19. FPROT Field Descriptions

Field	Description
7 FPOPEN	**Flash Protection Operation Enable** — The FPOPEN bit determines the protection function for program or erase operations as shown in Table 9-20 for the P-Flash block. 0 When FPOPEN is clear, the FPHDIS and FPLDIS bits define unprotected address ranges as specified by the corresponding FPHS and FPLS bits 1 When FPOPEN is set, the FPHDIS and FPLDIS bits enable protection for the address range specified by the corresponding FPHS and FPLS bits
6 RNV[6]	**Reserved Nonvolatile Bit** — The RNV bit should remain in the erased state for future enhancements.
5 FPHDIS	**Flash Protection Higher Address Range Disable** — The FPHDIS bit determines whether there is a protected/unprotected area in a specific region of the P-Flash memory ending with global address 0x7F_FFFF. 0 Protection/Unprotection enabled 1 Protection/Unprotection disabled
4–3 FPHS[1:0]	**Flash Protection Higher Address Size** — The FPHS bits determine the size of the protected/unprotected area in P-Flash memory as shown inTable 9-21. The FPHS bits can only be written to while the FPHDIS bit is set.
2 FPLDIS	**Flash Protection Lower Address Range Disable** — The FPLDIS bit determines whether there is a protected/unprotected area in a specific region of the P-Flash memory beginning with global address 0x7F_8000. 0 Protection/Unprotection enabled 1 Protection/Unprotection disabled
1–0 FPLS[1:0]	**Flash Protection Lower Address Size** — The FPLS bits determine the size of the protected/unprotected area in P-Flash memory as shown in Table 9-22. The FPLS bits can only be written to while the FPLDIS bit is set.

Table 9-20. P-Flash Protection Function

FPOPEN	FPHDIS	FPLDIS	Function[1]
1	1	1	No P-Flash Protection
1	1	0	Protected Low Range
1	0	1	Protected High Range
1	0	0	Protected High and Low Ranges
0	1	1	Full P-Flash Memory Protected
0	1	0	Unprotected Low Range
0	0	1	Unprotected High Range
0	0	0	Unprotected High and Low Ranges

1. For range sizes, refer to Table 9-21 and Table 9-22.

Table 9-21. P-Flash Protection Higher Address Range

FPHS[1:0]	Global Address Range	Protected Size
00	0x7F_F800–0x7F_FFFF	2 Kbytes
01	0x7F_F000–0x7F_FFFF	4 Kbytes
10	0x7F_E000–0x7F_FFFF	8 Kbytes
11	0x7F_C000–0x7F_FFFF	16 Kbytes

Table 9-22. P-Flash Protection Lower Address Range

FPLS[1:0]	Global Address Range	Protected Size
00	0x7F_8000–0x7F_83FF	1 Kbyte
01	0x7F_8000–0x7F_87FF	2 Kbytes
10	0x7F_8000–0x7F_8FFF	4 Kbytes
11	0x7F_8000–0x7F_9FFF	8 Kbytes

All possible P-Flash protection scenarios are shown in Figure 9-14. Although the protection scheme is loaded from the Flash memory at global address 0x7F_FF0C during the reset sequence, it can be changed by the user. The P-Flash protection scheme can be used by applications requiring reprogramming in single chip mode while providing as much protection as possible if reprogramming is not required.

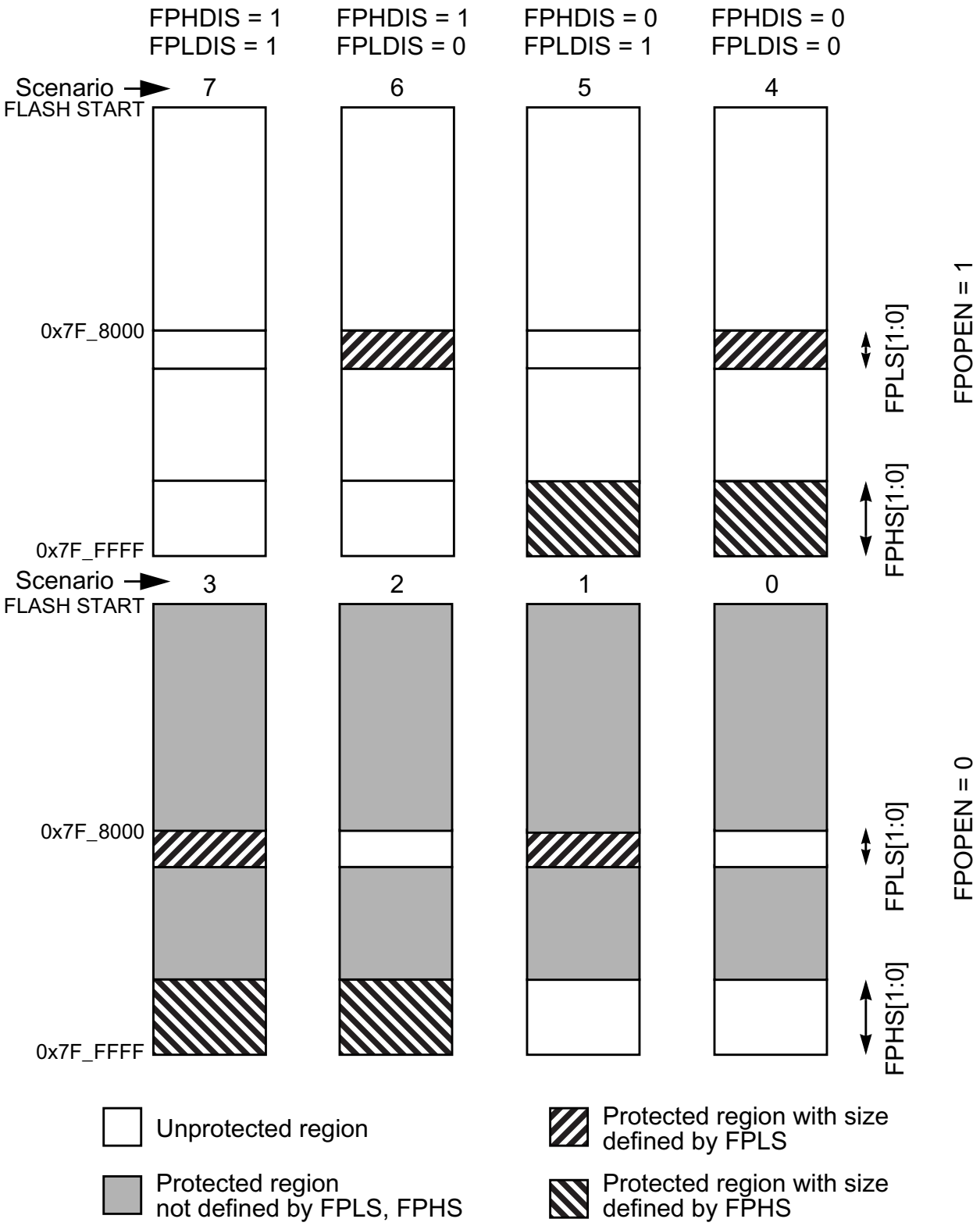


Figure 9-14. P-Flash Protection Scenarios

9.3.2.9.1 P-Flash Protection Restrictions

The general guideline is that P-Flash protection can only be added and not removed. Table 9-23 specifies all valid transitions between P-Flash protection scenarios. Any attempt to write an invalid scenario to the FPROT register will be ignored. The contents of the FPROT register reflect the active protection scenario. See the FPHS and FPLS bit descriptions for additional restrictions.

Table 9-23. P-Flash Protection Scenario Transitions

From Protection Scenario	To Protection Scenario[1]							
	0	1	2	3	4	5	6	7
0	X	X	X	X				
1		X		X				
2			X	X				
3				X				
4				X	X			
5			X	X	X	X		
6		X		X	X		X	
7	X	X	X	X	X	X	X	X

1. Allowed transitions marked with X, see Figure 9-14 for a definition of the scenarios.

9.3.2.10 EEE Protection Register (EPROT)

The EPROT register defines which buffer RAM EEE partition areas are protected against writes.

Offset Module Base + 0x0009

	7	6	5	4	3	2	1	0
R	EPOPEN	RNV[6:4]			EPDIS	EPS[2:0]		
W								
Reset	F	F	F	F	F	F	F	F

= Unimplemented or Reserved

Figure 9-15. EEE Protection Register (EPROT)

All bits in the EPROT register are readable and writable except for RNV[6:4] which are only readable. The EPOPEN and EPDIS bits can only be written to the protected state. The EPS bits can be written anytime until the EPDIS bit is cleared. If the EPOPEN bit is cleared, the state of the EPDIS and EPS bits is irrelevant.

During the reset sequence, the EPROT register is loaded from the EEE protection byte in the Flash configuration field at global address 0x7F_FF0D located in P-Flash memory (see Table 9-3) as indicated by reset condition F in Figure 9-15. To change the EEE protection that will be loaded during the reset sequence, the P-Flash sector containing the EEE protection byte must be unprotected, then the EEE protection byte must be programmed. If a double bit fault is detected while reading the P-Flash phrase

containing the EEE protection byte during the reset sequence, the EPOPEN bit will be cleared and remaining bits in the EPROT register will be set to leave the buffer RAM EEE partition fully protected.

Trying to write data to any protected area in the buffer RAM EEE partition will result in a protection violation error and the EPVIOLIF flag will be set in the FERSTAT register. Trying to write data to any protected area in the buffer RAM partitioned for user access will not be prevented and the EPVIOLIF flag in the FERSTAT register will not set.

Table 9-24. EPROT Field Descriptions

Field	Description
7 EPOPEN	**Enables writes to the Buffer RAM partitioned for EEE** 0 The entire buffer RAM EEE partition is protected from writes 1 Unprotected buffer RAM EEE partition areas are enabled for writes
6–4 RNV[6:4]	**Reserved Nonvolatile Bits** — The RNV bits should remain in the erased state for future enhancements
3 EPDIS	**Buffer RAM Protection Address Range Disable** — The EPDIS bit determines whether there is a protected area in a specific region of the buffer RAM EEE partition. 0 Protection enabled 1 Protection disabled
2–0 EPS[2:0]	**Buffer RAM Protection Size** — The EPS[2:0] bits determine the size of the protected area in the buffer RAM EEE partition as shown inTable 9-21. The EPS bits can only be written to while the EPDIS bit is set.

Table 9-25. Buffer RAM EEE Partition Protection Address Range

EPS[2:0]	Global Address Range	Protected Size
000	0x13_FFC0 – 0x13_FFFF	64 bytes
001	0x13_FF80 – 0x13_FFFF	128 bytes
010	0x13_FF40 – 0x13_FFFF	192 bytes
011	0x13_FF00 – 0x13_FFFF	256 bytes
100	0x13_FEC0 – 0x13_FFFF	320 bytes
101	0x13_FE80 – 0x13_FFFF	384 bytes
110	0x13_FE40 – 0x13_FFFF	448 bytes
111	0x13_FE00 – 0x13_FFFF	512 bytes

9.3.2.11 Flash Common Command Object Register (FCCOB)

The FCCOB is an array of six words addressed via the CCOBIX index found in the FCCOBIX register. Byte wide reads and writes are allowed to the FCCOB register.

Offset Module Base + 0x000A

	7	6	5	4	3	2	1	0
R W	CCOB[15:8]							
Reset	0	0	0	0	0	0	0	0

Figure 9-16. Flash Common Command Object High Register (FCCOBHI)

Offset Module Base + 0x000B

	7	6	5	4	3	2	1	0
R W	CCOB[7:0]							
Reset	0	0	0	0	0	0	0	0

Figure 9-17. Flash Common Command Object Low Register (FCCOBLO)

9.3.2.11.1 FCCOB - NVM Command Mode

NVM command mode uses the indexed FCCOB register to provide a command code and its relevant parameters to the Memory Controller. The user first sets up all required FCCOB fields and then initiates the command's execution by writing a 1 to the CCIF bit in the FSTAT register (a 1 written by the user clears the CCIF command completion flag to 0). When the user clears the CCIF bit in the FSTAT register all FCCOB parameter fields are locked and cannot be changed by the user until the command completes (as evidenced by the Memory Controller returning CCIF to 1). Some commands return information to the FCCOB register array.

The generic format for the FCCOB parameter fields in NVM command mode is shown in Table 9-26. The return values are available for reading after the CCIF flag in the FSTAT register has been returned to 1 by the Memory Controller. Writes to the unimplemented parameter fields (CCOBIX = 110 and CCOBIX = 111) are ignored with reads from these fields returning 0x0000.

Table 9-26 shows the generic Flash command format. The high byte of the first word in the CCOB array contains the command code, followed by the parameters for this specific Flash command. For details on the FCCOB settings required by each command, see the Flash command descriptions in Section 9.4.2.

Table 9-26. FCCOB - NVM Command Mode (Typical Usage)

CCOBIX[2:0]	Byte	FCCOB Parameter Fields (NVM Command Mode)
000	HI	FCMD[7:0] defining Flash command
	LO	0, Global address [22:16]
001	HI	Global address [15:8]
	LO	Global address [7:0]
010	HI	Data 0 [15:8]
	LO	Data 0 [7:0]

Table 9-26. FCCOB - NVM Command Mode (Typical Usage)

CCOBIX[2:0]	Byte	FCCOB Parameter Fields (NVM Command Mode)
011	HI	Data 1 [15:8]
	LO	Data 1 [7:0]
100	HI	Data 2 [15:8]
	LO	Data 2 [7:0]
101	HI	Data 3 [15:8]
	LO	Data 3 [7:0]

9.3.2.12 EEE Tag Counter Register (ETAG)

The ETAG register contains the number of outstanding words in the buffer RAM EEE partition that need to be programmed into the D-Flash EEE partition. The ETAG register is decremented prior to the related tagged word being programmed into the D-Flash EEE partition. All tagged words have been programmed into the D-Flash EEE partition once all bits in the ETAG register read 0 and the MGBUSY flag in the FSTAT register reads 0.

Offset Module Base + 0x000C

	7	6	5	4	3	2	1	0
R	ETAG[15:8]							
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 9-18. EEE Tag Counter High Register (ETAGHI)

Offset Module Base + 0x000D

	7	6	5	4	3	2	1	0
R	ETAG[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 9-19. EEE Tag Counter Low Register (ETAGLO)

All ETAG bits are readable but not writable and are cleared by the Memory Controller.

9.3.2.13 Flash ECC Error Results Register (FECCR)

The FECCR registers contain the result of a detected ECC fault for both single bit and double bit faults. The FECCR register provides access to several ECC related fields as defined by the ECCRIX index bits in the FECCRIX register (see Section 9.3.2.4). Once ECC fault information has been stored, no other fault

information will be recorded until the specific ECC fault flag has been cleared. In the event of simultaneous ECC faults, the priority for fault recording is:

1. Double bit fault over single bit fault
2. CPU over XGATE

Offset Module Base + 0x000E

	7	6	5	4	3	2	1	0
R	ECCR[15:8]							
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 9-20. Flash ECC Error Results High Register (FECCRHI)

Offset Module Base + 0x000F

	7	6	5	4	3	2	1	0
R	ECCR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 9-21. Flash ECC Error Results Low Register (FECCRLO)

All FECCR bits are readable but not writable.

Table 9-27. FECCR Index Settings

ECCRIX[2:0]	FECCR Register Content		
	Bits [15:8]	**Bit[7]**	**Bits[6:0]**
000	Parity bits read from Flash block	CPU or XGATE source identity	Global address [22:16]
001	Global address [15:0]		
010	Data 0 [15:0]		
011	Data 1 [15:0] (P-Flash only)		
100	Data 2 [15:0] (P-Flash only)		
101	Data 3 [15:0] (P-Flash only)		
110	Not used, returns 0x0000 when read		
111	Not used, returns 0x0000 when read		

Table 9-28. FECCR Index=000 Bit Descriptions

Field	Description
15:8 PAR[7:0]	**ECC Parity Bits** — Contains the 8 parity bits from the 72 bit wide P-Flash data word or the 6 parity bits, allocated to PAR[5:0], from the 22 bit wide D-Flash word with PAR[7:6]=00.
7 XBUS01	**Bus Source Identifier** — The XBUS01 bit determines whether the ECC error was caused by a read access from the CPU or XGATE. 0 ECC Error happened on the CPU access 1 ECC Error happened on the XGATE access
6–0 GADDR[22:16]	**Global Address** — The GADDR[22:16] field contains the upper seven bits of the global address having caused the error.

The P-Flash word addressed by ECCRIX = 001 contains the lower 16 bits of the global address. The following four words addressed by ECCRIX = 010 to 101 contain the 64-bit wide data phrase. The four data words and the parity byte are the uncorrected data read from the P-Flash block.

The D-Flash word addressed by ECCRIX = 001 contains the lower 16 bits of the global address. The uncorrected 16-bit data word is addressed by ECCRIX = 010.

9.3.2.14 Flash Option Register (FOPT)

The FOPT register is the Flash option register.

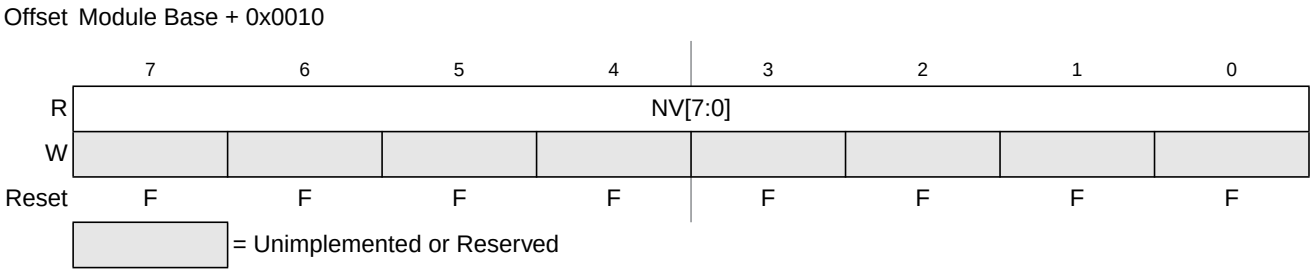


Figure 9-22. Flash Option Register (FOPT)

All bits in the FOPT register are readable but are not writable.

During the reset sequence, the FOPT register is loaded from the Flash nonvolatile byte in the Flash configuration field at global address 0x7F_FF0E located in P-Flash memory (see Table 9-3) as indicated by reset condition F in Figure 9-22. If a double bit fault is detected while reading the P-Flash phrase containing the Flash nonvolatile byte during the reset sequence, all bits in the FOPT register will be set.

Table 9-29. FOPT Field Descriptions

Field	Description
7–0 NV[7:0]	**Nonvolatile Bits** — The NV[7:0] bits are available as nonvolatile bits. Refer to the device user guide for proper use of the NV bits.

9.3.2.15 Flash Reserved0 Register (FRSV0)

This Flash register is reserved for factory testing.

Offset Module Base + 0x0011

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 9-23. Flash Reserved0 Register (FRSV0)

All bits in the FRSV0 register read 0 and are not writable.

9.3.2.16 Flash Reserved1 Register (FRSV1)

This Flash register is reserved for factory testing.

Offset Module Base + 0x0012

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 9-24. Flash Reserved1 Register (FRSV1)

All bits in the FRSV1 register read 0 and are not writable.

9.3.2.17 Flash Reserved2 Register (FRSV2)

This Flash register is reserved for factory testing.

Offset Module Base + 0x0013

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 9-25. Flash Reserved2 Register (FRSV2)

All bits in the FRSV2 register read 0 and are not writable.

9.4 Functional Description

9.4.1 Flash Command Operations

Flash command operations are used to modify Flash memory contents or configure module resources for EEE operation.

The next sections describe:

- How to write the FCLKDIV register that is used to generate a time base (FCLK) derived from OSCCLK for Flash program and erase command operations
- The command write sequence used to set Flash command parameters and launch execution
- Valid Flash commands available for execution

9.4.1.1 Writing the FCLKDIV Register

Prior to issuing any Flash program or erase command after a reset, the user is required to write the FCLKDIV register to divide OSCCLK down to a target FCLK of 1 MHz. Table 9-9 shows recommended values for the FDIV field based on OSCCLK frequency.

NOTE

Programming or erasing the Flash memory cannot be performed if the bus clock runs at less than 1 MHz. Setting FDIV too high can destroy the Flash memory due to overstress. Setting FDIV too low can result in incomplete programming or erasure of the Flash memory cells.

When the FCLKDIV register is written, the FDIVLD bit is set automatically. If the FDIVLD bit is 0, the FCLKDIV register has not been written since the last reset. If the FCLKDIV register has not been written, any Flash program or erase command loaded during a command write sequence will not execute and the ACCERR bit in the FSTAT register will set.

9.4.1.2 Command Write Sequence

The Memory Controller will launch all valid Flash commands entered using a command write sequence.

Before launching a command, the ACCERR and FPVIOL bits in the FSTAT register must be clear (see Section 9.3.2.7) and the CCIF flag should be tested to determine the status of the current command write sequence. If CCIF is 0, the previous command write sequence is still active, a new command write sequence cannot be started, and all writes to the FCCOB register are ignored.

CAUTION

Writes to any Flash register must be avoided while a Flash command is active (CCIF=0) to prevent corruption of Flash register contents and Memory Controller behavior.

9.4.1.2.1 Define FCCOB Contents

The FCCOB parameter fields must be loaded with all required parameters for the Flash command being executed. Access to the FCCOB parameter fields is controlled via the CCOBIX bits in the FCCOBIX register (see Section 9.3.2.3).

The contents of the FCCOB parameter fields are transferred to the Memory Controller when the user clears the CCIF command completion flag in the FSTAT register (writing 1 clears the CCIF to 0). The CCIF flag will remain clear until the Flash command has completed. Upon completion, the Memory Controller will return CCIF to 1 and the FCCOB register will be used to communicate any results. The flow for a generic command write sequence is shown in Figure 9-26.

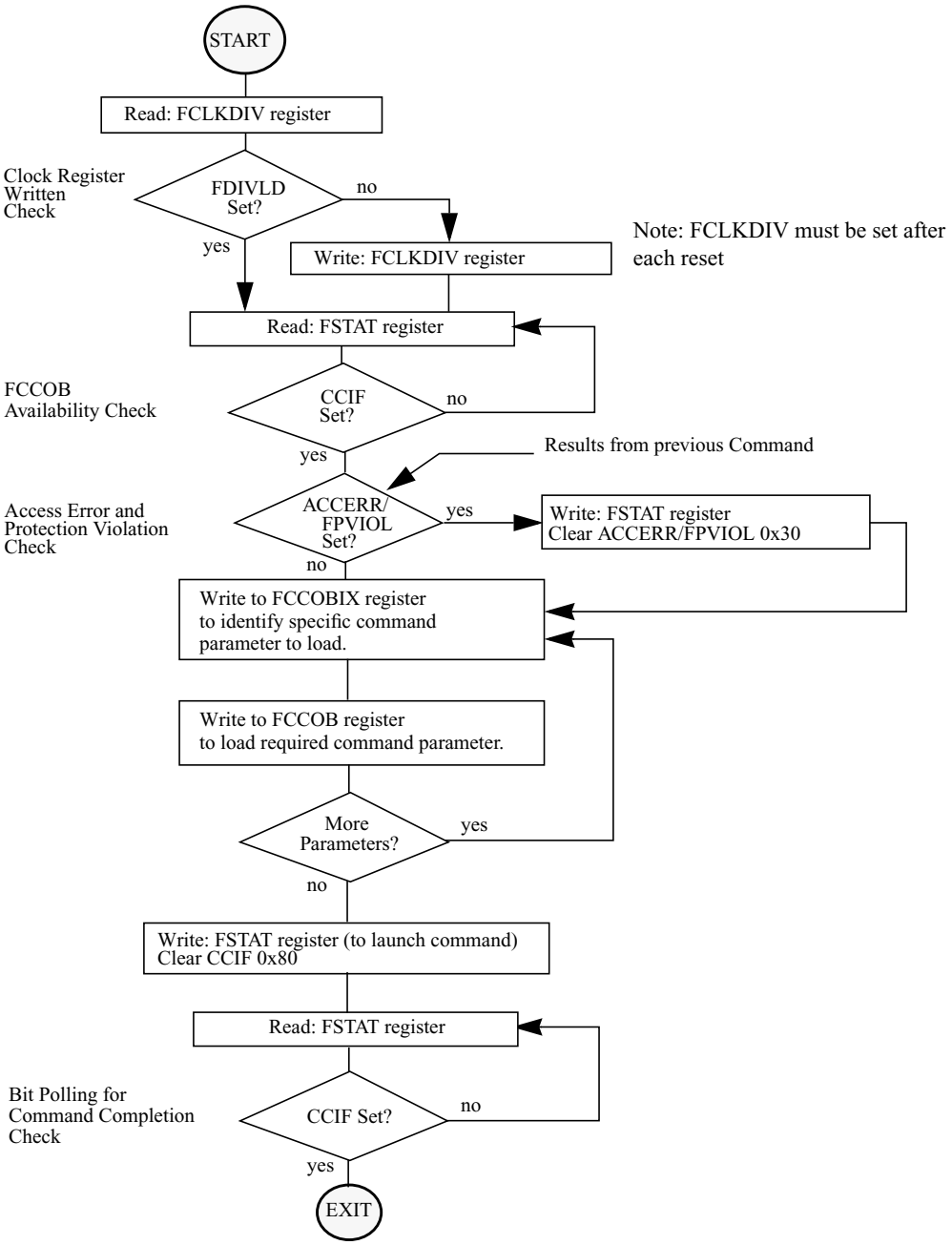


Figure 9-26. Generic Flash Command Write Sequence Flowchart

9.4.1.3 Valid Flash Module Commands

Table 9-30. Flash Commands by Mode

FCMD	Command	Unsecured				Secured			
		NS (1)	NX (2)	SS(3)	ST(4)	NS (5)	NX (6)	SS(7)	ST(8)
0x01	Erase Verify All Blocks	*	*	*	*	*	*	*	*
0x02	Erase Verify Block	*	*	*	*	*	*	*	*
0x03	Erase Verify P-Flash Section	*	*	*	*	*			
0x04	Read Once	*	*	*	*	*			
0x05	Load Data Field	*	*	*	*	*			
0x06	Program P-Flash	*	*	*	*	*			
0x07	Program Once	*	*	*	*	*			
0x08	Erase All Blocks			*	*			*	*
0x09	Erase P-Flash Block	*	*	*	*	*			
0x0A	Erase P-Flash Sector	*	*	*	*	*			
0x0B	Unsecure Flash			*	*			*	*
0x0C	Verify Backdoor Access Key	*				*			
0x0D	Set User Margin Level	*	*	*	*	*			
0x0E	Set Field Margin Level			*	*				
0x0F	Full Partition D-Flash			*	*				
0x10	Erase Verify D-Flash Section	*	*	*	*	*			
0x11	Program D-Flash	*	*	*	*	*			
0x12	Erase D-Flash Sector	*	*	*	*	*			
0x13	Enable EEPROM Emulation	*	*	*	*	*	*	*	*
0x14	Disable EEPROM Emulation	*	*	*	*	*	*	*	*
0x15	EEPROM Emulation Query	*	*	*	*	*	*	*	*
0x20	Partition D-Flash	*	*	*	*	*	*	*	*

1. Unsecured Normal Single Chip mode.
2. Unsecured Normal Expanded mode.
3. Unsecured Special Single Chip mode.
4. Unsecured Special Mode.
5. Secured Normal Single Chip mode.
6. Secured Normal Expanded mode.
7. Secured Special Single Chip mode.
8. Secured Special Mode.

9.4.1.4 P-Flash Commands

Table 9-31 summarizes the valid P-Flash commands along with the effects of the commands on the P-Flash block and other resources within the Flash module.

Table 9-31. P-Flash Commands

FCMD	Command	Function on P-Flash Memory
0x01	Erase Verify All Blocks	Verify that all P-Flash (and D-Flash) blocks are erased.
0x02	Erase Verify Block	Verify that a P-Flash block is erased.
0x03	Erase Verify P-Flash Section	Verify that a given number of words starting at the address provided are erased.
0x04	Read Once	Read a dedicated 64 byte field in the nonvolatile information register in P-Flash block 0 that was previously programmed using the Program Once command.
0x05	Load Data Field	Load data for simultaneous multiple P-Flash block operations.
0x06	Program P-Flash	Program a phrase in a P-Flash block and any previously loaded phrases for any other P-Flash block (see Load Data Field command).
0x07	Program Once	Program a dedicated 64 byte field in the nonvolatile information register in P-Flash block 0 that is allowed to be programmed only once.
0x08	Erase All Blocks	Erase all P-Flash (and D-Flash) blocks. An erase of all Flash blocks is only possible when the FPLDIS, FPHDIS, and FPOPEN bits in the FPROT register and the EPDIS and EPOPEN bits in the EPROT register are set prior to launching the command.
0x09	Erase P-Flash Block	Erase a single P-Flash block. An erase of the full P-Flash block is only possible when FPLDIS, FPHDIS and FPOPEN bits in the FPROT register are set prior to launching the command.
0x0A	Erase P-Flash Sector	Erase all bytes in a P-Flash sector.
0x0B	Unsecure Flash	Supports a method of releasing MCU security by erasing all P-Flash (and D-Flash) blocks and verifying that all P-Flash (and D-Flash) blocks are erased.
0x0C	Verify Backdoor Access Key	Supports a method of releasing MCU security by verifying a set of security keys.
0x0D	Set User Margin Level	Specifies a user margin read level for all P-Flash blocks.
0x0E	Set Field Margin Level	Specifies a field margin read level for all P-Flash blocks (special modes only).

9.4.1.5 D-Flash and EEE Commands

Table 9-32 summarizes the valid D-Flash and EEE commands along with the effects of the commands on the D-Flash block and EEE operation.

Table 9-32. D-Flash Commands

FCMD	Command	Function on D-Flash Memory
0x01	Erase Verify All Blocks	Verify that all D-Flash (and P-Flash) blocks are erased.
0x02	Erase Verify Block	Verify that the D-Flash block is erased.

Table 9-32. D-Flash Commands

FCMD	Command	Function on D-Flash Memory
0x08	Erase All Blocks	Erase all D-Flash (and P-Flash) blocks. An erase of all Flash blocks is only possible when the FPLDIS, FPHDIS, and FPOPEN bits in the FPROT register and the EPDIS and EPOPEN bits in the EPROT register are set prior to launching the command.
0x0B	Unsecure Flash	Supports a method of releasing MCU security by erasing all D-Flash (and P-Flash) blocks and verifying that all D-Flash (and P-Flash) blocks are erased.
0x0D	Set User Margin Level	Specifies a user margin read level for the D-Flash block.
0x0E	Set Field Margin Level	Specifies a field margin read level for the D-Flash block (special modes only).
0x0F	Full Partition D-Flash	Erase the D-Flash block and partition an area of the D-Flash block for user access.
0x10	Erase Verify D-Flash Section	Verify that a given number of words starting at the address provided are erased.
0x11	Program D-Flash	Program up to four words in the D-Flash block.
0x12	Erase D-Flash Sector	Erase all bytes in a sector of the D-Flash block.
0x13	Enable EEPROM Emulation	Enable EEPROM emulation where writes to the buffer RAM EEE partition will be copied to the D-Flash EEE partition.
0x14	Disable EEPROM Emulation	Suspend all current erase and program activity related to EEPROM emulation but leave current EEE tags set.
0x15	EEPROM Emulation Query	Returns EEE partition and status variables.
0x20	Partition D-Flash	Partition an area of the D-Flash block for user access.

9.4.2 Flash Command Description

This section provides details of all available Flash commands launched by a command write sequence. The ACCERR bit in the FSTAT register will be set during the command write sequence if any of the following illegal steps are performed, causing the command not to be processed by the Memory Controller:

- Starting any command write sequence that programs or erases Flash memory before initializing the FCLKDIV register
- Writing an invalid command as part of the command write sequence
- For additional possible errors, refer to the error handling table provided for each command

If a Flash block is read during execution of an algorithm (CCIF = 0) on that same block, the read operation will return invalid data. If the SFDIF or DFDIF flags were not previously set when the invalid read operation occurred, both the SFDIF and DFDIF flags will be set and the FECCR registers will be loaded with the global address used in the invalid read operation with the data and parity fields set to all 0.

If the ACCERR or FPVIOL bits are set in the FSTAT register, the user must clear these bits before starting any command write sequence (see Section 9.3.2.7).

CAUTION

A Flash word or phrase must be in the erased state before being programmed. Cumulative programming of bits within a Flash word or phrase is not allowed.

9.4.2.1 Erase Verify All Blocks Command

The Erase Verify All Blocks command will verify that all P-Flash and D-Flash blocks have been erased.

Table 9-33. Erase Verify All Blocks Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x01	Not required

Upon clearing CCIF to launch the Erase Verify All Blocks command, the Memory Controller will verify that the entire Flash memory space is erased. The CCIF flag will set after the Erase Verify All Blocks operation has completed.

Table 9-34. Erase Verify All Blocks Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if a Load Data Field command sequence is currently active
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read
	MGSTAT0	Set if any non-correctable errors have been encountered during the read
FERSTAT	EPVIOLIF	None

9.4.2.2 Erase Verify Block Command

The Erase Verify Block command allows the user to verify that an entire P-Flash or D-Flash block has been erased. The FCCOB upper global address bits determine which block must be verified.

Table 9-35. Erase Verify Block Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x02	Global address [22:16] of the Flash block to be verified.

Upon clearing CCIF to launch the Erase Verify Block command, the Memory Controller will verify that the selected P-Flash or D-Flash block is erased. The CCIF flag will set after the Erase Verify Block operation has completed.

Table 9-36. Erase Verify Block Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if an invalid global address [22:16] is supplied
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read
	MGSTAT0	Set if any non-correctable errors have been encountered during the read
FERSTAT	EPVIOLIF	None

9.4.2.3 Erase Verify P-Flash Section Command

The Erase Verify P-Flash Section command will verify that a section of code in the P-Flash memory is erased. The Erase Verify P-Flash Section command defines the starting point of the code to be verified and the number of phrases. The section to be verified cannot cross a 256 Kbyte boundary in the P-Flash memory space.

Table 9-37. Erase Verify P-Flash Section Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x03	Global address [22:16] of a P-Flash block
001	Global address [15:0] of the first phrase to be verified	
010	Number of phrases to be verified	

Upon clearing CCIF to launch the Erase Verify P-Flash Section command, the Memory Controller will verify the selected section of Flash memory is erased. The CCIF flag will set after the Erase Verify P-Flash Section operation has completed.

Table 9-38. Erase Verify P-Flash Section Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 010 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 9-30)
		Set if an invalid global address [22:0] is supplied
		Set if a misaligned phrase address is supplied (global address [2:0] != 000)
		Set if the requested section crosses a 256 Kbyte boundary
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read
	MGSTAT0	Set if any non-correctable errors have been encountered during the read

Table 9-38. Erase Verify P-Flash Section Command Error Handling

Register	Error Bit	Error Condition
FERSTAT	EPVIOLIF	None

9.4.2.4 Read Once Command

The Read Once command provides read access to a reserved 64 byte field (8 phrases) located in the nonvolatile information register of P-Flash block 0. The Read Once field is programmed using the Program Once command described in Section 9.4.2.7. The Read Once command must not be executed from the Flash block containing the Program Once reserved field to avoid code runaway.

Table 9-39. Read Once Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x04	Not Required
001	Read Once phrase index (0x0000 - 0x0007)	
010	Read Once word 0 value	
011	Read Once word 1 value	
100	Read Once word 2 value	
101	Read Once word 3 value	

Upon clearing CCIF to launch the Read Once command, a Read Once phrase is fetched and stored in the FCCOB indexed register. The CCIF flag will set after the Read Once operation has completed. Valid phrase index values for the Read Once command range from 0x0000 to 0x0007. During execution of the Read Once command, any attempt to read addresses within P-Flash block 0 will return invalid data.

Table 9-40. Read Once Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 9-30)
		Set if an invalid phrase index is supplied
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read
	MGSTAT0	Set if any non-correctable errors have been encountered during the read
FERSTAT	EPVIOLIF	None

9.4.2.5 Load Data Field Command

The Load Data Field command is executed to provide FCCOB parameters for multiple P-Flash blocks for a future simultaneous program operation in the P-Flash memory space.

Table 9-41. Load Data Field Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x05	Global address [22:16] to identify P-Flash block
001	Global address [15:0] of phrase location to be programmed(1)	
010	Word 0	
011	Word 1	
100	Word 2	
101	Word 3	

1. Global address [2:0] must be 000

Upon clearing CCIF to launch the Load Data Field command, the FCCOB registers will be transferred to the Memory Controller and be programmed in the block specified at the global address given with a future Program P-Flash command launched on a P-Flash block. The CCIF flag will set after the Load Data Field operation has completed. Note that once a Load Data Field command sequence has been initiated, the Load Data Field command sequence will be cancelled if any command other than Load Data Field or the future Program P-Flash is launched. Similarly, if an error occurs after launching a Load Data Field or Program P-Flash command, the associated Load Data Field command sequence will be cancelled.

Table 9-42. Load Data Field Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 101 at command launch
		Set if command not available in current mode (see Table 9-30)
		Set if an invalid global address [22:0] is supplied
		Set if a misaligned phrase address is supplied (global address [2:0] != 000)
		Set if a Load Data Field command sequence is currently active and the selected block has previously been selected in the same command sequence
		Set if a Load Data Field command sequence is currently active and global address [17:0] does not match that previously supplied in the same command sequence
	FPVIOL	Set if the global address [22:0] points to a protected area
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

9.4.2.6 Program P-Flash Command

The Program P-Flash operation will program a previously erased phrase in the P-Flash memory using an embedded algorithm.

CAUTION

A P-Flash phrase must be in the erased state before being programmed. Cumulative programming of bits within a Flash phrase is not allowed.

Table 9-43. Program P-Flash Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x06	Global address [22:16] to identify P-Flash block
001	Global address [15:0] of phrase location to be programmed(1)	
010	Word 0 program value	
011	Word 1 program value	
100	Word 2 program value	
101	Word 3 program value	

1. Global address [2:0] must be 000

Upon clearing CCIF to launch the Program P-Flash command, the Memory Controller will program the data words to the supplied global address and will then proceed to verify the data words read back as expected. The CCIF flag will set after the Program P-Flash operation has completed.

Table 9-44. Program P-Flash Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 101 at command launch
		Set if command not available in current mode (see Table 9-30)
		Set if an invalid global address [22:0] is supplied
		Set if a misaligned phrase address is supplied (global address [2:0] != 000)
		Set if a Load Data Field command sequence is currently active and the selected block has previously been selected in the same command sequence
		Set if a Load Data Field command sequence is currently active and global address [17:0] does not match that previously supplied in the same command sequence
	FPVIOL	Set if the global address [22:0] points to a protected area
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation
FERSTAT	EPVIOLIF	None

9.4.2.7 Program Once Command

The Program Once command restricts programming to a reserved 64 byte field (8 phrases) in the nonvolatile information register located in P-Flash block 0. The Program Once reserved field can be read using the Read Once command as described in Section 9.4.2.4. The Program Once command must only be issued once since the nonvolatile information register in P-Flash block 0 cannot be erased. The Program

Once command must not be executed from the Flash block containing the Program Once reserved field to avoid code runaway.

Table 9-45. Program Once Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x07	Not Required
001	Program Once phrase index (0x0000 - 0x0007)	
010	Program Once word 0 value	
011	Program Once word 1 value	
100	Program Once word 2 value	
101	Program Once word 3 value	

Upon clearing CCIF to launch the Program Once command, the Memory Controller first verifies that the selected phrase is erased. If erased, then the selected phrase will be programmed and then verified with read back. The CCIF flag will remain clear, setting only after the Program Once operation has completed.

The reserved nonvolatile information register accessed by the Program Once command cannot be erased and any attempt to program one of these phrases a second time will not be allowed. Valid phrase index values for the Program Once command range from 0x0000 to 0x0007. During execution of the Program Once command, any attempt to read addresses within P-Flash block 0 will return invalid data.

Table 9-46. Program Once Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 101 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 9-30)
		Set if an invalid phrase index is supplied
		Set if the requested phrase has already been programmed(1)
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation
FERSTAT	EPVIOLIF	None

1. If a Program Once phrase is initially programmed to 0xFFFF_FFFF_FFFF_FFFF, the Program Once command will be allowed to execute again on that same phrase.

9.4.2.8 Erase All Blocks Command

The Erase All Blocks operation will erase the entire P-Flash and D-Flash memory space including the EEE nonvolatile information register.

Table 9-47. Erase All Blocks Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x08	Not required

Upon clearing CCIF to launch the Erase All Blocks command, the Memory Controller will erase the entire Flash memory space and verify that it is erased. If the Memory Controller verifies that the entire Flash memory space was properly erased, security will be released. During the execution of this command (CCIF=0) the user must not write to any Flash module register. The CCIF flag will set after the Erase All Blocks operation has completed.

Table 9-48. Erase All Blocks Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 9-30)
	FPVIOL	Set if any area of the P-Flash memory is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation
FERSTAT	EPVIOLIF	Set if any area of the buffer RAM EEE partition is protected

9.4.2.9 Erase P-Flash Block Command

The Erase P-Flash Block operation will erase all addresses in a P-Flash block.

Table 9-49. Erase P-Flash Block Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x09	Global address [22:16] to identify P-Flash block
001	Global address [15:0] in P-Flash block to be erased	

Upon clearing CCIF to launch the Erase P-Flash Block command, the Memory Controller will erase the selected P-Flash block and verify that it is erased. The CCIF flag will set after the Erase P-Flash Block operation has completed.

Table 9-50. Erase P-Flash Block Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 9-30)
		Set if an invalid global address [22:16] is supplied
	FPVIOL	Set if an area of the selected P-Flash block is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation
FERSTAT	EPVIOLIF	None

9.4.2.10 Erase P-Flash Sector Command

The Erase P-Flash Sector operation will erase all addresses in a P-Flash sector.

Table 9-51. Erase P-Flash Sector Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x0A	Global address [22:16] to identify P-Flash block to be erased
001	Global address [15:0] anywhere within the sector to be erased. Refer to Section 9.1.2.1 for the P-Flash sector size.	

Upon clearing CCIF to launch the Erase P-Flash Sector command, the Memory Controller will erase the selected Flash sector and then verify that it is erased. The CCIF flag will be set after the Erase P-Flash Sector operation has completed.

Table 9-52. Erase P-Flash Sector Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 9-30)
		Set if an invalid global address [22:16] is supplied
		Set if a misaligned phrase address is supplied (global address [2:0] != 000)
	FPVIOL	Set if the selected P-Flash sector is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation
FERSTAT	EPVIOLIF	None

9.4.2.11 Unsecure Flash Command

The Unsecure Flash command will erase the entire P-Flash and D-Flash memory space and, if the erase is successful, will release security.

Table 9-53. Unsecure Flash Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x0B	Not required

Upon clearing CCIF to launch the Unsecure Flash command, the Memory Controller will erase the entire P-Flash and D-Flash memory space and verify that it is erased. If the Memory Controller verifies that the entire Flash memory space was properly erased, security will be released. If the erase verify is not successful, the Unsecure Flash operation sets MGSTAT1 and terminates without changing the security state. During the execution of this command (CCIF=0) the user must not write to any Flash module register. The CCIF flag is set after the Unsecure Flash operation has completed.

Table 9-54. Unsecure Flash Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 9-30)
	FPVIOL	Set if any area of the P-Flash memory is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation
FERSTAT	EPVIOLIF	Set if any area of the buffer RAM EEE partition is protected

9.4.2.12 Verify Backdoor Access Key Command

The Verify Backdoor Access Key command will only execute if it is enabled by the KEYEN bits in the FSEC register (see Table 9-11). The Verify Backdoor Access Key command releases security if user-supplied keys match those stored in the Flash security bytes of the Flash configuration field (see Table 9-3). The Verify Backdoor Access Key command must not be executed from the Flash block containing the backdoor comparison key to avoid code runaway.

Table 9-55. Verify Backdoor Access Key Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x0C	Not required
001	Key 0	
010	Key 1	
011	Key 2	
100	Key 3	

Upon clearing CCIF to launch the Verify Backdoor Access Key command, the Memory Controller will check the FSEC KEYEN bits to verify that this command is enabled. If not enabled, the Memory Controller sets the ACCERR bit in the FSTAT register and terminates. If the command is enabled, the Memory Controller compares the key provided in FCCOB to the backdoor comparison key in the Flash configuration field with Key 0 compared to 0x7F_FF00, etc. If the backdoor keys match, security will be released. If the backdoor keys do not match, security is not released and all future attempts to execute the Verify Backdoor Access Key command are aborted (set ACCERR) until a reset occurs. The CCIF flag is set after the Verify Backdoor Access Key operation has completed.

Table 9-56. Verify Backdoor Access Key Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 100 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if an incorrect backdoor key is supplied
		Set if backdoor key access has not been enabled (KEYEN[1:0] != 10, see Section 9.3.2.2)
		Set if the backdoor key has mismatched since the last reset
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

9.4.2.13 Set User Margin Level Command

The Set User Margin Level command causes the Memory Controller to set the margin level for future read operations of a specific P-Flash or D-Flash block.

Table 9-57. Set User Margin Level Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x0D	Global address [22:16] to identify the Flash block
001	Margin level setting	

Upon clearing CCIF to launch the Set User Margin Level command, the Memory Controller will set the user margin level for the targeted block and then set the CCIF flag.

Valid margin level settings for the Set User Margin Level command are defined in Table 9-58.

Table 9-58. Valid Set User Margin Level Settings

CCOB (CCOBIX=001)	Level Description
0x0000	Return to Normal Level
0x0001	User Margin-1 Level(1)

Table 9-58. Valid Set User Margin Level Settings

CCOB (CCOBIX=001)	Level Description
0x0002	User Margin-0 Level(2)

1. Read margin to the erased state
2. Read margin to the programmed state

Table 9-59. Set User Margin Level Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 9-30)
		Set if an invalid global address [22:16] is supplied
		Set if an invalid margin level setting is supplied
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

NOTE

User margin levels can be used to check that Flash memory contents have adequate margin for normal level read operations. If unexpected results are encountered when checking Flash memory contents at user margin levels, a potential loss of information has been detected.

9.4.2.14 Set Field Margin Level Command

The Set Field Margin Level command, valid in special modes only, causes the Memory Controller to set the margin level specified for future read operations of a specific P-Flash or D-Flash block.

Table 9-60. Set Field Margin Level Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x0E	Global address [22:16] to identify the Flash block
001	Margin level setting	

Upon clearing CCIF to launch the Set Field Margin Level command, the Memory Controller will set the field margin level for the targeted block and then set the CCIF flag.

Valid margin level settings for the Set Field Margin Level command are defined in Table 9-61.

Table 9-61. Valid Set Field Margin Level Settings

CCOB (CCOBIX=001)	Level Description
0x0000	Return to Normal Level
0x0001	User Margin-1 Level(1)
0x0002	User Margin-0 Level(2)
0x0003	Field Margin-1 Level[1]
0x0004	Field Margin-0 Level[2]

1. Read margin to the erased state
2. Read margin to the programmed state

Table 9-62. Set Field Margin Level Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 9-30)
		Set if an invalid global address [22:16] is supplied
		Set if an invalid margin level setting is supplied
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

CAUTION

Field margin levels must only be used during verify of the initial factory programming.

NOTE

Field margin levels can be used to check that Flash memory contents have adequate margin for data retention at the normal level setting. If unexpected results are encountered when checking Flash memory contents at field margin levels, the Flash memory contents should be erased and reprogrammed.

9.4.2.15 Full Partition D-Flash Command

The Full Partition D-Flash command allows the user to allocate sectors within the D-Flash block for applications and a partition within the buffer RAM for EEPROM access. The D-Flash block consists of 128 sectors with 256 bytes per sector.

Table 9-63. Full Partition D-Flash Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x0F	Not required
001	Number of 256 byte sectors for the D-Flash user partition (DFPART)	
010	Number of 256 byte sectors for buffer RAM EEE partition (ERPART)	

Upon clearing CCIF to launch the Full Partition D-Flash command, the following actions are taken to define a partition within the D-Flash block for direct access (DFPART) and a partition within the buffer RAM for EEE use (ERPART):

- Validate the DFPART and ERPART values provided:
  - DFPART <= 128 (maximum number of 256 byte sectors in D-Flash block)
  - ERPART <= 16 (maximum number of 256 byte sectors in buffer RAM)
  - If ERPART > 0, 128 - DFPART >= 12 (minimum number of 256 byte sectors in the D-Flash block required to support EEE)
  - If ERPART > 0, ((128-DFPART)/ERPART) >= 8 (minimum ratio of D-Flash EEE space to buffer RAM EEE space to support EEE)
- Erase the D-Flash block and the EEE nonvolatile information register
- Program DFPART to the EEE nonvolatile information register at global address 0x12_0000 (see Table 9-7)
- Program a duplicate DFPART to the EEE nonvolatile information register at global address 0x12_0002 (see Table 9-7)
- Program ERPART to the EEE nonvolatile information register at global address 0x12_0004 (see Table 9-7)
- Program a duplicate ERPART to the EEE nonvolatile information register at global address 0x12_0006 (see Table 9-7)

The D-Flash user partition will start at global address 0x10_0000. The buffer RAM EEE partition will end at global address 0x13_FFFF. After the Full Partition D-Flash operation has completed, the CCIF flag will set.

Running the Full Partition D-Flash command a second time will result in the previous partition values and the entire D-Flash memory being erased. The data value written corresponds to the number of 256 byte sectors allocated for either direct D-Flash access (DFPART) or buffer RAM EEE access (ERPART).

Table 9-64. Full Partition D-Flash Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 010 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 9-30)
		Set if an invalid DFPART or ERPART selection is supplied
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read
	MGSTAT0	Set if any non-correctable errors have been encountered during the read
FERSTAT	EPVIOLIF	None

9.4.2.16 Erase Verify D-Flash Section Command

The Erase Verify D-Flash Section command will verify that a section of code in the D-Flash user partition is erased. The Erase Verify D-Flash Section command defines the starting point of the data to be verified and the number of words.

Table 9-65. Erase Verify D-Flash Section Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x10	Global address [22:16] to identify the D-Flash block
001	Global address [15:0] of the first word to be verified	
010	Number of words to be verified	

Upon clearing CCIF to launch the Erase Verify D-Flash Section command, the Memory Controller will verify the selected section of D-Flash memory is erased. The CCIF flag will set after the Erase Verify D-Flash Section operation has completed.

Table 9-66. Erase Verify D-Flash Section Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 010 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 9-30)
		Set if an invalid global address [22:0] is supplied
		Set if a misaligned word address is supplied (global address [0] != 0)
		Set if the global address [22:0] points to an area of the D-Flash EEE partition
		Set if the requested section breaches the end of the D-Flash block or goes into the D-Flash EEE partition
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read
	MGSTAT0	Set if any non-correctable errors have been encountered during the read
FERSTAT	EPVIOLIF	None

9.4.2.17 Program D-Flash Command

The Program D-Flash operation programs one to four previously erased words in the D-Flash user partition. The Program D-Flash operation will confirm that the targeted location(s) were successfully programmed upon completion.

CAUTION

A Flash word must be in the erased state before being programmed. Cumulative programming of bits within a Flash word is not allowed.

Table 9-67. Program D-Flash Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x11	Global address [22:16] to identify the D-Flash block
001	Global address [15:0] of word to be programmed	
010	Word 0 program value	
011	Word 1 program value, if desired	
100	Word 2 program value, if desired	
101	Word 3 program value, if desired	

Upon clearing CCIF to launch the Program D-Flash command, the user-supplied words will be transferred to the Memory Controller and be programmed. The CCOBIX index value at Program D-Flash command launch determines how many words will be programmed in the D-Flash block. No protection checks are made in the Program D-Flash operation on the D-Flash block, only access error checks. The CCIF flag is set when the operation has completed.

Table 9-68. Program D-Flash Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] < 010 at command launch
		Set if CCOBIX[2:0] > 101 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 9-30)
		Set if an invalid global address [22:0] is supplied
		Set if a misaligned word address is supplied (global address [0] != 0)
		Set if the global address [22:0] points to an area in the D-Flash EEE partition
		Set if the requested group of words breaches the end of the D-Flash block or goes into the D-Flash EEE partition
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation
FERSTAT	EPVIOLIF	None

9.4.2.18 Erase D-Flash Sector Command

The Erase D-Flash Sector operation will erase all addresses in a sector of the D-Flash user partition.

Table 9-69. Erase D-Flash Sector Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x12	Global address [22:16] to identify D-Flash block
001	Global address [15:0] anywhere within the sector to be erased. See Section 9.1.2.2 for D-Flash sector size.	

Upon clearing CCIF to launch the Erase D-Flash Sector command, the Memory Controller will erase the selected Flash sector and verify that it is erased. The CCIF flag will set after the Erase D-Flash Sector operation has completed.

Table 9-70. Erase D-Flash Sector Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 9-30)
		Set if an invalid global address [22:0] is supplied
		Set if a misaligned word address is supplied (global address [0] != 0)
		Set if the global address [22:0] points to the D-Flash EEE partition
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation
FERSTAT	EPVIOLIF	None

9.4.2.19 Enable EEPROM Emulation Command

The Enable EEPROM Emulation command causes the Memory Controller to enable EEE activity. EEE activity is disabled after any reset.

Table 9-71. Enable EEPROM Emulation Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x13	Not required

Upon clearing CCIF to launch the Enable EEPROM Emulation command, the CCIF flag will set after the Memory Controller enables EEE operations using the contents of the EEE tag RAM and tag counter. The Full Partition D-Flash or the Partition D-Flash command must be run prior to launching the Enable EEPROM Emulation command.

Table 9-72. Enable EEPROM Emulation Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if Full Partition D-Flash or Partition D-Flash command not previously run
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

9.4.2.20 Disable EEPROM Emulation Command

The Disable EEPROM Emulation command causes the Memory Controller to suspend current EEE activity.

Table 9-73. Disable EEPROM Emulation Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x14	Not required

Upon clearing CCIF to launch the Disable EEPROM Emulation command, the Memory Controller will halt EEE operations at the next convenient point without clearing the EEE tag RAM or tag counter before setting the CCIF flag.

Table 9-74. Disable EEPROM Emulation Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if Full Partition D-Flash or Partition D-Flash command not previously run
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

9.4.2.21 EEPROM Emulation Query Command

The EEPROM Emulation Query command returns EEE partition and status variables.

Table 9-75. EEPROM Emulation Query Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x15	Not required
001	Return DFPART	
010	Return ERPART	
011	Return ECOUNT(1)	
100	Return Dead Sector Count	Return Ready Sector Count

1. Indicates sector erase count

Upon clearing CCIF to launch the EEPROM Emulation Query command, the CCIF flag will set after the EEE partition and status variables are stored in the FCCOBIX register.If the Emulation Query command is executed prior to partitioning (Partition D-Flash Command Section 9.4.2.15), the following reset values are returned: DFPART = 0x_FFFF, ERPART = 0x_FFFF, ECOUNT = 0x_FFFF, Dead Sector Count = 0x_00, Ready Sector Count = 0x_00.

Table 9-76. EEPROM Emulation Query Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 9-30)
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

9.4.2.22 Partition D-Flash Command

The Partition D-Flash command allows the user to allocate sectors within the D-Flash block for applications and a partition within the buffer RAM for EEPROM access. The D-Flash block consists of 128 sectors with 256 bytes per sector. The Erase All Blocks command must be run prior to launching the Partition D-Flash command.

Table 9-77. Partition D-Flash Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x20	Not required
001	Number of 256 byte sectors for the D-Flash user partition (DFPART)	
010	Number of 256 byte sectors for buffer RAM EEE partition (ERPART)	

Upon clearing CCIF to launch the Partition D-Flash command, the following actions are taken to define a partition within the D-Flash block for direct access (DFPART) and a partition within the buffer RAM for EEE use (ERPART):

- Validate the DFPART and ERPART values provided:
  - DFPART <= 128 (maximum number of 256 byte sectors in D-Flash block)
  - ERPART <= 16 (maximum number of 256 byte sectors in buffer RAM)
  - If ERPART > 0, 128 - DFPART >= 12 (minimum number of 256 byte sectors in the D-Flash block required to support EEE)
  - If ERPART > 0, ((128-DFPART)/ERPART) >= 8 (minimum ratio of D-Flash EEE space to buffer RAM EEE space to support EEE)
- Erase verify the D-Flash block and the EEE nonvolatile information register
- Program DFPART to the EEE nonvolatile information register at global address 0x12_0000 (see Table 9-7)

- Program a duplicate DFPART to the EEE nonvolatile information register at global address 0x12_0002 (see Table 9-7)
- Program ERPART to the EEE nonvolatile information register at global address 0x12_0004 (see Table 9-7)
- Program a duplicate ERPART to the EEE nonvolatile information register at global address 0x12_0006 (see Table 9-7)

The D-Flash user partition will start at global address 0x10_0000. The buffer RAM EEE partition will end at global address 0x13_FFFF. After the Partition D-Flash operation has completed, the CCIF flag will set.

Running the Partition D-Flash command a second time will result in the ACCERR bit within the FSTAT register being set. The data value written corresponds to the number of 256 byte sectors allocated for either direct D-Flash access (DFPART) or buffer RAM EEE access (ERPART).

Table 9-78. Partition D-Flash Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 010 at command launch
		Set if a Load Data Field command sequence is currently active
		Set if command not available in current mode (see Table 9-30)
		Set if partitions have already been defined
		Set if an invalid DFPART or ERPART selection is supplied
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read
	MGSTAT0	Set if any non-correctable errors have been encountered during the read
FERSTAT	EPVIOLIF	None

9.4.3 Interrupts

The Flash module can generate an interrupt when a Flash command operation has completed or when a Flash command operation has detected an EEE error or an ECC fault.

Table 9-79. Flash Interrupt Sources

Interrupt Source	Interrupt Flag	Local Enable	Global (CCR) Mask
Flash Command Complete	CCIF (FSTAT register)	CCIE (FCNFG register)	I Bit
Flash EEE Erase Error	ERSERIF (FERSTAT register)	ERSERIE (FERCNFG register)	I Bit
Flash EEE Program Error	PGMERIF (FERSTAT register)	PGMERIE (FERCNFG register)	I Bit
Flash EEE Protection Violation	EPVIOLIF (FERSTAT register)	EPVIOLIE (FERCNFG register)	I Bit
Flash EEE Error Type 1 Violation	ERSVIF1 (FERSTAT register)	ERSVIE1 (FERCNFG register)	I Bit
Flash EEE Error Type 0 Violation	ERSVIF0 (FERSTAT register)	ERSVIE0 (FERCNFG register)	I Bit
ECC Double Bit Fault on Flash Read	DFDIF (FERSTAT register)	DFDIE (FERCNFG register)	I Bit
ECC Single Bit Fault on Flash Read	SFDIF (FERSTAT register)	SFDIE (FERCNFG register)	I Bit

NOTE

Vector addresses and their relative interrupt priority are determined at the MCU level.

9.4.3.1 Description of Flash Interrupt Operation

The Flash module uses the CCIF flag in combination with the CCIE interrupt enable bit to generate the Flash command interrupt request. The Flash module uses the ERSEIF, PGMEIF, EPVIOLIF, ERSVIF1, ERSVIF0, DFDIF and SFDIF flags in combination with the ERSEIE, PGMEIE, EPVIOLIE, ERSVIE1, ERSVIE0, DFDIE and SFDIE interrupt enable bits to generate the Flash error interrupt request. For a detailed description of the register bits involved, refer to Section 9.3.2.5, “Flash Configuration Register (FCNFG)”, Section 9.3.2.6, “Flash Error Configuration Register (FERCNFG)”, Section 9.3.2.7, “Flash Status Register (FSTAT)”, and Section 9.3.2.8, “Flash Error Status Register (FERSTAT)”.

The logic used for generating the Flash module interrupts is shown in Figure 9-27.

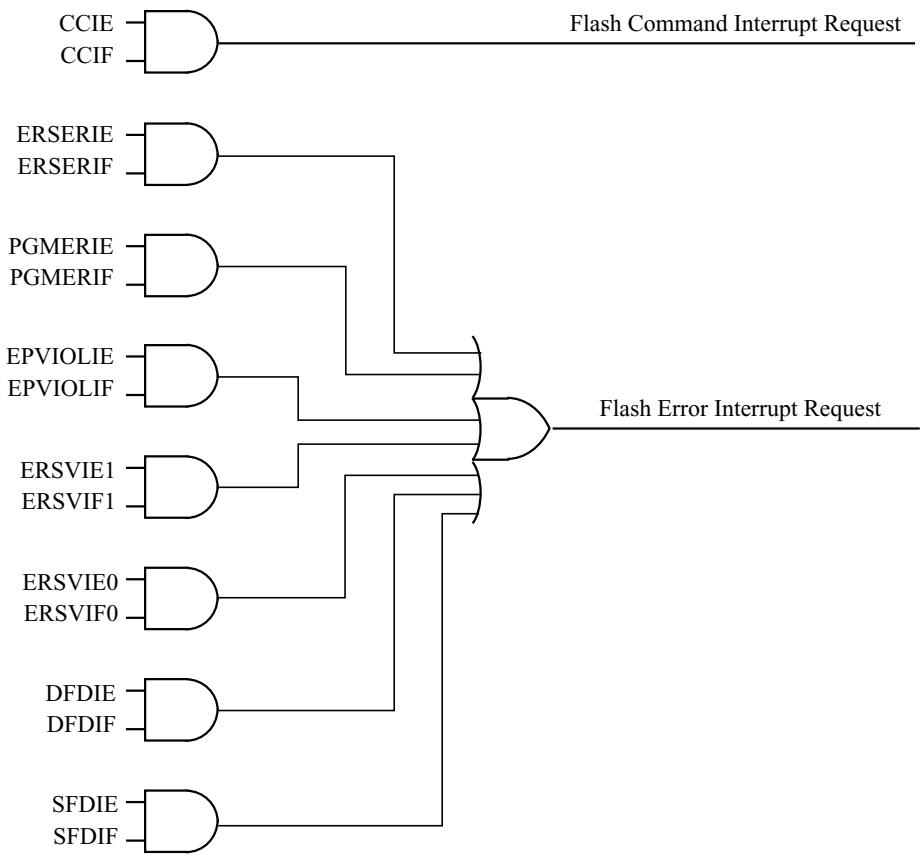


Figure 9-27. Flash Module Interrupts Implementation

9.4.4 Wait Mode

The Flash module is not affected if the MCU enters wait mode. The Flash module can recover the MCU from wait via the CCIF interrupt (see Section 9.4.3, “Interrupts”).

9.4.5 Stop Mode

If a Flash command is active (CCIF = 0) or an EE-Emulation operation is pending when the MCU requests stop mode, the current Flash operation will be completed before the CPU is allowed to enter stop mode.

9.5 Security

The Flash module provides security information to the MCU. The Flash security state is defined by the SEC bits of the FSEC register (see Table 9-12). During reset, the Flash module initializes the FSEC register using data read from the security byte of the Flash configuration field at global address 0x7F_FF0F.

The security state out of reset can be permanently changed by programming the security byte of the Flash configuration field. This assumes that you are starting from a mode where the necessary P-Flash erase and program commands are available and that the upper region of the P-Flash is unprotected. If the Flash security byte is successfully programmed, its new value will take affect after the next MCU reset.

The following subsections describe these security-related subjects:

- Unsecuring the MCU using Backdoor Key Access
- Unsecuring the MCU in Special Single Chip Mode using BDM
- Mode and Security Effects on Flash Command Availability

9.5.1 Unsecuring the MCU using Backdoor Key Access

The MCU may be unsecured by using the backdoor key access feature which requires knowledge of the contents of the backdoor keys (four 16-bit words programmed at addresses 0x7F_FF00–0x7F_FF07). If the KEYEN[1:0] bits are in the enabled state (see Section 9.3.2.2), the Verify Backdoor Access Key command (see Section 9.4.2.12) allows the user to present four prospective keys for comparison to the keys stored in the Flash memory via the Memory Controller. If the keys presented in the Verify Backdoor Access Key command match the backdoor keys stored in the Flash memory, the SEC bits in the FSEC register (see Table 9-12) will be changed to unsecure the MCU. Key values of 0x0000 and 0xFFFF are not permitted as backdoor keys. While the Verify Backdoor Access Key command is active, P-Flash block 0 will not be available for read access and will return invalid data.

The user code stored in the P-Flash memory must have a method of receiving the backdoor keys from an external stimulus. This external stimulus would typically be through one of the on-chip serial ports.

If the KEYEN[1:0] bits are in the enabled state (see Section 9.3.2.2), the MCU can be unsecured by the backdoor key access sequence described below:

1. Follow the command sequence for the Verify Backdoor Access Key command as explained in Section 9.4.2.12
2. If the Verify Backdoor Access Key command is successful, the MCU is unsecured and the SEC[1:0] bits in the FSEC register are forced to the unsecure state of 10

The Verify Backdoor Access Key command is monitored by the Memory Controller and an illegal key will prohibit future use of the Verify Backdoor Access Key command. A reset of the MCU is the only method to re-enable the Verify Backdoor Access Key command.

After the backdoor keys have been correctly matched, the MCU will be unsecured. After the MCU is unsecured, the sector containing the Flash security byte can be erased and the Flash security byte can be reprogrammed to the unsecure state, if desired.

In the unsecure state, the user has full control of the contents of the backdoor keys by programming addresses 0x7F_FF00–0x7F_FF07 in the Flash configuration field.

The security as defined in the Flash security byte (0x7F_FF0F) is not changed by using the Verify Backdoor Access Key command sequence. The backdoor keys stored in addresses 0x7F_FF00–0x7F_FF07 are unaffected by the Verify Backdoor Access Key command sequence. After the next reset of the MCU, the security state of the Flash module is determined by the Flash security byte

(0x7F_FF0F). The Verify Backdoor Access Key command sequence has no effect on the program and erase protections defined in the Flash protection register, FPROT.

9.5.2 Unsecuring the MCU in Special Single Chip Mode using BDM

The MCU can be unsecured in special single chip mode by erasing the P-Flash and D-Flash memory by one of the following methods:

- Reset the MCU into special single chip mode, delay while the erase test is performed by the BDM, send BDM commands to disable protection in the P-Flash and D-Flash memory, and execute the Erase All Blocks command write sequence to erase the P-Flash and D-Flash memory.
- Reset the MCU into special expanded wide mode, disable protection in the P-Flash and D-Flash memory and run code from external memory to execute the Erase All Blocks command write sequence to erase the P-Flash and D-Flash memory.

After the CCIF flag sets to indicate that the Erase All Blocks operation has completed, reset the MCU into special single chip mode. The BDM will execute the Erase Verify All Blocks command write sequence to verify that the P-Flash and D-Flash memory is erased. If the P-Flash and D-Flash memory are verified as erased the MCU will be unsecured. All BDM commands will be enabled and the Flash security byte may be programmed to the unsecure state by the following method:

- Send BDM commands to execute a ‘Program P-Flash’ command sequence to program the Flash security byte to the unsecured state and reset the MCU.

9.5.3 Mode and Security Effects on Flash Command Availability

The availability of Flash module commands depends on the MCU operating mode and security state as shown in Table 9-30.

9.6 Initialization

On each system reset the Flash module executes a reset sequence which establishes initial values for the Flash Block Configuration Parameters, the FPROT and DFPROT protection registers, and the FOPT and FSEC registers. The Flash module reverts to built-in default values that leave the module in a fully protected and secured state if errors are encountered during execution of the reset sequence. If a double bit fault is detected during the reset sequence, both MGSTAT bits in the FSTAT register will be set. The ACCERR bit in the FSTAT register is set if errors are encountered while initializing the EEE buffer ram during the reset sequence.

CCIF remains clear throughout the reset sequence. The Flash module holds off all CPU access for the initial portion of the reset sequence. While Flash reads are possible when the hold is removed, writes to the FCCOBIX, FCCOBHI, and FCCOBLO registers are ignored to prevent command activity while the Memory Controller remains busy. Completion of the reset sequence is marked by setting CCIF high which enables writes to the FCCOBIX, FCCOBHI, and FCCOBLO registers to launch any available Flash command.

If a reset occurs while any Flash command is in progress, that command will be immediately aborted. The state of the word being programmed or the sector/block being erased is not guaranteed.

Chapter 10
128 KByte Flash Module (S12XFTM128K2XFV1)

Table 10-1. Revision History

Revision Number	Revision Date	Sections Affected	Description of Changes
V01.00	12 Dec 2007		- Initial version
V01.01	19 Dec 2007	10.4.2/10-379 10.4.2/10-379 10.4.2/10-379 10.4.2/10-379 10.3.1/10-348 10.1.3/10-346 10.3.1/10-348 10.3.1/10-348	- Removed Load Data Field command 0x05 - Updated Command Error Handling tables based on parent-child relationship with FTM512K3 - Corrected Error Handling table for Full Partition D-Flash, Partition D-Flash, and EEPROM Emulation Query commands - Corrected maximum allowed ERPART for Full Partition D-Flash and Partition D-Flash commands - Corrected P-Flash IFR Accessibility table - Corrected Buffer RAM size in Feature List - Corrected EEE Resource Memory Map - Corrected P-Flash Memory Map
V01.02	25 Sep 2009	10.1/10-344 10.3.2.1/10-355 10.4.2.4/10-382 10.4.2.6/10-383 10.4.2.11/10-387 10.4.2.11/10-387 10.4.2.11/10-387 10.4.2.19/10-396 10.3.2/10-353 10.3.2.1/10-355 10.4.1.2/10-374 10.6/10-402	- Change references for D-Flash from 16 Kbytes to 32 Kbytes - Clarify single bit fault correction for P-Flash phrase - Expand FDIV vs OSCCLK Frequency table - Add statement concerning code runaway when executing Read Once command from Flash block containing associated fields - Add statement concerning code runaway when executing Program Once command from Flash block containing associated fields - Add statement concerning code runaway when executing Verify Backdoor Access Key command from Flash block containing associated fields - Relate Key 0 to associated Backdoor Comparison Key address - Change “power down reset” to “reset” - Add ACCERR condition for Disable EEPROM Emulation command The following changes were made to clarify module behavior related to Flash register access during reset sequence and while Flash commands are active: - Add caution concerning register writes while command is active - Writes to FCLKDIV are allowed during reset sequence while CCIF is clear - Add caution concerning register writes while command is active - Writes to FCCOBIX, FCCOBHI, FCCOBLO registers are ignored during reset sequence

10.1 Introduction

The FTM128K2XF module implements the following:

- 128 Kbytes of P-Flash (Program Flash) memory, consisting of 2 physical Flash blocks, intended primarily for nonvolatile code storage
- 32 Kbytes of D-Flash (Data Flash) memory, consisting of 1 physical Flash block, that can be used as nonvolatile storage to support the built-in hardware scheme for emulated EEPROM, as basic Flash memory primarily intended for nonvolatile data storage, or as a combination of both
- 2 Kbytes of buffer RAM, consisting of 1 physical RAM block, that can be used as emulated EEPROM using a built-in hardware scheme, as basic RAM, or as a combination of both

The Flash memory is ideal for single-supply applications allowing for field reprogramming without requiring external high voltage sources for program or erase operations. The Flash module includes a memory controller that executes commands to modify Flash memory contents or configure module resources for emulated EEPROM operation. The user interface to the memory controller consists of the indexed Flash Common Command Object (FCCOB) register which is written to with the command, global address, data, and any required command parameters. The memory controller must complete the execution of a command before the FCCOB register can be written to with a new command.

CAUTION

A Flash word or phrase must be in the erased state before being programmed. Cumulative programming of bits within a Flash word or phrase is not allowed.

The RAM and Flash memory may be read as bytes, aligned words, or misaligned words. Read access time is one bus cycle for bytes and aligned words, and two bus cycles for misaligned words. For Flash memory, an erased bit reads 1 and a programmed bit reads 0.

It is not possible to read from a Flash block while any command is executing on that specific Flash block. It is possible to read from a Flash block while a command is executing on a different Flash block.

Both P-Flash and D-Flash memories are implemented with Error Correction Codes (ECC) that can resolve single bit faults and detect double bit faults. For P-Flash memory, the ECC implementation requires that programming be done on an aligned 8 byte basis (a Flash phrase). Since P-Flash memory is always read by phrase, only one single bit fault in the phrase containing the byte or word accessed will be corrected.

10.1.1 Glossary

Buffer RAM — The buffer RAM constitutes the volatile memory store required for EEE. Memory space in the buffer RAM not required for EEE can be partitioned to provide volatile memory space for applications.

Command Write Sequence — An MCU instruction sequence to execute built-in algorithms (including program and erase) on the Flash memory.

D-Flash Memory — The D-Flash memory constitutes the nonvolatile memory store required for EEE. Memory space in the D-Flash memory not required for EEE can be partitioned to provide nonvolatile memory space for applications.

D-Flash Sector — The D-Flash sector is the smallest portion of the D-Flash memory that can be erased. The D-Flash sector consists of four 64 byte rows for a total of 256 bytes.

EEE (Emulated EEPROM) — A method to emulate the small sector size features and endurance characteristics associated with an EEPROM.

EEE IFR — Nonvolatile information register located in the D-Flash block that contains data required to partition the D-Flash memory and buffer RAM for EEE. The EEE IFR is visible in the global memory map by setting the EEEIFRON bit in the MMCCTL1 register.

NVM Command Mode — An NVM mode using the CPU to setup the FCCOB register to pass parameters required for Flash command execution.

Phrase — An aligned group of four 16-bit words within the P-Flash memory. Each phrase includes eight ECC bits for single bit fault correction and double bit fault detection within the phrase.

P-Flash Memory — The P-Flash memory constitutes the main nonvolatile memory store for applications.

P-Flash Sector — The P-Flash sector is the smallest portion of the P-Flash memory that can be erased. Each P-Flash sector contains 1024 bytes.

Program IFR — Nonvolatile information register located in the P-Flash block that contains the Device ID, Version ID, and the Program Once field. The Program IFR is visible in the global memory map by setting the PGMIFRON bit in the MMCCTL1 register.

10.1.2 Features

10.1.2.1 P-Flash Features

- 128 Kbytes of P-Flash memory composed of two 64 Kbyte Flash blocks. The 64 Kbyte Flash blocks are each divided into 64 sectors of 1024 bytes.
- Single bit fault correction and double bit fault detection within a 64-bit phrase during read operations
- Automated program and erase algorithm with verify and generation of ECC parity bits
- Fast sector erase and phrase program operation
- Ability to program up to one phrase in each P-Flash block simultaneously
- Flexible protection scheme to prevent accidental program or erase of P-Flash memory

10.1.2.2 D-Flash Features

- Up to 32 Kbytes of D-Flash memory with 256 byte sectors for user access
- Dedicated commands to control access to the D-Flash memory over EEE operation

- Single bit fault correction and double bit fault detection within a word during read operations
- Automated program and erase algorithm with verify and generation of ECC parity bits
- Fast sector erase and word program operation
- Ability to program up to four words in a burst sequence

10.1.2.3 Emulated EEPROM Features

- Up to 2 Kbytes of emulated EEPROM (EEE) accessible as 2 Kbytes of RAM
- Flexible protection scheme to prevent accidental program or erase of data
- Automatic EEE file handling using an internal Memory Controller
- Automatic transfer of valid EEE data from D-Flash memory to buffer RAM on reset
- Ability to monitor the number of outstanding EEE related buffer RAM words left to be programmed into D-Flash memory
- Ability to disable EEE operation and allow priority access to the D-Flash memory
- Ability to cancel all pending EEE operations and allow priority access to the D-Flash memory

10.1.2.4 User Buffer RAM Features

- Up to 2 Kbytes of RAM for user access

10.1.2.5 Other Flash Module Features

- No external high-voltage power supply required for Flash memory program and erase operations
- Interrupt generation on Flash command completion and Flash error detection
- Security mechanism to prevent unauthorized access to the Flash memory

10.1.3 Block Diagram

The block diagram of the Flash module is shown in Figure 10-1.

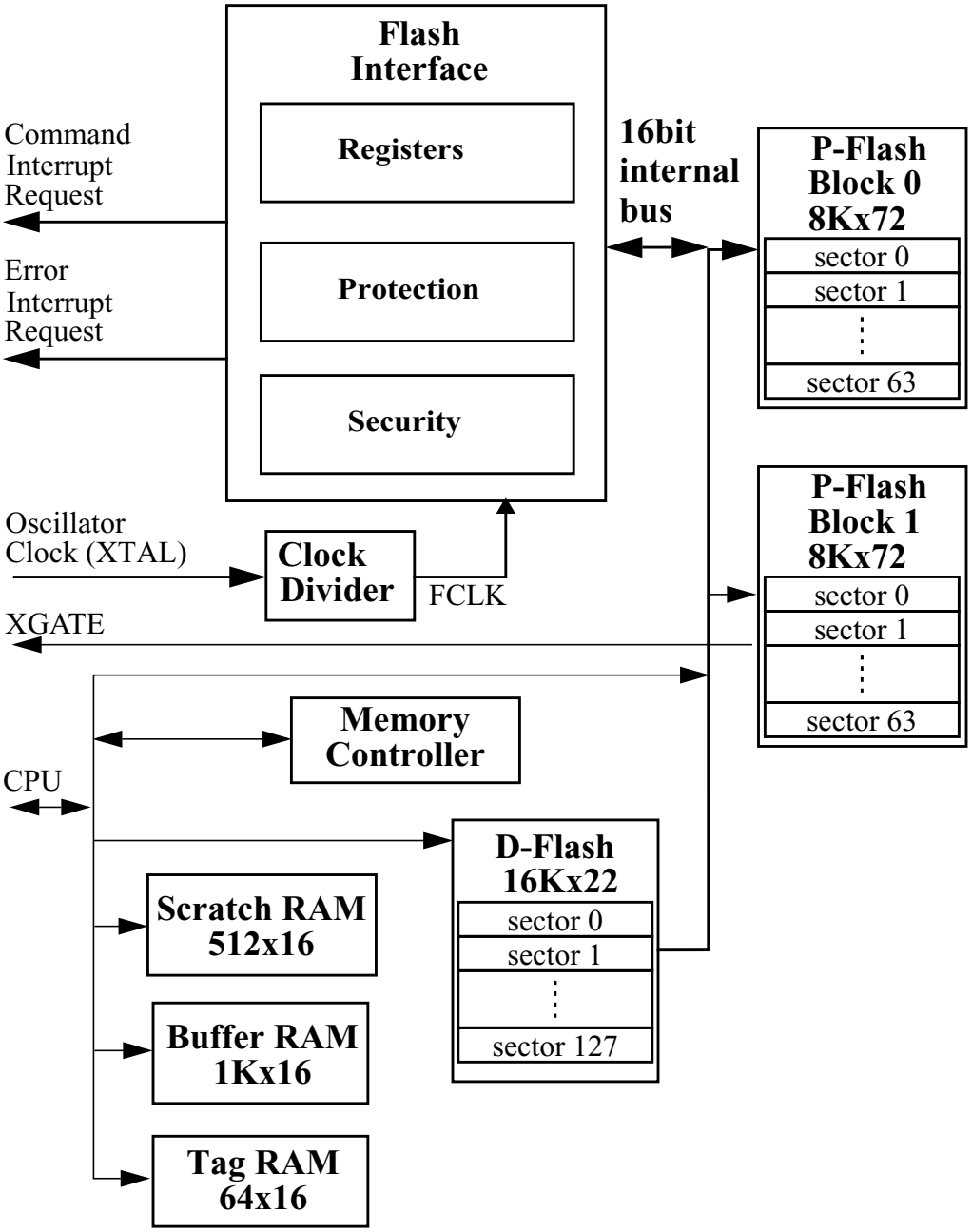


Figure 10-1. FTM128K2 Block Diagram

10.2 External Signal Description

The Flash module contains no signals that connect off-chip.

10.3 Memory Map and Registers

This section describes the memory map and registers for the Flash module. Read data from unimplemented memory space in the Flash module is undefined. Write access to unimplemented or reserved memory space in the Flash module will be ignored by the Flash module.

10.3.1 Module Memory Map

The S12X architecture places the P-Flash memory between global addresses 0x78_0000 and 0x7F_FFFF as shown in Table 10-2. The P-Flash memory map is shown in Figure 10-2.

Table 10-2. P-Flash Memory Addressing

Global Address	Size (Bytes)	Description
0x7F_0000 – 0x7F_FFFF	64 K	P-Flash Block 0 Contains Flash Configuration Field (see Table 10-3)
0x79_0000 – 0x7E_FFFF	384 K	No P-Flash Memory
0x78_0000 – 0x78_FFFF	64 K	P-Flash Block 1

The FPROT register, described in Section 10.3.2.9, can be set to protect regions in the Flash memory from accidental program or erase. Three separate memory regions, one growing upward from global address 0x7F_8000 in the Flash memory (called the lower region), one growing downward from global address 0x7F_FFFF in the Flash memory (called the higher region), and the remaining addresses in the Flash memory, can be activated for protection. The Flash memory addresses covered by these protectable regions are shown in the P-Flash memory map. The higher address region is mainly targeted to hold the boot loader code since it covers the vector space. Default protection settings as well as security information that allows the MCU to restrict access to the Flash module are stored in the Flash configuration field as described in Table 10-3.

Table 10-3. Flash Configuration Field(1)

Global Address	Size (Bytes)	Description
0x7F_FF00 – 0x7F_FF07	8	Backdoor Comparison Key Refer to Section 10.4.2.11, “Verify Backdoor Access Key Command,” and Section 10.5.1, “Unsecuring the MCU using Backdoor Key Access”
0x7F_FF08 – 0x7F_FF0B(2)	4	Reserved
0x7F_FF0C[2]	1	P-Flash Protection byte. Refer to Section 10.3.2.9, “P-Flash Protection Register (FPROT)”
0x7F_FF0D[2]	1	EEE Protection byte Refer to Section 10.3.2.10, “EEE Protection Register (EPROT)”
0x7F_FF0E[2]	1	Flash Nonvolatile byte Refer to Section 10.3.2.14, “Flash Option Register (FOPT)”

Table 10-3. Flash Configuration Field[1]

Global Address	Size (Bytes)	Description
0x7F_FF0F[2]	1	Flash Security byte Refer to Section 10.3.2.2, “Flash Security Register (FSEC)”

1. Older versions may have swapped protection byte addresses
2. 0x7FF08 - 0x7F_FF0F form a Flash phrase and must be programmed in a single command write sequence. Each byte in the 0x7F_FF08 - 0x7F_FF0B reserved field should be programmed to 0xFF.

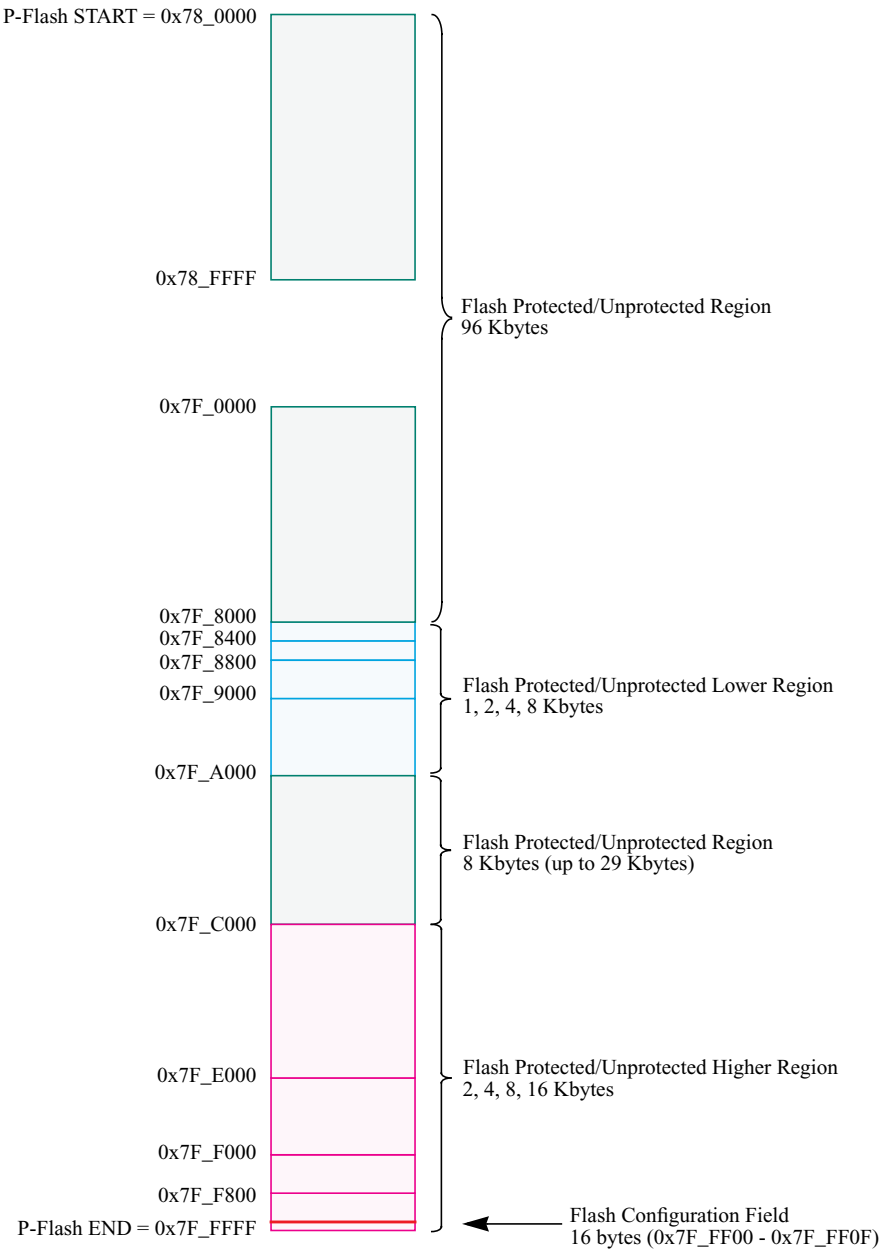


Figure 10-2. P-Flash Memory Map

Table 10-4. Program IFR Fields

Global Address (PGMIFRON)	Size (Bytes)	Field Description
0x40_0000 – 0x40_0007	8	Device ID
0x40_0008 – 0x40_00E7	224	Reserved
0x40_00E8 – 0x40_00E9	2	Version ID
0x40_00EA – 0x40_00FF	22	Reserved
0x40_0100 – 0x40_013F	64	Program Once Field Refer to Section 10.4.2.6, “Program Once Command”
0x40_0140 – 0x40_01FF	192	Reserved

Table 10-5. P-Flash IFR Accessibility

Global Address (PGMIFRON)	Size (Bytes)	Accessed From
0x40_0000 – 0x40_01FF	512	XBUS0 (PBLK0)(1)
0x40_0200 – 0x40_03FF	512	Unimplemented
0x40_0400 – 0x40_05FF	512	Unimplemented
0x40_0600 – 0x40_07FF	512	XBUS1 (PBLK1)

1. Refer to Table 10-4 for more details.

Table 10-6. EEE Resource Fields

Global Address	Size (Bytes)	Description
0x10_0000 – 0x10_7FFF	32,768	D-Flash Memory (User and EEE)
0x10_8000 – 0x11_FFFF	98,304	Reserved
0x12_0000 – 0x12_007F	128	EEE Nonvolatile Information Register (EEEIFRON(1) = 1)
0x12_0080 – 0x12_0FFF	3,968	Reserved
0x12_1000 – 0x12_1F7F	3,968	Reserved
0x12_1F80 – 0x12_1FFF	128	EEE Tag RAM (TMGRAMON1 = 1)
0x12_2000 – 0x12_3BFF	7,168	Reserved
0x12_3C00 – 0x12_3FFF	1,024	Memory Controller Scratch RAM (TMGRAMON1 = 1)
0x12_4000 – 0x12_DFFF	40,960	Reserved
0x12_E000 – 0x12_FFFF	8,192	Reserved
0x13_0000 – 0x13_F7FF	63,488	Reserved
0x13_F800 – 0x13_FFFF	2,048	Buffer RAM (User and EEE)

1. MMCCTL1 register bit

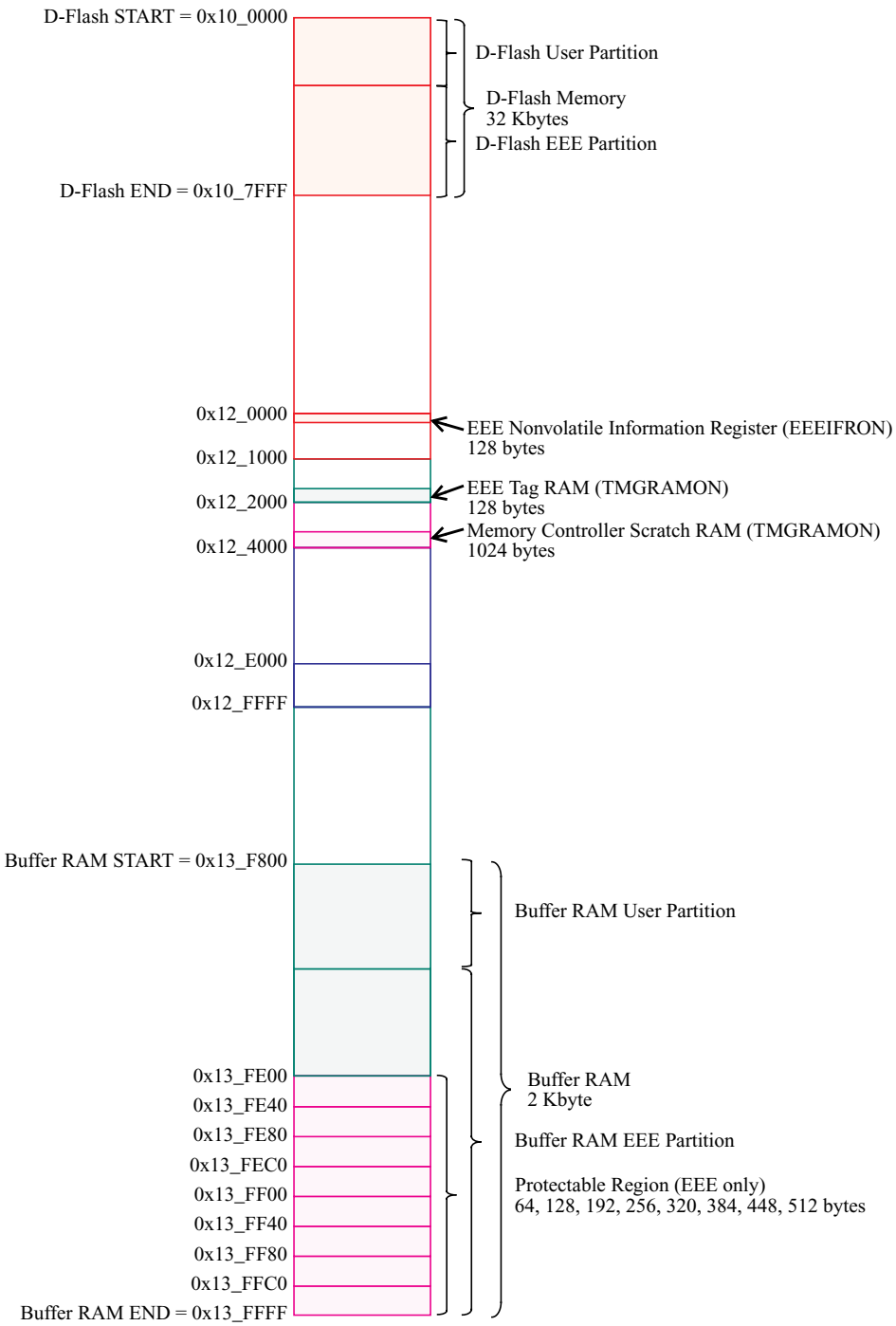


Figure 10-3. EEE Resource Memory Map

The Full Partition D-Flash command (see Section 10.4.2.14) is used to program the EEE nonvolatile information register fields where address 0x12_0000 defines the D-Flash partition for user access and address 0x12_0004 defines the buffer RAM partition for EEE operations.

Table 10-7. EEE Nonvolatile Information Register Fields

Global Address (EEEIFRON)	Size (Bytes)	Description
0x12_0000 – 0x12_0001	2	D-Flash User Partition (DFPART) Refer to Section 10.4.2.14, “Full Partition D-Flash Command”
0x12_0002 – 0x12_0003	2	D-Flash User Partition (duplicate(1))
0x12_0004 – 0x12_0005	2	Buffer RAM EEE Partition (ERPART) Refer to Section 10.4.2.14, “Full Partition D-Flash Command”
0x12_0006 – 0x12_0007	2	Buffer RAM EEE Partition (duplicate1)
0x12_0008 – 0x12_007F	120	Reserved

1. Duplicate value used if primary value generates a double bit fault when read during the reset sequence.

10.3.2 Register Descriptions

The Flash module contains a set of 20 control and status registers located between Flash module base + 0x0000 and 0x0013. A summary of the Flash module registers is given in Figure 10-4 with detailed descriptions in the following subsections.

CAUTION

Writes to any Flash register must be avoided while a Flash command is active (CCIF=0) to prevent corruption of Flash register contents and Memory Controller behavior.

Address & Name		7	6	5	4	3	2	1	0
0x0000 FCLKDIV	R	FDIVLD	FDIV6	FDIV5	FDIV4	FDIV3	FDIV2	FDIV1	FDIV0
	W								
0x0001 FSEC	R	KEYEN1	KEYEN0	RNV5	RNV4	RNV3	RNV2	SEC1	SEC0
	W								
0x0002 FCCOBIX	R	0	0	0	0	0	CCOBIX2	CCOBIX1	CCOBIX0
	W								
0x0003 FECCRIX	R	0	0	0	0	0	ECCRIX2	ECCRIX1	ECCRIX0
	W								
0x0004 FCNFG	R	CCIE	0	0	IGNSF	0	0	FDFD	FSFD
	W								

Figure 10-4. FTM128K2XF Register Summary

Address & Name		7	6	5	4	3	2	1	0
0x0005 FERCNFG	R	ERSERIE	PGMERIE	0	EPVIOLIE	ERSVIE1	ERSVIE0	DFDIE	SFDIE
	W								
0x0006 FSTAT	R	CCIF	0	ACCERR	FPVIOL	MGBUSY	RSVD	MGSTAT1	MGSTAT0
	W								
0x0007 FERSTAT	R	ERSERIF	PGMERIF	0	EPVIOLIF	ERSVIF1	ERSVIF0	DFDIF	SFDIF
	W								
0x0008 FPROT	R	FPOPEN	RNV6	FPHDIS	FPHS1	FPHS0	FPLDIS	FPLS1	FPLS0
	W								
0x0009 EPROT	R	EPOPEN	RNV6	RNV5	RNV4	EPDIS	EPS2	EPS1	EPS0
	W								
0x000A FCCOBHI	R	CCOB15	CCOB14	CCOB13	CCOB12	CCOB11	CCOB10	CCOB9	CCOB8
	W								
0x000B FCCOBLO	R	CCOB7	CCOB6	CCOB5	CCOB4	CCOB3	CCOB2	CCOB1	CCOB0
	W								
0x000C ETAGHI	R	ETAG15	ETAG14	ETAG13	ETAG12	ETAG11	ETAG10	ETAG9	ETAG8
	W								
0x000D ETAGLO	R	ETAG7	ETAG6	ETAG5	ETAG4	ETAG3	ETAG2	ETAG1	ETAG0
	W								
0x000E FECCRHI	R	ECCR15	ECCR14	ECCR13	ECCR12	ECCR11	ECCR10	ECCR9	ECCR8
	W								
0x000F FECCRLO	R	ECCR7	ECCR6	ECCR5	ECCR4	ECCR3	ECCR2	ECCR1	ECCR0
	W								
0x0010 FOPT	R	NV7	NV6	NV5	NV4	NV3	NV2	NV1	NV0
	W								
0x0011 FRSV0	R	0	0	0	0	0	0	0	0
	W								
0x0012 FRSV1	R	0	0	0	0	0	0	0	0
	W								

Figure 10-4. FTM128K2XF Register Summary (continued)

Address & Name		7	6	5	4	3	2	1	0
0x0013 FRSV2	R	0	0	0	0	0	0	0	0
	W								

= Unimplemented or Reserved

Figure 10-4. FTM128K2XF Register Summary (continued)

10.3.2.1 Flash Clock Divider Register (FCLKDIV)

The FCLKDIV register is used to control timed events in program and erase algorithms.

Offset Module Base + 0x0000

	7	6	5	4	3	2	1	0
R	FDIVLD	FDIV[6:0]						
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 10-5. Flash Clock Divider Register (FCLKDIV)

All bits in the FCLKDIV register are readable, bits 6–0 are write once and bit 7 is not writable.

Table 10-8. FCLKDIV Field Descriptions

Field	Description
7 FDIVLD	**Clock Divider Loaded** 0 FCLKDIV register has not been written 1 FCLKDIV register has been written since the last reset
6–0 FDIV[6:0]	**Clock Divider Bits** — FDIV[6:0] must be set to effectively divide OSCCLK down to generate an internal Flash clock, FCLK, with a target frequency of 1 MHz for use by the Flash module to control timed events during program and erase algorithms. Table 10-9 shows recommended values for FDIV[6:0] based on OSCCLK frequency. Please refer to Section 10.4.1, “Flash Command Operations,” for more information.

CAUTION

The FCLKDIV register should never be written while a Flash command is executing (CCIF=0). The FCLKDIV register is writable during the Flash reset sequence even though CCIF is clear.

Table 10-9. FDIV vs OSCCLK Frequency

OSCCLK Frequency (MHz)		FDIV[6:0]	OSCCLK Frequency (MHz)		FDIV[6:0]	OSCCLK Frequency (MHz)		FDIV[6:0]
MIN(1)	MAX(2)		MIN[1]	MAX[2]		MIN[1]	MAX[2]	
			33.60	34.65	0x20	67.20	68.25	0x40
1.60	2.10	0x01	34.65	35.70	0x21	68.25	69.30	0x41
2.40	3.15	0x02	35.70	36.75	0x22	69.30	70.35	0x42
3.20	4.20	0x03	36.75	37.80	0x23	70.35	71.40	0x43
4.20	5.25	0x04	37.80	38.85	0x24	71.40	72.45	0x44
5.25	6.30	0x05	38.85	39.90	0x25	72.45	73.50	0x45
6.30	7.35	0x06	39.90	40.95	0x26	73.50	74.55	0x46
7.35	8.40	0x07	40.95	42.00	0x27	74.55	75.60	0x47
8.40	9.45	0x08	42.00	43.05	0x28	75.60	76.65	0x48
9.45	10.50	0x09	43.05	44.10	0x29	76.65	77.70	0x49
10.50	11.55	0x0A	44.10	45.15	0x2A	77.70	78.75	0x4A
11.55	12.60	0x0B	45.15	46.20	0x2B	78.75	79.80	0x4B
12.60	13.65	0x0C	46.20	47.25	0x2C	79.80	80.85	0x4C
13.65	14.70	0x0D	47.25	48.30	0x2D	80.85	81.90	0x4D
14.70	15.75	0x0E	48.30	49.35	0x2E	81.90	82.95	0x4E
15.75	16.80	0x0F	49.35	50.40	0x2F	82.95	84.00	0x4F
16.80	17.85	0x10	50.40	51.45	0x30	84.00	85.05	0x50
17.85	18.90	0x11	51.45	52.50	0x31	85.05	86.10	0x51
18.90	19.95	0x12	52.50	53.55	0x32	86.10	87.15	0x52
19.95	21.00	0x13	53.55	54.60	0x33	87.15	88.20	0x53
21.00	22.05	0x14	54.60	55.65	0x34	88.20	89.25	0x54
22.05	23.10	0x15	55.65	56.70	0x35	89.25	90.30	0x55
23.10	24.15	0x16	56.70	57.75	0x36	90.30	91.35	0x56
24.15	25.20	0x17	57.75	58.80	0x37	91.35	92.40	0x57
25.20	26.25	0x18	58.80	59.85	0x38	92.40	93.45	0x58
26.25	27.30	0x19	59.85	60.90	0x39	93.45	94.50	0x59
27.30	28.35	0x1A	60.90	61.95	0x3A	94.50	95.55	0x5A
28.35	29.40	0x1B	61.95	63.00	0x3B	95.55	96.60	0x5B
29.40	30.45	0x1C	63.00	64.05	0x3C	96.60	97.65	0x5C
30.45	31.50	0x1D	64.05	65.10	0x3D	97.65	98.70	0x5D
31.50	32.55	0x1E	65.10	66.15	0x3E	98.70	99.75	0x5E
32.55	33.60	0x1F	66.15	67.20	0x3F	99.75	100.80	0x5F

1. FDIV shown generates an FCLK frequency of >0.8 MHz

2. FDIV shown generates an FCLK frequency of 1.05 MHz

10.3.2.2 Flash Security Register (FSEC)

The FSEC register holds all bits associated with the security of the MCU and Flash module.

Offset Module Base + 0x0001

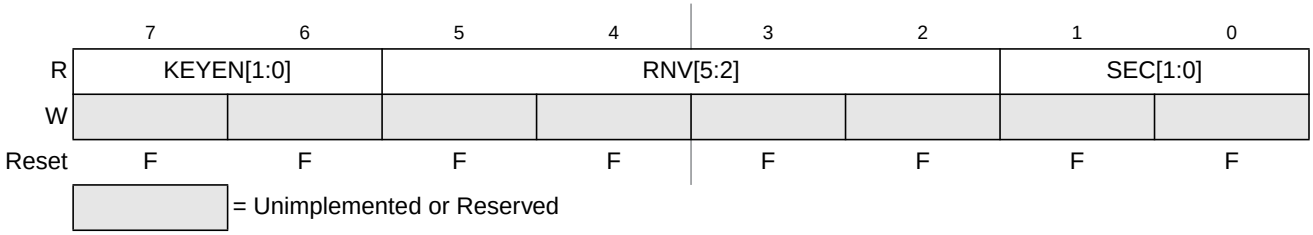


Figure 10-6. Flash Security Register (FSEC)

All bits in the FSEC register are readable but not writable.

During the reset sequence, the FSEC register is loaded with the contents of the Flash security byte in the Flash configuration field at global address 0x7F_FF0F located in P-Flash memory (see Table 10-3) as indicated by reset condition F in Figure 10-6. If a double bit fault is detected while reading the P-Flash phrase containing the Flash security byte during the reset sequence, all bits in the FSEC register will be set to leave the Flash module in a secured state with backdoor key access disabled.

Table 10-10. FSEC Field Descriptions

Field	Description
7–6 KEYEN[1:0]	**Backdoor Key Security Enable Bits** — The KEYEN[1:0] bits define the enabling of backdoor key access to the Flash module as shown in Table 10-11.
5–2 RNV[5:2}	**Reserved Nonvolatile Bits** — The RNV bits should remain in the erased state for future enhancements.
1–0 SEC[1:0]	**Flash Security Bits** — The SEC[1:0] bits define the security state of the MCU as shown in Table 10-12. If the Flash module is unsecured using backdoor key access, the SEC bits are forced to 10.

Table 10-11. Flash KEYEN States

KEYEN[1:0]	Status of Backdoor Key Access
00	DISABLED
01	DISABLED(1)
10	ENABLED
11	DISABLED

1. Preferred KEYEN state to disable backdoor key access.

Table 10-12. Flash Security States

SEC[1:0]	Status of Security
00	SECURED
01	SECURED(1)
10	UNSECURED
11	SECURED

1. Preferred SEC state to set MCU to secured state.

The security function in the Flash module is described in Section 10.5.

10.3.2.3 Flash CCOB Index Register (FCCOBIX)

The FCCOBIX register is used to index the FCCOB register for Flash memory operations.

Offset Module Base + 0x0002

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	CCOBIX[2:0]		
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 10-7. FCCOB Index Register (FCCOBIX)

CCOBIX bits are readable and writable while remaining bits read 0 and are not writable.

Table 10-13. FCCOBIX Field Descriptions

Field	Description
2–0 CCOBIX[1:0]	**Common Command Register Index**— The CCOBIX bits are used to select which word of the FCCOB register array is being read or written to. See Section 10.3.2.11, “Flash Common Command Object Register (FCCOB),” for more details.

10.3.2.4 Flash ECCR Index Register (FECCRIX)

The FECCRIX register is used to index the FECCR register for ECC fault reporting.

Offset Module Base + 0x0003

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	ECCRIX[2:0]		
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 10-8. FECCR Index Register (FECCRIX)

ECCRIX bits are readable and writable while remaining bits read 0 and are not writable.

Table 10-14. FECCRIX Field Descriptions

Field	Description
2-0 ECCRIX[2:0]	**ECC Error Register Index**— The ECCRIX bits are used to select which word of the FECCR register array is being read. See Section 10.3.2.13, “Flash ECC Error Results Register (FECCR),” for more details.

10.3.2.5 Flash Configuration Register (FCNFG)

The FCNFG register enables the Flash command complete interrupt and forces ECC faults on Flash array read access from the CPU or XGATE.

Offset Module Base + 0x0004

	7	6	5	4	3	2	1	0
R	CCIE	0	0	IGNSF	0	0	FDFD	FSFD
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 10-9. Flash Configuration Register (FCNFG)

CCIE, IGNSF, FDFD, and FSFD bits are readable and writable while remaining bits read 0 and are not writable.

Table 10-15. FCNFG Field Descriptions

Field	Description
7 CCIE	**Command Complete Interrupt Enable** — The CCIE bit controls interrupt generation when a Flash command has completed. 0 Command complete interrupt disabled 1 An interrupt will be requested whenever the CCIF flag in the FSTAT register is set (see Section 10.3.2.7)
4 IGNSF	**Ignore Single Bit Fault** — The IGNSF controls single bit fault reporting in the FERSTAT register (see Section 10.3.2.8). 0 All single bit faults detected during array reads are reported 1 Single bit faults detected during array reads are not reported and the single bit fault interrupt will not be generated

Table 10-15. FCNFG Field Descriptions (continued)

Field	Description
1 FDFD	**Force Double Bit Fault Detect** — The FDFD bit allows the user to simulate a double bit fault during Flash array read operations and check the associated interrupt routine. The FDFD bit is cleared by writing a 0 to FDFD. The FECCR registers will not be updated during the Flash array read operation with FDFD set unless an actual double bit fault is detected. 0 Flash array read operations will set the DFDIF flag in the FERSTAT register only if a double bit fault is detected 1 Any Flash array read operation will force the DFDIF flag in the FERSTAT register to be set (see Section 10.3.2.7) and an interrupt will be generated as long as the DFDIE interrupt enable in the FERCNFG register is set (see Section 10.3.2.6)
0 FSFD	**Force Single Bit Fault Detect** — The FSFD bit allows the user to simulate a single bit fault during Flash array read operations and check the associated interrupt routine. The FSFD bit is cleared by writing a 0 to FSFD. The FECCR registers will not be updated during the Flash array read operation with FSFD set unless an actual single bit fault is detected. 0 Flash array read operations will set the SFDIF flag in the FERSTAT register only if a single bit fault is detected 1 Flash array read operation will force the SFDIF flag in the FERSTAT register to be set (see Section 10.3.2.7) and an interrupt will be generated as long as the SFDIE interrupt enable in the FERCNFG register is set (see Section 10.3.2.6)

10.3.2.6 Flash Error Configuration Register (FERCNFG)

The FERCNFG register enables the Flash error interrupts for the FERSTAT flags.

Offset Module Base + 0x0005

	7	6	5	4	3	2	1	0
R	ERSERIE	PGMERIE	0	EPVIOLIE	ERSVIE1	ERSVIE0	DFDIE	SFDIE
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 10-10. Flash Error Configuration Register (FERCNFG)

All assigned bits in the FERCNFG register are readable and writable.

Table 10-16. FERCNFG Field Descriptions

Field	Description
7 ERSERIE	**EEE Erase Error Interrupt Enable** — The ERSERIE bit controls interrupt generation when a failure is detected during an EEE erase operation. 0 ERSERIF interrupt disabled 1 An interrupt will be requested whenever the ERSERIF flag is set (see Section 10.3.2.8)
6 PGMERIE	**EEE Program Error Interrupt Enable** — The PGMERIE bit controls interrupt generation when a failure is detected during an EEE program operation. 0 PGMERIF interrupt disabled 1 An interrupt will be requested whenever the PGMERIF flag is set (see Section 10.3.2.8)
4 EPVIOLIE	**EEE Protection Violation Interrupt Enable** — The EPVIOLIE bit controls interrupt generation when a protection violation is detected during a write to the buffer RAM EEE partition. 0 EPVIOLIF interrupt disabled 1 An interrupt will be requested whenever the EPVIOLIF flag is set (see Section 10.3.2.8)

Table 10-16. FERCNFG Field Descriptions (continued)

Field	Description
3 ERSVIE1	**EEE Error Type 1 Interrupt Enable** — The ERSVIE1 bit controls interrupt generation when a change state error is detected during an EEE operation. 0 ERSVIF1 interrupt disabled 1 An interrupt will be requested whenever the ERSVIF1 flag is set (see Section 10.3.2.8)
2 ERSVIE0	**EEE Error Type 0 Interrupt Enable** — The ERSVIE0 bit controls interrupt generation when a sector format error is detected during an EEE operation. 0 ERSVIF0 interrupt disabled 1 An interrupt will be requested whenever the ERSVIF0 flag is set (see Section 10.3.2.8)
1 DFDIE	**Double Bit Fault Detect Interrupt Enable** — The DFDIE bit controls interrupt generation when a double bit fault is detected during a Flash block read operation. 0 DFDIF interrupt disabled 1 An interrupt will be requested whenever the DFDIF flag is set (see Section 10.3.2.8)
0 SFDIE	**Single Bit Fault Detect Interrupt Enable** — The SFDIE bit controls interrupt generation when a single bit fault is detected during a Flash block read operation. 0 SFDIF interrupt disabled whenever the SFDIF flag is set (see Section 10.3.2.8) 1 An interrupt will be requested whenever the SFDIF flag is set (see Section 10.3.2.8)

10.3.2.7 Flash Status Register (FSTAT)

The FSTAT register reports the operational status of the Flash module.

Offset Module Base + 0x0006

	7	6	5	4	3	2	1	0
R	CCIF	0	ACCERR	FPVIOL	MGBUSY	RSVD	MGSTAT[1:0]	
W								
Reset	1	0	0	0	0	0	0(1)	0[1]

= Unimplemented or Reserved

Figure 10-11. Flash Status Register (FSTAT)

1. Reset value can deviate from the value shown if a double bit fault is detected during the reset sequence (see Section 10.6).

CCIF, ACCERR, and FPVIOL bits are readable and writable, MGBUSY and MGSTAT bits are readable but not writable, while remaining bits read 0 and are not writable.

Table 10-17. FSTAT Field Descriptions

Field	Description
7 CCIF	**Command Complete Interrupt Flag** — The CCIF flag indicates that a Flash command has completed. The CCIF flag is cleared by writing a 1 to CCIF to launch a command and CCIF will stay low until command completion or command violation. 0 Flash command in progress 1 Flash command has completed
5 ACCERR	**Flash Access Error Flag** — The ACCERR bit indicates an illegal access has occurred to the Flash memory caused by either a violation of the command write sequence (see Section 10.4.1.2) or issuing an illegal Flash command or when errors are encountered while initializing the EEE buffer ram during the reset sequence. While ACCERR is set, the CCIF flag cannot be cleared to launch a command. The ACCERR bit is cleared by writing a 1 to ACCERR. Writing a 0 to the ACCERR bit has no effect on ACCERR. 0 No access error detected 1 Access error detected
4 FPVIOL	**Flash Protection Violation Flag** —The FPVIOL bit indicates an attempt was made to program or erase an address in a protected area of P-Flash memory during a command write sequence. The FPVIOL bit is cleared by writing a 1 to FPVIOL. Writing a 0 to the FPVIOL bit has no effect on FPVIOL. While FPVIOL is set, it is not possible to launch a command or start a command write sequence. 0 No protection violation detected 1 Protection violation detected
3 MGBUSY	**Memory Controller Busy Flag** — The MGBUSY flag reflects the active state of the Memory Controller. 0 Memory Controller is idle 1 Memory Controller is busy executing a Flash command (CCIF = 0) or is handling internal EEE operations
2 RSVD	**Reserved Bit** — This bit is reserved and always reads 0.
1–0 MGSTAT[1:0]	**Memory Controller Command Completion Status Flag** — One or more MGSTAT flag bits are set if an error is detected during execution of a Flash command or during the Flash reset sequence. See Section 10.4.2, “Flash Command Description,” and Section 10.6, “Initialization” for details.

10.3.2.8 Flash Error Status Register (FERSTAT)

The FERSTAT register reflects the error status of internal Flash operations.

Offset Module Base + 0x0007

	7	6	5	4	3	2	1	0
R	ERSERIF	PGMERIF	0	EPVIOLIF	ERSVIF1	ERSVIF0	DFDIF	SFDIF
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 10-12. Flash Error Status Register (FERSTAT)

All flags in the FERSTAT register are readable and only writable to clear the flag.

Table 10-18. FERSTAT Field Descriptions

Field	Description
7 ERSERIF	**EEE Erase Error Interrupt Flag** — The setting of the ERSERIF flag occurs due to an error in a Flash erase command that resulted in the erase operation not being successful during EEE operations. The ERSERIF flag is cleared by writing a 1 to ERSERIF. Writing a 0 to the ERSERIF flag has no effect on ERSERIF. While ERSERIF is set, it is possible to write to the buffer RAM EEE partition but the data written will not be transferred to the D-Flash EEE partition. 0 Erase command successfully completed on the D-Flash EEE partition 1 Erase command failed on the D-Flash EEE partition
6 PGMERIF	**EEE Program Error Interrupt Flag** — The setting of the PGMERIF flag occurs due to an error in a Flash program command that resulted in the program operation not being successful during EEE operations. The PGMERIF flag is cleared by writing a 1 to PGMERIF. Writing a 0 to the PGMERIF flag has no effect on PGMERIF. While PGMERIF is set, it is possible to write to the buffer RAM EEE partition but the data written will not be transferred to the D-Flash EEE partition. 0 Program command successfully completed on the D-Flash EEE partition 1 Program command failed on the D-Flash EEE partition
4 EPVIOLIF	**EEE Protection Violation Interrupt Flag** —The setting of the EPVIOLIF flag indicates an attempt was made to write to a protected area of the buffer RAM EEE partition. The EPVIOLIF flag is cleared by writing a 1 to EPVIOLIF. Writing a 0 to the EPVIOLIF flag has no effect on EPVIOLIF. While EPVIOLIF is set, it is possible to write to the buffer RAM EEE partition as long as the address written to is not in a protected area. 0 No EEE protection violation 1 EEE protection violation detected
3 ERSVIF1	**EEE Error Interrupt 1 Flag** —The setting of the ERSVIF1 flag indicates that the memory controller was unable to change the state of a D-Flash EEE sector. The ERSVIF1 flag is cleared by writing a 1 to ERSVIF1. Writing a 0 to the ERSVIF1 flag has no effect on ERSVIF1. While ERSVIF1 is set, it is possible to write to the buffer RAM EEE partition but the data written will not be transferred to the D-Flash EEE partition. 0 No EEE sector state change error detected 1 EEE sector state change error detected
2 ERSVIF0	**EEE Error Interrupt 0 Flag** —The setting of the ERSVIF0 flag indicates that the memory controller was unable to format a D-Flash EEE sector for EEE use. The ERSVIF0 flag is cleared by writing a 1 to ERSVIF0. Writing a 0 to the ERSVIF0 flag has no effect on ERSVIF0. While ERSVIF0 is set, it is possible to write to the buffer RAM EEE partition but the data written will not be transferred to the D-Flash EEE partition. 0 No EEE sector format error detected 1 EEE sector format error detected
1 DFDIF	**Double Bit Fault Detect Interrupt Flag** — The setting of the DFDIF flag indicates that a double bit fault was detected in the stored parity and data bits during a Flash array read operation or that a Flash array read operation was attempted on a Flash block that was under a Flash command operation. The DFDIF flag is cleared by writing a 1 to DFDIF. Writing a 0 to DFDIF has no effect on DFDIF. 0 No double bit fault detected 1 Double bit fault detected or an invalid Flash array read operation attempted
0 SFDIF	**Single Bit Fault Detect Interrupt Flag** — With the IGNSF bit in the FCNFG register clear, the SFDIF flag indicates that a single bit fault was detected in the stored parity and data bits during a Flash array read operation or that a Flash array read operation was attempted on a Flash block that was under a Flash command operation. The SFDIF flag is cleared by writing a 1 to SFDIF. Writing a 0 to SFDIF has no effect on SFDIF. 0 No single bit fault detected 1 Single bit fault detected and corrected or an invalid Flash array read operation attempted

10.3.2.9 P-Flash Protection Register (FPROT)

The FPROT register defines which P-Flash sectors are protected against program and erase operations.

Offset Module Base + 0x0008

	7	6	5	4	3	2	1	0
R	FPOPEN	RNV6	FPHDIS	FPHS[1:0]		FPLDIS	FPLS[1:0]	
W								
Reset	F	F	F	F	F	F	F	F

= Unimplemented or Reserved

Figure 10-13. Flash Protection Register (FPROT)

The (unreserved) bits of the FPROT register are writable with the restriction that the size of the protected region can only be increased (see Section 10.3.2.9.1, “P-Flash Protection Restrictions,” and Table 10-23).

During the reset sequence, the FPROT register is loaded with the contents of the P-Flash protection byte in the Flash configuration field at global address 0x7F_FF0C located in P-Flash memory (see Table 10-3) as indicated by reset condition ‘F’ in Figure 10-13. To change the P-Flash protection that will be loaded during the reset sequence, the upper sector of the P-Flash memory must be unprotected, then the P-Flash protection byte must be reprogrammed. If a double bit fault is detected while reading the P-Flash phrase containing the P-Flash protection byte during the reset sequence, the FPOPEN bit will be cleared and remaining bits in the FPROT register will be set to leave the P-Flash memory fully protected.

Trying to alter data in any protected area in the P-Flash memory will result in a protection violation error and the FPVIOL bit will be set in the FSTAT register. The block erase of a P-Flash block is not possible if any of the P-Flash sectors contained in the same P-Flash block are protected.

Table 10-19. FPROT Field Descriptions

Field	Description
7 FPOPEN	**Flash Protection Operation Enable** — The FPOPEN bit determines the protection function for program or erase operations as shown in Table 10-20 for the P-Flash block. 0 When FPOPEN is clear, the FPHDIS and FPLDIS bits define unprotected address ranges as specified by the corresponding FPHS and FPLS bits 1 When FPOPEN is set, the FPHDIS and FPLDIS bits enable protection for the address range specified by the corresponding FPHS and FPLS bits
6 RNV[6]	**Reserved Nonvolatile Bit** — The RNV bit should remain in the erased state for future enhancements.
5 FPHDIS	**Flash Protection Higher Address Range Disable** — The FPHDIS bit determines whether there is a protected/unprotected area in a specific region of the P-Flash memory ending with global address 0x7F_FFFF. 0 Protection/Unprotection enabled 1 Protection/Unprotection disabled
4–3 FPHS[1:0]	**Flash Protection Higher Address Size** — The FPHS bits determine the size of the protected/unprotected area in P-Flash memory as shown inTable 10-21. The FPHS bits can only be written to while the FPHDIS bit is set.
2 FPLDIS	**Flash Protection Lower Address Range Disable** — The FPLDIS bit determines whether there is a protected/unprotected area in a specific region of the P-Flash memory beginning with global address 0x7F_8000. 0 Protection/Unprotection enabled 1 Protection/Unprotection disabled
1–0 FPLS[1:0]	**Flash Protection Lower Address Size** — The FPLS bits determine the size of the protected/unprotected area in P-Flash memory as shown in Table 10-22. The FPLS bits can only be written to while the FPLDIS bit is set.

Table 10-20. P-Flash Protection Function

FPOPEN	FPHDIS	FPLDIS	Function[1]
1	1	1	No P-Flash Protection
1	1	0	Protected Low Range
1	0	1	Protected High Range
1	0	0	Protected High and Low Ranges
0	1	1	Full P-Flash Memory Protected
0	1	0	Unprotected Low Range
0	0	1	Unprotected High Range
0	0	0	Unprotected High and Low Ranges

1. For range sizes, refer to Table 10-21 and Table 10-22.

Table 10-21. P-Flash Protection Higher Address Range

FPHS[1:0]	Global Address Range	Protected Size
00	0x7F_F800–0x7F_FFFF	2 Kbytes
01	0x7F_F000–0x7F_FFFF	4 Kbytes
10	0x7F_E000–0x7F_FFFF	8 Kbytes
11	0x7F_C000–0x7F_FFFF	16 Kbytes

Table 10-22. P-Flash Protection Lower Address Range

FPLS[1:0]	Global Address Range	Protected Size
00	0x7F_8000–0x7F_83FF	1 Kbyte
01	0x7F_8000–0x7F_87FF	2 Kbytes
10	0x7F_8000–0x7F_8FFF	4 Kbytes
11	0x7F_8000–0x7F_9FFF	8 Kbytes

All possible P-Flash protection scenarios are shown in Figure 10-14. Although the protection scheme is loaded from the Flash memory at global address 0x7F_FF0C during the reset sequence, it can be changed by the user. The P-Flash protection scheme can be used by applications requiring reprogramming in single chip mode while providing as much protection as possible if reprogramming is not required.

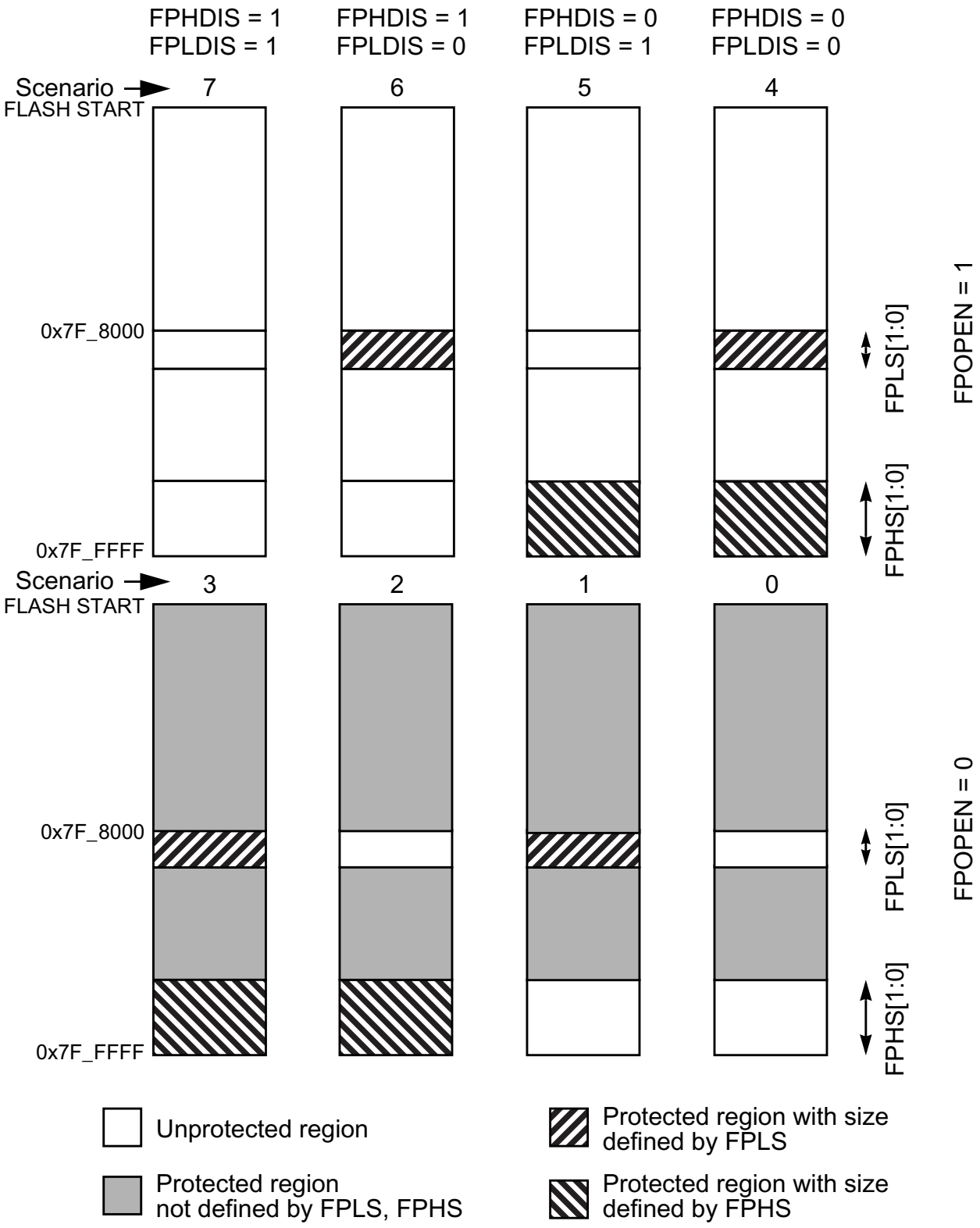


Figure 10-14. P-Flash Protection Scenarios

10.3.2.9.1 P-Flash Protection Restrictions

The general guideline is that P-Flash protection can only be added and not removed. Table 10-23 specifies all valid transitions between P-Flash protection scenarios. Any attempt to write an invalid scenario to the FPROT register will be ignored. The contents of the FPROT register reflect the active protection scenario. See the FPHS and FPLS bit descriptions for additional restrictions.

Table 10-23. P-Flash Protection Scenario Transitions

From Protection Scenario	To Protection Scenario[1]							
	0	1	2	3	4	5	6	7
0	X	X	X	X				
1		X		X				
2			X	X				
3				X				
4				X	X			
5			X	X	X	X		
6		X		X	X		X	
7	X	X	X	X	X	X	X	X

1. Allowed transitions marked with X, see Figure 10-14 for a definition of the scenarios.

10.3.2.10 EEE Protection Register (EPROT)

The EPROT register defines which buffer RAM EEE partition areas are protected against writes.

Offset Module Base + 0x0009

	7	6	5	4	3	2	1	0
R	EPOPEN	RNV[6:4]			EPDIS	EPS[2:0]		
W								
Reset	F	F	F	F	F	F	F	F

= Unimplemented or Reserved

Figure 10-15. EEE Protection Register (EPROT)

All bits in the EPROT register are readable and writable except for RNV[6:4] which are only readable. The EPOPEN and EPDIS bits can only be written to the protected state. The EPS bits can be written anytime until the EPDIS bit is cleared. If the EPOPEN bit is cleared, the state of the EPDIS and EPS bits is irrelevant.

During the reset sequence, the EPROT register is loaded from the EEE protection byte in the Flash configuration field at global address 0x7F_FF0D located in P-Flash memory (see Table 10-3) as indicated by reset condition F in Figure 10-15. To change the EEE protection that will be loaded during the reset sequence, the P-Flash sector containing the EEE protection byte must be unprotected, then the EEE protection byte must be programmed. If a double bit fault is detected while reading the P-Flash phrase

containing the EEE protection byte during the reset sequence, the EPOPEN bit will be cleared and remaining bits in the EPROT register will be set to leave the buffer RAM EEE partition fully protected.

Trying to write data to any protected area in the buffer RAM EEE partition will result in a protection violation error and the EPVIOLIF flag will be set in the FERSTAT register. Trying to write data to any protected area in the buffer RAM partitioned for user access will not be prevented and the EPVIOLIF flag in the FERSTAT register will not set.

Table 10-24. EPROT Field Descriptions

Field	Description
7 EPOPEN	**Enables writes to the Buffer RAM partitioned for EEE** 0 The entire buffer RAM EEE partition is protected from writes 1 Unprotected buffer RAM EEE partition areas are enabled for writes
6–4 RNV[6:4]	**Reserved Nonvolatile Bits** — The RNV bits should remain in the erased state for future enhancements
3 EPDIS	**Buffer RAM Protection Address Range Disable** — The EPDIS bit determines whether there is a protected area in a specific region of the buffer RAM EEE partition. 0 Protection enabled 1 Protection disabled
2–0 EPS[2:0]	**Buffer RAM Protection Size** — The EPS[2:0] bits determine the size of the protected area in the buffer RAM EEE partition as shown inTable 10-21. The EPS bits can only be written to while the EPDIS bit is set.

Table 10-25. Buffer RAM EEE Partition Protection Address Range

EPS[2:0]	Global Address Range	Protected Size
000	0x13_FFC0 – 0x13_FFFF	64 bytes
001	0x13_FF80 – 0x13_FFFF	128 bytes
010	0x13_FF40 – 0x13_FFFF	192 bytes
011	0x13_FF00 – 0x13_FFFF	256 bytes
100	0x13_FEC0 – 0x13_FFFF	320 bytes
101	0x13_FE80 – 0x13_FFFF	384 bytes
110	0x13_FE40 – 0x13_FFFF	448 bytes
111	0x13_FE00 – 0x13_FFFF	512 bytes

10.3.2.11 Flash Common Command Object Register (FCCOB)

The FCCOB is an array of six words addressed via the CCOBIX index found in the FCCOBIX register. Byte wide reads and writes are allowed to the FCCOB register.

Offset Module Base + 0x000A

	7	6	5	4	3	2	1	0
R W	CCOB[15:8]							
Reset	0	0	0	0	0	0	0	0

Figure 10-16. Flash Common Command Object High Register (FCCOBHI)

Offset Module Base + 0x000B

	7	6	5	4	3	2	1	0
R W	CCOB[7:0]							
Reset	0	0	0	0	0	0	0	0

Figure 10-17. Flash Common Command Object Low Register (FCCOBLO)

10.3.2.11.1 FCCOB - NVM Command Mode

NVM command mode uses the indexed FCCOB register to provide a command code and its relevant parameters to the Memory Controller. The user first sets up all required FCCOB fields and then initiates the command's execution by writing a 1 to the CCIF bit in the FSTAT register (a 1 written by the user clears the CCIF command completion flag to 0). When the user clears the CCIF bit in the FSTAT register all FCCOB parameter fields are locked and cannot be changed by the user until the command completes (as evidenced by the Memory Controller returning CCIF to 1). Some commands return information to the FCCOB register array.

The generic format for the FCCOB parameter fields in NVM command mode is shown in Table 10-26. The return values are available for reading after the CCIF flag in the FSTAT register has been returned to 1 by the Memory Controller. Writes to the unimplemented parameter fields (CCOBIX = 110 and CCOBIX = 111) are ignored with reads from these fields returning 0x0000.

Table 10-26 shows the generic Flash command format. The high byte of the first word in the CCOB array contains the command code, followed by the parameters for this specific Flash command. For details on the FCCOB settings required by each command, see the Flash command descriptions in Section 10.4.2.

Table 10-26. FCCOB - NVM Command Mode (Typical Usage)

CCOBIX[2:0]	Byte	FCCOB Parameter Fields (NVM Command Mode)
000	HI	FCMD[7:0] defining Flash command
	LO	0, Global address [22:16]
001	HI	Global address [15:8]
	LO	Global address [7:0]
010	HI	Data 0 [15:8]
	LO	Data 0 [7:0]

Table 10-26. FCCOB - NVM Command Mode (Typical Usage)

CCOBIX[2:0]	Byte	FCCOB Parameter Fields (NVM Command Mode)
011	HI	Data 1 [15:8]
	LO	Data 1 [7:0]
100	HI	Data 2 [15:8]
	LO	Data 2 [7:0]
101	HI	Data 3 [15:8]
	LO	Data 3 [7:0]

10.3.2.12 EEE Tag Counter Register (ETAG)

The ETAG register contains the number of outstanding words in the buffer RAM EEE partition that need to be programmed into the D-Flash EEE partition. The ETAG register is decremented prior to the related tagged word being programmed into the D-Flash EEE partition. All tagged words have been programmed into the D-Flash EEE partition once all bits in the ETAG register read 0 and the MGBUSY flag in the FSTAT register reads 0.

Offset Module Base + 0x000C

	7	6	5	4	3	2	1	0
R	ETAG[15:8]							
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 10-18. EEE Tag Counter High Register (ETAGHI)

Offset Module Base + 0x000D

	7	6	5	4	3	2	1	0
R	ETAG[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 10-19. EEE Tag Counter Low Register (ETAGLO)

All ETAG bits are readable but not writable and are cleared by the Memory Controller.

10.3.2.13 Flash ECC Error Results Register (FECCR)

The FECCR registers contain the result of a detected ECC fault for both single bit and double bit faults. The FECCR register provides access to several ECC related fields as defined by the ECCRIX index bits in the FECCRIX register (see Section 10.3.2.4). Once ECC fault information has been stored, no other

fault information will be recorded until the specific ECC fault flag has been cleared. In the event of simultaneous ECC faults, the priority for fault recording is:

1. Double bit fault over single bit fault
2. CPU over XGATE

Offset Module Base + 0x000E

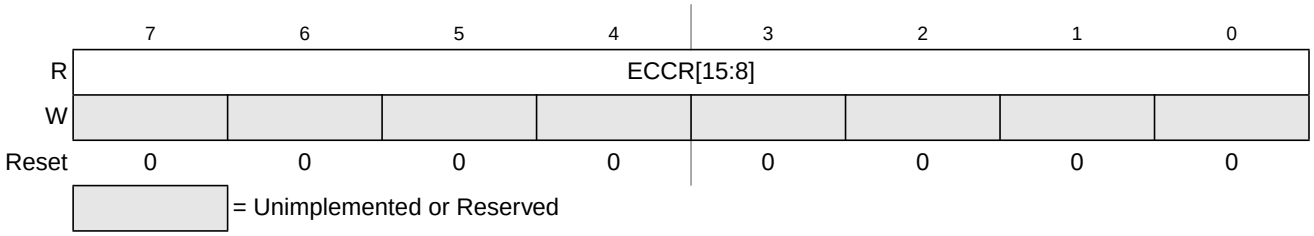


Figure 10-20. Flash ECC Error Results High Register (FECCRHI)

Offset Module Base + 0x000F

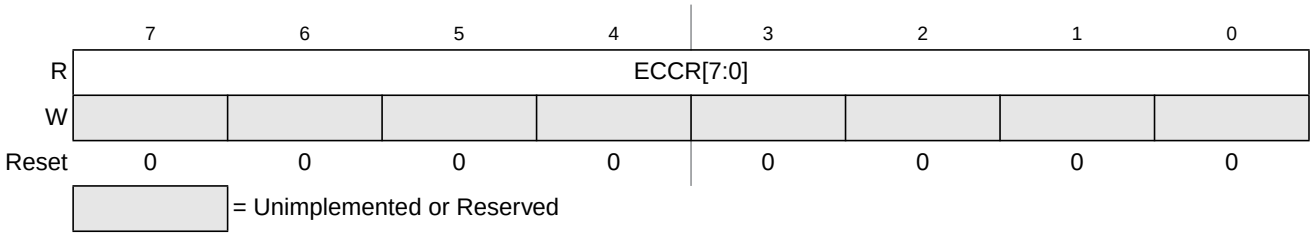


Figure 10-21. Flash ECC Error Results Low Register (FECCRLO)

All FECCR bits are readable but not writable.

Table 10-27. FECCR Index Settings

ECCRIX[2:0]	FECCR Register Content		
	Bits [15:8]	**Bit[7]**	**Bits[6:0]**
000	Parity bits read from Flash block	CPU or XGATE source identity	Global address [22:16]
001	Global address [15:0]		
010	Data 0 [15:0]		
011	Data 1 [15:0] (P-Flash only)		
100	Data 2 [15:0] (P-Flash only)		
101	Data 3 [15:0] (P-Flash only)		
110	Not used, returns 0x0000 when read		
111	Not used, returns 0x0000 when read		

Table 10-28. FECCR Index=000 Bit Descriptions

Field	Description
15:8 PAR[7:0]	**ECC Parity Bits** — Contains the 8 parity bits from the 72 bit wide P-Flash data word or the 6 parity bits, allocated to PAR[5:0], from the 22 bit wide D-Flash word with PAR[7:6]=00.
7 XBUS01	**Bus Source Identifier** — The XBUS01 bit determines whether the ECC error was caused by a read access from the CPU or XGATE. 0 ECC Error happened on the CPU access 1 ECC Error happened on the XGATE access
6–0 GADDR[22:16]	**Global Address** — The GADDR[22:16] field contains the upper seven bits of the global address having caused the error.

The P-Flash word addressed by ECCRIX = 001 contains the lower 16 bits of the global address. The following four words addressed by ECCRIX = 010 to 101 contain the 64-bit wide data phrase. The four data words and the parity byte are the uncorrected data read from the P-Flash block.

The D-Flash word addressed by ECCRIX = 001 contains the lower 16 bits of the global address. The uncorrected 16-bit data word is addressed by ECCRIX = 010.

10.3.2.14 Flash Option Register (FOPT)

The FOPT register is the Flash option register.

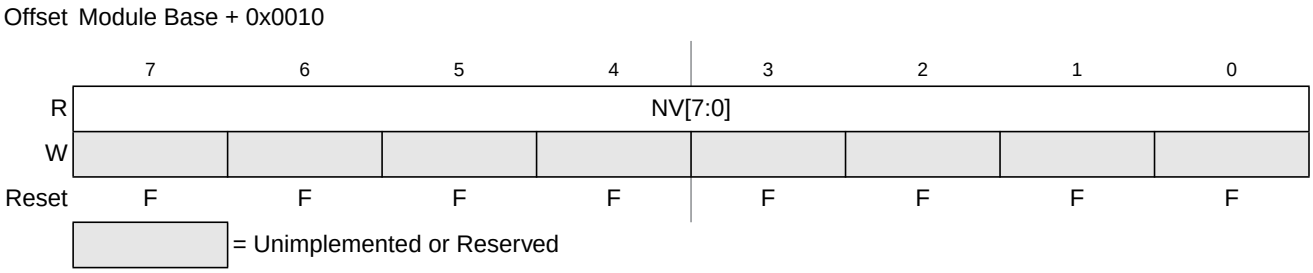


Figure 10-22. Flash Option Register (FOPT)

All bits in the FOPT register are readable but are not writable.

During the reset sequence, the FOPT register is loaded from the Flash nonvolatile byte in the Flash configuration field at global address 0x7F_FF0E located in P-Flash memory (see Table 10-3) as indicated by reset condition F in Figure 10-22. If a double bit fault is detected while reading the P-Flash phrase containing the Flash nonvolatile byte during the reset sequence, all bits in the FOPT register will be set.

Table 10-29. FOPT Field Descriptions

Field	Description
7–0 NV[7:0]	**Nonvolatile Bits** — The NV[7:0] bits are available as nonvolatile bits. Refer to the device user guide for proper use of the NV bits.

10.3.2.15 Flash Reserved0 Register (FRSV0)

This Flash register is reserved for factory testing.

Offset Module Base + 0x0011

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 10-23. Flash Reserved0 Register (FRSV0)

All bits in the FRSV0 register read 0 and are not writable.

10.3.2.16 Flash Reserved1 Register (FRSV1)

This Flash register is reserved for factory testing.

Offset Module Base + 0x0012

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 10-24. Flash Reserved1 Register (FRSV1)

All bits in the FRSV1 register read 0 and are not writable.

10.3.2.17 Flash Reserved2 Register (FRSV2)

This Flash register is reserved for factory testing.

Offset Module Base + 0x0013

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 10-25. Flash Reserved2 Register (FRSV2)

All bits in the FRSV2 register read 0 and are not writable.

10.4 Functional Description

10.4.1 Flash Command Operations

Flash command operations are used to modify Flash memory contents or configure module resources for EEE operation.

The next sections describe:

- How to write the FCLKDIV register that is used to generate a time base (FCLK) derived from OSCCLK for Flash program and erase command operations
- The command write sequence used to set Flash command parameters and launch execution
- Valid Flash commands available for execution

10.4.1.1 Writing the FCLKDIV Register

Prior to issuing any Flash program or erase command after a reset, the user is required to write the FCLKDIV register to divide OSCCLK down to a target FCLK of 1 MHz. Table 10-9 shows recommended values for the FDIV field based on OSCCLK frequency.

NOTE

Programming or erasing the Flash memory cannot be performed if the bus clock runs at less than 1 MHz. Setting FDIV too high can destroy the Flash memory due to overstress. Setting FDIV too low can result in incomplete programming or erasure of the Flash memory cells.

When the FCLKDIV register is written, the FDIVLD bit is set automatically. If the FDIVLD bit is 0, the FCLKDIV register has not been written since the last reset. If the FCLKDIV register has not been written, any Flash program or erase command loaded during a command write sequence will not execute and the ACCERR bit in the FSTAT register will set.

10.4.1.2 Command Write Sequence

The Memory Controller will launch all valid Flash commands entered using a command write sequence.

Before launching a command, the ACCERR and FPVIOL bits in the FSTAT register must be clear (see Section 10.3.2.7) and the CCIF flag should be tested to determine the status of the current command write sequence. If CCIF is 0, the previous command write sequence is still active, a new command write sequence cannot be started, and all writes to the FCCOB register are ignored.

CAUTION

Writes to any Flash register must be avoided while a Flash command is active (CCIF=0) to prevent corruption of Flash register contents and Memory Controller behavior.

10.4.1.2.1 Define FCCOB Contents

The FCCOB parameter fields must be loaded with all required parameters for the Flash command being executed. Access to the FCCOB parameter fields is controlled via the CCOBIX bits in the FCCOBIX register (see Section 10.3.2.3).

The contents of the FCCOB parameter fields are transferred to the Memory Controller when the user clears the CCIF command completion flag in the FSTAT register (writing 1 clears the CCIF to 0). The CCIF flag will remain clear until the Flash command has completed. Upon completion, the Memory Controller will return CCIF to 1 and the FCCOB register will be used to communicate any results. The flow for a generic command write sequence is shown in Figure 10-26.

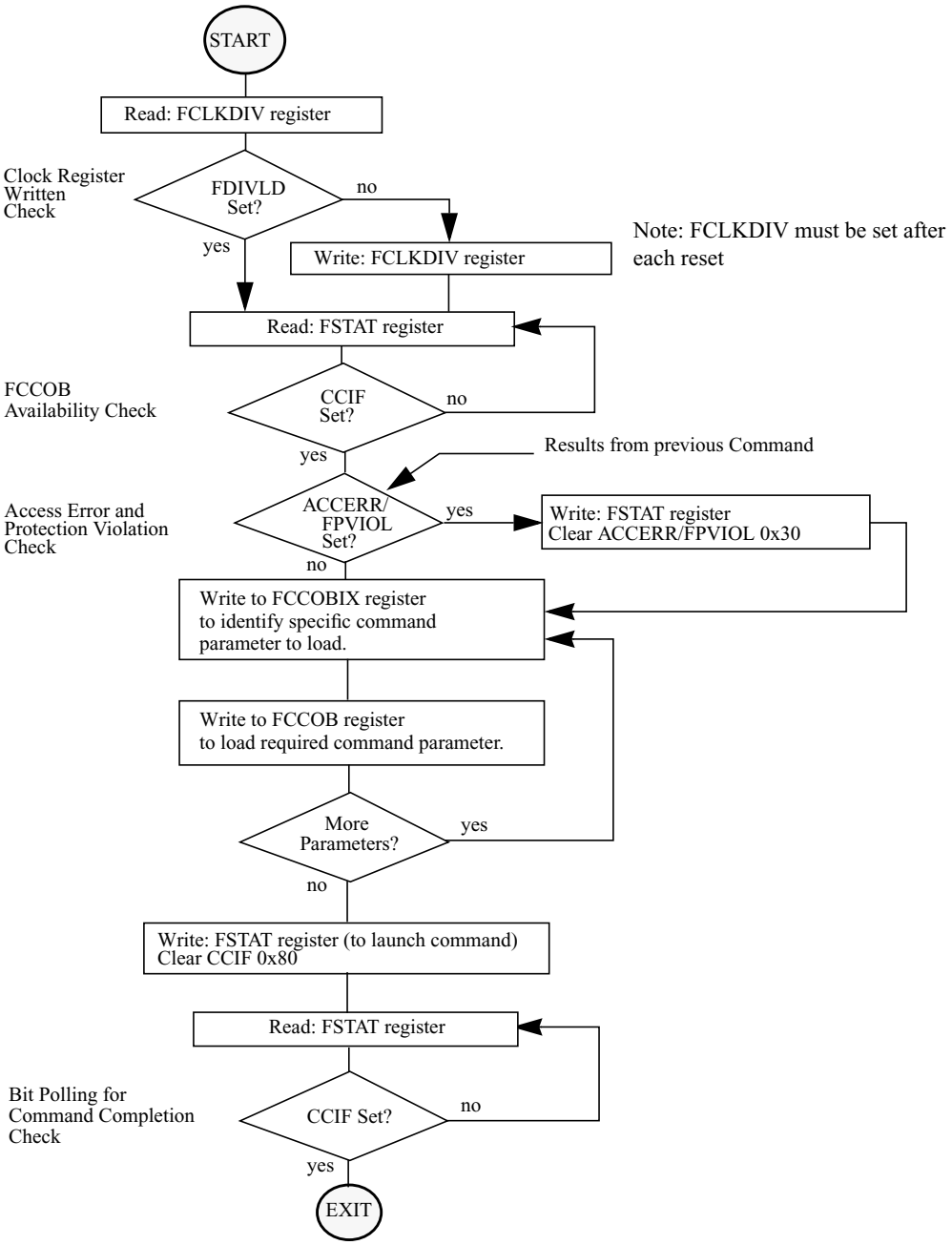


Figure 10-26. Generic Flash Command Write Sequence Flowchart

10.4.1.3 Valid Flash Module Commands

Table 10-30. Flash Commands by Mode

FCMD	Command	Unsecured				Secured			
		NS (1)	NX (2)	SS(3)	ST(4)	NS (5)	NX (6)	SS(7)	ST(8)
0x01	Erase Verify All Blocks	*	*	*	*	*	*	*	*
0x02	Erase Verify Block	*	*	*	*	*	*	*	*
0x03	Erase Verify P-Flash Section	*	*	*	*	*			
0x04	Read Once	*	*	*	*	*			
0x05	Reserved	*	*	*	*	*			
0x06	Program P-Flash	*	*	*	*	*			
0x07	Program Once	*	*	*	*	*			
0x08	Erase All Blocks			*	*			*	*
0x09	Erase P-Flash Block	*	*	*	*	*			
0x0A	Erase P-Flash Sector	*	*	*	*	*			
0x0B	Unsecure Flash			*	*			*	*
0x0C	Verify Backdoor Access Key	*				*			
0x0D	Set User Margin Level	*	*	*	*	*			
0x0E	Set Field Margin Level			*	*				
0x0F	Full Partition D-Flash			*	*				
0x10	Erase Verify D-Flash Section	*	*	*	*	*			
0x11	Program D-Flash	*	*	*	*	*			
0x12	Erase D-Flash Sector	*	*	*	*	*			
0x13	Enable EEPROM Emulation	*	*	*	*	*	*	*	*
0x14	Disable EEPROM Emulation	*	*	*	*	*	*	*	*
0x15	EEPROM Emulation Query	*	*	*	*	*	*	*	*
0x20	Partition D-Flash	*	*	*	*	*	*	*	*

1. Unsecured Normal Single Chip mode.
2. Unsecured Normal Expanded mode.
3. Unsecured Special Single Chip mode.
4. Unsecured Special Mode.
5. Secured Normal Single Chip mode.
6. Secured Normal Expanded mode.
7. Secured Special Single Chip mode.
8. Secured Special Mode.

10.4.1.4 P-Flash Commands

Table 10-31 summarizes the valid P-Flash commands along with the effects of the commands on the P-Flash block and other resources within the Flash module.

Table 10-31. P-Flash Commands

FCMD	Command	Function on P-Flash Memory
0x01	Erase Verify All Blocks	Verify that all P-Flash (and D-Flash) blocks are erased.
0x02	Erase Verify Block	Verify that a P-Flash block is erased.
0x03	Erase Verify P-Flash Section	Verify that a given number of words starting at the address provided are erased.
0x04	Read Once	Read a dedicated 64 byte field in the nonvolatile information register in P-Flash block 0 that was previously programmed using the Program Once command.
0x06	Program P-Flash	Program a phrase in a P-Flash block.
0x07	Program Once	Program a dedicated 64 byte field in the nonvolatile information register in P-Flash block 0 that is allowed to be programmed only once.
0x08	Erase All Blocks	Erase all P-Flash (and D-Flash) blocks. An erase of all Flash blocks is only possible when the FPLDIS, FPHDIS, and FPOPEN bits in the FPROT register and the EPDIS and EPOPEN bits in the EPROT register are set prior to launching the command.
0x09	Erase P-Flash Block	Erase a single P-Flash block. An erase of the full P-Flash block is only possible when FPLDIS, FPHDIS and FPOPEN bits in the FPROT register are set prior to launching the command.
0x0A	Erase P-Flash Sector	Erase all bytes in a P-Flash sector.
0x0B	Unsecure Flash	Supports a method of releasing MCU security by erasing all P-Flash (and D-Flash) blocks and verifying that all P-Flash (and D-Flash) blocks are erased.
0x0C	Verify Backdoor Access Key	Supports a method of releasing MCU security by verifying a set of security keys.
0x0D	Set User Margin Level	Specifies a user margin read level for all P-Flash blocks.
0x0E	Set Field Margin Level	Specifies a field margin read level for all P-Flash blocks (special modes only).

10.4.1.5 D-Flash and EEE Commands

Table 10-32 summarizes the valid D-Flash and EEE commands along with the effects of the commands on the D-Flash block and EEE operation.

Table 10-32. D-Flash Commands

FCMD	Command	Function on D-Flash Memory
0x01	Erase Verify All Blocks	Verify that all D-Flash (and P-Flash) blocks are erased.
0x02	Erase Verify Block	Verify that the D-Flash block is erased.
0x08	Erase All Blocks	Erase all D-Flash (and P-Flash) blocks. An erase of all Flash blocks is only possible when the FPLDIS, FPHDIS, and FPOPEN bits in the FPROT register and the EPDIS and EPOPEN bits in the EPROT register are set prior to launching the command.

Table 10-32. D-Flash Commands

FCMD	Command	Function on D-Flash Memory
0x0B	Unsecure Flash	Supports a method of releasing MCU security by erasing all D-Flash (and P-Flash) blocks and verifying that all D-Flash (and P-Flash) blocks are erased.
0x0D	Set User Margin Level	Specifies a user margin read level for the D-Flash block.
0x0E	Set Field Margin Level	Specifies a field margin read level for the D-Flash block (special modes only).
0x0F	Full Partition D-Flash	Erase the D-Flash block and partition an area of the D-Flash block for user access.
0x10	Erase Verify D-Flash Section	Verify that a given number of words starting at the address provided are erased.
0x11	Program D-Flash	Program up to four words in the D-Flash block.
0x12	Erase D-Flash Sector	Erase all bytes in a sector of the D-Flash block.
0x13	Enable EEPROM Emulation	Enable EEPROM emulation where writes to the buffer RAM EEE partition will be copied to the D-Flash EEE partition.
0x14	Disable EEPROM Emulation	Suspend all current erase and program activity related to EEPROM emulation but leave current EEE tags set.
0x15	EEPROM Emulation Query	Returns EEE partition and status variables.
0x20	Partition D-Flash	Partition an area of the D-Flash block for user access.

10.4.2 Flash Command Description

This section provides details of all available Flash commands launched by a command write sequence. The ACCERR bit in the FSTAT register will be set during the command write sequence if any of the following illegal steps are performed, causing the command not to be processed by the Memory Controller:

- Starting any command write sequence that programs or erases Flash memory before initializing the FCLKDIV register
- Writing an invalid command as part of the command write sequence
- For additional possible errors, refer to the error handling table provided for each command

If a Flash block is read during execution of an algorithm (CCIF = 0) on that same block, the read operation will return invalid data. If the SFDIF or DFDIF flags were not previously set when the invalid read operation occurred, both the SFDIF and DFDIF flags will be set and the FECCR registers will be loaded with the global address used in the invalid read operation with the data and parity fields set to all 0.

If the ACCERR or FPVIOL bits are set in the FSTAT register, the user must clear these bits before starting any command write sequence (see Section 10.3.2.7).

CAUTION

A Flash word or phrase must be in the erased state before being programmed. Cumulative programming of bits within a Flash word or phrase is not allowed.

10.4.2.1 Erase Verify All Blocks Command

The Erase Verify All Blocks command will verify that all P-Flash and D-Flash blocks have been erased.

Table 10-33. Erase Verify All Blocks Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x01	Not required

Upon clearing CCIF to launch the Erase Verify All Blocks command, the Memory Controller will verify that the entire Flash memory space is erased. The CCIF flag will set after the Erase Verify All Blocks operation has completed.

Table 10-34. Erase Verify All Blocks Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read(1)
	MGSTAT0	Set if any non-correctable errors have been encountered during the read[1]
FERSTAT	EPVIOLIF	None

1. As found in the memory map for FTM512K3.

10.4.2.2 Erase Verify Block Command

The Erase Verify Block command allows the user to verify that an entire P-Flash or D-Flash block has been erased. The FCCOB upper global address bits determine which block must be verified.

Table 10-35. Erase Verify Block Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x02	Global address [22:16] of the Flash block to be verified.

Upon clearing CCIF to launch the Erase Verify Block command, the Memory Controller will verify that the selected P-Flash or D-Flash block is erased. The CCIF flag will set after the Erase Verify Block operation has completed.

Table 10-36. Erase Verify Block Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if an invalid global address [22:16] is supplied(1)
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read(2)
	MGSTAT0	Set if any non-correctable errors have been encountered during the read2
FERSTAT	EPVIOLIF	None

1. As defined by the memory map for FTM512K3.
2. As found in the memory map for FTM512K3.

10.4.2.3 Erase Verify P-Flash Section Command

The Erase Verify P-Flash Section command will verify that a section of code in the P-Flash memory is erased. The Erase Verify P-Flash Section command defines the starting point of the code to be verified and the number of phrases.

Table 10-37. Erase Verify P-Flash Section Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x03	Global address [22:16] of a P-Flash block
001	Global address [15:0] of the first phrase to be verified	
010	Number of phrases to be verified	

Upon clearing CCIF to launch the Erase Verify P-Flash Section command, the Memory Controller will verify the selected section of Flash memory is erased. The CCIF flag will set after the Erase Verify P-Flash Section operation has completed.

Table 10-38. Erase Verify P-Flash Section Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 010 at command launch
		Set if command not available in current mode (see Table 10-30)
		Set if an invalid global address [22:0] is supplied(1)
		Set if a misaligned phrase address is supplied (global address [2:0] != 000)
		Set if the requested section crosses a 256 Kbyte boundary
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read(2)
	MGSTAT0	Set if any non-correctable errors have been encountered during the read2
FERSTAT	EPVIOLIF	None

1. As defined by the memory map for FTM512K3.
2. As found in the memory map for FTM512K3.

10.4.2.4 Read Once Command

The Read Once command provides read access to a reserved 64 byte field (8 phrases) located in the nonvolatile information register of P-Flash block 0. The Read Once field is programmed using the Program Once command described in Section 10.4.2.6. The Read Once command must not be executed from the Flash block containing the Program Once reserved field to avoid code runaway.

Table 10-39. Read Once Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x04	Not Required
001	Read Once phrase index (0x0000 - 0x0007)	
010	Read Once word 0 value	
011	Read Once word 1 value	
100	Read Once word 2 value	
101	Read Once word 3 value	

Upon clearing CCIF to launch the Read Once command, a Read Once phrase is fetched and stored in the FCCOB indexed register. The CCIF flag will set after the Read Once operation has completed. Valid phrase index values for the Read Once command range from 0x0000 to 0x0007. During execution of the Read Once command, any attempt to read addresses within P-Flash block 0 will return invalid data.

Table 10-40. Read Once Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if command not available in current mode (see Table 10-30)
		Set if an invalid phrase index is supplied
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read
	MGSTAT0	Set if any non-correctable errors have been encountered during the read
FERSTAT	EPVIOLIF	None

10.4.2.5 Program P-Flash Command

The Program P-Flash operation will program a previously erased phrase in the P-Flash memory using an embedded algorithm.

CAUTION

A P-Flash phrase must be in the erased state before being programmed. Cumulative programming of bits within a Flash phrase is not allowed.

Table 10-41. Program P-Flash Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x06	Global address [22:16] to identify P-Flash block
001	Global address [15:0] of phrase location to be programmed(1)	
010	Word 0 program value	
011	Word 1 program value	
100	Word 2 program value	
101	Word 3 program value	

1. Global address [2:0] must be 000

Upon clearing CCIF to launch the Program P-Flash command, the Memory Controller will program the data words to the supplied global address and will then proceed to verify the data words read back as expected. The CCIF flag will set after the Program P-Flash operation has completed.

Table 10-42. Program P-Flash Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 101 at command launch
		Set if command not available in current mode (see Table 10-30)
		Set if an invalid global address [22:0] is supplied(1)
		Set if a misaligned phrase address is supplied (global address [2:0] != 000)
	FPVIOL	Set if the global address [22:0] points to a protected area
	MGSTAT1	Set if any errors have been encountered during the verify operation(2)
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation2
FERSTAT	EPVIOLIF	None

1. As defined by the memory map for FTM512K3.
2. As found in the memory map for FTM512K3.

10.4.2.6 Program Once Command

The Program Once command restricts programming to a reserved 64 byte field (8 phrases) in the nonvolatile information register located in P-Flash block 0. The Program Once reserved field can be read using the Read Once command as described in Section 10.4.2.4. The Program Once command must only be issued once since the nonvolatile information register in P-Flash block 0 cannot be erased. The Program

Once command must not be executed from the Flash block containing the Program Once reserved field to avoid code runaway.

Table 10-43. Program Once Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x07	Not Required
001	Program Once phrase index (0x0000 - 0x0007)	
010	Program Once word 0 value	
011	Program Once word 1 value	
100	Program Once word 2 value	
101	Program Once word 3 value	

Upon clearing CCIF to launch the Program Once command, the Memory Controller first verifies that the selected phrase is erased. If erased, then the selected phrase will be programmed and then verified with read back. The CCIF flag will remain clear, setting only after the Program Once operation has completed.

The reserved nonvolatile information register accessed by the Program Once command cannot be erased and any attempt to program one of these phrases a second time will not be allowed. Valid phrase index values for the Program Once command range from 0x0000 to 0x0007. During execution of the Program Once command, any attempt to read addresses within P-Flash block 0 will return invalid data.

Table 10-44. Program Once Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 101 at command launch
		Set if command not available in current mode (see Table 10-30)
		Set if an invalid phrase index is supplied
		Set if the requested phrase has already been programmed(1)
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation
FERSTAT	EPVIOLIF	None

1. If a Program Once phrase is initially programmed to 0xFFFF_FFFF_FFFF_FFFF, the Program Once command will be allowed to execute again on that same phrase.

10.4.2.7 Erase All Blocks Command

The Erase All Blocks operation will erase the entire P-Flash and D-Flash memory space including the EEE nonvolatile information register.

Table 10-45. Erase All Blocks Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x08	Not required

Upon clearing CCIF to launch the Erase All Blocks command, the Memory Controller will erase the entire Flash memory space and verify that it is erased. If the Memory Controller verifies that the entire Flash memory space was properly erased, security will be released. During the execution of this command (CCIF=0) the user must not write to any Flash module register. The CCIF flag will set after the Erase All Blocks operation has completed.

Table 10-46. Erase All Blocks Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if command not available in current mode (see Table 10-30)
	FPVIOL	Set if any area of the P-Flash memory is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation(1)
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation[1]
FERSTAT	EPVIOLIF	Set if any area of the buffer RAM EEE partition is protected

1. As found in the memory map for FTM512K3.

10.4.2.8 Erase P-Flash Block Command

The Erase P-Flash Block operation will erase all addresses in a P-Flash block.

Table 10-47. Erase P-Flash Block Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x09	Global address [22:16] to identify P-Flash block
001	Global address [15:0] in P-Flash block to be erased	

Upon clearing CCIF to launch the Erase P-Flash Block command, the Memory Controller will erase the selected P-Flash block and verify that it is erased. The CCIF flag will set after the Erase P-Flash Block operation has completed.

Table 10-48. Erase P-Flash Block Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if command not available in current mode (see Table 10-30)
		Set if an invalid global address [22:16] is supplied(1)
	FPVIOL	Set if an area of the selected P-Flash block is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation(2)
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation2
FERSTAT	EPVIOLIF	None

1. As defined by the memory map for FTM512K3.
2. As found in the memory map for FTM512K3.

10.4.2.9 Erase P-Flash Sector Command

The Erase P-Flash Sector operation will erase all addresses in a P-Flash sector.

Table 10-49. Erase P-Flash Sector Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x0A	Global address [22:16] to identify P-Flash block to be erased
001	Global address [15:0] anywhere within the sector to be erased. Refer to Section 10.1.2.1 for the P-Flash sector size.	

Upon clearing CCIF to launch the Erase P-Flash Sector command, the Memory Controller will erase the selected Flash sector and then verify that it is erased. The CCIF flag will be set after the Erase P-Flash Sector operation has completed.

Table 10-50. Erase P-Flash Sector Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if command not available in current mode (see Table 10-30)
		Set if an invalid global address [22:16] is supplied(1)
		Set if a misaligned phrase address is supplied (global address [2:0] != 000)
	FPVIOL	Set if the selected P-Flash sector is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation
FERSTAT	EPVIOLIF	None

1. As defined by the memory map for FTM512K3.

10.4.2.10 Unsecure Flash Command

The Unsecure Flash command will erase the entire P-Flash and D-Flash memory space and, if the erase is successful, will release security.

Table 10-51. Unsecure Flash Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x0B	Not required

Upon clearing CCIF to launch the Unsecure Flash command, the Memory Controller will erase the entire P-Flash and D-Flash memory space and verify that it is erased. If the Memory Controller verifies that the entire Flash memory space was properly erased, security will be released. If the erase verify is not successful, the Unsecure Flash operation sets MGSTAT1 and terminates without changing the security state. During the execution of this command (CCIF=0) the user must not write to any Flash module register. The CCIF flag is set after the Unsecure Flash operation has completed.

Table 10-52. Unsecure Flash Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if command not available in current mode (see Table 10-30)
	FPVIOL	Set if any area of the P-Flash memory is protected
	MGSTAT1	Set if any errors have been encountered during the verify operation(1)
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation[1]
FERSTAT	EPVIOLIF	Set if any area of the buffer RAM EEE partition is protected

1. As found in the memory map for FTM512K3.

10.4.2.11 Verify Backdoor Access Key Command

The Verify Backdoor Access Key command will only execute if it is enabled by the KEYEN bits in the FSEC register (see Table 10-11). The Verify Backdoor Access Key command releases security if user-supplied keys match those stored in the Flash security bytes of the Flash configuration field (see Table 10-3). The Verify Backdoor Access Key command must not be executed from the Flash block containing the backdoor comparison key to avoid code runaway.

Table 10-53. Verify Backdoor Access Key Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x0C	Not required
001	Key 0	
010	Key 1	

Table 10-53. Verify Backdoor Access Key Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters
011	Key 2
100	Key 3

Upon clearing CCIF to launch the Verify Backdoor Access Key command, the Memory Controller will check the FSEC KEYEN bits to verify that this command is enabled. If not enabled, the Memory Controller sets the ACCERR bit in the FSTAT register and terminates. If the command is enabled, the Memory Controller compares the key provided in FCCOB to the backdoor comparison key in the Flash configuration field with Key 0 compared to 0x7F_FF00, etc. If the backdoor keys match, security will be released. If the backdoor keys do not match, security is not released and all future attempts to execute the Verify Backdoor Access Key command are aborted (set ACCERR) until a reset occurs. The CCIF flag is set after the Verify Backdoor Access Key operation has completed.

Table 10-54. Verify Backdoor Access Key Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 100 at command launch
		Set if an incorrect backdoor key is supplied
		Set if backdoor key access has not been enabled (KEYEN[1:0] != 10, see Section 10.3.2.2)
		Set if the backdoor key has mismatched since the last reset
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

10.4.2.12 Set User Margin Level Command

The Set User Margin Level command causes the Memory Controller to set the margin level for future read operations of a specific P-Flash or D-Flash block.

Table 10-55. Set User Margin Level Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x0D	Global address [22:16] to identify the Flash block
001	Margin level setting	

Upon clearing CCIF to launch the Set User Margin Level command, the Memory Controller will set the user margin level for the targeted block and then set the CCIF flag.

Valid margin level settings for the Set User Margin Level command are defined in Table 10-56.

Table 10-56. Valid Set User Margin Level Settings

CCOB (CCOBIX=001)	Level Description
0x0000	Return to Normal Level
0x0001	User Margin-1 Level(1)
0x0002	User Margin-0 Level(2)

1. Read margin to the erased state
2. Read margin to the programmed state

Table 10-57. Set User Margin Level Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if command not available in current mode (see Table 10-30)
		Set if an invalid global address [22:16] is supplied(1)
		Set if an invalid margin level setting is supplied
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

1. As defined by the memory map for FTM512K3.

NOTE

User margin levels can be used to check that Flash memory contents have adequate margin for normal level read operations. If unexpected results are encountered when checking Flash memory contents at user margin levels, a potential loss of information has been detected.

10.4.2.13 Set Field Margin Level Command

The Set Field Margin Level command, valid in special modes only, causes the Memory Controller to set the margin level specified for future read operations of a specific P-Flash or D-Flash block.

Table 10-58. Set Field Margin Level Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x0E	Global address [22:16] to identify the Flash block
001	Margin level setting	

Upon clearing CCIF to launch the Set Field Margin Level command, the Memory Controller will set the field margin level for the targeted block and then set the CCIF flag.

Valid margin level settings for the Set Field Margin Level command are defined in Table 10-59.

Table 10-59. Valid Set Field Margin Level Settings

CCOB (CCOBIX=001)	Level Description
0x0000	Return to Normal Level
0x0001	User Margin-1 Level(1)
0x0002	User Margin-0 Level(2)
0x0003	Field Margin-1 Level[1]
0x0004	Field Margin-0 Level[2]

1. Read margin to the erased state
2. Read margin to the programmed state

Table 10-60. Set Field Margin Level Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if command not available in current mode (see Table 10-30)
		Set if an invalid global address [22:16] is supplied(1)
		Set if an invalid margin level setting is supplied
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

1. As defined by the memory map for FTM512K3.

CAUTION

Field margin levels must only be used during verify of the initial factory programming.

NOTE

Field margin levels can be used to check that Flash memory contents have adequate margin for data retention at the normal level setting. If unexpected results are encountered when checking Flash memory contents at field margin levels, the Flash memory contents should be erased and reprogrammed.

10.4.2.14 Full Partition D-Flash Command

The Full Partition D-Flash command allows the user to allocate sectors within the D-Flash block for applications and a partition within the buffer RAM for EEPROM access. The D-Flash block consists of 128 sectors with 256 bytes per sector.

Table 10-61. Full Partition D-Flash Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x0F	Not required
001	Number of 256 byte sectors for the D-Flash user partition (DFPART)	
010	Number of 256 byte sectors for buffer RAM EEE partition (ERPART)	

Upon clearing CCIF to launch the Full Partition D-Flash command, the following actions are taken to define a partition within the D-Flash block for direct access (DFPART) and a partition within the buffer RAM for EEE use (ERPART):

- Validate the DFPART and ERPART values provided:
  - DFPART <= 128 (maximum number of 256 byte sectors in D-Flash block)
  - ERPART <= 8 (maximum number of 256 byte sectors in buffer RAM)
  - If ERPART > 0, 128 - DFPART >= 12 (minimum number of 256 byte sectors in the D-Flash block required to support EEE)
  - If ERPART > 0, ((128-DFPART)/ERPART) >= 8 (minimum ratio of D-Flash EEE space to buffer RAM EEE space to support EEE)
- Erase the D-Flash block and the EEE nonvolatile information register
- Program DFPART to the EEE nonvolatile information register at global address 0x12_0000 (see Table 10-7)
- Program a duplicate DFPART to the EEE nonvolatile information register at global address 0x12_0002 (see Table 10-7)
- Program ERPART to the EEE nonvolatile information register at global address 0x12_0004 (see Table 10-7)
- Program a duplicate ERPART to the EEE nonvolatile information register at global address 0x12_0006 (see Table 10-7)

The D-Flash user partition will start at global address 0x10_0000. The buffer RAM EEE partition will end at global address 0x13_FFFF. After the Full Partition D-Flash operation has completed, the CCIF flag will set.

Running the Full Partition D-Flash command a second time will result in the previous partition values and the entire D-Flash memory being erased. The data value written corresponds to the number of 256 byte sectors allocated for either direct D-Flash access (DFPART) or buffer RAM EEE access (ERPART).

Table 10-62. Full Partition D-Flash Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 010 at command launch
		Set if command not available in current mode (see Table 10-30)
		Set if an invalid DFPART or ERPART selection is supplied(1)
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read
	MGSTAT0	Set if any non-correctable errors have been encountered during the read
FERSTAT	EPVIOLIF	None

1. As defined by the maximum ERPART for FTM512K3.

10.4.2.15 Erase Verify D-Flash Section Command

The Erase Verify D-Flash Section command will verify that a section of code in the D-Flash user partition is erased. The Erase Verify D-Flash Section command defines the starting point of the data to be verified and the number of words.

Table 10-63. Erase Verify D-Flash Section Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x10	Global address [22:16] to identify the D-Flash block
001	Global address [15:0] of the first word to be verified	
010	Number of words to be verified	

Upon clearing CCIF to launch the Erase Verify D-Flash Section command, the Memory Controller will verify the selected section of D-Flash memory is erased. The CCIF flag will set after the Erase Verify D-Flash Section operation has completed.

Table 10-64. Erase Verify D-Flash Section Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 010 at command launch
		Set if command not available in current mode (see Table 10-30)
		Set if an invalid global address [22:0] is supplied
		Set if a misaligned word address is supplied (global address [0] != 0)
		Set if the global address [22:0] points to an area of the D-Flash EEE partition
		Set if the requested section breaches the end of the D-Flash block or goes into the D-Flash EEE partition
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read
	MGSTAT0	Set if any non-correctable errors have been encountered during the read
FERSTAT	EPVIOLIF	None

10.4.2.16 Program D-Flash Command

The Program D-Flash operation programs one to four previously erased words in the D-Flash user partition. The Program D-Flash operation will confirm that the targeted location(s) were successfully programmed upon completion.

CAUTION

A Flash word must be in the erased state before being programmed. Cumulative programming of bits within a Flash word is not allowed.

Table 10-65. Program D-Flash Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x11	Global address [22:16] to identify the D-Flash block
001	Global address [15:0] of word to be programmed	
010	Word 0 program value	
011	Word 1 program value, if desired	
100	Word 2 program value, if desired	
101	Word 3 program value, if desired	

Upon clearing CCIF to launch the Program D-Flash command, the user-supplied words will be transferred to the Memory Controller and be programmed. The CCOBIX index value at Program D-Flash command launch determines how many words will be programmed in the D-Flash block. No protection checks are made in the Program D-Flash operation on the D-Flash block, only access error checks. The CCIF flag is set when the operation has completed.

Table 10-66. Program D-Flash Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] < 010 at command launch
		Set if CCOBIX[2:0] > 101 at command launch
		Set if command not available in current mode (see Table 10-30)
		Set if an invalid global address [22:0] is supplied
		Set if a misaligned word address is supplied (global address [0] != 0)
		Set if the global address [22:0] points to an area in the D-Flash EEE partition
		Set if the requested group of words breaches the end of the D-Flash block or goes into the D-Flash EEE partition
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation
FERSTAT	EPVIOLIF	None

10.4.2.17 Erase D-Flash Sector Command

The Erase D-Flash Sector operation will erase all addresses in a sector of the D-Flash user partition.

Table 10-67. Erase D-Flash Sector Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x12	Global address [22:16] to identify D-Flash block
001	Global address [15:0] anywhere within the sector to be erased. See Section 10.1.2.2 for D-Flash sector size.	

Upon clearing CCIF to launch the Erase D-Flash Sector command, the Memory Controller will erase the selected Flash sector and verify that it is erased. The CCIF flag will set after the Erase D-Flash Sector operation has completed.

Table 10-68. Erase D-Flash Sector Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 001 at command launch
		Set if command not available in current mode (see Table 10-30)
		Set if an invalid global address [22:0] is supplied
		Set if a misaligned word address is supplied (global address [0] != 0)
		Set if the global address [22:0] points to the D-Flash EEE partition
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the verify operation
	MGSTAT0	Set if any non-correctable errors have been encountered during the verify operation
FERSTAT	EPVIOLIF	None

10.4.2.18 Enable EEPROM Emulation Command

The Enable EEPROM Emulation command causes the Memory Controller to enable EEE activity. EEE activity is disabled after any reset.

Table 10-69. Enable EEPROM Emulation Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x13	Not required

Upon clearing CCIF to launch the Enable EEPROM Emulation command, the CCIF flag will set after the Memory Controller enables EEE operations using the contents of the EEE tag RAM and tag counter. The Full Partition D-Flash or the Partition D-Flash command must be run prior to launching the Enable EEPROM Emulation command.

Table 10-70. Enable EEPROM Emulation Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if Full Partition D-Flash or Partition D-Flash command not previously run
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

10.4.2.19 Disable EEPROM Emulation Command

The Disable EEPROM Emulation command causes the Memory Controller to suspend current EEE activity.

Table 10-71. Disable EEPROM Emulation Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x14	Not required

Upon clearing CCIF to launch the Disable EEPROM Emulation command, the Memory Controller will halt EEE operations at the next convenient point without clearing the EEE tag RAM or tag counter before setting the CCIF flag.

Table 10-72. Disable EEPROM Emulation Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if Full Partition D-Flash or Partition D-Flash command not previously run
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

10.4.2.20 EEPROM Emulation Query Command

The EEPROM Emulation Query command returns EEE partition and status variables.

Table 10-73. EEPROM Emulation Query Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x15	Not required
001	Return DFPART	
010	Return ERPART	
011	Return ECOUNT(1)	
100	Return Dead Sector Count	Return Ready Sector Count

1. Indicates sector erase count

Upon clearing CCIF to launch the EEPROM Emulation Query command, the CCIF flag will set after the EEE partition and status variables are stored in the FCCOBIX register.If the Emulation Query command is executed prior to partitioning (Partition D-Flash Command Section 10.4.2.14), the following reset values are returned: DFPART = 0x_FFFF, ERPART = 0x_FFFF, ECOUNT = 0x_FFFF, Dead Sector Count = 0x_00, Ready Sector Count = 0x_00.

Table 10-74. EEPROM Emulation Query Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 000 at command launch
		Set if command not available in current mode (see Table 10-30)
	FPVIOL	None
	MGSTAT1	None
	MGSTAT0	None
FERSTAT	EPVIOLIF	None

10.4.2.21 Partition D-Flash Command

The Partition D-Flash command allows the user to allocate sectors within the D-Flash block for applications and a partition within the buffer RAM for EEPROM access. The D-Flash block consists of 64 sectors with 256 bytes per sector. The Erase All Blocks command must be run prior to launching the Partition D-Flash command.

Table 10-75. Partition D-Flash Command FCCOB Requirements

CCOBIX[2:0]	FCCOB Parameters	
000	0x20	Not required
001	Number of 256 byte sectors for the D-Flash user partition (DFPART)	
010	Number of 256 byte sectors for buffer RAM EEE partition (ERPART)	

Upon clearing CCIF to launch the Partition D-Flash command, the following actions are taken to define a partition within the D-Flash block for direct access (DFPART) and a partition within the buffer RAM for EEE use (ERPART):

- Validate the DFPART and ERPART values provided:
  - DFPART <= 128 (maximum number of 256 byte sectors in D-Flash block)
  - ERPART <= 8 (maximum number of 256 byte sectors in buffer RAM)
  - If ERPART > 0, 128 - DFPART >= 12 (minimum number of 256 byte sectors in the D-Flash block required to support EEE)
  - If ERPART > 0, ((128-DFPART)/ERPART) >= 8 (minimum ratio of D-Flash EEE space to buffer RAM EEE space to support EEE)
- Erase verify the D-Flash block and the EEE nonvolatile information register
- Program DFPART to the EEE nonvolatile information register at global address 0x12_0000 (see Table 10-7)
- Program a duplicate DFPART to the EEE nonvolatile information register at global address 0x12_0002 (see Table 10-7)

- Program ERPART to the EEE nonvolatile information register at global address 0x12_0004 (see Table 10-7)
- Program a duplicate ERPART to the EEE nonvolatile information register at global address 0x12_0006 (see Table 10-7)

The D-Flash user partition will start at global address 0x10_0000. The buffer RAM EEE partition will end at global address 0x13_FFFF. After the Partition D-Flash operation has completed, the CCIF flag will set.

Running the Partition D-Flash command a second time will result in the ACCERR bit within the FSTAT register being set. The data value written corresponds to the number of 256 byte sectors allocated for either direct D-Flash access (DFPART) or buffer RAM EEE access (ERPART).

Table 10-76. Partition D-Flash Command Error Handling

Register	Error Bit	Error Condition
FSTAT	ACCERR	Set if CCOBIX[2:0] != 010 at command launch
		Set if command not available in current mode (see Table 10-30)
		Set if partitions have already been defined
		Set if an invalid DFPART or ERPART selection is supplied(1)
	FPVIOL	None
	MGSTAT1	Set if any errors have been encountered during the read
	MGSTAT0	Set if any non-correctable errors have been encountered during the read
FERSTAT	EPVIOLIF	None

1. As defined by the maximum ERPART for FTM512K3.

10.4.3 Interrupts

The Flash module can generate an interrupt when a Flash command operation has completed or when a Flash command operation has detected an EEE error or an ECC fault.

Table 10-77. Flash Interrupt Sources

Interrupt Source	Interrupt Flag	Local Enable	Global (CCR) Mask
Flash Command Complete	CCIF (FSTAT register)	CCIE (FCNFG register)	I Bit
Flash EEE Erase Error	ERSERIF (FERSTAT register)	ERSERIE (FERCNFG register)	I Bit
Flash EEE Program Error	PGMERIF (FERSTAT register)	PGMERIE (FERCNFG register)	I Bit
Flash EEE Protection Violation	EPVIOLIF (FERSTAT register)	EPVIOLIE (FERCNFG register)	I Bit
Flash EEE Error Type 1 Violation	ERSVIF1 (FERSTAT register)	ERSVIE1 (FERCNFG register)	I Bit
Flash EEE Error Type 0 Violation	ERSVIF0 (FERSTAT register)	ERSVIE0 (FERCNFG register)	I Bit
ECC Double Bit Fault on Flash Read	DFDIF (FERSTAT register)	DFDIE (FERCNFG register)	I Bit
ECC Single Bit Fault on Flash Read	SFDIF (FERSTAT register)	SFDIE (FERCNFG register)	I Bit

NOTE

Vector addresses and their relative interrupt priority are determined at the MCU level.

10.4.3.1 Description of Flash Interrupt Operation

The Flash module uses the CCIF flag in combination with the CCIE interrupt enable bit to generate the Flash command interrupt request. The Flash module uses the ERSEIF, PGMEIF, EPVIOLIF, ERSVIF1, ERSVIF0, DFDIF and SFDIF flags in combination with the ERSEIE, PGMEIE, EPVIOLIE, ERSVIE1, ERSVIE0, DFDIE and SFDIE interrupt enable bits to generate the Flash error interrupt request. For a detailed description of the register bits involved, refer to Section 10.3.2.5, “Flash Configuration Register (FCNFG)”, Section 10.3.2.6, “Flash Error Configuration Register (FERCNFG)”, Section 10.3.2.7, “Flash Status Register (FSTAT)”, and Section 10.3.2.8, “Flash Error Status Register (FERSTAT)”.

The logic used for generating the Flash module interrupts is shown in Figure 10-27.

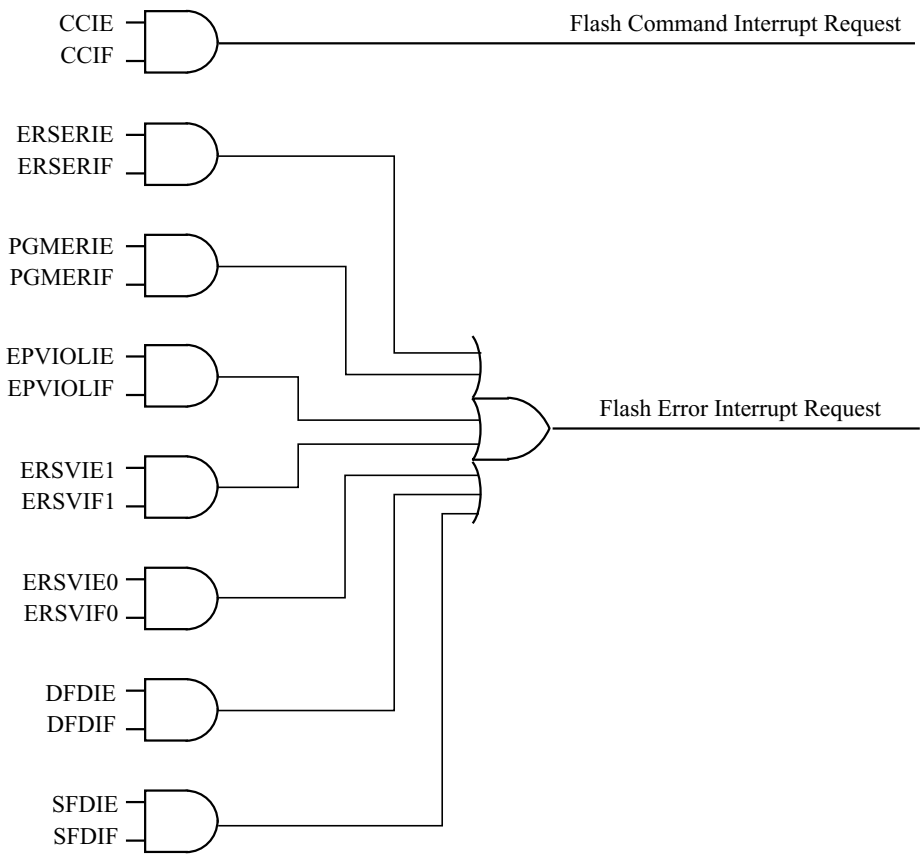


Figure 10-27. Flash Module Interrupts Implementation

10.4.4 Wait Mode

The Flash module is not affected if the MCU enters wait mode. The Flash module can recover the MCU from wait via the CCIF interrupt (see Section 10.4.3, “Interrupts”).

10.4.5 Stop Mode

If a Flash command is active (CCIF = 0) or an EE-Emulation operation is pending when the MCU requests stop mode, the current Flash operation will be completed before the CPU is allowed to enter stop mode.

10.5 Security

The Flash module provides security information to the MCU. The Flash security state is defined by the SEC bits of the FSEC register (see Table 10-12). During reset, the Flash module initializes the FSEC register using data read from the security byte of the Flash configuration field at global address 0x7F_FF0F.

The security state out of reset can be permanently changed by programming the security byte of the Flash configuration field. This assumes that you are starting from a mode where the necessary P-Flash erase and program commands are available and that the upper region of the P-Flash is unprotected. If the Flash security byte is successfully programmed, its new value will take affect after the next MCU reset.

The following subsections describe these security-related subjects:

- Unsecuring the MCU using Backdoor Key Access
- Unsecuring the MCU in Special Single Chip Mode using BDM
- Mode and Security Effects on Flash Command Availability

10.5.1 Unsecuring the MCU using Backdoor Key Access

The MCU may be unsecured by using the backdoor key access feature which requires knowledge of the contents of the backdoor keys (four 16-bit words programmed at addresses 0x7F_FF00–0x7F_FF07). If the KEYEN[1:0] bits are in the enabled state (see Section 10.3.2.2), the Verify Backdoor Access Key command (see Section 10.4.2.11) allows the user to present four prospective keys for comparison to the keys stored in the Flash memory via the Memory Controller. If the keys presented in the Verify Backdoor Access Key command match the backdoor keys stored in the Flash memory, the SEC bits in the FSEC register (see Table 10-12) will be changed to unsecure the MCU. Key values of 0x0000 and 0xFFFF are not permitted as backdoor keys. While the Verify Backdoor Access Key command is active, P-Flash block 0 will not be available for read access and will return invalid data.

The user code stored in the P-Flash memory must have a method of receiving the backdoor keys from an external stimulus. This external stimulus would typically be through one of the on-chip serial ports.

If the KEYEN[1:0] bits are in the enabled state (see Section 10.3.2.2), the MCU can be unsecured by the backdoor key access sequence described below:

1. Follow the command sequence for the Verify Backdoor Access Key command as explained in Section 10.4.2.11
2. If the Verify Backdoor Access Key command is successful, the MCU is unsecured and the SEC[1:0] bits in the FSEC register are forced to the unsecure state of 10

The Verify Backdoor Access Key command is monitored by the Memory Controller and an illegal key will prohibit future use of the Verify Backdoor Access Key command. A reset of the MCU is the only method to re-enable the Verify Backdoor Access Key command.

After the backdoor keys have been correctly matched, the MCU will be unsecured. After the MCU is unsecured, the sector containing the Flash security byte can be erased and the Flash security byte can be reprogrammed to the unsecure state, if desired.

In the unsecure state, the user has full control of the contents of the backdoor keys by programming addresses 0x7F_FF00–0x7F_FF07 in the Flash configuration field.

The security as defined in the Flash security byte (0x7F_FF0F) is not changed by using the Verify Backdoor Access Key command sequence. The backdoor keys stored in addresses 0x7F_FF00–0x7F_FF07 are unaffected by the Verify Backdoor Access Key command sequence. After the next reset of the MCU, the security state of the Flash module is determined by the Flash security byte

(0x7F_FF0F). The Verify Backdoor Access Key command sequence has no effect on the program and erase protections defined in the Flash protection register, FPROT.

10.5.2 Unsecuring the MCU in Special Single Chip Mode using BDM

The MCU can be unsecured in special single chip mode by erasing the P-Flash and D-Flash memory by one of the following methods:

- Reset the MCU into special single chip mode, delay while the erase test is performed by the BDM, send BDM commands to disable protection in the P-Flash and D-Flash memory, and execute the Erase All Blocks command write sequence to erase the P-Flash and D-Flash memory.
- Reset the MCU into special expanded wide mode, disable protection in the P-Flash and D-Flash memory and run code from external memory to execute the Erase All Blocks command write sequence to erase the P-Flash and D-Flash memory.

After the CCIF flag sets to indicate that the Erase All Blocks operation has completed, reset the MCU into special single chip mode. The BDM will execute the Erase Verify All Blocks command write sequence to verify that the P-Flash and D-Flash memory is erased. If the P-Flash and D-Flash memory are verified as erased the MCU will be unsecured. All BDM commands will be enabled and the Flash security byte may be programmed to the unsecure state by the following method:

- Send BDM commands to execute a ‘Program P-Flash’ command sequence to program the Flash security byte to the unsecured state and reset the MCU.

10.5.3 Mode and Security Effects on Flash Command Availability

The availability of Flash module commands depends on the MCU operating mode and security state as shown in Table 10-30.

10.6 Initialization

On each system reset the Flash module executes a reset sequence which establishes initial values for the Flash Block Configuration Parameters, the FPROT and DFPROT protection registers, and the FOPT and FSEC registers. The Flash module reverts to built-in default values that leave the module in a fully protected and secured state if errors are encountered during execution of the reset sequence. If a double bit fault is detected during the reset sequence, both MGSTAT bits in the FSTAT register will be set. The ACCERR bit in the FSTAT register is set if errors are encountered while initializing the EEE buffer ram during the reset sequence.

CCIF remains clear throughout the reset sequence. The Flash module holds off all CPU access for the initial portion of the reset sequence. While Flash reads are possible when the hold is removed, writes to the FCCOBIX, FCCOBHI, and FCCOBLO registers are ignored to prevent command activity while the Memory Controller remains busy. Completion of the reset sequence is marked by setting CCIF high which enables writes to the FCCOBIX, FCCOBHI, and FCCOBLO registers to launch any available Flash command.

If a reset occurs while any Flash command is in progress, that command will be immediately aborted. The state of the word being programmed or the sector/block being erased is not guaranteed.

Chapter 11
Memory Mapping Control (S12XMMCV4) SUPPORTING FLEXRAY

Revision History

Rev. No. (Item No.)	Date (Submitted By)	Sections Affected	Substantial Change(s)
v04.06	15-Nov-06		- Adding AUTOSAR Compliance concerning illegal CPU accesses
v04.07	02-Apr-07		- Adapting the MMC context to support S12XS family
v04.08	04-May-07		- Clarifying RPAGE usage for less than 12KB RAMSIZE. - Some Cleanups

11.1 Introduction

This section describes the functionality of the module mapping control (MMC) sub-block of the S12X platform. The block diagram of the MMC is shown in Figure 11-1.

The MMC module controls the multi-master priority accesses, the selection of internal resources and external space. Internal buses, including internal memories and peripherals, are controlled in this module. The local address space for each master is translated to a global memory space.

11.1.1 Terminology

Table 11-1. Acronyms and Abbreviations

Logic level “1”	Voltage that corresponds to Boolean true state
Logic level “0”	Voltage that corresponds to Boolean false state
0x	Represents hexadecimal number
x	Represents logic level ’don’t care’
Byte	8-bit data
word	16-bit data
local address	based on the 64KB Memory Space (16-bit address)
global address	based on the 8MB Memory Space (23-bit address)
Aligned address	Address on even boundary
Mis-aligned address	Address on odd boundary
Bus Clock	System Clock. Refer to CRG Block Guide.
expanded modes	Normal Expanded Mode Emulation Single-Chip Mode Emulation Expanded Mode Special Test Mode
single-chip modes	Normal Single-Chip Mode Special Single-Chip Mode
emulation modes	Emulation Single-Chip Mode Emulation Expanded Mode
normal modes	Normal Single-Chip Mode Normal Expanded Mode
special modes	Special Single-Chip Mode Special Test Mode
NS	Normal Single-Chip Mode
SS	Special Single-Chip Mode
NX	Normal Expanded Mode
ES	Emulation Single-Chip Mode
EX	Emulation Expanded Mode
ST	Special Test Mode
Unimplemented areas	Areas which are accessible by the pages (RPAGE,PPAGE,EPAGE) and not implemented
External Space	Area which is accessible in the global address range 14_0000 to 3F_FFFF
external resource	Resources (Emulator, Application) connected to the MCU via the external bus on expanded modes (Unimplemented areas and External Space)
PRR	Port Replacement Registers
PRU	Port Replacement Unit located on the emulator side
MCU	MicroController Unit
NVM	Non-volatile Memory; Flash, EEPROM or ROM
IFR	Information Row sector located on the top of NVM. For Test purposes.
FLEXRAY	FlexRay IP Integration module

11.1.2 Features

The main features of this block are:

- Paging capability to support a global 8MB memory address space

- Bus arbitration between the masters CPU, BDM, FLEXRAY and XGATE
- Simultaneous accesses to different resources[1] (internal, external, and peripherals) (see Figure 11-1)
- Resolution of target bus access collision
- MCU operation mode control
- MCU security control
- Separate memory map schemes for each master CPU, BDM, FLEXRAY and XGATE
- ROM control bits to enable the on-chip FLASH or ROM selection
- Port replacement registers access control
- Generation of system reset when CPU accesses an unimplemented address (i.e., an address which does not belong to any of the on-chip modules) in single-chip modes

11.1.3 S12X Memory Mapping

The S12X architecture implements a number of memory mapping schemes including

- a CPU 8MB global map, defined using a global page (GPAGE) register and dedicated 23-bit address load/store instructions.
- a BDM 8MB global map, defined using a global page (BDMGPR) register and dedicated 23-bit address load/store instructions.
- a FLEXRAY 8 MByte global map.
- a (CPU or BDM) 64KB local map, defined using specific resource page (RPAGE, EPAGE and PPAGE) registers and the default instruction set. The 64KB visible at any instant can be considered as the local map accessed by the 16-bit (CPU or BDM) address.
- The XGATE 64 Kbyte local map.

The MMC module performs translation of the different memory mapping schemes to the specific global (physical) memory implementation.

11.1.4 Modes of Operation

This subsection lists and briefly describes all operating modes supported by the MMC.

11.1.4.1 Power Saving Modes

- Run mode

  MMC is functional during normal run mode.
- Wait mode

  MMC is functional during wait mode.
- Stop mode

  MMC is inactive during stop mode.

1. Resources are also called targets.

11.1.4.2 Functional Modes

- Single chip modes

  In normal and special single chip mode the internal memory is used. External bus is not active.
- Expanded modes

  Address, data, and control signals are activated in normal expanded and special test modes when accessing the external bus. Access to internal resources will not cause activity on the external bus.
- Emulation modes

  External bus is active to emulate, via an external tool, the normal expanded or the normal single chip mode.

11.1.5 Block Diagram

[1] shows a block diagram of the MMC.

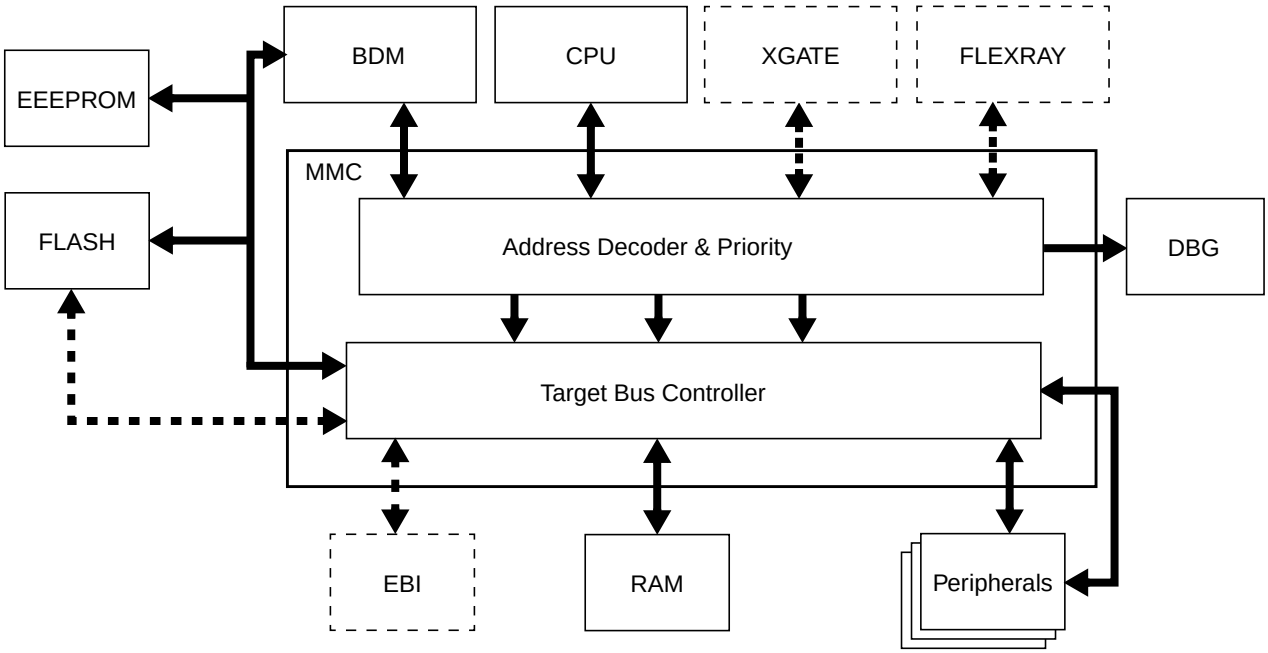


Figure 11-1. MMC Block Diagram

11.2 External Signal Description

The user is advised to refer to the SoC Guide for port configuration and location of external bus signals. Some pins may not be bonded out in all implementations.

Table 11-2 and Table 11-3 outline the pin names and functions. It also provides a brief description of their operation.

1. Doted blocks and lines are optional. Please refer to the SoC Guide for their availlibilities.

Table 11-2. External Input Signals Associated with the MMC

Signal	I/O	Description	Availability
MODC	I	Mode input	Latched after $\overline{\text{RESET}}$ (active low)
MODB	I	Mode input	Latched after $\overline{\text{RESET}}$ (active low)
MODA	I	Mode input	Latched after $\overline{\text{RESET}}$ (active low)
EROMCTL	I	EROM control input	Latched after $\overline{\text{RESET}}$ (active low)
ROMCTL	I	ROM control input	Latched after $\overline{\text{RESET}}$ (active low)

Table 11-3. External Output Signals Associated with the MMC

Signal	I/O	Description	Available in Modes					
			NS	SS	NX	ES	EX	ST
CS0	O	Chip select line 0	(see Table 11-4)					
CS1	O	Chip select line 1						
CS2	O	Chip select line 2						
CS3	O	Chip select line 3						

11.3 Memory Map and Registers

11.3.1 Module Memory Map

A summary of the registers associated with the MMC block is shown in Figure 11-2. Detailed descriptions of the registers and bits are given in the subsections that follow.

Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x000A	MMCCTL0	R	CS3E1	CS3E0	CS2E1	CS2E0	CS1E1	CS1E0	CS0E1	CS0E0
		W								
0x000B	MODE	R	MODC	MODB	MODA	0	0	0	0	0
		W								
0x0010	GPAGE	R	0	GP6	GP5	GP4	GP3	GP2	GP1	GP0
		W								
0x0011	DIRECT	R	DP15	DP14	DP13	DP12	DP11	DP10	DP9	DP8
		W								
0x0012	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0013	MMCCTL1	R	TGMRAMON	0	EEEIFRON	PGMIFRON	RAMHM	EROMON	ROMHM	ROMON
		W								
0x0014	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0015	PPAGE	R	PIX7	PIX6	PIX5	PIX4	PIX3	PIX2	PIX1	PIX0
		W								
0x0016	RPAGE	R	RP7	RP6	RP5	RP4	RP3	RP2	RP1	RP0
		W								
0x0017	EPAGE	R	EP7	EP6	EP5	EP4	EP3	EP2	EP1	EP0
		W								

= Unimplemented or Reserved

Figure 11-2. MMC Register Summary

11.3.2 Register Descriptions

11.3.2.1 MMC Control Register (MMCCTL0)

Address: 0x000A PRR

	7	6	5	4	3	2	1	0
R W	CS3E1	CS3E0	CS2E1	CS2E0	CS1E1	CS1E0	CS0E1	CS0E0
Reset	0	0	0	0	0	0	0	$\overline{\text{ROMON}}$[1]

1. ROMON is bit[0] of the register MMCTL1 (see Figure 11-10)

= Unimplemented or Reserved

Figure 11-3. MMC Control Register (MMCCTL0)

Read: Anytime. In emulation modes read operations will return the data from the external bus. In all other modes the data is read from this register.

Write: Anytime. In emulation modes write operations will also be directed to the external bus.

Table 11-4. Chip Selects Function Activity

Register Bit	Chip Modes					
	NS	SS	NX	ES	EX	ST
CS0E[1:0], CS1E[1:0], CS2E[1:0], CS3E[1:0]	Disabled(1)	Disabled	Enabled(2)	Disabled	Enabled	Disabled

1. Disabled: feature always inactive.
2. Enabled: activity is controlled by the appropriate register bit value.

The MMCCTL0 register is used to control external bus functions, like:

- Availability of chip selects. (See Table 11-4 and Table 11-5)
- Control of different external stretch mechanism. For more detail refer to the S12X_EBI BlockGuide.

CAUTION

XGATE write access to this register during an CPU access which makes use of this register could lead to unexpected results.

Table 11-5. MMCCTL0 Field Descriptions

Field	Description
7–6 CS3E[1:0]	**Chip Select 3 Enables** — These bits enable the external chip select $\overline{CS3}$ output which is asserted during accesses to specific external addresses. The associated global address range is shown in Table 11-6 and Figure 11-17. Chip select 3 is only active if enabled in Normal Expanded mode, Emulation Expanded mode. The function disabled in all other operating modes. 00 Chip select 3 is disabled 01,10,11 Chip select 3 is enabled
5–4 CS2E[1:0]	**Chip Select 2 Enables** — These bits enable the external chip select $\overline{CS2}$ output which is asserted during accesses to specific external addresses. The associated global address range is shown in Table 11-6 and Figure 11-17. Chip select 2 is only active if enabled in Normal Expanded mode, Emulation Expanded mode. The function disabled in all other operating modes. 00 Chip select 2 is disabled 01,10,11 Chip select 2 is enabled
3–2 CS1E[1:0]	**Chip Select 1 Enables** — These bits enable the external chip select $\overline{CS1}$ output which is asserted during accesses to specific external addresses. The associated global address range is shown in Table 11-6 and Figure 11-17. Chip select 1 is only active if enabled in Normal Expanded mode, Emulation Expanded mode. The function disabled in all other operating modes. 00 Chip select 1 is disabled 01,10,11 Chip select 1 is enabled
1–0 CS0E[1:0]	**Chip Select 0 Enables** — These bits enable the external chip select $\overline{CS0}$ output which is asserted during accesses to specific external addresses. The associated global address range is shown in Table 11-6 and Figure 11-17. Chip select 0 is only active if enabled in Normal Expanded mode, Emulation Expanded mode. The function disabled in all other operating modes. 00 Chip select 0 is disabled 01,10,11 Chip select 0 is enabled

Table 11-6 shows the address boundaries of each chip select and the relationship with the implemented resources (internal) parameters.

Table 11-6. Global Chip Selects Memory Space

Chip Selects	Bottom Address	Top Address
$\overline{CS3}$	0x00_0800	0x0F_FFFF minus RAMSIZE(1)
$\overline{CS2}$(2)	0x14_0000	0x1F_FFFF
$\overline{CS1}$	0x20_0000	0x3F_FFFF
$\overline{CS0}$(3)	0x40_0000	0x7F_FFFF minus FLASHSIZE(4)

1. External RPAGE accesses in (NX, EX)
2. When ROMHM is set (see ROMHM in Table 11-15) the $\overline{CS2}$ is asserted in the space occupied by this on-chip memory block.
3. When the internal NVM is enabled (see ROMON in Section 11.3.2.5, “MMC Control Register (MMCCTL1)) the $\overline{CS0}$ is not asserted in the space occupied by this on-chip memory block.
4. External PPAGE accesses in (NX, EX)

11.3.2.2 Mode Register (MODE)

Address: 0x000B PRR

	7	6	5	4	3	2	1	0
R	MODC	MODB	MODA	0	0	0	0	0
W								
Reset	MODC[1]	MODB[1]	MODA[1]	0	0	0	0	0

1. External signal (see Table 11-2).

= Unimplemented or Reserved

Figure 11-4. Mode Register (MODE)

Read: Anytime. In emulation modes read operations will return the data read from the external bus. In all other modes the data are read from this register.

Write: Only if a transition is allowed (see Figure 11-5). In emulation modes write operations will be also directed to the external bus.

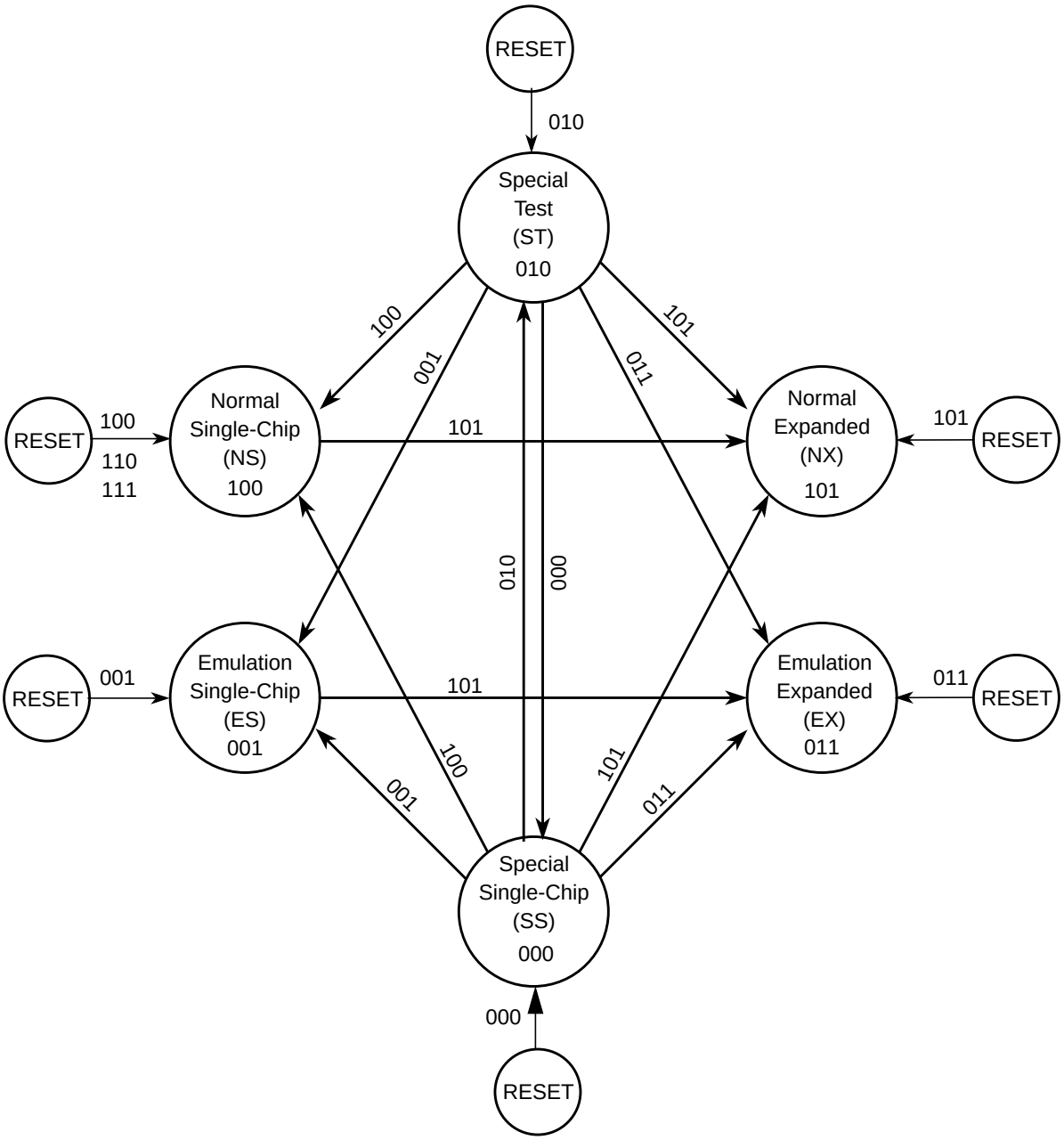
The MODE bits of the MODE register are used to establish the MCU operating mode.

CAUTION

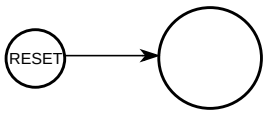
XGATE write access to this register during an CPU access which makes use of this register could lead to unexpected results.

Table 11-7. MODE Field Descriptions

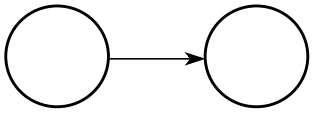
Field	Description
7–5 MODC, MODB, MODA	**Mode Select Bits** — These bits control the current operating mode during $\overline{\text{RESET}}$ high (inactive). The external mode pins MODC, MODB, and MODA determine the operating mode during $\overline{\text{RESET}}$ low (active). The state of the pins is latched into the respective register bits after the $\overline{\text{RESET}}$ signal goes inactive (see Figure 11-4). Write restrictions exist to disallow transitions between certain modes. Figure 11-5 illustrates all allowed mode changes. Attempting non authorized transitions will not change the MODE bits, but it will block further writes to these register bits except in special modes. Both transitions from normal single-chip mode to normal expanded mode and from emulation single-chip to emulation expanded mode are only executed by writing a value of 3'b101 (write once). Writing any other value will not change the MODE bits, but will block further writes to these register bits. Changes of operating modes are not allowed when the device is secured, but it will block further writes to these register bits except in special modes. In emulation modes reading this address returns data from the external bus which has to be driven by the emulator. It is therefore responsibility of the emulator hardware to provide the expected value (i.e. a value corresponding to normal single chip mode while the device is in emulation single-chip mode or a value corresponding to normal expanded mode while the device is in emulation expanded mode).



Transition done by external pins (MODC, MODB, MODA)



Transition done by write access to the MODE register



110
111

Illegal (MODC, MODB, MODA) pin values.

Do not use. (Reserved for future use).

Figure 11-5. Mode Transition Diagram when MCU is Unsecured

11.3.2.3 Global Page Index Register (GPAGE)

Address: 0x0010

	7	6	5	4	3	2	1	0
R	0	GP6	GP5	GP4	GP3	GP2	GP1	GP0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 11-6. Global Page Index Register (GPAGE)

Read: Anytime

Write: Anytime

The global page index register is used to construct a 23 bit address in the global map format. It is only used when the CPU is executing a global instruction (GLDAA, GLDAB, GLDD, GLDS, GLDX, GLDY,GSTAA, GSTAB, GSTD, GSTS, GSTX, GSTY) (see CPU Block Guide). The generated global address is the result of concatenation of the CPU local address [15:0] with the GPAGE register [22:16] (see Figure 11-7).

CAUTION

XGATE write access to this register during an CPU access which makes use of this register could lead to unexpected results.

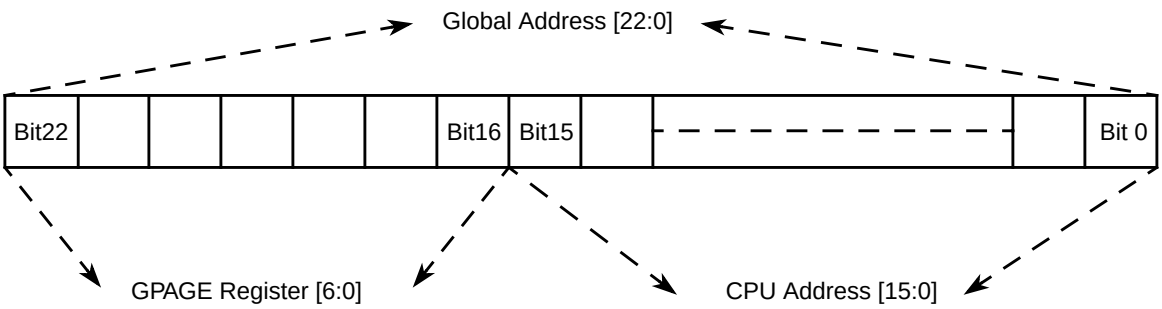


Figure 11-7. GPAGE Address Mapping

Table 11-8. GPAGE Field Descriptions

Field	Description
6–0 GP[6:0]	**Global Page Index Bits 6–0** — These page index bits are used to select which of the 128 64KB pages is to be accessed.

Example 11-1. This example demonstrates usage of the GPAGE register

```
LDX     #0x5000          ;Set GPAGE offset to the value of 0x5000
MOVB    #0x14, GPAGE     ;Initialize GPAGE register with the value of 0x14
GLDAA   X                ;Load Accu A from the global address 0x14_5000
```

11.3.2.4 Direct Page Register (DIRECT)

Address: 0x0011

	7	6	5	4	3	2	1	0
R W	DP15	DP14	DP13	DP12	DP11	DP10	DP9	DP8
Reset	0	0	0	0	0	0	0	0

Figure 11-8. Direct Register (DIRECT)

Read: Anytime

Write: anytime in special modes, one time only in other modes.

This register determines the position of the 256B direct page within the memory map.It is valid for both global and local mapping scheme.

Table 11-9. DIRECT Field Descriptions

Field	Description
7–0 DP[15:8]	**Direct Page Index Bits 15–8** — These bits are used by the CPU when performing accesses using the direct addressing mode. The bits from this register form bits [15:8] of the address (see Figure 11-9).

CAUTION

XGATE write access to this register during an CPU access which makes use of this register could lead to unexpected results.

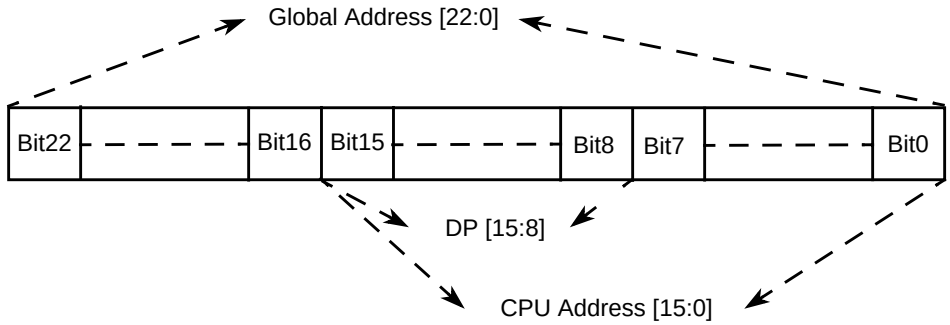


Figure 11-9. DIRECT Address Mapping

Bits [22:16] of the global address will be formed by the GPAGE[6:0] bits in case the CPU executes a global instruction in direct addressing mode or by the appropriate local address to the global address expansion (refer to Section 11.4.2.1.1, “Expansion of the Local Address Map).

Example 11-2. This example demonstrates usage of the Direct Addressing Mode

```
MOVB      #0x80,DIRECT      ;Set DIRECT register to 0x80. Write once only.
                            ;Global data accesses to the range 0xXX_80XX can be direct.
                            ;Logical data accesses to the range 0x80XX are direct.

LDY       <00               ;Load the Y index register from 0x8000 (direct access).
                            ;< operator forces direct access on some assemblers but in
                            ;many cases assemblers are “direct page aware” and can
```

```
;automatically select direct mode.
```

11.3.2.5 MMC Control Register (MMCCTL1)

Address: 0x0013 PRR

	7	6	5	4	3	2	1	0
R	MGRAMON	0	EEEIFRON	PGMIFRON	RAMHM	EROMON	ROMHM	ROMON
W								
Reset	0	0	0	0	0	EROMCTL	0	ROMCTL

= Unimplemented or Reserved

Figure 11-10. MMC Control Register (MMCCTL1)

Read: Anytime. In emulation modes read operations will return the data from the external bus. In all other modes the data are read from this register.

Write: Refer to each bit description. In emulation modes write operations will also be directed to the external bus.

CAUTION

XGATE write access to this register during an CPU access which makes use of this register could lead to unexpected results.

Table 11-10. MMCCTL1 Field Descriptions

Field	Description
7 MGRAMON	**Flash Memory Controller Tag RAM and SCRATCH RAM visible in the global memory map** Write: Anytime This bit is used to made the Flash Memory Controller Tag RAM and SCRATCH RAM visible in the global memory map. 0 Not visible in the global memory map. 1 Visible in the global memory map.
5 EEEIFRON	**EEE Information Row (IFR) visible in the global memory map** Write: Anytime This bit is used to made the IFR sector of EEE DATA FLASH visible in the global memory map. 0 Not visible in the global memory map. 1 Visible in the global memory map.
4 PGMIFRON	**Program Information Row (IFR) visible in the global memory map** Write: Anytime This bit is used to made the IFR sector of the Program Flash visible in the global memory map. 0 Not visible in the global memory map. 1 Visible in the global memory map.
3 RAMHM	**RAM only in higher Half of the global memory map** Write: Once in normal and emulation modes and anytime in special modes 0 Accesses to 0x4000–0x7FFF in the CPU/BDM local memory map will be mapped to 0x14_4000-0x14_7FFF in the global memory space (external access). 1 Accesses to 0x4000–0x7FFF in the CPU/BDM local memory map will be mapped to 0x0F_C000-0x0F_FFFF in the global memory space (RAM area).

Table 11-10. MMCCTL1 Field Descriptions (continued)

Field	Description
2 EROMON	**Enables emulated Flash or ROM memory in the global memory map** Write: Never This bit is used in some modes to define the placement of the Emulated Flash or ROM (Refer to Table 11-11) 0 Disables the emulated Flash or ROM in the global memory map. 1 Enables the emulated Flash or ROM in the global memory map.
1 ROMHM	**FLASH or ROM only in higher Half of the global memory map** Write: Once in normal and emulation modes and anytime in special modes 0 The fixed page of Flash or ROM can be accessed in the lower half of the memory map. Accesses to 0x4000–0x7FFF in the CPU/BDM local memory map will be mapped to 0x7F_4000-0x7F_7FFF in the global memory space. 1 Disables access to the Flash or ROM in the lower half of the memory map.These physical locations of the Flash or ROM can still be accessed through the program page window. Accesses to 0x4000–0x7FFF in the CPU/BDM local memory map will depends on the value of the RAMHM bit (Refer to Table 11-18).
0 ROMON	**Enable FLASH or ROM in the global memory map** Write: Once in normal and emulation modes and anytime in special modes. This bit is used in some modes to define the placement of the ROM (Refer to Table 11-11) 0 Disables the Flash or ROM from the memory map. 1 Enables the Flash or ROM in the global memory map.

EROMON and ROMON control the visibility of the Flash in the global memory map for CPU or BDM (not for XGATE). Both local and global memory maps are affected.

Table 11-11. Data Sources when CPU or BDM is Accessing Flash Area

Chip Modes	ROMON	EROMON	DATA SOURCE(1)	Stretch(2)
Normal Single Chip	X	X	Internal Flash	N
Special Single Chip				
Emulation Single Chip	X	0	Emulation Memory	N
	X	1	Internal Flash	
Normal Expanded	0	X	External Application	Y
	1	X	Internal Flash	N
Emulation Expanded	0	X	External Application	Y
	1	0	Emulation Memory	N
	1	1	Internal Flash	
Special Test	0	X	External Application	N
	1	X	Internal Flash	

1. Internal Flash means Flash resources inside the MCU are read/written.
   Emulation memory means resources inside the emulator are read/written (PRU registers, flash replacement, RAM, EEPROM and register space are always considered internal).
   External application means resources residing outside the MCU are read/written.
2. The external access stretch mechanism is part of the EBI module (refer to EBI Block Guide for details).

11.3.2.6 Program Page Index Register (PPAGE)

Address: 0x0015

	7	6	5	4	3	2	1	0
R W	PIX7	PIX6	PIX5	PIX4	PIX3	PIX2	PIX1	PIX0
Reset	1	1	1	1	1	1	1	0

Figure 11-11. Program Page Index Register (PPAGE)

Read: Anytime

Write: Anytime

These eight index bits are used to page 16KB blocks into the Flash page window located in the local (CPU or BDM) memory map from address 0x8000 to address 0xBFFF (see Figure 11-12). This supports accessing up to 4MB of Flash (in the Global map) within the 64KB Local map. The PPAGE register is effectively used to construct paged Flash addresses in the Local map format. The CPU has special access to read and write this register directly during execution of CALL and RTC instructions..

CAUTION

XGATE write access to this register during an CPU access which makes use of this register could lead to unexpected results.

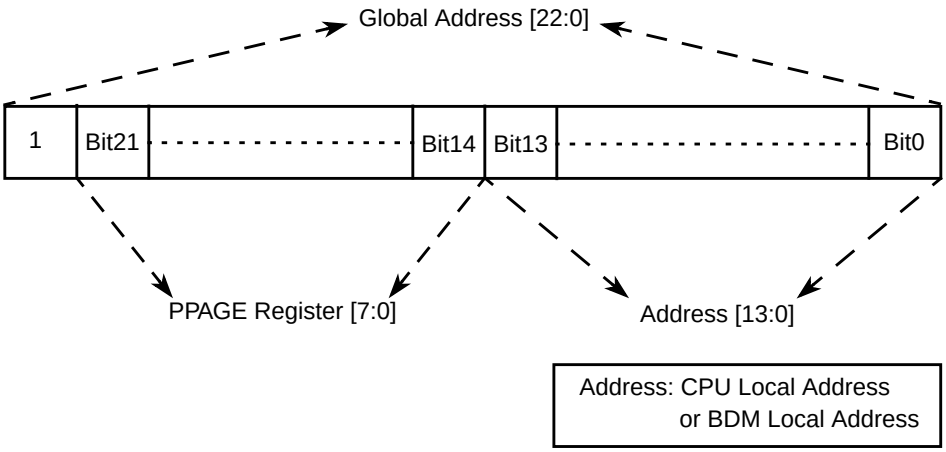


Figure 11-12. PPAGE Address Mapping

NOTE

Writes to this register using the special access of the CALL and RTC instructions will be complete before the end of the instruction execution.

Table 11-12. PPAGE Field Descriptions

Field	Description
7–0 PIX[7:0]	**Program Page Index Bits 7–0** — These page index bits are used to select which of the 256 FLASH or ROM array pages is to be accessed in the Program Page Window.

The fixed 16K page from 0x4000–0x7FFF (when ROMHM = 0) is the page number 0xFD.

The reset value of 0xFE ensures that there is linear Flash space available between addresses 0x4000 and 0xFFFF out of reset.

The fixed 16K page from 0xC000-0xFFFF is the page number 0xFF.

11.3.2.7 RAM Page Index Register (RPAGE)

Address: 0x0016

	7	6	5	4	3	2	1	0
R W	RP7	RP6	RP5	RP4	RP3	RP2	RP1	RP0
Reset	1	1	1	1	1	1	0	1

Figure 11-13. RAM Page Index Register (RPAGE)

Read: Anytime

Write: Anytime

These eight index bits are used to page 4KB blocks into the RAM page window located in the local (CPU or BDM) memory map from address 0x1000 to address 0x1FFF (see Figure 11-14). This supports

accessing up to 1022KB of RAM (in the Global map) within the 64KB Local map. The RAM page index register is effectively used to construct paged RAM addresses in the Local map format.

CAUTION

XGATE write access to this register during an CPU access which makes use of this register could lead to unexpected results.

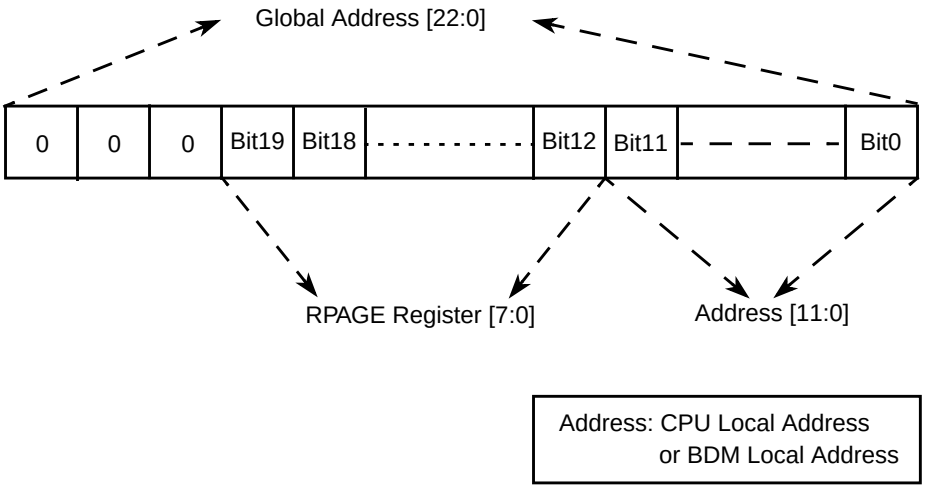


Figure 11-14. RPAGE Address Mapping

NOTE

Because RAM page 0 has the same global address as the register space, it is possible to write to registers through the RAM space when RPAGE = 0x00.

Table 11-13. RPAGE Field Descriptions

Field	Description
7–0 RP[7:0]	**RAM Page Index Bits 7–0** — These page index bits are used to select which of the 256 RAM array pages is to be accessed in the RAM Page Window.

The reset value of 0xFD ensures that there is a linear RAM space available between addresses 0x1000 and 0x3FFF out of reset.

The fixed 4K page from 0x2000–0x2FFF of RAM is equivalent to page 254 (page number 0xFE).

The fixed 4K page from 0x3000–0x3FFF of RAM is equivalent to page 255 (page number 0xFF).

NOTE

The page 0xFD (reset value) contains unimplemented area in the range not occupied by RAM if RAMSIZE is less than 12KB (Refer to Section 11.4.2.3, “Implemented Memory Map).

The two fixed 4KB pages (0xFE, 0xFF) contain unimplemented area in the range not occupied by RAM if RAMSIZE is less than 8KB (Refer to Section 11.4.2.3, “Implemented Memory Map).

11.3.2.8 EEPROM Page Index Register (EPAGE)

Address: 0x0017

	7	6	5	4	3	2	1	0
R W	EP7	EP6	EP5	EP4	EP3	EP2	EP1	EP0
Reset	1	1	1	1	1	1	1	0

Figure 11-15. EEPROM Page Index Register (EPAGE)

Read: Anytime

Write: Anytime

These eight index bits are used to page 1KB blocks into the EEPROM page window located in the local (CPU or BDM) memory map from address 0x0800 to address 0x0BFF (see Figure 11-16). This supports accessing up to 256KB of EEPROM (in the Global map) within the 64KB Local map. The EEPROM page index register is effectively used to construct paged EEPROM addresses in the Local map format.

CAUTION

XGATE write access to this register during an CPU access which makes use of this register could lead to unexpected results.

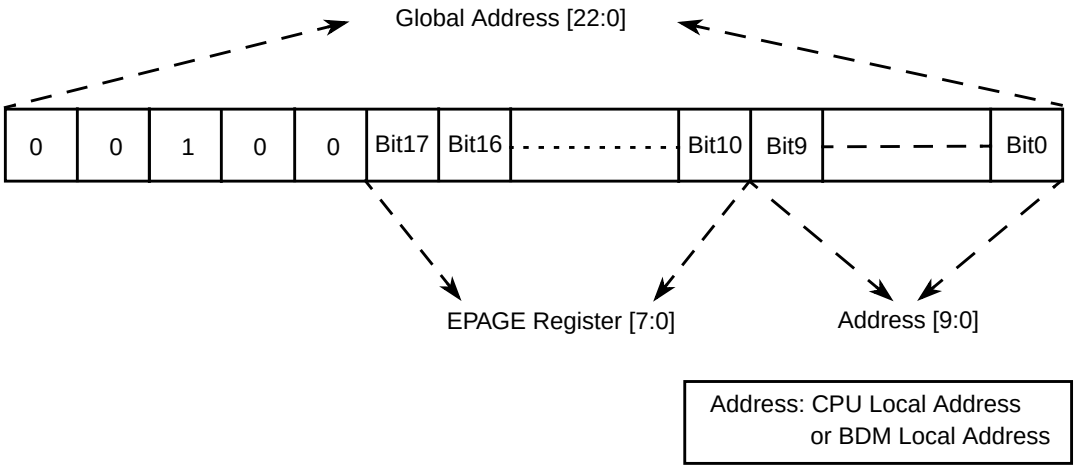


Figure 11-16. EPAGE Address Mapping

Table 11-14. EPAGE Field Descriptions

Field	Description
7–0 EP[7:0]	**EEPROM Page Index Bits 7–0** — These page index bits are used to select which of the 256 EEPROM array pages is to be accessed in the EEPROM Page Window.

The reset value of 0xFE ensures that there is a linear EEPROM space available between addresses 0x0800 and 0x0FFF out of reset.

The fixed 1K page 0x0C00–0x0FFF of EEPROM is equivalent to page 255 (page number 0xFF).

11.4 Functional Description

The MMC block performs several basic functions of the S12X sub-system operation: MCU operation modes, priority control, address mapping, select signal generation and access limitations for the system. Each aspect is described in the following subsections.

11.4.1 MCU Operating Mode

- Normal single-chip mode

  There is no external bus in this mode. The MCU program is executed from the internal memory and no external accesses are allowed.
- Special single-chip mode

  This mode is generally used for debugging single-chip operation, boot-strapping or security related operations. The active background debug mode is in control of the CPU code execution and the BDM firmware is waiting for serial commands sent through the BKGD pin. There is no external bus in this mode.
- Emulation single-chip mode

  Tool vendors use this mode for emulation systems in which the user's target application is normal single-chip mode. Code is executed from external or internal memory depending on the set-up of the EROMON bit (see Section 11.3.2.5, "MMC Control Register (MMCCTL1)). The external bus is active in both cases to allow observation of internal operations (internal visibility).

- Normal expanded mode

  The external bus interface is configured as an up to 23-bit address bus, 8 or 16-bit data bus with dedicated bus control and status signals. This mode allows 8 or 16-bit external memory and peripheral devices to be interfaced to the system. The fastest external bus rate is half of the internal bus rate. An external signal can be used in this mode to cause the external bus to wait as desired by the external logic.
- Emulation expanded mode

  Tool vendors use this mode for emulation systems in which the user's target application is normal expanded mode.
- Special test mode

  This mode is an expanded mode for factory test.

11.4.2 Memory Map Scheme

11.4.2.1 CPU and BDM Memory Map Scheme

The BDM firmware lookup tables and BDM register memory locations share addresses with other modules; however they are not visible in the global memory map during user's code execution. The BDM memory resources are enabled only during the READ_BD and WRITE_BD access cycles to distinguish between accesses to the BDM memory area and accesses to the other modules. (Refer to BDM Block Guide for further details).

When the MCU enters active BDM mode, the BDM firmware lookup tables and the BDM registers become visible in the local memory map in the range 0xFF00-0xFFFF (global address 0x7F_FF00 - 0x7F_FFFF) and the CPU begins execution of firmware commands or the BDM begins execution of hardware commands. The resources which share memory space with the BDM module will not be visible in the global memory map during active BDM mode.

Please note that after the MCU enters active BDM mode the BDM firmware lookup tables and the BDM registers will also be visible between addresses 0xBF00 and 0xBFFF if the PPAGE register contains value of 0xFF.

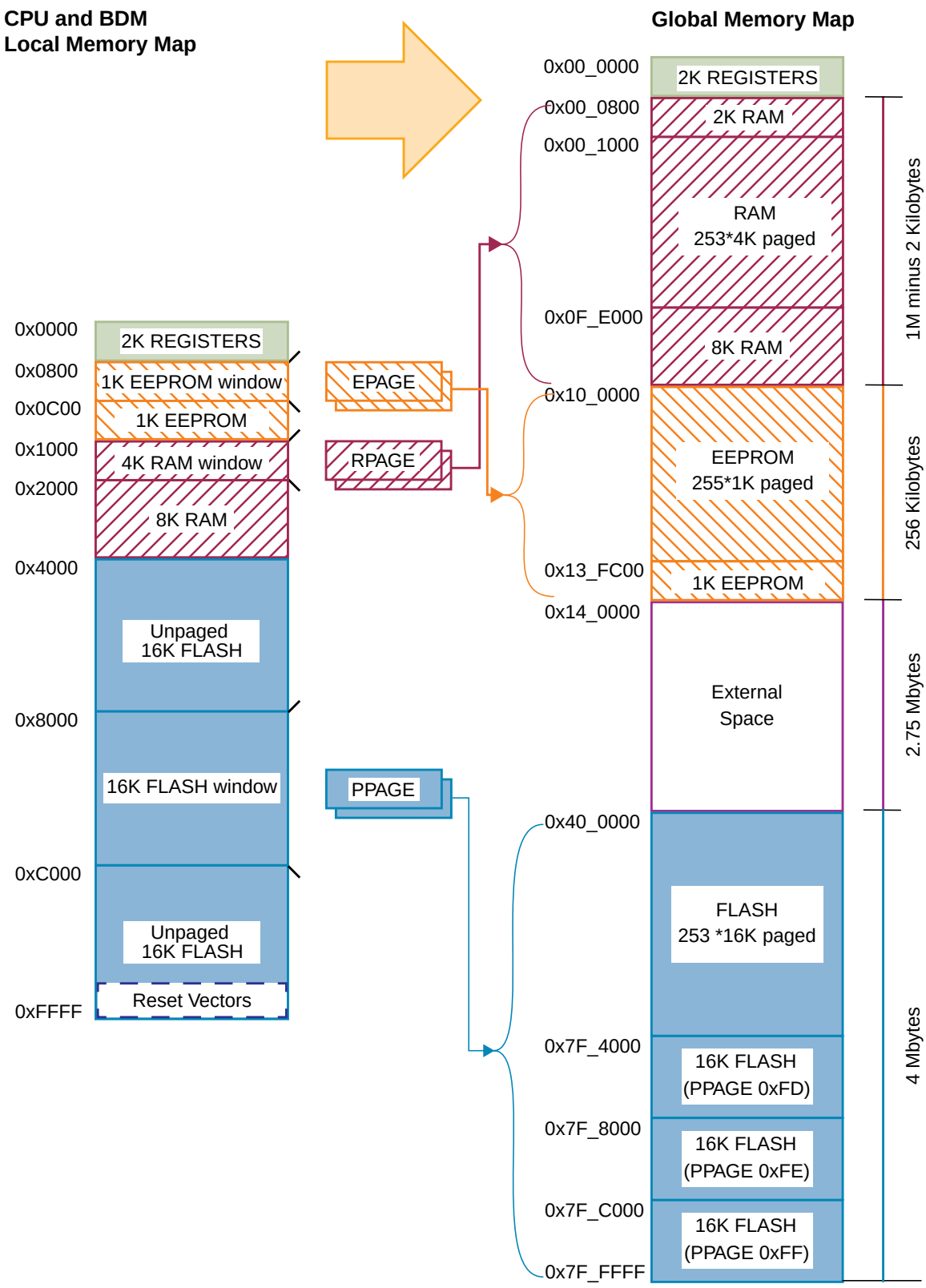


Figure 11-17. Expansion of the Local Address Map

11.4.2.1.1 Expansion of the Local Address Map

Expansion of the CPU Local Address Map

The program page index register in MMC allows accessing up to 4MB of FLASH or ROM in the global memory map by using the eight page index bits to page 256 16KB blocks into the program page window located from address 0x8000 to address 0xBFFF in the local CPU memory map.

The page value for the program page window is stored in the PPAGE register. The value of the PPAGE register can be read or written by normal memory accesses as well as by the CALL and RTC instructions (see Section 11.5.1, “CALL and RTC Instructions).

Control registers, vector space and parts of the on-chip memories are located in unpaged portions of the 64KB local CPU address space.

The starting address of an interrupt service routine must be located in unpaged memory unless the user is certain that the PPAGE register will be set to the appropriate value when the service routine is called. However an interrupt service routine can call other routines that are in paged memory. The upper 16KB block of the local CPU memory space (0xC000–0xFFFF) is unpaged. It is recommended that all reset and interrupt vectors point to locations in this area or to the other unpaged sections of the local CPU memory map.

Table 11-15 summarizes mapping of the address bus in Flash/External space based on the address, the PPAGE register value and value of the ROMHM bit in the MMCCTL1 register.

Table 11-15. Global FLASH/ROM Allocated

Local CPU Address	ROMHM	External Access	Global Address
0x4000–0x7FFF	0	No	0x7F_4000 –0x7F_7FFF
	1	Yes	0x14_4000–0x14_7FFF
0x8000–0xBFFF	N/A	No(1)	0x40_0000–0x7F_FFFF
	N/A	Yes[1]	
0xC000–0xFFFF	N/A	No	0x7F_C000–0x7F_FFFF

1. The internal or the external bus is accessed based on the size of the memory resources implemented on-chip. Please refer to Figure 1-23 for further details.

The RAM page index register allows accessing up to 1MB minus 2KB of RAM in the global memory map by using the eight RPAGE index bits to page 4KB blocks into the RAM page window located in the local CPU memory space from address 0x1000 to address 0x1FFF. The EEPROM page index register EPAGE allows accessing up to 256KB of EEPROM in the system by using the eight EPAGE index bits to page 1KB blocks into the EEPROM page window located in the local CPU memory space from address 0x0800 to address 0x0BFF.

Expansion of the BDM Local Address Map

PPAGE, RPAGE, and EPAGE registers are also used for the expansion of the BDM local address to the global address. These registers can be read and written by the BDM.

The BDM expansion scheme is the same as the CPU expansion scheme.

11.4.2.2 Global Addresses Based on the Global Page

CPU Global Addresses Based on the Global Page

The seven global page index bits allow access to the full 8MB address map that can be accessed with 23 address bits. This provides an alternative way to access all of the various pages of FLASH, RAM and EEE as well as additional external memory.

The GPAGE Register is used only when the CPU is executing a global instruction (see Section 11.3.2.3, “Global Page Index Register (GPAGE)). The generated global address is the result of concatenation of the CPU local address [15:0] with the GPAGE register [22:16] (see Figure 11-7).

BDM Global Addresses Based on the Global Page

The seven BDMGPR Global Page index bits allow access to the full 8MB address map that can be accessed with 23 address bits. This provides an alternative way to access all of the various pages of FLASH, RAM and EEE as well as additional external memory.

The BDM global page index register (BDMGPR) is used only in the case the CPU is executing a firmware command which uses a global instruction (like GLDD, GSTD) or by a BDM hardware command (like WRITE_W, WRITE_BYTE, READ_W, READ_BYTE). See the BDM Block Guide for further details.

The generated global address is a result of concatenation of the BDM local address with the BDMGPR register [22:16] in the case of a hardware command or concatenation of the CPU local address and the BDMGPR register [22:16] in the case of a firmware command (see Figure 11-18).

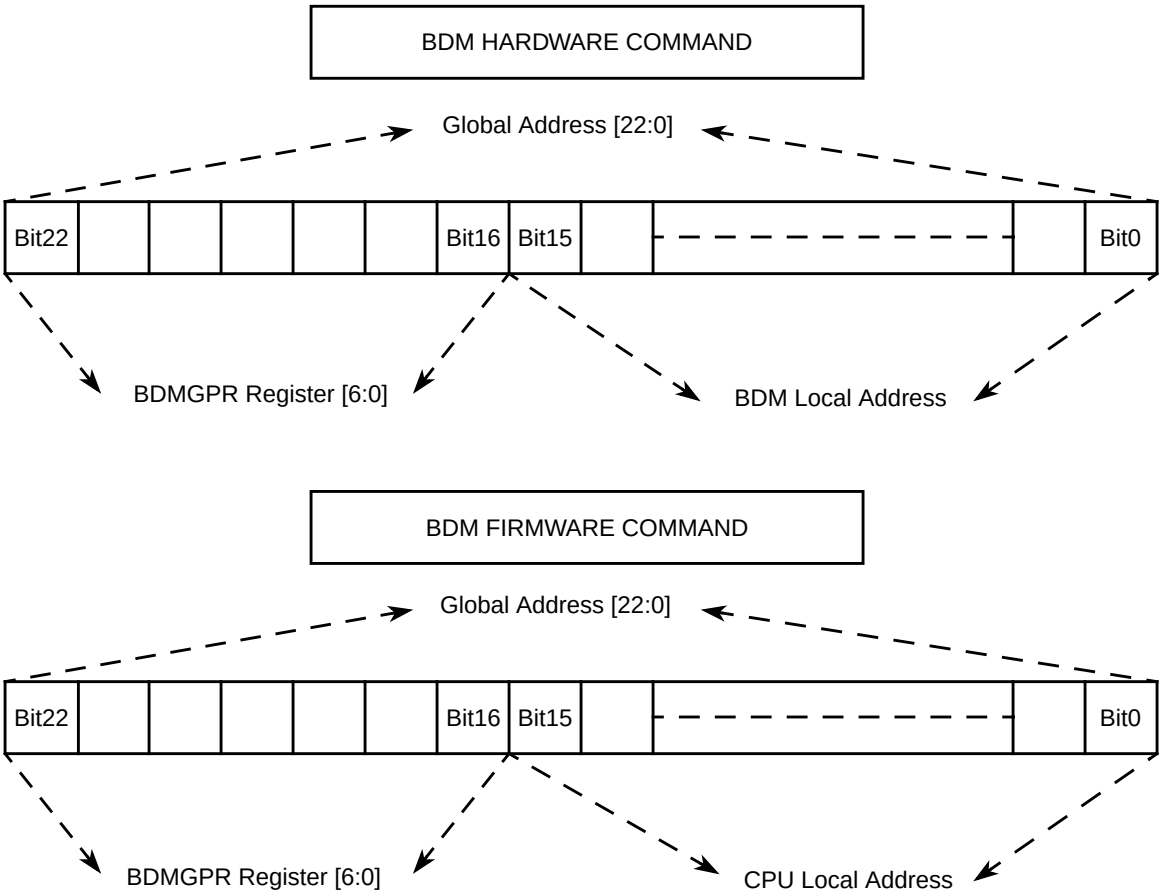


Figure 11-18. BDMGPR Address Mapping

11.4.2.3 Implemented Memory Map

The global memory spaces reserved for the internal resources (RAM, EEE, and FLASH) are not determined by the MMC module. Size of the individual internal resources are however fixed in the design of the device cannot be changed by the user. Please refer to the SoC Guide for further details. Figure and Table 11-16 show the memory spaces occupied by the on-chip resources. Please note that the memory spaces have fixed top addresses.

Table 11-16. Global Implemented Memory Space

Internal Resource	$Address
RAM	RAM_LOW = 0x10_0000 minus RAMSIZE(1)
FLASH	FLASH_LOW = 0x80_0000 minus FLASHSIZE(2)

1. RAMSIZE is the hexadecimal value of RAM SIZE in Bytes
2. FLASHSIZE is the hexadecimal value of FLASH SIZE in Bytes

When the device is operating in expanded modes except emulation single-chip mode, accesses to global addresses which are not occupied by the on-chip resources (unimplemented areas or external memory space) result in accesses to the external bus (see Figure).

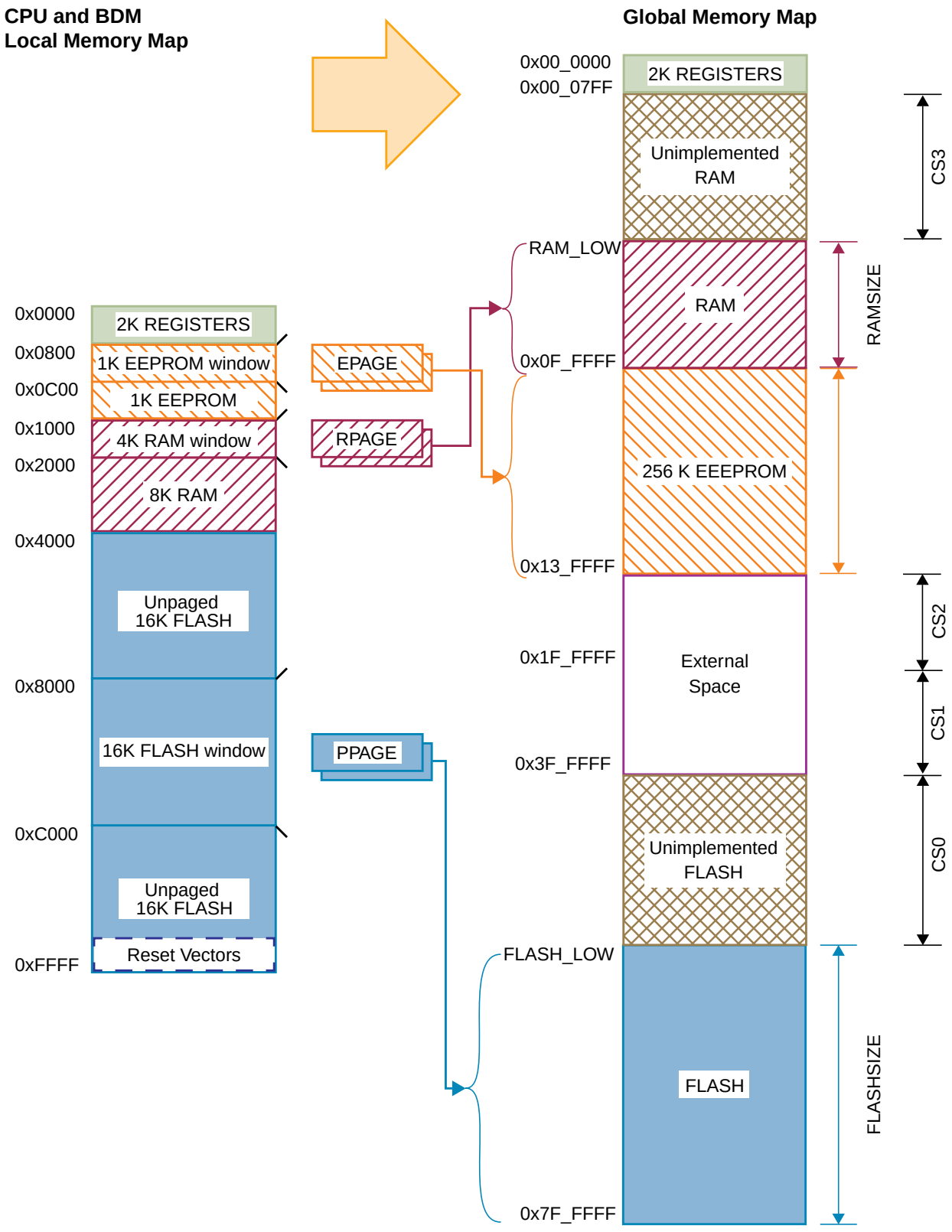
In emulation single-chip mode, accesses to global addresses which are not occupied by the on-chip resources (unimplemented areas) result in accesses to the external bus. CPU accesses to global addresses which are occupied by external memory space result in an illegal access reset (system reset) in case of no MPU error. BDM accesses to the external space are performed but the data will be undefined.

In single-chip modes accesses by the CPU (except for firmware commands) to any of the unimplemented areas (see Figure) will result in an illegal access reset (system reset) in case of no MPU error. BDM accesses to the unimplemented areas are allowed but the data will be undefined.No misaligned word access from the BDM module will occur; these accesses are blocked in the BDM module (Refer to BDM Block Guide).

Misaligned word access to the last location of RAM is performed but the data will be undefined.

Misaligned word access to the last location of any global page (64KB) by any global instruction, is performed by accessing the last byte of the page and the first byte of the same page, considering the above mentioned misaligned access cases.

The non-internal resources (unimplemented areas or external space) are used to generate the chip selects (CS0,CS1,CS2 and CS3) (see Figure), which are only active in normal expanded, emulation expanded (see Section 11.3.2.1, “MMC Control Register (MMCCTL0)).


CPU and BDM
Local Memory Map
Global Memory Map
0x0000
0x0800
0x0C00
0x1000
0x2000
0x4000
0x8000
0xC000
0xFFFF
2K REGISTERS
1K EEPROM window
1K EEPROM
4K RAM window
8K RAM
Unpaged
16K FLASH
16K FLASH window
Unpaged
16K FLASH
Reset Vectors
EPAGE
RPAGE
PPAGE
0x00_0000
0x00_07FF
2K REGISTERS
Unimplemented
RAM
RAM_LOW
RAM
0x0F_FFFF
256 K EEEPROM
0x13_FFFF
0x1F_FFFF
External
Space
0x3F_FFFF
Unimplemented
FLASH
FLASH_LOW
FLASH
0x7F_FFFF
CS3
RAMSIZE
CS2
CS1
CS0
FLASHSIZE

S12X CPU & BDM Global Address Mapping The non-internal resources (unimplemented areas or external space) are used to generate the chip selects (CS0,CS1,CS2 and CS3) (see Figure), which are only active in normal expanded, emulation expanded (see Section 11.3.2.1, “MMC Control Register (MMCCTL0)).

11.4.2.4 XGATE Memory Map Scheme

11.4.2.4.1 Expansion of the XGATE Local Address Map

The XGATE 64 Kbyte memory space allows access to internal resources only (Registers, RAM, and FLASH). The 2 Kilobyte register address range is the same register address range as for the CPU and the BDM module (see Table 11-17).

XGATE can access the FLASH in single chip modes, even when the MCU is secured. In expanded modes, XGATE can not access the FLASH when MCU is secured.

The local address of the XGATE RAM access is translated to the global RAM address range. The XGATE shares the RAM resource with the CPU and the BDM module (see Table 11-17).

XGATE RAM size (XGRAMSIZE) may be lower or equal to the MCU RAM size (RAMSIZE).

In case of XGATE RAM size less than 32 Kbytes (see Figure 11-19), the gap in the xgate local memory map will result in an illegal RAM access (see Section 11.4.3.1, “Illegal XGATE Accesses)

The local address of the XGATE FLASH access is always translated to the global address 0x78_0800 - 0x78_7FFF.

Example 11-3. is a general example of the XGATE memory map implementation.

Table 11-17. XGATE Implemented Memory Space

Internal Resource	$Address
XGATE RAM	XGRAM_LOW = 0x0F_0000 plus (0x1_0000 minus XGRAMSIZE)(1)

1. XGRAMSIZE is the hexadecimal value of XGATE RAM SIZE in bytes.

Example 11-3.

The MCU FLASHSIZE is 64 Kbytes (0x10000) and MCU RAMSIZE is 32 Kbytes (0x8000).

The XGATE RAMSIZE is 16 Kbytes (0x4000).

The space occupied by the XGATE RAM in the global address space will be:

Bottom address: (0x10_0000 minus 0x4000) = 0x0F_C000

Top address: 0x0F_FFFF

XGATE accesses to local address range 0x0800–0x7FFF will result always in accesses to the following FLASH block in the global address space:

Bottom address: 0x78_0800

Top address: 0x78_7FFF

The gap range in the local memory map 0x8000–0xBFFF will be translated in the global address space:

0x0F_8000 - 0x0F_BFFF (illegal xgate access to system RAM).

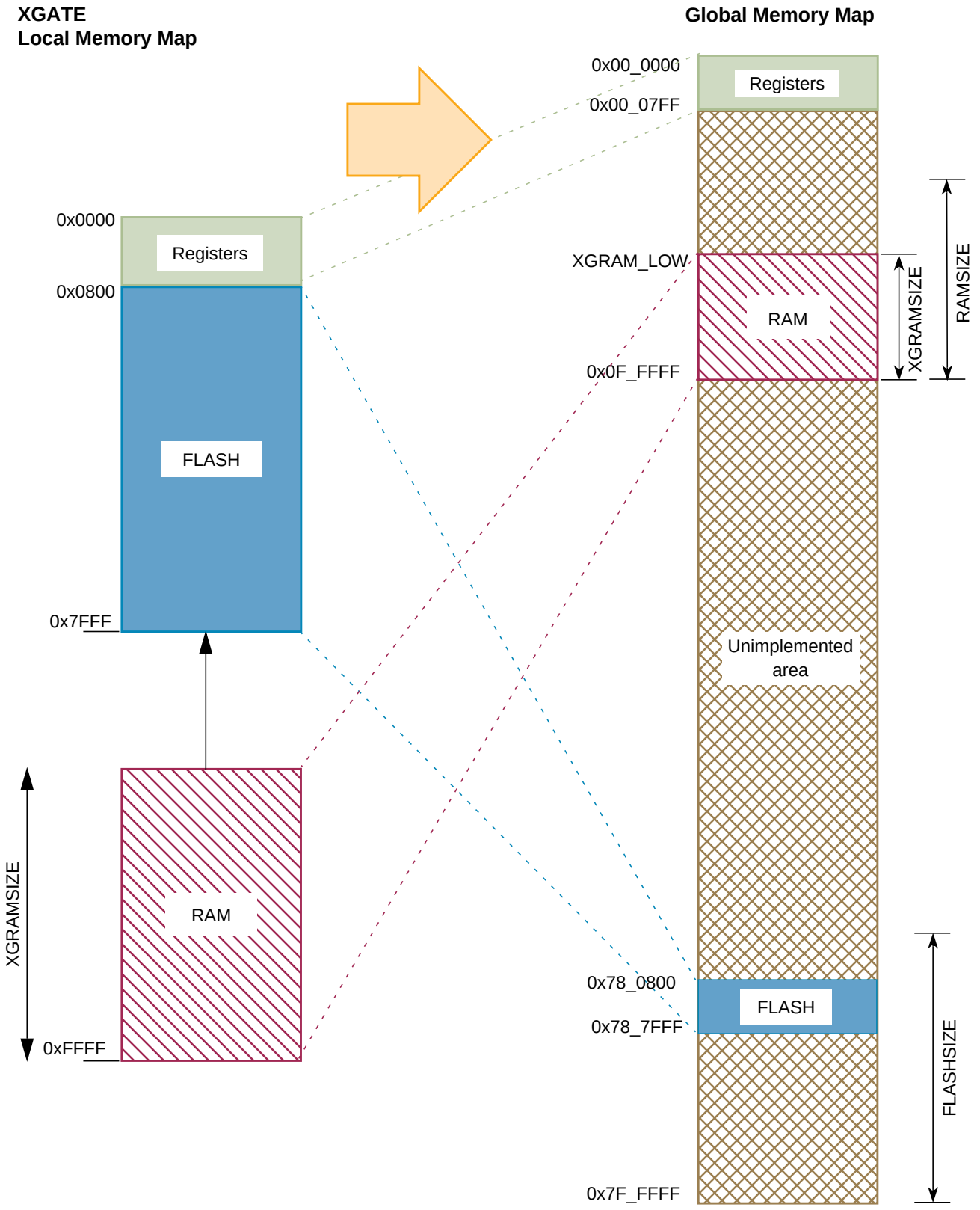


Figure 11-19. XGATE Global Address Mapping

11.4.2.5 FLEXRAY Memory Map Scheme

11.4.2.5.1 Expansion of the FLEXRAY Local Address Map

The FLEXRAY memory space allows access to internal resources only (RAM).

The local address of the FLEXRAY RAM access is connected to the global RAM address range. The FLEXRAY could share the RAM resource with the CPU, XGATE and the BDM module (Refer to the MPU Block Guide).

FLEXRAY RAM size (FLXRAMSIZE) may be lower or equal to the MCU RAM size (RAMSIZE).

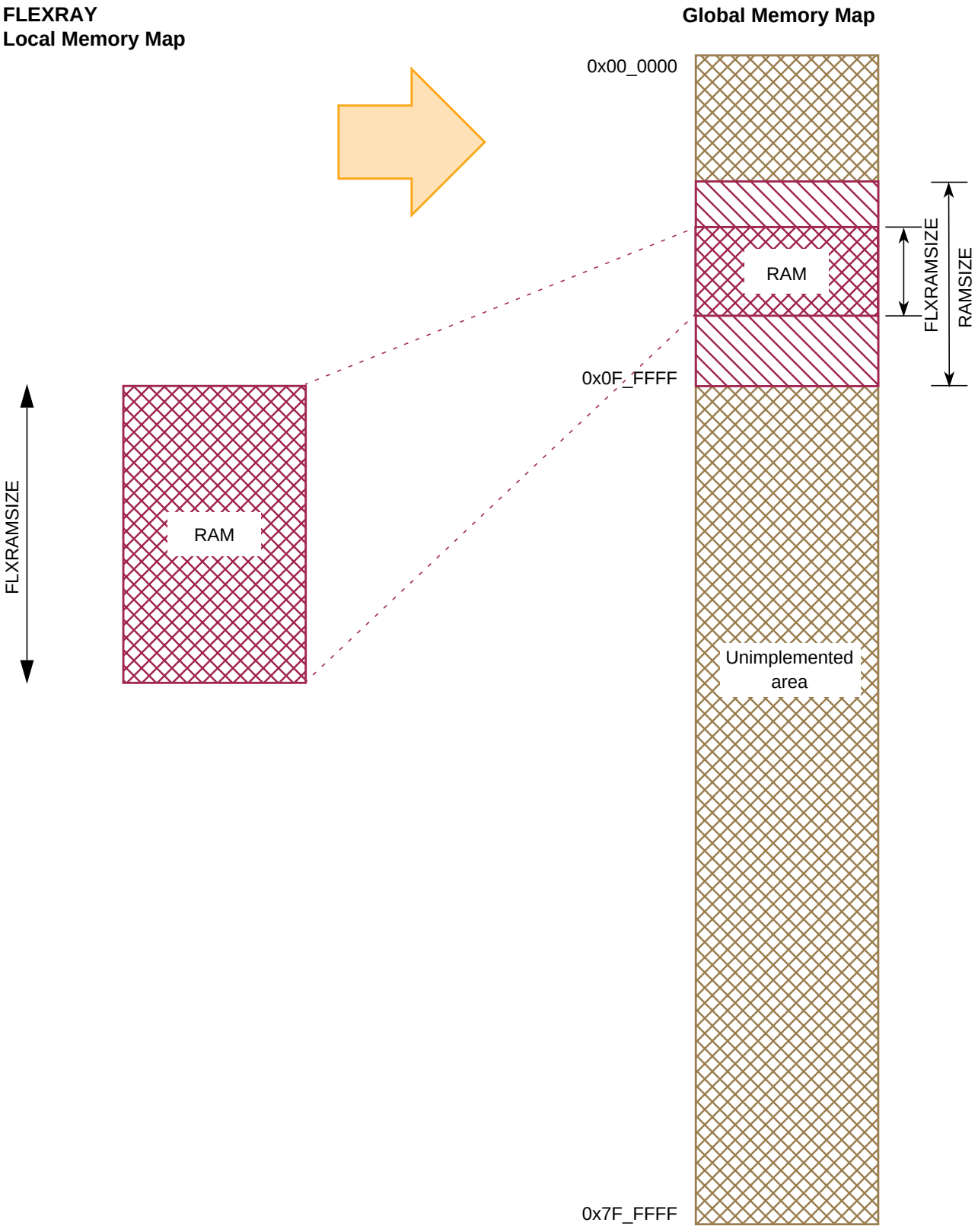


Figure 11-20. FLEXRAY Global Address Mapping

11.4.2.6 Memory Configuration

Two bits in the MMCCTL1 register (ROMHM, RAMHM) configure the mapping of the local address (0x4000-0x7FFF) in the global memory map.

ROMHM, RAMHM are write once in normal and emulation modes and anytime in special modes.

Three areas are identified (See Figure 11-21):

- Program FLASH (0x7F_4000-0x7F_7FFF) when ROMHM = 0.
- External Space (0x14_4000-0x14_7FFF) when ROMHM = 1 and RAMHM = 0.
- XSRAM Space (0x0F_C000-0x0F_FFFF) when ROMHM = 1 and RAMHM = 1.

Table 11-18 shows the translation from the local memory map to the global memory map taking in consideration the different configurations of ROMHM and RAMHM.

Table 11-18. ROMHM and RAMHM Address Location

Local Address	ROMHM	RAMHM	Global Address	Location
0x4000 - 0x7FFF	0	X	0x7F_4000 - 0x7F_7FFF	Internal Flash
	1	0	0x14_4000 - 0x14_7FFF	External Space
	1	1	0x0F_C000 - 0x0F_FFFF	Bottom of the Implemented RAM
0x2000 - 0x3FFF			0x0F_A000 - 0x0F_BFFF	Fixed up to 8K RAM
0x2000 - 0x3FFF	1	0	0x0F_E000 - 0x0F_FFFF	Fixed up to 8K RAM

Table 11-19 describes the application note of the RAM configuration and its dedicated global address.

Table 11-19. RAM Configuration

phase	RPAGE	ROMHM	RAMHM	RAM AREA	Global Address
After reset	RPAGE = 0xFD (Reset value)	0	0	12 Kilobytes	0x0F_D000 - 0x0F_FFFF
During setup	RPAGE = 0xFD (Reset value)	1	1	24 Kilobytes	0x0F_A000 - 0x0F_FFFF
Normal Operation	(0x00 <= RPAGE <= 0xF9)	1	1	28 Kilobytes	0x00_0000 - 0x0F_9FFF
	(0xFA <= RPAGE <= 0xFF)	1	1	24 Kilobytes	0x0F_A000 - 0x0F_FFFF

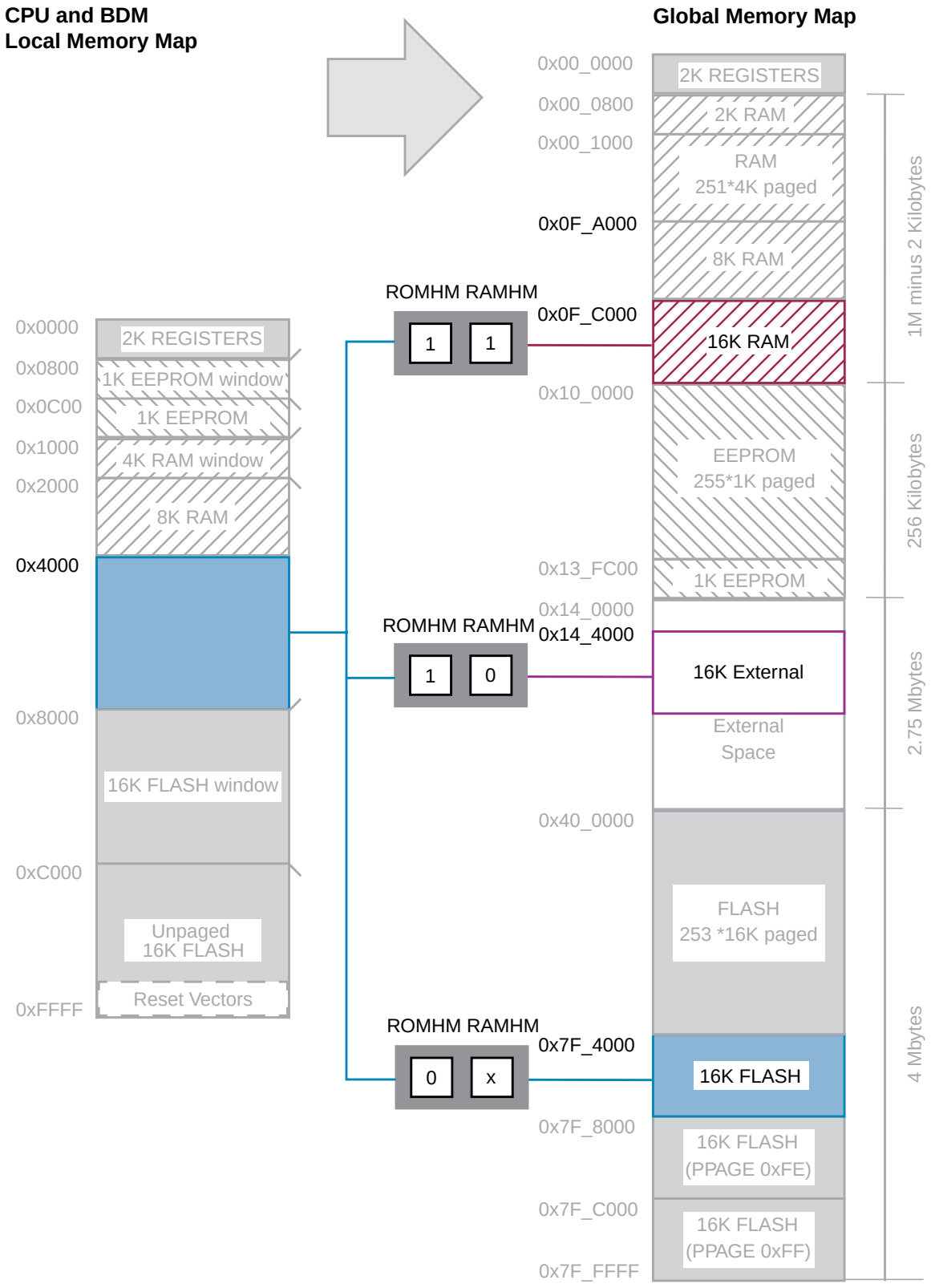


Figure 11-21. ROMHM, RAMHM Memory Configuration

11.4.2.6.1 System XSRAM

System XSRAM has two ways to be accessed by the CPU. One is by the programming of RPAGE and the fixed XSRAM areas configured by the values of ROMHM, RAMHM, or by the usage of the global instruction and the usage of GPAGE.

Figure 11-22 shows the memory map for the implemented XSRAM. The size of the implemented XSRAM is done by the device definition and denoted by RAMSIZE.

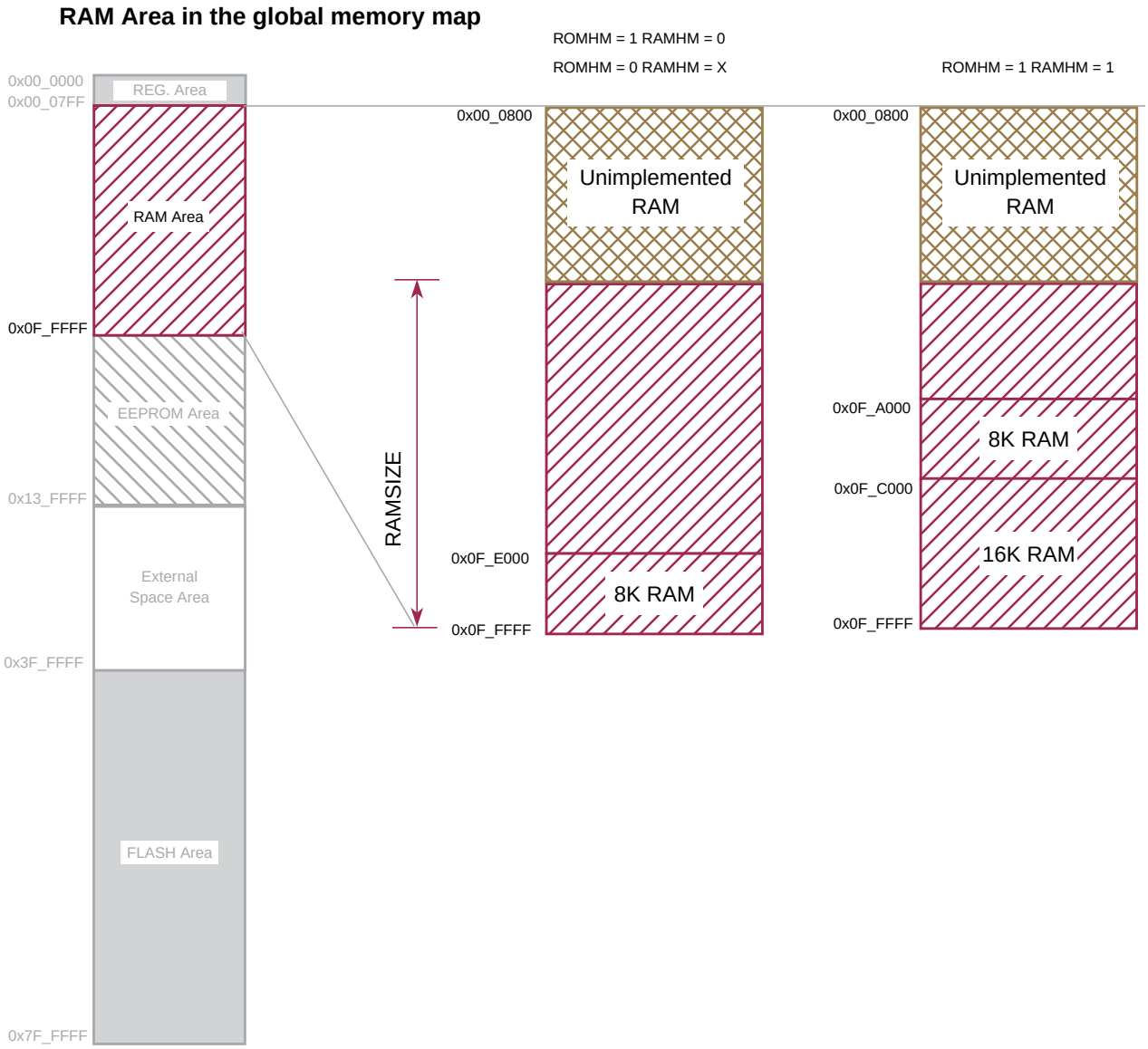


Figure 11-22. S12XE System RAM in the global memory map

11.4.3 Chip Access Restrictions

CPU, FLEXRAY and XGATE accesses are watched in the memory protection unit (See MPU Block Guide). In case of access violation, the suspect master is acknowledged with an indication of an error; the victim target will not be accessed.

Other violations MPU is not handling are listed below.

11.4.3.1 Illegal XGATE Accesses

A possible access error is flagged by the MMC and signalled to XGATE under the following conditions:

- XGATE performs misaligned word (in case of load-store or opcode or vector fetch accesses).
- XGATE accesses the register space (in case of opcode or vector fetch).
- XGATE performs a write to Flash in any modes (in case of load-store access).
- XGATE performs an access to a secured Flash in expanded modes (in case of load-store or opcode or vector fetch accesses).

For further details refer to the XGATE Block Guide.

11.4.4 Chip Bus Control

The MMC controls the address buses and the data buses that interface the S12X masters (CPU, BDM, FLEXRAY and XGATE) with the rest of the system (master buses). In addition the MMC handles all CPU read data bus swapping operations. All internal and external resources are connected to specific target buses (see Figure 11-23[1]).

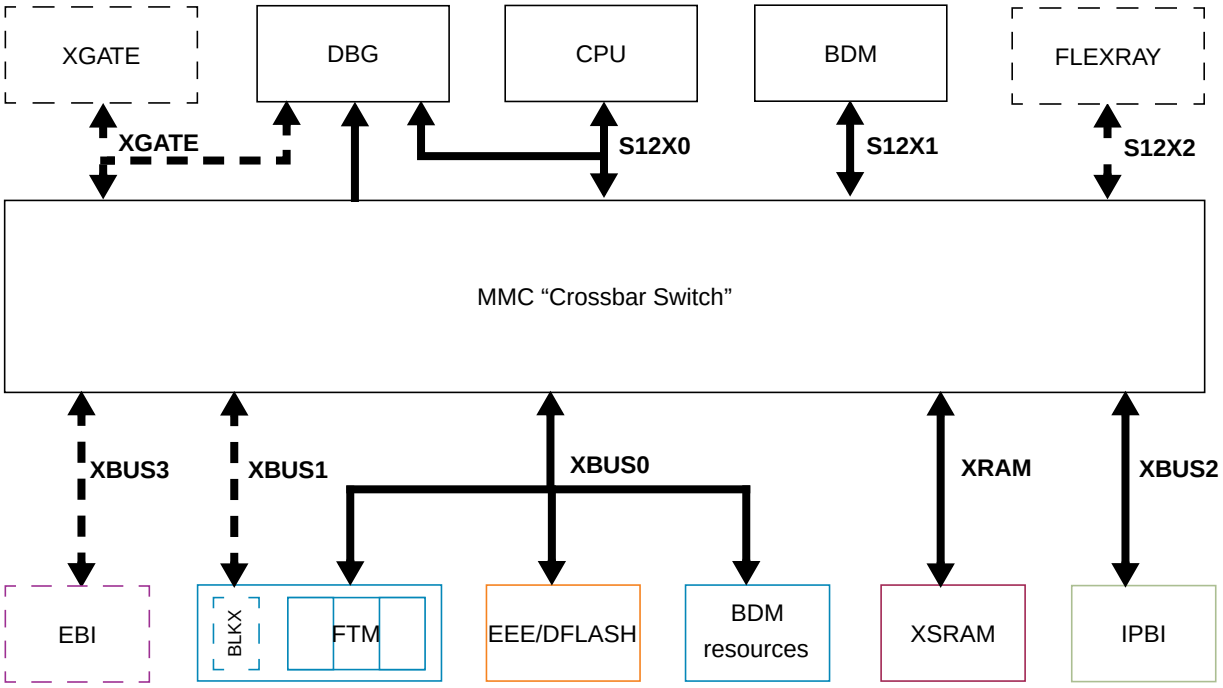


Figure 11-23. MMC Block Diagram

1. Doted blocks and lines are optional. Please refer to the SoC Guide for their availlibilities.

11.4.4.1 Master Bus Prioritization regarding access conflicts on Target Buses

The arbitration scheme allows only one master to be connected to a target at any given time. The following rules apply when prioritizing accesses from different masters to the same target bus:

- High priority[1] FLEXRAY access to XSRAM has priority over CPU, XGATE and BDM.
- CPU always has priority over BDM, FLEXRAY and XGATE.
- XGATE access to PRU registers constitutes a special case. It is always granted and stalls the CPU for its duration.
- XGATE has priority over FLEXRAY and BDM.
- BDM has priority over CPU, FLEXRAY and XGATE when its access is stalled for more than 128 cycles. In the later case the suspect master will be stalled after finishing the current operation and the BDM will gain access to the bus.
- In emulation modes all internal accesses are visible on the external bus as well and the external bus is used during access to the PRU registers.

11.5 Initialization/Application Information

11.5.1 CALL and RTC Instructions

CALL and RTC instructions are uninterruptible CPU instructions that automate page switching in the program page window. The CALL instruction is similar to the JSR instruction, but the subroutine that is called can be located anywhere in the local address space or in any Flash or ROM page visible through the program page window. The CALL instruction calculates and stacks a return address, stacks the current PPAGE value and writes a new instruction-supplied value to the PPAGE register. The PPAGE value controls which of the 256 possible pages is visible through the 16KB program page window in the 64KB local CPU memory map. Execution then begins at the address of the called subroutine.

During the execution of the CALL instruction, the CPU performs the following steps:

1. Writes the current PPAGE value into an internal temporary register and writes the new instruction-supplied PPAGE value into the PPAGE register
2. Calculates the address of the next instruction after the CALL instruction (the return address) and pushes this 16-bit value onto the stack
3. Pushes the temporarily stored PPAGE value onto the stack
4. Calculates the effective address of the subroutine, refills the queue and begins execution at the new address

This sequence is uninterruptible. There is no need to inhibit interrupts during the CALL instruction execution. A CALL instruction can be performed from any address to any other address in the local CPU memory space.

The PPAGE value supplied by the instruction is part of the effective address of the CPU. For all addressing mode variations (except indexed-indirect modes) the new page value is provided by an immediate operand in the instruction. In indexed-indirect variations of the CALL instruction a pointer specifies memory locations where the new page value and the address of the called subroutine are stored. Using indirect

1. FLEXRAY has two priority access types, one called high priority access and the other called normal access.

addressing for both the new page value and the address within the page allows usage of values calculated at run time rather than immediate values that must be known at the time of assembly.

The RTC instruction terminates subroutines invoked by a CALL instruction. The RTC instruction unstacks the PPAGE value and the return address and refills the queue. Execution resumes with the next instruction after the CALL instruction.

During the execution of an RTC instruction the CPU performs the following steps:

1. Pulls the previously stored PPAGE value from the stack
2. Pulls the 16-bit return address from the stack and loads it into the PC
3. Writes the PPAGE value into the PPAGE register
4. Refills the queue and resumes execution at the return address

This sequence is uninterruptible. The RTC can be executed from anywhere in the local CPU memory space.

The CALL and RTC instructions behave like JSR and RTS instruction, they however require more execution cycles. Usage of JSR/RTS instructions is therefore recommended when possible and CALL/RTC instructions should only be used when needed. The JSR and RTS instructions can be used to access subroutines that are already present in the local CPU memory map (i.e. in the same page in the program memory page window for example). However calling a function located in a different page requires usage of the CALL instruction. The function must be terminated by the RTC instruction. Because the RTC instruction restores contents of the PPAGE register from the stack, functions terminated with the RTC instruction must be called using the CALL instruction even when the correct page is already present in the memory map. This is to make sure that the correct PPAGE value will be present on stack at the time of the RTC instruction execution.

11.5.2 Port Replacement Registers (PRRs)

Registers used for emulation purposes must be rebuilt by the in-circuit emulator hardware to achieve full emulation of single chip mode operation. These registers are called port replacement registers (PRRs) (see Table 1-25). PRRs are accessible from CPU, BDM and XGATE using different access types (word aligned, word-misaligned and byte).

Each access to PRRs will be extended to 2 bus cycles for write or read accesses independent of the operating mode. In emulation modes all write operations result in simultaneous writing to the internal registers (peripheral access) and to the emulated registers (external access) located in the PRU in the emulator. All read operations are performed from external registers (external access) in emulation modes. In all other modes the read operations are performed from the internal registers (peripheral access).

Due to internal visibility of CPU accesses the CPU will be halted during XGATE or BDM access to any PRR. This rule applies also in normal modes to ensure that operation of the device is the same as in emulation modes.

A summary of PRR accesses:

- An aligned word access to a PRR will take 2 bus cycles.

- A misaligned word access to a PRRs will take 4 cycles. If one of the two bytes accessed by the misaligned word access is not a PRR, the access will take only 3 cycles.
- A byte access to a PRR will take 2 cycles.

Table 11-20. PRR Listing

PRR Name	PRR Local Address	PRR Location
PORTA	0x0000	PIM
PORTB	0x0001	PIM
DDRA	0x0002	PIM
DDRB	0x0003	PIM
PORTC	0x0004	PIM
PORTD	0x0005	PIM
DDRC	0x0006	PIM
DDRD	0x0007	PIM
PORTE	0x0008	PIM
DDRE	0x0009	PIM
MMCCTL0	0x000A	MMC
MODE	0x000B	MMC
PUCR	0x000C	PIM
RDRIV	0x000D	PIM
EBICTL0	0x000E	EBI
EBICTL1	0x000F	EBI
Reserved	0x0012	MMC
MMCCTL1	0x0013	MMC
ECLKCTL	0x001C	PIM
Reserved	0x001D	PIM
PORTK	0x0032	PIM
DDRK	0x0033	PIM

11.5.3 On-Chip ROM Control

The MCU offers two modes to support emulation. In the first mode (called generator) the emulator provides the data instead of the internal FLASH and traces the CPU actions. In the other mode (called observer) the internal FLASH provides the data and all internal actions are made visible to the emulator.

11.5.3.1 ROM Control in Single-Chip Modes

In single-chip modes the MCU has no external bus. All memory accesses and program fetches are internal (see Figure 11-24).

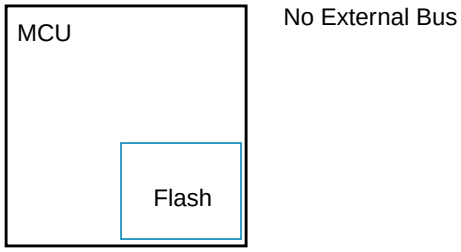


Figure 11-24. ROM in Single Chip Modes

11.5.3.2 ROM Control in Emulation Single-Chip Mode

In emulation single-chip mode the external bus is connected to the emulator. If the EROMON bit is set, the internal FLASH provides the data and the emulator can observe all internal CPU actions on the external bus. If the EROMON bit is cleared, the emulator provides the data (generator) and traces the all CPU actions (see Figure 11-25).

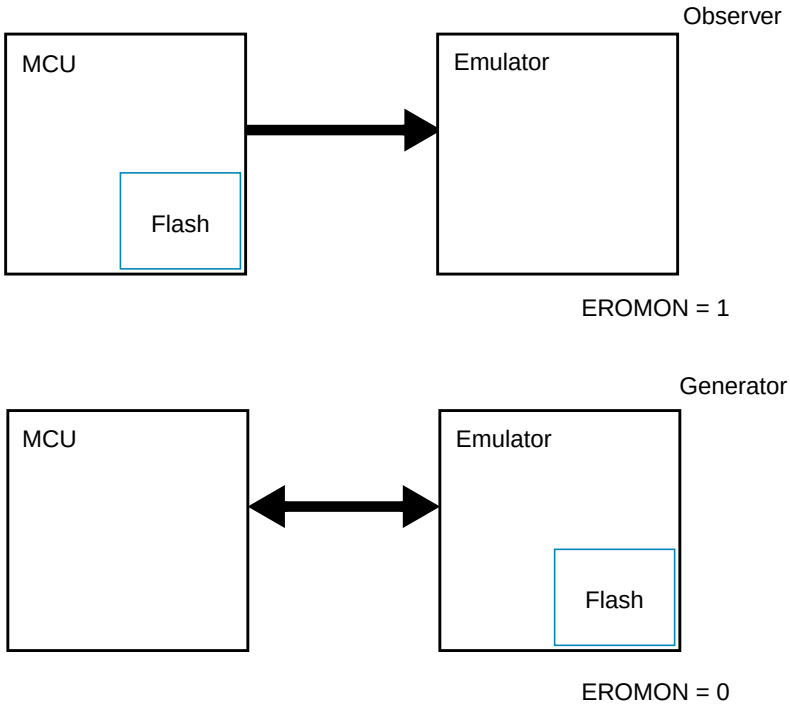


Figure 11-25. ROM in Emulation Single-Chip Mode

11.5.3.3 ROM Control in Normal Expanded Mode

In normal expanded mode the external bus will be connected to the application. If the ROMON bit is set, the internal FLASH provides the data. If the ROMON bit is cleared, the application memory provides the data (see Figure 11-26).

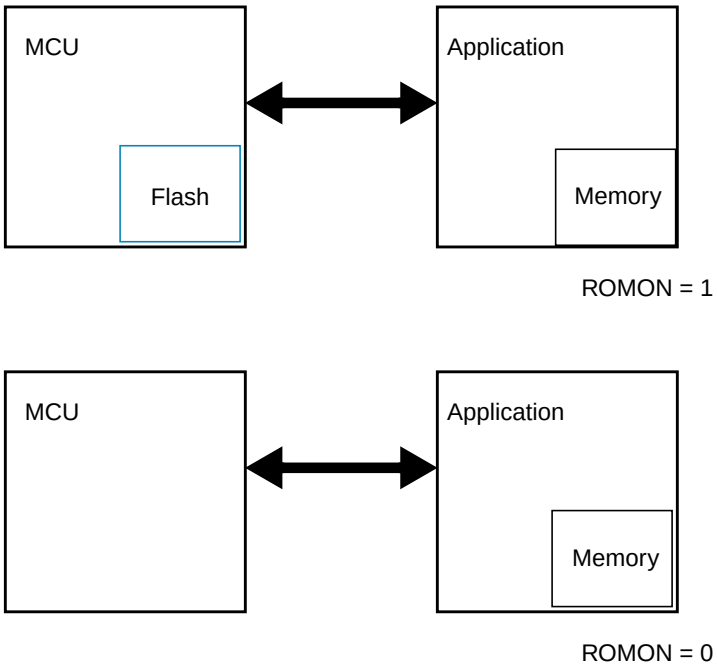


Figure 11-26. ROM in Normal Expanded Mode

11.5.3.4 ROM Control in Emulation Expanded Mode

In emulation expanded mode the external bus will be connected to the emulator and to the application. If the ROMON bit is set, the internal FLASH provides the data. If the EROMON bit is set as well the emulator observes all CPU internal actions, otherwise the emulator provides the data and traces all CPU actions (see Figure 11-27). When the ROMON bit is cleared, the application memory provides the data and the emulator will observe the CPU internal actions (see Figure 11-28).

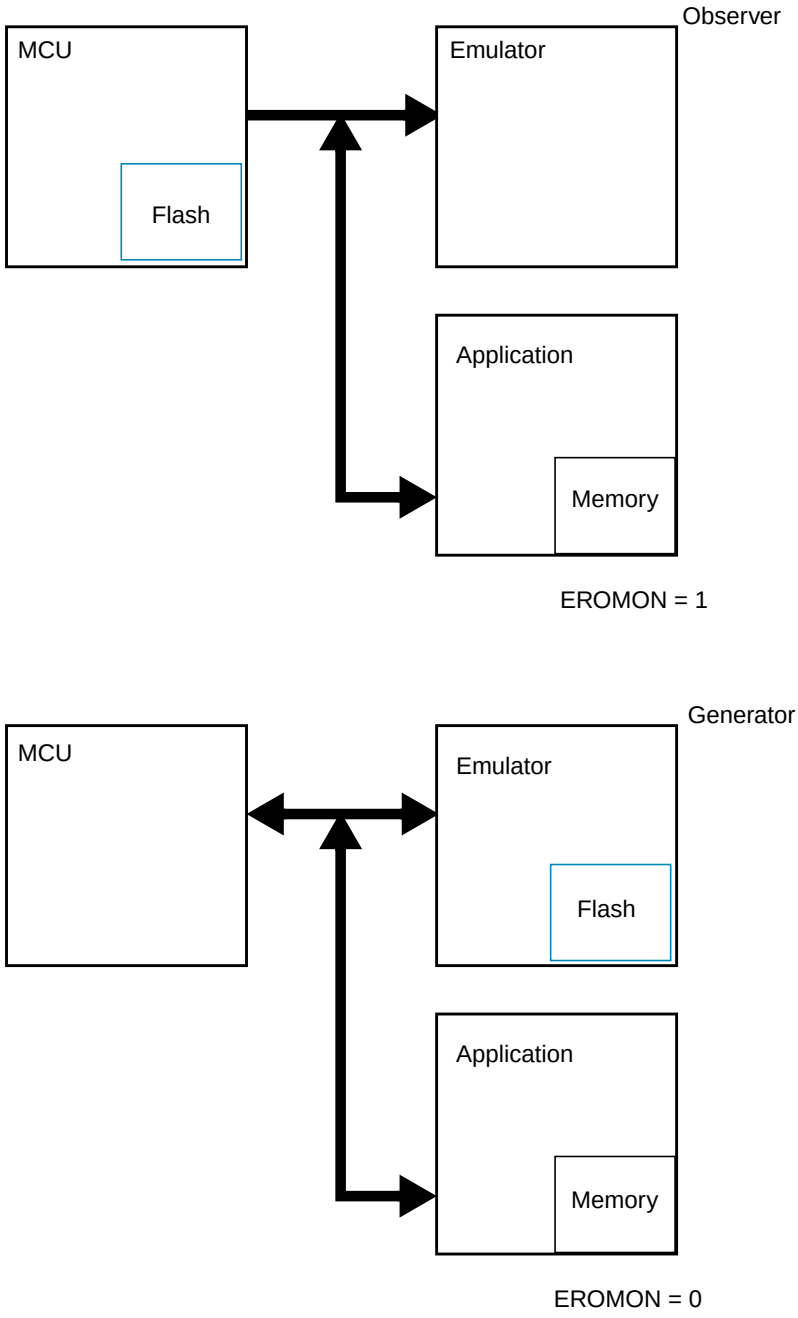


Figure 11-27. ROMON = 1 in Emulation Expanded Mode

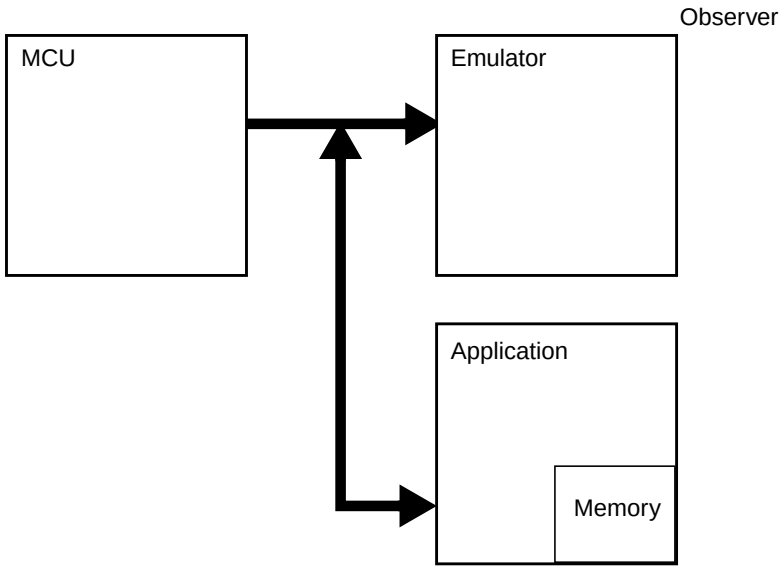


Figure 11-28. ROMON = 0 in Emulation Expanded Mode

11.5.3.5 ROM Control in Special Test Mode

In special test mode the external bus is connected to the application. If the ROMON bit is set, the internal FLASH provides the data, otherwise the application memory provides the data (see Figure 11-29).

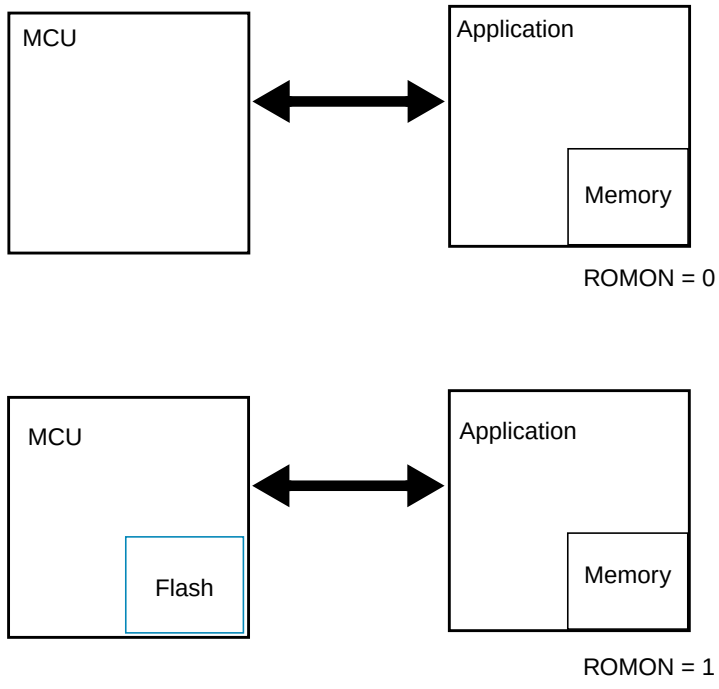


Figure 11-29. ROM in Special Test Mode

Chapter 12
Clock Generation Module using IPLL (CGMIPLL) Block Description

Revision History

Version Number	Revision Date	Effective Date	Author	Description of Changes
V01.00	4 Oct.. 06	4 Oct. 06		Initial release

12.1 Introduction

This specification describes the function of the Clock Generation Module using an internal PLL (CGMIPLL).

12.1.1 Features

The main features of this block are:

- Phase Locked Loop (IPLL) frequency multiplier with internal filter
  - Reference divider
  - optional divide by 2 of VCO frequency
  - Configurable internal filter (no external pin)
  - Optional frequency modulation for defined jitter and reduced emission
  - Automatic frequency lock detector
  - Interrupt request on entry or exit from locked condition

12.1.2 Modes of Operation

This subsection lists and briefly describes all operating modes supported by the CGMIPLL.

- Off Mode

  Default after reset, as PLLON bit is zero
- Run Mode

  PLLON bit is one, CGMIPLL registers must be configured for desired target frequency.
- Stop Mode

In Stop Model the IPLL is always off.

12.1.3 Block Diagram

Figure 12-1 shows a block diagram of the CGMIPLL.

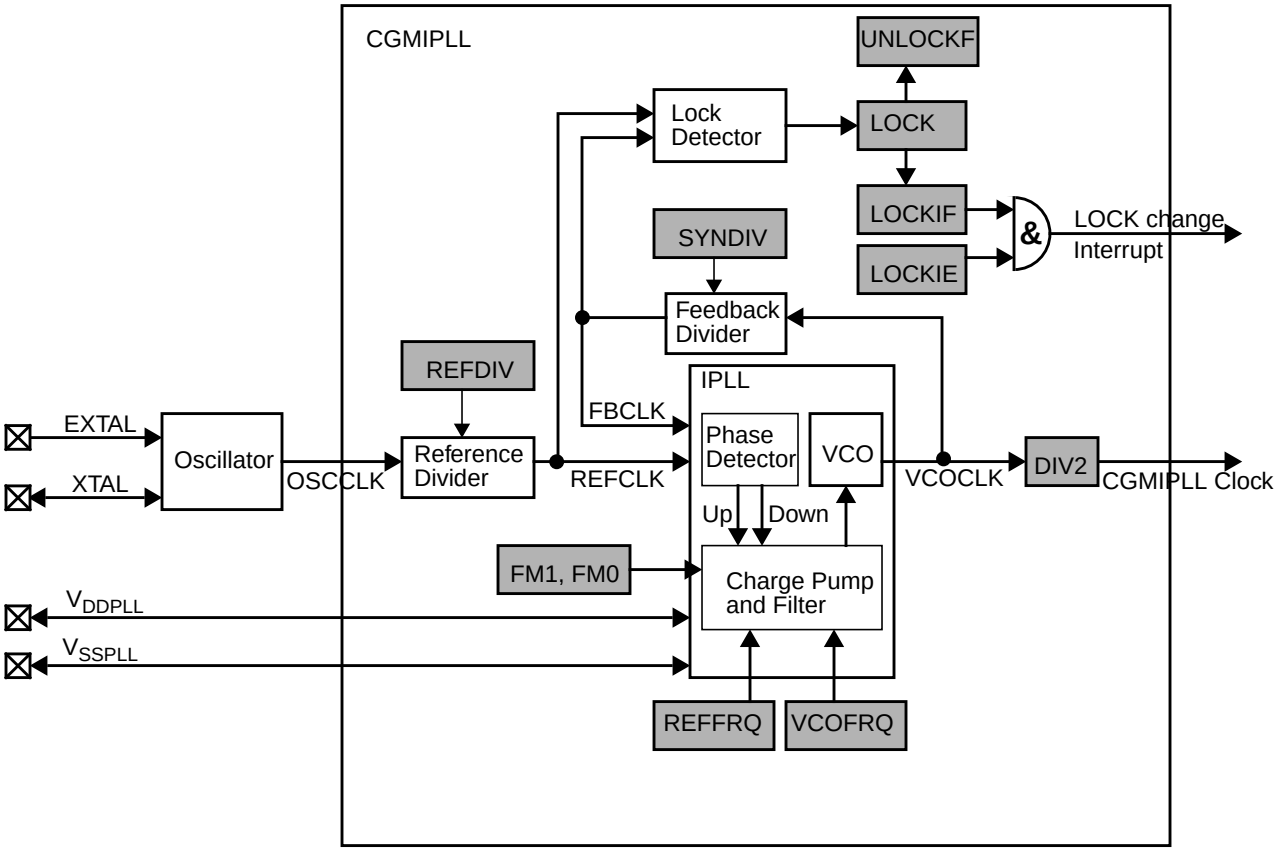


Figure 12-1. Block diagram of CGMIPLL

12.2 Signal Description

This section lists and describes the signals that connect off chip.

12.2.1 V_{DDPLL}, V_{SSPLL}

These pins provides operating voltage (V_{DDPLL}) and ground (V_{SSPLL}) for the IPLL circuitry. This allows the supply voltage to the IPLL to be independently bypassed. Even if IPLL usage is not required V_{DDPLL} and V_{SSPLL} must be connected properly.

12.3 Memory Map and Registers

This section provides a detailed description of all registers accessible in the CGMIPLL.

12.3.1 Module Memory Map

Figure 12-2 gives an overview on all CGMIPLL registers.

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000	CGMSYNR	R	VCOFRQ[1:0]		SYNDIV[5:0]					
		W								
0x0001	CGMREFDV	R	REFFRQ[1:0]		REFDIV[5:0]					
		W								
0x0002	RESERVED	R	0	0	0	0	0	0	0	0
		W								
0x0003	CGMFLG	R	LOCKIE	0	0	LOCKIF	LOCK	0	0	UNLOCKF
		W								
0x0004	CGMCTL	R	0	0	0	0	DIV2	FM1	FM0	PLLON
		W								
0x0005	CGMTEST0[2]	R	0	0	0	0	0	0	0	0
		W								
0x0006	CGMTEST1[2]	R	0	0	0	0	0	0	0	0
		W								
0x0007	CGMTEST2[2]	R	0	0	0	0	0	0	0	0
		W								

2. CGMTEST0, CGMTEST1 and CGMTEST2 registers are intended for factory test purposes only.

= Unimplemented or Reserved

Figure 12-2. CGMIPLL Register Summary

NOTE

Register Address = Base Address + Address Offset, where the Base Address is defined at the MCU level and the Address Offset is defined at the module level.

12.3.2 Register Descriptions

This section describes in address order all the CGMIPLL registers and their individual bits.

12.3.2.1 CGMIPLL Synthesizer Register (CGMSYNR)

The SYNR register controls the multiplication factor of the IPLL and selects the VCO frequency range.

Module Base + 0x0000

	7	6	5	4	3	2	1	0
R W	VCOFRQ[1:0]		SYNDIV[5:0]					
Reset	0	0	0	0	0	0	0	0

Figure 12-3. CGMIPLL Synthesizer Register (CGMSYNR)

Read: Anytime

Write: Anytime

Writing the CGMSYNR register clears the LOCK status bit.

$$f_{VCO} = 2 \times f_{OSC} \times \frac{(SYNDIV + 1)}{(REFDIV + 1)}$$

$$f_{CGMIPLL} = f_{VCO} \quad \text{(IF DIV2=0)}$$

$$f_{CGMIPLL} = \frac{f_{VCO}}{2} \quad \text{(IF DIV2=1)}$$

NOTE

f_{VCO} must be within the specified VCO frequency lock range. $f_{CGMIPLL}$ must not exceed the specified maximum.

The VCOFRQ[1:0] bit are used to configure the VCO gain for optimal stability and lock time. For correct IPLL operation the VCOFRQ[1:0] bits have to be selected according to the actual target VCOCLK frequency as shown in Table 12-1. Setting the VCOFRQ[1:0] bits wrong can result in a non functional IPLL (no locking and/or insufficient stability).

Table 12-1. VCO Clock Frequency Selection

VCOCLK Frequency Ranges	VCOFRQ[1:0]
32MHz <= f_{VCO}<= 48MHz	00
48MHz < f_{VCO}<= 80MHz	01
Reserved	10
80MHz < f_{VCO} <= 120MHz	11

12.3.2.2 CGMIPLL Reference Divider Register (CGMREFDV)

The REFDV register provides a finer granularity for the IPLL multiplier steps.

Module Base + 0x0001

	7	6	5	4	3	2	1	0
R	REFFRQ[1:0]		REFDIV[5:0]					
W								
Reset	0	0	0	0	0	0	0	0

Figure 12-4. CGMIPLL Reference Divider Register (CGMREFDV)

Read: Anytime

Write: Anytime

Writing the CGMREFDV register clears the LOCK status bit.

$$f_{REF} = \frac{f_{OSC}}{(REFDIV + 1)}$$

The REFFRQ[1:0] bit are used to configure the internal PLL filter for optimal stability and lock time. For correct IPLL operation the REFFRQ[1:0] bits have to be selected according to the actual REFCLK frequency as shown in Figure 12-2. Setting the REFFRQ[1:0] bits wrong can result in a non functional IPLL (no locking and/or insufficient stability).

Table 12-2. Reference Clock Frequency Selection

REFCLK Frequency Ranges	REFFRQ[1:0]
1MHz <= f_{REF} <= 2MHz	00
2MHz < f_{REF} <= 6MHz	01
6MHz < f_{REF} <= 12MHz	10
f_{REF} >12MHz	11

12.3.2.3 CGMIPLL Flags Register (CGMFLG)

This register provides CGMIPLL status bits and flags.

Module Base + 0x0003

	7	6	5	4	3	2	1	0
R	LOCKIE	0	0	LOCKIF	LOCK	0	0	UNLOCKF
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 12-5. CGMIPLL Flags Register (CGMFLG)

Read: Anytime

Write: Refer to each bit for individual write conditions

Table 12-3. CGMFLG Field Descriptions

Field	Description
7 LOCKIE	**Lock Interrupt Enable Bit** 0 LOCK interrupt requests are disabled. 1 Interrupt will be requested whenever LOCKIF is set.
4 LOCKIF	IPLL **Lock Interrupt Flag** —LOCKIF is set to 1 when LOCK status bit changes. This flag can only be cleared by writing a 1. Writing a 0 has no effect.If enabled (LOCKIE=1), LOCKIF causes an interrupt request. 0 No change in LOCK status bit. 1 LOCK status bit has changed.
3 LOCK	**IPLL Lock Status Bit** — LOCK reflects the current state of IPLL lock condition. Writes have no effect. Writing registers CGMSYNR or CGMREFDV or CGMCTL clears the LOCK status bit. 0 VCOCLK is not within the desired tolerance of the target frequency. 1 VCOCLK is within the desired tolerance of the target frequency.
0 UNLOCKF	IPLL **Unlock Flag** —UN LOCKF flag is set to 1 when LOCK status bit changes from locked (one) to unlocked (zero). This flag can only be cleared by writing a 1. Writing a 0 has no effect. 0 No change from locked (one) to unlocked (zero). 1 LOCK bit has changed from locked (one) to unlocked (zero).

12.3.2.4 CGMIPLL Control Register (CGMCTL)

Module Base + 0x0004

	7	6	5	4	3	2	1	0
R	0	0	0	0	DIV2	FM1	FM0	PLLON
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 12-6. CGMIPLL Control Register (CGMCTL)

Read: Anytime

Write: Anytime

Writing the CGMCTL register clears the LOCK status bit.

Table 12-4. CGMCTL Field Descriptions

Field	Description
4 DIV2	**VCOCLK divide by 2 Bit** 0 CGMIPLL Clock equals VCOCLK. 1 CGMIPLL Clock is half the frequency of VCOCLK.

Table 12-4. CGMCTL Field Descriptions (continued)

Field	Description
2, 1 FM1, FM0	IPLL **Frequency Modulation Enable Bit** — FM1 and FM0 enable additional frequency modulation on the VCOCLK. This is to reduce noise emission. The modulation frequency is f_{ref} divided by 16. See Table 12-5 for coding.
0 PLLON	**Phase Lock Loop On Bit** — PLLON turns on the IPLL circuitry. 0 IPLL is turned off. 1 IPLL is turned on.

Table 12-5. FM Amplitude selection

FM1	FM0	FM Amplitude / f_{VCO} Variation
0	0	FM off
0	1	±1%
1	0	±2%
1	1	±4%

12.3.2.5 Reserved Register (CGMTEST0)

NOTE

This reserved register is designed for factory test purposes only, and is not intended for general user access. Writing to this register when in special modes can alter the CGMIPLL's functionality.

Module Base + 0x0005

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 12-7. Reserved Register (CGMTEST0)

Read: Always read $00 except in special modes

Write: Only in special modes

12.3.2.6 Reserved Register (CGMTEST1)

NOTE

This reserved register is designed for factory test purposes only, and is not intended for general user access. Writing to this register when in special test modes can alter the CGMIPLL's functionality.

Module Base + 0x0006

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 12-8. Reserved Register (CGMTEST1)

Read: Always read $00 except in special modes

Write: Only in special modes

12.3.2.7 Reserved Register (CGMTEST2)

NOTE

This reserved register is designed for factory test purposes only, and is not intended for general user access. Writing to this register when in special modes can alter the CGMIPLL's functionality.

Module Base + 0x0007

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 12-9. Reserved Register (CGMTEST2)

Read: always read $00 except in special modes

Write: only in special modes

12.4 Functional Description

12.4.1 Examples of IPLL divider settings

Several examples of IPLL divider settings are shown in Table 12-6. Shaded rows indicated that these settings are not recommended. The following rules help to achieve optimum stability and shortest lock time:

- Use lowest possible f_{VCO} / f_{REF} ratio (SYNDIV value).
- Use highest possible REFCLK frequency f_{REF}.

Table 12-6. Examples of IPLL Divider Settings

f_{OSC}	REFDIV[5:0]	f_{REF}	REFFRQ[1:0]	SYNDIV[5:0]	f_{VCO}	VCOFRQ[1:0]	DIV2	$f_{CGMIPLL}$
4MHz	$00	4MHz	01	$09	80MHz	01	0	80MHz
8MHz	$00	8MHz	10	$04	80MHz	01	0	80MHz
4MHz	$00	4MHz	01	$03	32MHz	00	1	16MHz
4MHz	$01	2MHz	00	$18	100MHz	11	0	100MHz
4MHz	$03	1MHz	00	$18	50MHz	01	0	50MHz
4MHz	$03	1MHz	00	$32	100MHz	11	0	100MHz

12.4.2 IPLL Operation

The oscillator output clock signal (OSCCLK) is fed through the reference programmable divider and is divided in a range of 1 to 64 (REFDIV+1) to output the REFCLK. The VCO output clock, (VCOCLK) is fed back through the programmable loop divider and is divided in a range of 2 to 128 in increments of [2 x (SYNDIV +1)] to output the FBCLK. The VCOCLK can by divided by 2 (DIV2 bit) to output the CGMIPLL Clock.

The phase detector then compares the FBCLK, with the REFCLK. Correction pulses are generated based on the phase difference between the two signals. The loop filter then slightly alters the DC voltage on the internal filter capacitor, based on the width and direction of the correction pulse.

The user must select the range of the REFCLK frequency and the range of the VCOCLK frequency to ensure that the correct IPLL loop bandwidth is set.

The lock detector compares the frequencies of the FBCLK, and the REFCLK. Therefore, the speed of the lock detector is directly proportional to the reference clock frequency. The circuit determines the lock condition based on this comparison.

If IPLL LOCK interrupt requests are enabled, the software can wait for an interrupt request and then check the LOCK bit. If interrupt requests are disabled, software can poll the LOCK bit continuously (during IPLL start-up, usually) or at periodic intervals.

- The LOCK bit is a read-only indicator of the locked state of the IPLL.

- The LOCK bit is set when the VCO frequency is within a certain tolerance, Δ_{Lock}, and is cleared when the VCO frequency is out of a certain tolerance, Δ_{unl}.
- Interrupt requests can occur if enabled (LOCKIE = 1) when the lock condition changes, toggling the LOCK bit.

12.5 Interrupts

The interrupts/reset vectors requested by the CGMIPLL are listed in Table 12-7. Refer to MCU specification for related vector addresses and priorities.

Table 12-7. CGMIPLL Interrupt Vectors

Interrupt Source	CCR Mask	Local Enable
LOCK interrupt	I bit	CGMFLG (LOCKIE)

12.5.1 Description of Interrupt Operation

12.5.1.1 IPLL Lock Interrupt

The CGMIPLL generates a IPLL Lock interrupt when the LOCK condition of the IPLL has changed, either from a locked state to an unlocked state or vice versa. Lock interrupts are locally disabled by setting the LOCKIE bit to zero. The IPLL Lock interrupt flag (LOCKIF) is set to1 when the LOCK condition has changed, and is cleared to 0 by writing a 1 to the LOCKIF bit.

Chapter 13 FlexRay Communication Controller (FLEXRAY)

Table 13-1. Revision History

Rev. No.	Revision Date	Effective Date	Author	Summary of Changes
1.5	7 Nov 2006	7 Nov 2006		Module Configuration Register (MCR) - updated description of CLKSEL bit Section 13.6.7, “Individual Message Buffer Search” - split message buffer priority table into static / dynamic segment - add statement of rx tx buffer pair in dynamic segment Section 13.6.3.7.2, “Receive FIFO Control Data” - removed Note on empty receive fifo update issue - added statement, that empty fifo can not be updated
1.6	02 Feb 07	02 Feb 07		Section 13.6.7, “Individual Message Buffer Search” - major update Section 13.7.6, “Message Buffer Search on Simple Message Buffer Configuration” - added to illustrate message buffer search Section 13.7.2, “Shut Down Sequence” - simplified description Section 13.1.6.3, “Stop Mode” - make shutdown mandatory added w1c indication to all flag bits, added rwm to CMT bit added flexray bus related minimum chi frequency Section 13.1.6, “Modes of Operation” - updated desciption Section 13.3, “Controller Host Interface Clocking” - added and provide minimum chi frequency Section 13.7.1, “Initialization Sequence -updated, changed shutdown sequence
1.7	10.Nov.10	10.Nov.10		Limited crystal oscillator clocking option to test only

13.1 Introduction

13.1.1 Reference

The following documents are referenced.

- *FlexRay Communications System Protocol Specification, Version 2.1 Rev A*

- *FlexRay Communications System Electrical Physical Layer Specification, Version 2.1 Rev A*

13.1.2 Glossary

This section provides a list of terms used in the description of the FlexRay block.

Table 13-2. List of Terms

Term	Definition
BCU	Buffer Control Unit. Handles message buffer access.
BMIF	Bus Master Interface. Provides master access to FlexRay memory block.
CC	Communication Controller
CDC	Clock Domain Crosser
CHI	Controller Host Interface
Cycle length in µT	The actual length of a cycle in µT for the ideal controller (+/- 0 ppm)
EBI	External Bus Interface
FRM	FlexRay Memory. Memory to store message buffer payload, header, and status, and to store synchronization frame related tables.
FSS	Frame Start Sequence
HIF	Host Interface. Provides host access to FlexRay block.
Host	The FlexRay CC host MCU
LUT	Look Up Table. Stores message buffer header index value.
MB	Message Buffer
MBIDX	Message Buffer Index: the position of a header field entry within the header area. If the header area is accessed as an array, this is the same as the array index of the entry.
MBNum	Message Buffer Number: Position of message buffer configuration registers within the register map. For example, Message Buffer Number 5 corresponds to the MBCCS5 register.
MCU	Microcontroller Unit
µT	Microtick
MT	Macrotick
MTS	Media Access Test Symbol
NIT	Network Idle Time
PE	Protocol Engine
POC	Protocol Operation Control. Each state of the POC is denoted by *POC:state*
Rx	Reception
SEQ	Sequencer Engine
TCU	Time Control Unit
Tx	Transmission

13.1.3 Color Coding

Throughout this chapter types of items are highlighted through the use of an italicized color font.

FlexRay protocol parameters, constants and variables are highlighted with *blue italics*. An example is the parameter *gdActionPointOffset*.

FlexRay protocol states are highlighted in *green italics*. An example is the state *POC:normal active*.

13.1.4 Overview

The FlexRay block is a FlexRay communication controller that implements the *FlexRay Communications System Protocol Specification, Version 2.1 Rev A*.

The FlexRay block has three main components:

- Controller host interface (CHI)
- Protocol engine (PE)
- Clock domain crossing unit (CDC)

A block diagram of the FlexRay block with its surrounding modules is given in Figure 13-1.

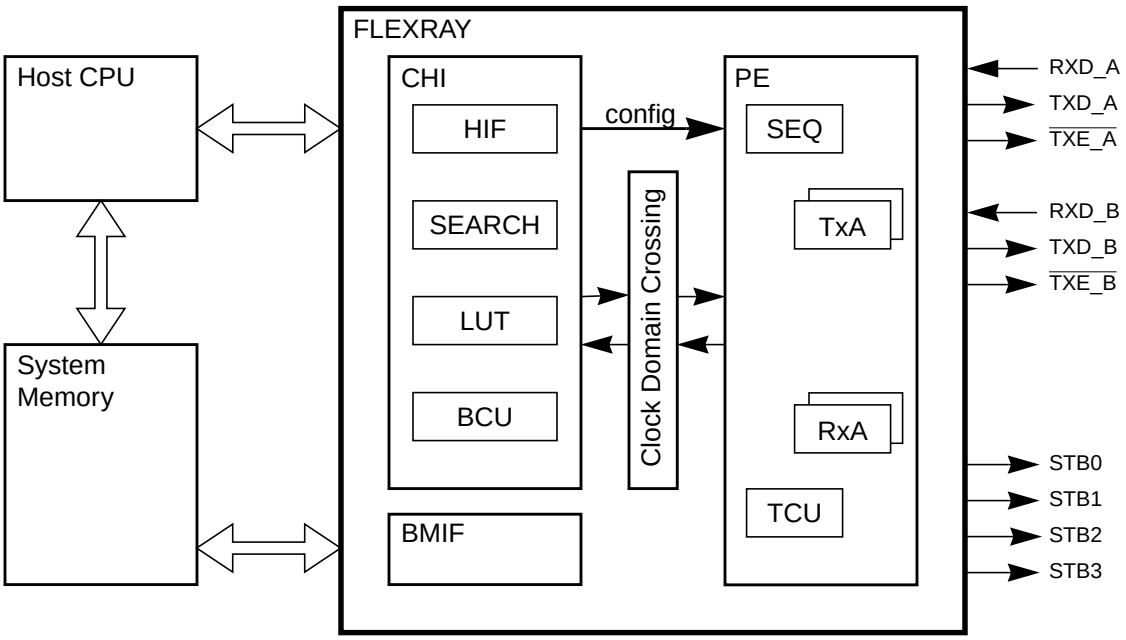


Figure 13-1. FLEXRAY Block Diagram

The protocol engine has two transmitter units TxA and TxB and two receiver units RxA and RxB for sending and receiving frames through the two FlexRay channels. The time control unit (TCU) is responsible for maintaining global clock synchronization to the FlexRay network. The overall activity of the PE is controlled by the sequencer engine (SEQ).

The controller host interface provides host access to the module's configuration, control, and status registers, as well as to the message buffer configuration, control, and status registers. The message buffers themselves, which contain the frame header and payload data received or to be transmitted, and the slot status information, are stored in the FlexRay Memory (FRM).

The clock domain crossing unit implements signal crossing from the CHI clock domain to the PE clock domain and vice versa, to allow for asynchronous PE and CHI clock domains.

The FlexRay block stores the frame header and payload data of frames received or of frames to be transmitted in the FRM. The application accesses the FRM to retrieve and provide the frames to be processed by the FlexRay block. In addition to the frame header and payload data, the FlexRay block stores the synchronization frame related tables in the FRM for application processing.

The FlexRay Memory is located in the system memory of the MCU. The FlexRay block has access to the FRM via its bus master interface (BMIF). The host provides the start address of the FRM window within the system memory by programming the System Memory Base Address High Register (SYMBADHR) and System Memory Base Address Low Register (SYMBADLR). All FRM related offsets are stored in offset registers. The physical address pointer into the FRM window of the MCU system memory is calculated using the offset values the FlexRay Memory base address.

NOTE

The FlexRay block does not provide a memory protection scheme for the FlexRay Memory.

13.1.5 Features

The FlexRay block provides the following features:

- *FlexRay Communications System Protocol Specification, Version 2.1 Rev A* compliant protocol implementation
- *FlexRay Communications System Electrical Physical Layer Specification, Version 2.1 Rev A* compliant bus driver interface
- single channel support
  - FlexRay Port A can be configured to be connected either to physical FlexRay channel A or physical FlexRay channel B.
- FlexRay bus data rates of 10 Mbit/s, 8 Mbit/s, 5 Mbit/s, and 2.5 Mbit/s supported
- internal oscillator or internal PLL clocking of the protocol engine
- 32 configurable message buffers with
  - individual frame ID filtering
  - individual channel ID filtering
  - individual cycle counter filtering
- message buffer header, status and payload data stored in dedicated FlexRay Memory
  - allows for flexible and efficient message buffer implementation
  - consistent data access ensured by means of buffer locking scheme
  - application can lock multiple buffers at the same time
- size of message buffer payload data section configurable from 0 up to 254 bytes
- two independent message buffer segments with configurable size of payload data section
  - each segment can contain message buffers assigned to the static segment and message buffers assigned to the dynamic segment at the same time
- zero padding for transmit message buffers in static segment
  - applied when the frame payload length exceeds the size of the message buffer data section

- transmit message buffers configurable with state/event semantics
- message buffers can be configured as
  - receive message buffer
  - single buffered transmit message buffer
  - double buffered transmit message buffer (combines two single buffered message buffer)
- individual message buffer reconfiguration supported
  - means provided to safely disable individual message buffers
  - disabled message buffers can be reconfigured
- two independent receive FIFOs
  - one receive FIFO per channel
  - up to 255 entries for each FIFO
  - global frame ID filtering, based on both value/mask filters and range filters
  - global channel ID filtering
  - global message ID filtering for the dynamic segment
- 4 configurable slot error counters
- 4 dedicated slot status indicators
  - used to observe slots without using receive message buffers
- measured value indicators for the clock synchronization
  - internal synchronization frame ID and synchronization frame measurement tables can be copied into the FlexRay Memory
- fractional macroticks are supported for clock correction
- maskable interrupt sources provided via individual and combined interrupt lines
- 1 absolute timer
- 1 timer that can be configured to absolute or relative

13.1.6 Modes of Operation

This section describes the basic operational power modes of the FlexRay block.

13.1.6.1 Disabled Mode

This is the mode the FlexRay block enters during hard reset. The FlexRay block indicates that it is in the Disabled Mode by negating the module enable bit MEN in the Module Configuration Register (MCR).

No communication is performed on the FlexRay bus.

All registers with the write access conditions *Any Time* and *Disabled Mode* can be accessed for writing as stated in Section 13.5.2, “Register Descriptions”.

The application configures the FlexRay block by accessing the configuration bits and fields in the Module Configuration Register (MCR).

13.1.6.1.1 Leave Disabled Mode

The FlexRay block leaves the Disabled Mode and enters the Normal Mode, when the application writes 1 to the module enable bit MEN in the Module Configuration Register (MCR)

NOTE

When the FlexRay block was enabled, it cannot be disabled the later on.

13.1.6.2 Normal Mode

In this mode the FlexRay block is fully functional. The FlexRay block indicates that it is in Normal Mode by asserting the module enable bit MEN in the Module Configuration Register (MCR).

13.1.6.2.1 Enter Normal Mode

This mode is entered when the application requests the FlexRay block to leave the Disabled Mode or when the MCU leaves the Stop Mode. If the Normal Mode was entered by leaving the Disabled Mode, the application has to perform the protocol initialization described in 13.7.1.2, “Protocol Initialization” to achieve full FlexRay functionality.

Depending on the values of the SCM, CHA, and CHB bits in the Module Configuration Register (MCR), the corresponding FlexRay bus driver ports are driven.

13.1.6.3 Stop Mode

The FlexRay block is in Stop Mode when the MCU is either in Full Stop or Pseudo Stop Mode. In this mode all FlexRay block clocks are stopped. No registers can be accessed. No communication is performed on the FlexRay Bus.

13.1.6.3.1 Enter Stop Mode

Before the application requests the MCU to enter one of the Stop Modes, it has to shut down the FlexRay block as described in Section 13.7.2, “Shut Down Sequence”.

NOTE

If the FlexRay block is stopped during transmission of data, it is not guaranteed, that the FlexRay ports return to its inactive state before the clocks are stopped. This can result in the lockup of the FlexRay Bus.

13.1.6.3.2 Leave Stop Mode

The FlexRay block leaves the Stop Mode when the MCU leaves its Stop Mode and all clocks are reapplied to the FlexRay block. The FlexRay block enters the operational mode it was in before going into Stop Mode.

If the FlexRay block enters the Normal Mode, the application has to put the protocol engine into the default config state by the following sequence:

d) issue the DEFAULT_CONFIG command via Protocol Operation Control Register (POCR)

e) wait for *POC:default config* in Protocol Status Register 0 (PSR0)

Subsequently the application can reconfigure and/or reintegrate the FlexRay node into the FlexRay cluster.

13.2 External Signal Description

This section lists and describes the FlexRay block signals, connected to external pins. These signals are summarized in Table 13-2 and described in detail in Section 13.2.1, “Detailed Signal Descriptions”.

NOTE

The off chip signals RXD_A, TXD_A, and $\overline{\text{TXE\_A}}$ are available on each package option. The availability of the other off chip signals depends on the package option.

Table 13-3. External Signal Properties

Name	Direction	Active	Reset	Function
RXD_A	Input	—	—	Receive Data Channel A
TXD_A	Output	—	1	Transmit Data Channel A
$\overline{\text{TXE\_A}}$	Output	Low	1	Transmit Enable Channel A
RXD_B	Input	—	—	Receive Data Channel B
TXD_B	Output	—	1	Transmit Data Channel B
$\overline{\text{TXE\_B}}$	Output	Low	1	Transmit Enable Channel B
STB0	Output	—	0	Debug Strobe Signal 0
STB1	Output	—	0	Debug Strobe Signal 1
STB2	Output	—	0	Debug Strobe Signal 2
STB3	Output	—	0	Debug Strobe Signal 3

13.2.1 Detailed Signal Descriptions

This section provides a detailed description of the FlexRay block signals, connected to external pins.

13.2.1.1 RXD_A — Receive Data Channel A

The RXD_A signal carries the receive data for channel A from the corresponding FlexRay bus driver.

13.2.1.2 TXD_A — Transmit Data Channel A

The TXD_A signal carries the transmit data for channel A to the corresponding FlexRay bus driver.

13.2.1.3 $\overline{\text{TXE\_A}}$ — Transmit Enable Channel A

The $\overline{\text{TXE\_A}}$ signal indicates to the FlexRay bus driver that the FlexRay block is attempting to transmit data on channel A.

13.2.1.4 RXD_B — Receive Data Channel B

The RXD_B signal carries the receive data for channel B from the corresponding FlexRay bus driver.

13.2.1.5 TXD_B — Transmit Data Channel B

The TXD_B signal carries the transmit data for channel B to the corresponding FlexRay bus driver

13.2.1.6 $\overline{\text{TXE\_B}}$ — Transmit Enable Channel B

The $\overline{\text{TXE\_B}}$ signal indicates to the FlexRay bus driver that the FlexRay block is attempting to transmit data on channel B.

13.2.1.7 STB3, STB2, STB1, STB0 — Strobe Signals

These signals provide the selected debug strobe signals. For details on the debug strobe signal selection refer to Section 13.6.16, “Strobe Signal Support”.

13.3 Controller Host Interface Clocking

The clock for the CHI is derived from the system bus clock and has the same phase and frequency. Since the FlexRay protocol requires data delivery at fixed points in time, the memory read cycles from the FRM must be finished after a fixed amount of time. To ensure this, a minimum frequency f_{chi} of the CHI clock is required, which is given in Equation 13-1.

$$f_{chi} \geq 16\text{MHz} \qquad \textit{Eqn. 13-1}$$

Additional requirements for the minimum frequency of the CHI clock result from the number of message buffer. The requirement is provides in Section 13.7.3, “Number of Usable Message Buffers”

13.4 Protocol Engine Clocking

The clock for the protocol engine can be generated by two sources. The first source is the internal crystal oscillator and the second source is an internal PLL. The clock source to be used is selected by the clock source select bit CLKSEL in the Module Configuration Register (MCR).

13.4.1 Oscillator Clocking

Applications must use the FlexRay IPLL as clock source for the FlexRay protocol engine. The option to use the crystal as clock source is only intended for test purposes.

13.4.2 PLL Clocking

If the protocol engine is clocked by the dedicated internal PLL, which is described in Chapter 12, “Clock Generation Module using IPLL (CGMIPLL) Block Description, the CGMIPLL must be programmed to generate an output clock with $f_{CGMIPLL}$ = 80 MHz.

13.4.3 PLL Lock Handling

For correct flexray bus functionality of the protocol engine, it is required that the PLL in the CGMIPLL module is locked while the protocol engine is driving data onto the flexray bus. The lock state of the PLL is indicated by the lock status bit LOCK in the CGMIPLL Flags Register (CGMFLG).

The FlexRay block drives its FlexRay transmit enable ports $\overline{\text{TXE\_A}}$ and $\overline{\text{TXE\_B}}$ to active state only if the PLL unlock flag UNLOCKF in the CGMIPLL Flags Register (CGMFLG) is 0. If the unlock flag UNLOCKF is 1, the FlexRay block drives its FlexRay transmit enable ports $\overline{\text{TXE\_A}}$ and $\overline{\text{TXE\_B}}$ to its inactive state 1.

13.4.3.1 PLL Loss of Lock

If the PLL goes from the locked state to the unlock state, the CGMIPLL module sets the lock bit LOCK to 0 and the unlock flag UNLOCKF to 1. As a result, the FlexRay block drives its FlexRay transmit enable ports $\overline{\text{TXE\_A}}$ and $\overline{\text{TXE\_B}}$ to its inactive states 1 and thus stops the transmission onto the FlexRay bus immediately. As a result of the loss of lock, the correct operation of the PE is no longer guaranteed. The application should perform a shutdown of the FlexRay module as described in 13.7.2, “Shut Down Sequence”.

13.4.3.2 PLL Gain of Lock

If the PLL goes from the unlocked state to the locked state, the CGMIPLL module sets the lock bit LOCK to 1. The unlock flag UNLOCKF is not cleared by the module, this has to be done by the application. After the clearing of the unlock flag UNLOCKF, the application can perform the FlexRay initialization as described in 13.7.1, “Initialization Sequence”.

13.5 Memory Map and Register Description

The FlexRay block occupies 512 bytes of address space starting atthe FlexRay block’s base address defined by the memory map of the MCU.

13.5.1 Memory Map

The complete memory map of the FlexRay block is shown in Table 13-3. The addresses presented here are the offsets relative to the FlexRay block base address which is defined by the MCU address map.

Table 13-4. FlexRay Memory Map (Sheet 1 of 4)

Offset	Register	Access
Module Configuration and Control		
0x0000	Module Version Register (MVR)	R
0x0002	Module Configuration Register (MCR)	R/W
0x0004	System Memory Base Address High Register (SYMBADHR)	R/W
0x0006	System Memory Base Address Low Register (SYMBADLR)	R/W
0x0008	Strobe Signal Control Register (STBSCR)	R/W
0x000A	Strobe Port Control Register (STBPCR)	R/W

Table 13-4. FlexRay Memory Map (Sheet 2 of 4)

Offset	Register	Access
0x000C	Message Buffer Data Size Register (MBDSR)	R/W
0x000E	Message Buffer Segment Size and Utilization Register (MBSSUTR)	R/W
Test Registers		
0x0010	Reserved	R
0x0012	Reserved	R
Interrupt and Error Handling		
0x0014	Protocol Operation Control Register (POCR)	R/W
0x0016	Global Interrupt Flag and Enable Register (GIFER)	R/W
0x0018	Protocol Interrupt Flag Register 0 (PIFR0)	R/W
0x001A	Protocol Interrupt Flag Register 1 (PIFR1)	R/W
0x001C	Protocol Interrupt Enable Register 0 (PIER0)	R/W
0x001E	Protocol Interrupt Enable Register 1 (PIER1)	R/W
0x0020	CHI Error Flag Register (CHIERFR)	R/W
0x0022	Message Buffer Interrupt Vector Register (MBIVEC)	R
0x0024	Channel A Status Error Counter Register (CASERCR)	R
0x0026	Channel B Status Error Counter Register (CBSERCR)	R
Protocol Status		
0x0028	Protocol Status Register 0 (PSR0)	R
0x002A	Protocol Status Register 1 (PSR1)	R
0x002C	Protocol Status Register 2 (PSR2)	R
0x002E	Protocol Status Register 3 (PSR3)	R/W
0x0030	Macrotick Counter Register (MTCTR)	R
0x0032	Cycle Counter Register (CYCTR)	R
0x0034	Slot Counter Channel A Register (SLTCTAR)	R
0x0036	Slot Counter Channel B Register (SLTCTBR)	R
0x0038	Rate Correction Value Register (RTCORVR)	R
0x003A	Offset Correction Value Register (OFCORVR)	R
0x003C	Combined Interrupt Flag Register (CIFRR)	R
0x003E	System Memory Access Time-Out Register (SYMATOR)	R/W
Sync Frame Counter and Tables		
0x0040	Sync Frame Counter Register (SFCNTR)	R
0x0042	Sync Frame Table Offset Register (SFTOR)	R/W
0x0044	Sync Frame Table Configuration, Control, Status Register (SFTCCSR)	R/W
Sync Frame Filter		
0x0046	Sync Frame ID Rejection Filter Register (SFIDRFR)	R/W
0x0048	Sync Frame ID Acceptance Filter Value Register (SFIDAFVR)	R/W
0x004A	Sync Frame ID Acceptance Filter Mask Register (SFIDAFMR)	R/W
Network Management Vector		
0x004C	Network Management Vector Register 0 (NMVR0)	R
0x004E	Network Management Vector Register 1 (NMVR1)	R
0x0050	Network Management Vector Register 2 (NMVR2)	R

Table 13-4. FlexRay Memory Map (Sheet 3 of 4)

Offset	Register	Access
0x0052	Network Management Vector Register 3 (NMVR3)	R
0x0054	Network Management Vector Register 4 (NMVR4)	R
0x0056	Network Management Vector Register 5 (NMVR5)	R
0x0058	Network Management Vector Length Register (NMVLR)	R/W
Timer Configuration		
0x005A	Timer Configuration and Control Register (TICCR)	R/W
0x005C	Timer 1 Cycle Set Register (TI1CYSR)	R/W
0x005E	Timer 1 Macrotick Offset Register (TI1MTOR)	R/W
0x0060	Timer 2 Configuration Register 0 (TI2CR0)	R/W
0x0062	Timer 2 Configuration Register 1 (TI2CR1)	R/W
Slot Status Configuration		
0x0064	Slot Status Selection Register (SSSR)	R/W
0x0066	Slot Status Counter Condition Register (SSCCR)	R/W
Slot Status		
0x0068	Slot Status Register 0 (SSR0)	R
0x006A	Slot Status Register 1 (SSR1)	R
0x006C	Slot Status Register 2 (SSR2)	R
0x006E	Slot Status Register 3 (SSR3)	R
0x0070	Slot Status Register 4 (SSR4)	R
0x0072	Slot Status Register 5 (SSR5)	R
0x0074	Slot Status Register 6 (SSR6)	R
0x0076	Slot Status Register 7 (SSR7)	R
0x0078	Slot Status Counter Register 0 (SSCR0)	R
0x007A	Slot Status Counter Register 1 (SSCR1)	R
0x007C	Slot Status Counter Register 2 (SSCR2)	R
0x007E	Slot Status Counter Register 3 (SSCR3)	R
MTS Generation		
0x0080	MTS A Configuration Register (MTSACFR)	R/W
0x0082	MTS B Configuration Register (MTSBCFR)	R/W
Shadow Buffer Configuration		
0x0084	Receive Shadow Buffer Index Register (RSBIR)	R/W
Receive FIFO — Configuration		
0x0086	Receive FIFO Selection Register (RFSR)	R/W
0x0088	Receive FIFO Start Index Register (RFSIR)	R/W
0x008A	Receive FIFO Depth and Size Register (RFDSR)	R/W
Receive FIFO - Status		
0x008C	Receive FIFO A Read Index Register (RFARIR)	R
0x008E	Receive FIFO B Read Index Register (RFBRIR)	R
Receive FIFO - Filter		
0x0090	Receive FIFO Message ID Acceptance Filter Value Register (RFMIDAFVR)	R/W

Table 13-4. FlexRay Memory Map (Sheet 4 of 4)

Offset	Register	Access
0x0092	Receive FIFO Message ID Acceptance Filter Mask Register (RFMIAFMR)	R/W
0x0094	Receive FIFO Frame ID Rejection Filter Value Register (RFFIDRFVR)	R/W
0x0096	Receive FIFO Frame ID Rejection Filter Mask Register (RFFIDRFMR)	R/W
0x0098	Receive FIFO Range Filter Configuration Register (RFRFCFR)	R/W
0x009A	Receive FIFO Range Filter Control Register (RFRFCTR)	R/W
Dynamic Segment Status		
0x009C	Last Dynamic Transmit Slot Channel A Register (LDTXSLAR)	R
0x009E	Last Dynamic Transmit Slot Channel B Register (LDTXSLBR)	R
Protocol Configuration		
0x00A0 ... 0x00DC	Protocol Configuration Register 0 (PCR0) ... Protocol Configuration Register 30 (PCR30)	R/W – R/W
0x00DE ... 0x00FE	Reserved	R
Message Buffers Configuration, Control, Status		
0x0100	Message Buffer Configuration, Control, Status Register 0 (MBCCSR0)	R/W
0x0102	Message Buffer Cycle Counter Filter Register 0 (MBCCFR0)	R/W
0x0104	Message Buffer Frame ID Register 0 (MBFIDR0)	R/W
0x0106	Message Buffer Index Register 0 (MBIDXR0)	R/W
...	...	...
0x01F8	Message Buffer Configuration, Control, Status Register 31 (MBCCSR31)	R/W
0x01FA	Message Buffer Cycle Counter Filter Register 31 (MBCCFR31)	R/W
0x01FC	Message Buffer Frame ID Register 31 (MBFIDR31)	R/W
0x01FE	Message Buffer Index Register 31 (MBIDXR31)	R/W

13.5.2 Register Descriptions

This section provides detailed descriptions of all registers in ascending address order, presented as 16-bit wide entities

Table 13-4 provides a key for the register figures and register tables.

Table 13-5. Register Access Conventions

Convention	Description
	Depending on its placement in the read or write row, indicates that the bit is not readable or not writeable.
R*	Reserved bit or field, will not be changed. Application must not write any value different from the reset value.
FIELDNAME	Identifies the field. Its presence in the read or write row indicates that it can be read or written.
Register Field Types	
rwm	A read/write bit that may be modified by a hardware in some fashion other than by a reset.
w1c	Write one to clear. A flag bit that can be read, is cleared by writing a one, writing 0 has no effect.

Table 13-5. Register Access Conventions (continued)

Convention	Description
Reset Value	
0	Resets to zero.
1	Resets to one.
–	Not defined after reset and not affected by reset.

13.5.2.1 Register Reset

All registers except the Message Buffer Cycle Counter Filter Registers (MBCCFRn), Message Buffer Frame ID Registers (MBFIDRn), and Message Buffer Index Registers (MBIDXRn) are reset to their reset value on system reset. The registers mentioned above are located in physical memory blocks and, thus, they are not affected by reset. For some register fields, additional reset conditions exist. These additional reset conditions are mentioned in the detailed description of the register. The additional reset conditions are explained in Table 13-5.

Table 13-6. Additional Register Reset Conditions

Condition	Description
Protocol RUN Command	The register field is reset when the application writes to RUN command “0101” to the POCCMD field in the Protocol Operation Control Register (POCR).
Message Buffer Disable	The register field is reset when the application has disabled the message buffer. This happens when the application writes 1 to the message buffer disable trigger bit MBCCSRn.EDT while the message buffer is enabled (MBCCSn.EDS = 1) and the FlexRay block grants the disable to the application by clearing the MBCCSRn.EDS bit.

13.5.2.2 Register Write Access

This section describes the write access restriction terms that apply to all registers.

13.5.2.2.1 Register Write Access Restriction

For each register bit and register field, the write access conditions are specified in the detailed register description. A description of the write access conditions is given in Table 13-6. If, for a specific register bit or field, none of the given write access conditions is fulfilled, any write attempt to this register bit or field is ignored without any notification. The values of the bits or fields are not changed. The condition term [A or B] indicates that the register or field can be written to if at least one of the conditions is fulfilled.

Table 13-7. Register Write Access Restrictions

Condition	Indication	Description
Any Time	-	No write access restriction.
Disabled Mode	MCR.MEN = 0	Write access only when the FlexRay block is in Disabled Mode.
Normal Mode	MCR.MEN = 1	Write access only when the FlexRay block is in Normal Mode.
POC:config	PSR0.PROTSTATE = *POC:config*	Write access only when the Protocol is in the *POC:config* state.
MB_DIS	MBCCSRn.EDS = 0	Write access only when the related Message Buffer is disabled.

Table 13-7. Register Write Access Restrictions

Condition	Indication	Description
MB_LCK	MBCCSRn.LCKS = 1	Write access only when the related Message Buffer is locked.

13.5.2.2.2 Register Write Access Requirements

For some of the registers, a 16-bit wide write access is required to ensure correct operation. This write access requirement is stated in the detailed register description for each register affected

13.5.2.2.3 Internal Register Access

The following memory mapped registers are used to access multiple internal registers.

- Strobe Signal Control Register (STBSCR)
- Slot Status Selection Register (SSSR)
- Slot Status Counter Condition Register (SSCCR)
- Receive Shadow Buffer Index Register (RSBIR)

Each of these memory mapped registers provides a SEL field and a WMD bit. The SEL field is used to select the internal register. The WMD bit controls the write mode. If the WMD bit is set to 0 during the write access, all fields of the internal register are updated. If the WMD bit set to 1, only the SEL field is changed. All other fields of the internal register remain unchanged. This allows for reading back the values of the selected internal register in a subsequent read access.

13.5.2.3 Module Version Register (MVR)

Module Base + 0x0000

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CHIVER								PEVER							
W																
Reset	1	0	0	0	1	0	0	0	0	1	1	0	0	1	1	0

Figure 13-2. Module Version Register (MVR)

This register provides the FlexRay block version number. The module version number is derived from the CHI version number and the PE version number.

Table 13-8. MVR Field Descriptions

Field	Description
15–8 CHIVER	**CHI Version Number** — This field provides the version number of the controller host interface.
7–0 PEVER	**PE Version Number** — This field provides the version number of the protocol engine.

13.5.2.4 Module Configuration Register (MCR)

Module Base + 0x0002

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MEN	0	SCM	CHB	CHA	SFFE	0	R*	0	0	0	CLKSEL	BITRATE			0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-3. Module Configuration Register (MCR)

Write: MEN, SCM, CHB, CHA, CLKSEL, BITRATE: Disabled Mode
SFFE: Disabled Mode or *POC:config*

This register defines the global configuration of the FlexRay block.

Table 13-9. MCR Field Descriptions

Field	Description
15 MEN	**Module Enable** — This bit indicates whether or not the FlexRay block is in the Disabled Mode. The application requests the FlexRay block to leave the Disabled Mode by writing 1 to this bit. Before leaving the Disabled Mode, the application must configure the SCM, CHB, CHA, TMODE, CLKSEL, BITRATE values. For details see Section 13.1.6, “Modes of Operation”. 0 Write: ignored, FlexRay block disable not possible Read: FlexRay block disabled 1 Write: enable FlexRay block Read: FlexRay block enabled **Note:** If the FlexRay block is enabled it can not be disabled.
13 SCM	**Single Channel Device Mode** — This control bit defines the channel device mode of the FlexRay block as described in Section 13.6.10, “Channel Device Modes”. 0 FlexRay block works in dual channel device mode 1 FlexRay block works in single channel device mode
12–11 CHB CHA	**Channel Enable** — protocol related parameter: *pChannels* The semantic of these control bits depends on the channel device mode controlled by the SCM bit and is given Table 13-9.
10 SFFE	**Synchronization Frame Filter Enable** — This bit controls the filtering for received synchronization frames. For details see Section 13.6.15, “Sync Frame Filtering”. 0 Synchronization frame filtering disabled 1 Synchronization frame filtering enabled
8 R*	**Reserved** — This bit is reserved. It is read as 0. Application must not write 1 to this bit.
4 CLKSEL	**Protocol Engine Clock Source Select** — This bit is used to select the clock source for the protocol engine. 0 PE clock source is generated by on-chip crystal oscillator. 1 PE clock source is generated by on-chip PLL.
3–1 BITRATE	**FlexRay Bus Bit Rate** — This bit field defines the bit rate of the flexray channels according to Table 13-10.

Table 13-10. FlexRay Channel Selection (Sheet 1 of 2)

SCM	CHB	CHA	Description
Dual Channel Device Modes			

Table 13-10. FlexRay Channel Selection (Sheet 2 of 2)

SCM	CHB	CHA	Description
0	0	0	ports RXD_A, TXD_A, and $\overline{\text{TXE\_A}}$ not driven by FlexRay block ports RXD_B, TXD_B, and $\overline{\text{TXE\_A}}$ not driven by FlexRay block PE channel 0 idle PE channel 1 idle
	0	1	ports RXD_A, TXD_A, and $\overline{\text{TXE\_A}}$ driven by FlexRay block ports RXD_B, TXD_B, and $\overline{\text{TXE\_A}}$ not driven by FlexRay block PE channel 0 active PE channel 1 idle
	1	0	ports RXD_A, TXD_A, and $\overline{\text{TXE\_A}}$ not driven by FlexRay block ports RXD_B, TXD_B, and $\overline{\text{TXE\_A}}$ driven by FlexRay block PE channel 0 idle PE channel 1 active
	1	1	ports RXD_A, TXD_A, and $\overline{\text{TXE\_A}}$ driven by FlexRay block ports RXD_B, TXD_B, and $\overline{\text{TXE\_A}}$ driven by FlexRay block PE channel 0 active PE channel 1 active
Single Channel Device Mode			
1	0	0	ports RXD_A, TXD_A, and $\overline{\text{TXE\_A}}$ not driven by FlexRay block ports RXD_B, TXD_B, and $\overline{\text{TXE\_A}}$ not driven by FlexRay block PE channel 0 idle PE channel 1 idle
	0	1	ports RXD_A, TXD_A, and $\overline{\text{TXE\_A}}$ driven by FlexRay block ports RXD_B, TXD_B, and $\overline{\text{TXE\_A}}$ not driven by FlexRay block PE channel 0 active PE channel 1 idle
	1	0	ports RXD_A, TXD_A, and $\overline{\text{TXE\_A}}$ driven by FlexRay block ports RXD_B, TXD_B, and $\overline{\text{TXE\_A}}$ not driven by FlexRay block PE channel 0 active, uses cCrcInit[B] (see Figure 13-135) PE channel 1 idle
	1	1	reserved

Table 13-11. FlexRay Channel Bit Rate Selection

MCR[BITRATE]	FlexRay Channel Bit Rate [Mbit/s]
000	10.0
001	5.0
010	2.5
011	8.0
100	reserved
101	reserved
110	reserved
111	reserved

13.5.2.5 System Memory Base Address High Register (SYMBADHR) and System Memory Base Address Low Register (SYMBADLR)

Module Base + 0x0004

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	SYS_MEM_BASE_ADDR[22:16]						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-4. System Memory Base Address High Register (SYMBADHR)

Write: Disabled Mode

Module Base + 0x0006

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SYS_MEM_BASE_ADDR[15:4]												0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-5. System Memory Base Address Low Register (SYMBADLR)

Write: Disabled Mode

NOTE

The system memory base address must be set before the FlexRay block is enabled.

The system memory base address registers define the base address of the FRM within the system memory. The base address is used by the BMIF to calculate the physical memory address for system memory accesses.

Table 13-12. SYMBADHR and SYMBADLR Field Descriptions

Field	Description
22–4 SYMBADHR SYMBADLR	This base address will be added to all system memory offset values stored in registers or calculated in the FlexRay block before the FlexRay block accesses the system memory via its bus master interface. The system memory base address must be aligned to an 16-byte boundary.

13.5.2.6 Strobe Signal Control Register (STBSCR)

Module Base + 0x0008 16-bit write access required

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	SEL							0	0	0	ENB	0	0	STBPSEL	
W	WMD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-6. Strobe Signal Control Register (STBSCR)

Write: Anytime

This register is used to assign the individual protocol timing related strobe signals given in Table 13-13 to the external strobe ports. Each strobe signal can be assigned to at most one strobe port. Each write access to registers overwrites the previously written ENB and STBPSEL values for the signal indicated by SEL. If more than one strobe signal is assigned to one strobe port, the current values of the strobe signals are combined with a binary OR and presented at the strobe port. If no strobe signal is assigned to a strobe port, the strobe port carries logic 0. For more detailed and timing information refer to Section 13.6.16, “Strobe Signal Support”.

NOTE

In single channel device mode, channel B related strobe signals are undefined and should not be assigned to the strobe ports.

Table 13-13. STBSCR Field Descriptions

Field	Description
15 WMD	**Write Mode** — This control bit defines the write mode of this register. 0 Write to all fields in this register on write access. 1 Write to SEL field only on write access.
14–8 SEL	**Strobe Signal Select** — This control field selects one of the strobe signals given in Table 13-13 to be enabled or disabled and assigned to one of the four strobe ports given in Table 13-13.
4 ENB	**Strobe Signal Enable** — The control bit is used to enable and to disable the strobe signal selected by STBSSEL. 0 Strobe signal is disabled and not assigned to any strobe port. 1 Strobe signal is enabled and assigned to the strobe port selected by STBPSEL.
1–0 STBPSEL	**Strobe Port Select** — This field selects the strobe port that the strobe signal selected by the SEL is assigned to. All strobe signals that are enabled and assigned to the same strobe port are combined with a binary OR operation. 00 assign selected signal to STB0 01 assign selected signal to STB1 10 assign selected signal to STB2 11 assign selected signal to STB3

Table 13-14. Strobe Signal Mapping (Sheet 1 of 3)

SEL		Description	Channel	Type	Offset[1]	Reference
dec	hex					
0	0x00	poc_startup_state[0] (for coding see PSR0[4])	-	value	0	MT start
1	0x01	poc_startup_state[1] (for coding see PSR0[5])				
2	0x02	poc_startup_state[2] (for coding see PSR0[6])				
3	0x03	poc_startup_state[3] (for coding see PSR0[7])				
4	0x04	poc_state[0] (for coding see PSR0[8])				
5	0x05	poc_state[1] (for coding see PSR0[9])				
6	0x06	poc_state[2] (for coding see PSR0[10])				
7	0x07	channel idle indicator	A	level	+5	RXD_A
8	0x08		B			RXD_B
9	0x09	receive data after glitch filtering	A	value	+4	RXD_A
10	0x0A		B			RXD_B

Table 13-14. Strobe Signal Mapping (Sheet 2 of 3)

SEL		Description	Channel	Type	Offset[1]	Reference
dec	hex					
11	0x0B	synchronization edge strobe	A	pulse	+4	RXD_A
12	0x0C		B			RXD_B
13	0x0D	header received	A	pulse	+4	RXD_A
14	0x0E		B			RXD_B
15	0x0F	wakeup symbol decoded	A	pulse	+5	RXD_A
16	0x10		B			RXD_B
17	0x11	MTS or CAS symbol decoded	A	pulse	+4	RXD_A
18	0x12		B			RXD_B
19	0x13	frame decoded	A	pulse	+4	RXD_A
20	0x14		B			RXD_B
21	0x15	channel idle detected	A	pulse	+4	RXD_A
22	0x16		B			RXD_B
23	0x17	start of communication element detected	A	pulse	+4	RXD_A
24	0x18		B			RXD_B
25	0x19	potential frame start channel	A	pulse	+4	RXD_A
26	0x1A		B			RXD_B
27	0x1B	wakeup collision detected	A	pulse	+5	RXD_A
28	0x1C		B			RXD_B
29	0x1D	content error detected	A	level	+4	RXD_A
30	0x1E		B			RXD_B
31	0x1F	syntax error detected	A	pulse	+4	RXD_A
32	0x20		B			RXD_B
33	0x21	start transmission of wakeup pattern	A	pulse	-1	TXD_A
34	0x22		B			TXD_B
35	0x23	start transmission of MTS or CAS symbol	A	pulse	-1	TXD_A
36	0x24		B			TXD_B
37	0x25	start of transmission	A	pulse	-1	TXD_A
38	0x26		B			TXD_B
39	0x27	end of transmission	A	pulse	-1	TXD_A
40	0x28		B			TXD_B
41	0x29	static segment indicator	-	level	0	MT start
42	0x2A	dynamic segment indicator	-	level	0	MT start
43	0x2B	symbol window indicator	-	level	0	MT start
44	0x2C	NIT indicator	-	level	0	MT start
45	0x2D	action point	-	pulse	-1	TXD_A
46	0x2E	sync calculation complete[2]	-	pulse	-	-
47	0x2F	start of offset correction	-	pulse	-2	MT start

Table 13-14. Strobe Signal Mapping (Sheet 3 of 3)

SEL		Description	Channel	Type	Offset[1]	Reference
dec	hex					
48	0x30	cycle count[0]	-	value	-2	MT start
49	0x31	cycle count[1]				
50	0x32	cycle count[2]				
51	0x33	cycle count[3]				
52	0x34	cycle count[4]				
53	0x35	cycle count[5]				
54	0x36	slot count[0]	A	value	0	MT start
55	0x37	slot count[1]				
56	0x38	slot count[2]				
57	0x39	slot count[3]				
58	0x3A	slot count[4]				
59	0x3B	slot count[5]				
60	0x3C	slot count[6]				
61	0x3D	slot count[7]				
62	0x3E	slot count[8]				
63	0x3F	slot count[9]				
64	0x40	slot count[10]				
65	0x41	slot count[0]	B	value	0	MT start
66	0x42	slot count[1]				
67	0x43	slot count[2]				
68	0x44	slot count[3]				
69	0x45	slot count[4]				
70	0x46	slot count[5]				
71	0x47	slot count[6]				
72	0x48	slot count[7]				
73	0x49	slot count[8]				
74	0x4A	slot count[9]				
75	0x4B	slot count[10]				
76	0x4C	cycle start	-	pulse	0	MT start
77	0x4D	slot start	A	pulse	0	MT start
78	0x4E		B			
79	0x4F	minislot start	-	pulse	0	MT start
80	0x50	arm	-	value	+1	MT start
81	0x51	mt	-	value	+1	MT start

1. Given in PE clock cycles
2. Indicates internal PE event not directly related to FlexRay bus timing

13.5.2.7 Strobe Port Control Register (STBPCR)

Module Base + 0x000A

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	STB3EN	STB2EN	STB1EN	STB0EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-7. Strobe Port Control Register (STBPCR)

Write: Anytime

This register is used to enable and disable the strobe port signals. Each disabled port will stay disabled even when strobe signals are assigned to it.

Table 13-15. STBPCR Field Descriptions

Field	Description
3 STB3EN	**Strobe Port 3 Enable** — This control bit defines whether the STB3 port is enabled or disabled. 0 Strobe port STB3 disabled 1 Strobe port STB3 enabled
2 STB2EN	**Strobe Port 2 Enable** — This control bit defines whether the STB2 port is enabled or disabled. 0 Strobe port STB2 disabled 1 Strobe port STB2 enabled
1 STB1EN	**Strobe Port 1 Enable** — This control bit defines whether the STB1 port is enabled or disabled. 0 Strobe port STB1 disabled 1 Strobe port STB1 enabled
0 STB0EN	**Strobe Port 0 Enable** — This control bit defines whether the STB0 port is enabled or disabled. 0 Strobe port STB0 disabled 1 Strobe port STB0 enabled

13.5.2.8 Message Buffer Data Size Register (MBDSR)

Module Base + 0x000C

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	MBSEG2DS							0	MBSEG1DS						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-8. Message Buffer Data Size Register (MBDSR)

Write: *POC:config*

This register defines the size of the message buffer data section for the two message buffer segments in a number of two-byte entities.

The FlexRay block provides two independent segments for the individual message buffers. All individual message buffers within one segment have to have the same size for the message buffer data section. This size can be different for the two message buffer segments.

Table 13-16. MBDSR Field Descriptions

Field	Description
14–8 MBSEG2DS	**Message Buffer Segment 2 Data Size** — The field defines the size of the message buffer data section in two-byte entities for message buffers within the *second* message buffer segment.
6–0 MBSEG1DS	**Message Buffer Segment 1 Data Size** — The field defines the size of the message buffer data section in two-byte entities for message buffers within the *first* message buffer segment.

13.5.2.9 Message Buffer Segment Size and Utilization Register (MBSSUTR)

Module Base + 0x000E

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	LAST_MB_SEG1					0	0	0	LAST_MB_UTIL				
W																
Reset	0	0	0	1	1	1	1	1	0	0	0	1	1	1	1	1

Figure 13-9. Message Buffer Segment Size and Utilization Register (MBSSUTR)

Write: *POC:config*

This register is used to define the last individual message buffer that belongs to the first message buffer segment and the number of the last used individual message buffer.

Table 13-17. MBSSUTR Field Descriptions

Field	Description
12–8 LAST_MB_SEG1	**Last Message Buffer In Segment 1** — This field defines the message buffer number of the last individual message buffer that is assigned to the *first* message buffer segment. The individual message buffers in the *first* segment correspond to the message buffer control registers MBCCSRn, MBCCFRn, MBFIDRn, MBIDXRn with n <= LAST_MB_SEG1. The first message buffer segment contains LAST_MB_SEG1+1 individual message buffers. **Note:** The *first* message buffer segment contains *at least* one individual message buffer. The individual message buffers in the *second* message buffer segment correspond to the message buffer control registers MBCCSRn, MBCCFRn, MBFIDRn, MBIDXRn with LAST_MB_SEG1 < n < 32. **Note:** If LAST_MB_SEG1 = 31 all individual message buffers belong to the *first* message buffer segment and the *second* message buffer segment is empty.
4–0 LAST_MB_UTIL	**Last Message Buffer Utilized** — This field defines the message buffer number of last utilized individual message buffer. The message buffer search engine examines all individual message buffer with a message buffer number n <= LAST_MB_UTIL. **Note:** If LAST_MB_UTIL=LAST_MB_SEG1 all individual message buffers belong to the *first* message buffer segment and the *second* message buffer segment is empty.

13.5.2.10 Protocol Operation Control Register (POCR)

Module Base + 0x0014

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	EOC_AP		ERC_AP		BSY	0	0	0	POCCMD			
W	WME								WMC							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-10. Protocol Operation Control Register (POCR)

Write: Normal Mode

The application uses this register to issue

- protocol control commands
- external clock correction commands

Protocol control commands are issued by writing to the POCCMD field. For more information on protocol control commands, see Section 13.7.4, “Protocol Control Command Execution”.

External clock correction commands are issued by writing to the EOC_AP and ERC_AP fields. For more information on external clock correction, refer to Section 13.6.11, “External Clock Synchronization”.

Table 13-18. POCR Field Descriptions (Sheet 1 of 2)

Field	Description
15 WME	**Write Mode External Correction** — This bit controls the write mode of the EOC_AP and ERC_AP fields. 0 Write to EOC_AP and ERC_AP fields on register write. 1 No write to EOC_AP and ERC_AP fields on register write.
11–10 EOC_AP	**External Offset Correction Application** — This field is used to trigger the application of the external offset correction value defined in the Protocol Configuration Register 29 (PCR29). 00 do not apply external offset correction value 01 reserved 10 subtract external offset correction value 11 add external offset correction value
9–8 ERC_AP	**External Rate Correction Application** — This field is used to trigger application of the external rate correction value defined in the Protocol Configuration Register 21 (PCR21) 00 do not apply external rate correction value 01 reserved 10 subtract external rate correction value 11 add external rate correction value

Table 13-18. POCR Field Descriptions (Sheet 2 of 2)

Field	Description
7 BSY WMC	**Protocol Control Command Write Busy** — This status bit indicates the acceptance of the protocol control command issued by the application via the POCCMD field. The FlexRay block sets this status bit when the application has issued a protocol control command via the POCCMD field. The FlexRay block clears this status bit when protocol control command was accepted by the PE.When the application issues a protocol control command while the BSY bit is asserted, the FlexRay block ignores this command, sets the protocol command ignored error flag PCMI_EF in the CHI Error Flag Register (CHIERFR), and will not change the value of the POCCMD field. 0 Command write idle, command accepted and ready to receive new protocol command. 1 Command write busy, command not yet accepted, not ready to receive new protocol command. **Write Mode Command** — This bit controls the write mode of the POCCMD field. 0 Write to POCCMD field on register write. 1 Do not write to POCCMD field on register write.
3–0 POCCMD	**Protocol Control Command** — The application writes to this field to issue a protocol control command to the PE. The FlexRay block sends the protocol command to the PE immediately. While the transfer is running, the BSY bit is set. 0000 ALLOW_COLDSTART — Immediately activate capability of node to cold start cluster. 0001 ALL_SLOTS — Delayed(1) transition to the all slots transmission mode. 0010 CONFIG — Immediately transition to the *POC:config* state. 0011 FREEZE — Immediately transition to the *POC:halt* state. 0100 READY, CONFIG_COMPLETE — Immediately transition to the *POC:ready* state. 0101 RUN — Immediately transition to the *POC:startup start* state. 0110 DEFAULT_CONFIG — Immediately transition to the *POC:default config* state. 0111 HALT — Delayed transition to the *POC:halt* state 1000 WAKEUP — Immediately initiate the wakeup procedure. 1001 reserved 1010 reserved 1011 reserved 1100 RESET(2) — Immediately reset the Protocol Engine. 1101 reserved 1110 reserved 1111 reserved

1. Delayed means on completion of current communication cycle.
2. Additional to *FlexRay Communications System Protocol Specification, Version 2.1 Rev A*

After sending the RESET command, it is mandatory to execute the command sequence described in Section 13.7.5, “Protocol Reset Command” immediately, to reach the DEFAULT CONFIG state correctly.

13.5.2.11 Global Interrupt Flag and Enable Register (GIFER)

Module Base + 0x0016

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MIF	PRIF	CHIF	WUPIF	FNEBIF	FNEAIF	RBIF	TBIF	MIE	PRIE	CHIE	WUPIE	FNEBIE	FNEAIE	RBIE	TBIE
W				w1c	w1c	w1c										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-11. Global Interrupt Flag and Enable Register (GIFER)

Write: Normal Mode

This register provides the means to control some of the interrupt request lines and provides the corresponding interrupt flags. The interrupt flags MIF, PRIF, CHIF, RBIF, and TBIF are the outcome of a binary OR of the related individual interrupt flags and interrupt enables. The generation scheme for these flags is depicted in Figure 13-144. For more details on interrupt generation, see Section 13.6.19, “Interrupt Support. These flags are cleared automatically when all of the corresponding interrupt flags or interrupt enables in the related interrupt flag and enable registers are cleared by the application.

Table 13-19. GIFER Field Descriptions (Sheet 1 of 2)

Field	Description
15 MIF	**Module Interrupt Flag** — This flag is set if at least one of the other interrupt flags is in this register is asserted and the related interrupt enable is asserted, too. The FlexRay block generates the module interrupt request if MIE is asserted. 0 No interrupt flag is asserted or no interrupt enable is set 1 At least one of the other interrupt flags in this register is asserted and the related interrupt bit is asserted, too
13 PRIF	**Protocol Interrupt Flag** — This flag is set if at least one of the individual protocol interrupt flags in the Protocol Interrupt Flag Register 0 (PIFR0) and Protocol Interrupt Flag Register 1 (PIFR1) is asserted and the related interrupt enable flag is asserted, too. The FlexRay block generates the combined protocol interrupt request if the PRIE flag is asserted. 0 All individual protocol interrupt flags are equal to 0 or no interrupt enable bit is set. 1 At least one of the individual protocol interrupt flags and the related interrupt enable is equal to 1.
13 CHIF	**CHI Interrupt Flag** — This flag is set if at least one of the individual CHI error flags in the CHI Error Flag Register (CHIERFR) is asserted and the chi error interrupt enable GIFER.CHIE is asserted. The FlexRay block generates the combined CHI error interrupt if the CHIE flag is asserted, too. 0 All CHI error flags are equal to 0 or the chi error interrupt is disabled 1 At least one CHI error flag is asserted and chi error interrupt is enabled
12 WUPIF	**Wakeup Interrupt Flag** — This flag is set when the FlexRay block has received a wakeup symbol on the FlexRay bus. The application can determine on which channel the wakeup symbol was received by reading the related wakeup flags WUB and WUA in the Protocol Status Register 3 (PSR3). The FlexRay block generates the wakeup interrupt request if the WUPIE flag is asserted. 0 No wakeup condition or interrupt disabled 1 Wakeup symbol received on FlexRay bus and interrupt enabled
11 FNEBIF	**Receive FIFO channel B Not Empty Interrupt Flag** — This flag is set when the receive FIFO for channel B is not empty. If the application writes 1 to this bit, the FlexRay block updates the FIFO status, increments or wraps the FIFO read index in the Receive FIFO B Read Index Register (RFBRIR) and clears the interrupt flag if the FIFO B is now empty. If the FIFO is still not empty, the FlexRay block sets this flag again. The FlexRay block generates the Receive FIFO B Not empty interrupt if the FNEBIE flag is asserted. 0 Receive FIFO B is empty or interrupt is disabled 1 Receive FIFO B is not empty and interrupt enabled
10 FNEAIF	**Receive FIFO channel A Not Empty Interrupt Flag** — This flag is set when the receive FIFO for channel A is not empty. If the application writes 1 to this bit, the FlexRay block updates the FIFO status, increments or wraps the FIFO read index in the Receive FIFO A Read Index Register (RFARIR) and clears the interrupt flag if the FIFO A is now empty. If the FIFO is still not empty, the FlexRay block sets this flag again. The FlexRay block generates the Receive FIFO A Not empty interrupt if the FNEAIE flag is asserted. 0 Receive FIFO A is empty or interrupt is disabled 1 Receive FIFO A is not empty and interrupt enabled

Table 13-19. GIFER Field Descriptions (Sheet 2 of 2)

Field	Description
9 RBIF	**Receive Message Buffer Interrupt Flag** — This flag is set if for at least one of the individual receive message buffers (MBCCSn.MTD = 0) both the interrupt flag MBIF and the interrupt enable bit MBIE in the corresponding Message Buffer Configuration, Control, Status Registers (MBCCSRn) are asserted. The application can not clear this RBIF flag directly. This flag is cleared by the FlexRay block when all of the interrupt flags MBIF of the individual receive message buffers are cleared by the application or if the application has cleared the interrupt enables bit MBIE. 0 None of the individual receive message buffers has the MBIF and MBIE flag asserted. 1 At least one individual receive message buffer has the MBIF and MBIE flag asserted.
8 TBIF	**Transmit Buffer Interrupt Flag** — This flag is set if for at least one of the individual single or double transmit message buffers (MBCCSn.MTD = 0) both the interrupt flag MBIF and the interrupt enable bit MBIE in the corresponding Message Buffer Configuration, Control, Status Registers (MBCCSRn) are equal to 1. The application can not clear this TBIF flag directly. This flag is cleared by the FlexRay block when either all of the individual interrupt flags MBIF of the individual transmit message buffers are cleared by the application or the host has cleared the interrupt enables bit MBIE. 0 None of the individual transmit message buffers has the MBIF and MBIE flag asserted. 1 At least one individual transmit message buffer has the MBIF and MBIE flag asserted.
7 MIE	**Module Interrupt Enable** — This flag controls if the module interrupt line is asserted when the MIF flag is set. 0 Disable interrupt line 1 Enable interrupt line
6 PRIE	**Protocol Interrupt Enable** — This flag controls if the protocol interrupt line is asserted when the PRIF flag is set. 0 Disable interrupt line 1 Enable interrupt line
5 CHIE	**CHI Interrupt Enable** — This flag controls if the CHI interrupt line is asserted when the CHIF flag is set. 0 Disable interrupt line 1 Enable interrupt line
4 WUPIE	**Wakeup Interrupt Enable** — This flag controls if the wakeup interrupt line is asserted when the WUPIF flag is set. 0 Disable interrupt line 1 Enable interrupt line
3 FNEBIE	**Receive FIFO channel B Not Empty Interrupt Enable** — This flag controls if the receive FIFO B interrupt line is asserted when the FNEBIF flag is set. 0 Disable interrupt line 1 Enable interrupt line
2 FNEAIE	**Receive FIFO channel A Not Empty Interrupt Enable** — This flag controls if the receive FIFO A interrupt line is asserted when the FNEAIF flag is set. 0 Disable interrupt line 1 Enable interrupt line
1 RBIE	**Receive Buffer Interrupt Enable** — This flag controls if the receive buffer interrupt line is asserted when the RBIF flag is set. 0 Disable interrupt line 1 Enable interrupt line
0 TBIE	**Transmit Interrupt Enable** — This flag controls if the transmit buffer interrupt line is asserted when the TBIF flag is set. 0 Disable interrupt line 1 Enable interrupt line

13.5.2.12 Protocol Interrupt Flag Register 0 (PIFR0)

Module Base + 0x0018

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FATL_IF	INTL_IF	ILCF_IF	CSA_IF	MRC_IF	MOC_IF	CCL_IF	MXS_IF	MTX_IF	LTXB_IF	LTXA_IF	TBVB_IF	TBVA_IF	TI2_IF	TI1_IF	CYS_IF
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-12. Protocol Interrupt Flag Register 0 (PIFR0)

Write: Normal Mode

The register holds one set of the protocol-related individual interrupt flags.

Table 13-20. PIFR0 Field Descriptions (Sheet 1 of 2)

Field	Description
15 FATL_IF	**Fatal Protocol Error Interrupt Flag** — This flag is set when the protocol engine has detected a fatal protocol error. In this case, the protocol engine goes into the *POC:halt* state immediately. The fatal protocol errors are: 1) *pLatestTx* violation, as described in the MAC process of the FlexRay protocol 2) transmission across slot boundary violation, as described in the FSP process of the FlexRay protocol 0 No such event. 1 Fatal protocol error detected.
14 INTL_IF	**Internal Protocol Error Interrupt Flag** — This flag is set when the protocol engine has detected an internal protocol error. In this case, the protocol engine goes into the *POC:halt* state immediately. An internal protocol error occurs when the protocol engine has not finished a calculation and a new calculation is requested. This can be caused by a hardware error. 0 No such event. 1 Internal protocol error detected.
13 ILCF_IF	**Illegal Protocol Configuration Interrupt Flag** — This flag is set when the protocol engine has detected an illegal protocol configuration parameter setting. In this case, the protocol engine goes into the *POC:halt* state immediately. The protocol engine checks the *listen_timeout* value programmed into the Protocol Configuration Register 14 (PCR14) and Protocol Configuration Register 15 (PCR15) when the CONFIG_COMPLETE command was sent by the application via the Protocol Operation Control Register (POCR). If the value of *listen_timeout* is equal to zero, the protocol configuration setting is considered as illegal. 0 No such event. 1 Illegal protocol configuration detected.
12 CSA_IF	**Cold Start Abort Interrupt Flag** — This flag is set when the configured number of allowed cold start attempts is reached and none of these attempts was successful. The number of allowed cold start attempts is configured by the coldstart_attempts field in the Protocol Configuration Register 3 (PCR3). 0 No such event. 1 Cold start aborted and no more coldstart attempts allowed.
11 MRC_IF	**Missing Rate Correction Interrupt Flag** — This flag is set when an insufficient number of measurements is available for rate correction at the end of the communication cycle. 0 No such event 1 Insufficient number of measurements for rate correction detected
10 MOC_IF	**Missing Offset Correction Interrupt Flag** — This flag is set when an insufficient number of measurements is available for offset correction. This is related to the MISSING_TERM event in the CSP process for offset correction in the FlexRay protocol. 0 No such event. 1 Insufficient number of measurements for offset correction detected.

Table 13-20. PIFR0 Field Descriptions (Sheet 2 of 2)

Field	Description
9 CCL_IF	**Clock Correction Limit Reached Interrupt Flag** — This flag is set when the internal calculated offset or rate calculation values have reached or exceeded its configured thresholds as given by the *offset_coorection_out* field in the Protocol Configuration Register 9 (PCR9) and the *rate_correction_out* field in the Protocol Configuration Register 14 (PCR14). 0 No such event. 1 Offset or rate correction limit reached.
8 MXS_IF	**Max Sync Frames Detected Interrupt Flag** — This flag is set when the number of synchronization frames detected in the current communication cycle exceeds the value of the *node_sync_max* field in the Protocol Configuration Register 30 (PCR30). 0 No such event. 1 More than *node_sync_max* sync frames detected. **Note:** Only synchronization frames that have passed the synchronization frame acceptance and rejection filters are taken into account.
7 MTX_IF	**Media Access Test Symbol Received Interrupt Flag** — This flag is set when the MTS symbol was received on channel A or channel B. 0 No such event. 1 MTS symbol received.
6 LTXB_IF	*pLatestTx* **Violation on Channel B Interrupt Flag** — This flag is set when the frame transmission on channel B in the dynamic segment exceeds the dynamic segment boundary. This is related to the *pLatestTx* violation, as described in the MAC process of the FlexRay protocol. 0 No such event. 1 *pLatestTx* violation occurred on channel B.
5 LTXA_IF	*pLatestTx* **Violation on Channel A Interrupt Flag** — This flag is set when the frame transmission on channel A in the dynamic segment exceeds the dynamic segment boundary. This is related to the *pLatestTx* violation as described in the MAC process of the FlexRay protocol. 0 No such event. 1 *pLatestTx* violation occurred on channel A.
4 TBVB_IF	**Transmission across boundary on channel B Interrupt Flag** — This flag is set when the frame transmission on channel B crosses the slot boundary. This is related to the transmission across slot boundary violation as described in the FSP process of the FlexRay protocol. 0 No such event. 1 Transmission across boundary violation occurred on channel B.
3 TBVA_IF	**Transmission across boundary on channel A Interrupt Flag** — This flag is set when the frame transmission on channel A crosses the slot boundary. This is related to the transmission across slot boundary violation as described in the FSP process of the FlexRay protocol. 0 No such event. 1 Transmission across boundary violation occurred on channel A.
2 TI2_IF	**Timer 2 Expired Interrupt Flag** — This flag is set whenever timer 2 expires. 0 No such event. 1 Timer 2 has reached its time limit.
1 TI1_IF	**Timer 1 Expired Interrupt Flag** — This flag is set whenever timer 1 expires. 0 No such event 1 Timer 1 has reached its time limit
0 CYS_IF	**Cycle Start Interrupt Flag** — This flag is set when a communication cycle starts. 0 No such event 1 Communication cycle started.

13.5.2.13 Protocol Interrupt Flag Register 1 (PIFR1)

Module Base + 0x001A

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EMC_IF	IPC_IF	PECF_IF	PSC_IF	SSI3_IF	SSI2_IF	SSI1_IF	SSI0_IF	0	0	EVT_IF	ODT_IF	0	0	0	0
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c			w1c	w1c				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-13. Protocol Interrupt Flag Register 1 (PIFR1)

Write: Normal Mode

The register holds one set of the protocol-related individual interrupt flags.

Table 13-21. PIFR1 Field Descriptions

Field	Description
15 EMC_IF	**Error Mode Changed Interrupt Flag** — This flag is set when the value of the ERRMODE bit field in the Protocol Status Register 0 (PSR0) is changed by the FlexRay block. 0 No such event. 1 ERRMODE field changed.
14 IPC_IF	**Illegal Protocol Control Command Interrupt Flag** — This flag is set when the PE tries to execute a protocol control command, which was issued via the POCCMD field of the Protocol Operation Control Register (POCR), and detects that this protocol control command is not allowed in the current protocol state. In this case the command is not executed. For more details, see Section 13.7.4, “Protocol Control Command Execution”. 0 No such event. 1 Illegal protocol control command detected.
13 PECF_IF	**Protocol Engine Communication Failure Interrupt Flag** — This flag is set if the FlexRay block has detected a communication failure between the protocol engine and the controller host interface 0 No such event. 1 Protocol Engine Communication Failure detected.
12 PSC_IF	**Protocol State Changed Interrupt Flag** — This flag is set when the protocol state in the PROTSTATE field in the Protocol Status Register 0 (PSR0) has changed. 0 No such event. 1 Protocol state changed.
11–8 SSI[3:0]_IF	**Slot Status Counter Incremented Interrupt Flag** — Each of these flags is set when the SLOTSTATUSCNT field in the corresponding Slot Status Counter Registers (SSCR0–SSCR3) is incremented. 0 No such event. 1 The corresponding slot status counter has incremented.
5 EVT_IF	**Even Cycle Table Written Interrupt Flag** — This flag is set if the FlexRay block has written the sync frame measurement / ID tables into the FlexRay Memory for the even cycle. 0 No such event. 1 Sync frame measurement table written
4 ODT_IF	**Odd Cycle Table Written Interrupt Flag** — This flag is set if the FlexRay block has written the sync frame measurement / ID tables into the FlexRay Memory for the odd cycle. 0 No such event. 1 Sync frame measurement table written

13.5.2.14 Protocol Interrupt Enable Register 0 (PIER0)

Module Base + 0x001C

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	FATL_IE	INTL_IE	ILCF_IE	CSA_IE	MRC_IE	MOC_IE	CCL_IE	MXS_IE	MTX_IE	LTXB_IE	LTXA_IE	TBVB_IE	TBVA_IE	TI2_IE	TI1_IE	CYS_IE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-14. Protocol Interrupt Enable Register 0 (PIER0)

Write: Anytime

This register defines whether or not the individual interrupt flags defined in the Protocol Interrupt Flag Register 0 (PIFR0) can generate a protocol interrupt request.

Table 13-22. PIER0 Field Descriptions

Field	Description
15 FATL_IE	**Fatal Protocol Error Interrupt Enable** — This bit controls FATL_IF interrupt request generation. 0 interrupt request generation disabled 1 interrupt request generation enabled
14 INTL_IE	**Internal Protocol Error Interrupt Enable** — This bit controls INTL_IF interrupt request generation. 0 interrupt request generation disabled 1 interrupt request generation enabled
13 ILCF_IE	**Illegal Protocol Configuration Interrupt Enable** — This bit controls ILCF_IF interrupt request generation. 0 interrupt request generation disabled 1 interrupt request generation enabled
12 CSA_IE	**Cold Start Abort Interrupt Enable** — This bit controls CSA_IF interrupt request generation. 0 interrupt request generation disabled 1 interrupt request generation enabled
11 MRC_IE	**Missing Rate Correction Interrupt Enable** — This bit controls MRC_IF interrupt request generation. 0 interrupt request generation disabled 1 interrupt request generation enabled
10 MOC_IE	**Missing Offset Correction Interrupt Enable** — This bit controls MOC_IF interrupt request generation. 0 interrupt request generation disabled 1 interrupt request generation enabled
9 CCL_IE	**Clock Correction Limit Reached Interrupt Enable** — This bit controls CCL_IF interrupt request generation. 0 interrupt request generation disabled 1 interrupt request generation enabled
8 MXS_IE	**Max Sync Frames Detected Interrupt Enable** — This bit controls MXS_IF interrupt request generation. 0 interrupt request generation disabled 1 interrupt request generation enabled
7 MTX_IE	**Media Access Test Symbol Received Interrupt Enable** — This bit controls MTX_IF interrupt request generation. 0 interrupt request generation disabled 1 interrupt request generation enabled
6 LTXB_IE	*pLatestTx* **Violation on Channel B Interrupt Enable** — This bit controls LTXB_IF interrupt request generation. 0 interrupt request generation disabled 1 interrupt request generation enabled
5 LTXA_IE	*pLatestTx* **Violation on Channel A Interrupt Enable** — This bit controls LTXA_IF interrupt request generation. 0 interrupt request generation disabled 1 interrupt request generation enabled

Table 13-22. PIER0 Field Descriptions (continued)

Field	Description
4 TBVB_IE	**Transmission across boundary on channel B Interrupt Enable** — This bit controls TBVB_IF interrupt request generation. 0 interrupt request generation disabled 1 interrupt request generation enabled
3 TBVA_IE	**Transmission across boundary on channel A Interrupt Enable** — This bit controls TBVA_IF interrupt request generation. 0 interrupt request generation disabled 1 interrupt request generation enabled
2 TI2_IE	**Timer 2 Expired Interrupt Enable** — This bit controls TI1_IF interrupt request generation. 0 interrupt request generation disabled 1 interrupt request generation enabled
1 TI1_IE	**Timer 1 Expired Interrupt Enable** — This bit controls TI1_IF interrupt request generation. 0 interrupt request generation disabled 1 interrupt request generation enabled
0 CYS_IE	**Cycle Start Interrupt Enable** — This bit controls CYC_IF interrupt request generation. 0 interrupt request generation disabled 1 interrupt request generation enabled

13.5.2.15 Protocol Interrupt Enable Register 1 (PIER1)

Module Base + 0x001E

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EMC_IE	IPC_IE	PECF_IE	PSC_IE	SSI3_IE	SSI2_IE	SSI1_IE	SSI0_IE	0	0	EVT_IE	ODT_IE	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-15. Protocol Interrupt Enable Register 1 (PIER1)

Write: Anytime

This register defines whether or not the individual interrupt flags defined in Protocol Interrupt Flag Register 1 (PIFR1) can generate a protocol interrupt request.

Table 13-23. PIER1 Field Descriptions

Field	Description
15 EMC_IE	**Error Mode Changed Interrupt Enable** — This bit controls EMC_IF interrupt request generation. 0 interrupt request generation disabled 1 interrupt request generation enabled
14 IPC_IE	**Illegal Protocol Control Command Interrupt Enable** — This bit controls IPC_IF interrupt request generation. 0 interrupt request generation disabled 1 interrupt request generation enabled
13 PECF_IE	**Protocol Engine Communication Failure Interrupt Enable** — This bit controls PECF_IF interrupt request generation. 0 interrupt request generation disabled 1 interrupt request generation enabled
12 PSC_IE	**Protocol State Changed Interrupt Enable** — This bit controls PSC_IF interrupt request generation. 0 interrupt request generation disabled 1 interrupt request generation enabled

Table 13-23. PIER1 Field Descriptions (continued)

Field	Description
11–8 SSI[3:0]_IE	**Slot Status Counter Incremented Interrupt Enable** — This bit controls SSI[3:0]_IF interrupt request generation. 0 interrupt request generation disabled 1 interrupt request generation enabled
5 EVT_IE	**Even Cycle Table Written Interrupt Enable** — This bit controls EVT_IF interrupt request generation. 0 interrupt request generation disabled 1 interrupt request generation enabled
4 ODT_IE	**Odd Cycle Table Written Interrupt Enable** — This bit controls ODT_IF interrupt request generation. 0 interrupt request generation disabled 1 interrupt request generation enabled

13.5.2.16 CHI Error Flag Register (CHIERFR)

Module Base + 0x0020

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FRLB_EF	FRLA_EF	PCMI_EF	FOVB_EF	FOVA_EF	MBS_EF	MBU_EF	LCK_EF	DBL_EF	SBCF_EF	FID_EF	DPL_EF	SPL_EF	NML_EF	NMF_EF	ILSA_EF
W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-16. CHI Error Flag Register (CHIERFR)

Write: Normal Mode

This register holds the CHI related error flags. The interrupt generation for each of these error flags is controlled by the CHI interrupt enable bit CHIE in the Global Interrupt Flag and Enable Register (GIFER).

Table 13-24. CHIERFR Field Descriptions (Sheet 1 of 3)

Field	Description
15 FRLB_EF	**Frame Lost Channel B Error Flag** — This flag is set if a complete frame was received on channel B but could not be stored in the selected individual message buffer because this message buffer is currently locked by the application. In this case, the frame and the related slot status information are lost. 0 No such event 1 Frame lost on channel B detected
14 FRLA_EF	**Frame Lost Channel A Error Flag** — This flag is set if a complete frame was received on channel A but could not be stored in the selected individual message buffer because this message buffer is currently locked by the application. In this case, the frame and the related slot status information are lost. 0 No such error 1 Frame lost on channel A detected
13 PCMI_EF	**Protocol Command Ignored Error Flag** — This flag is set if the application has issued a POC command by writing to the POCCMD field in the Protocol Operation Control Register (POCR) while the BSY flag is equal to 1. In this case the command is ignored by the FlexRay block and is lost. 0 No such error 1 POC command ignored

Table 13-24. CHIERFR Field Descriptions (Sheet 2 of 3)

Field	Description
12 FOVB_EF	**Receive FIFO Overrun Channel B Error Flag** — This flag is set when an overrun of the Receive FIFO for channel B occurred. This error occurs if a semantically valid frame was received on channel B and matches the all criteria to be appended to the FIFO for channel B but the FIFO is full. In this case, the received frame and its related slot status information is lost. 0 No such error 1 Receive FIFO overrun on channel B has been detected
11 FOVA_EF	**Receive FIFO Overrun Channel A Error Flag** — This flag is set when an overrun of the Receive FIFO for channel A occurred. This error occurs if a semantically valid frame was received on channel A and matches the all criteria to be appended to the FIFO for channel A but the FIFO is full. In this case, the received frame and its related slot status information is lost. 0 No such error 1 Receive FIFO overrun on channel B has been detected
10 MSB_EF	**Message Buffer Search Error Flag** — This flag is set if the message buffer search engine is still running while the next search cycle must be started due to the FlexRay protocol timing. In this case, not all message buffers are considered while searching. 0 No such event 1 Search engine active while search start appears
9 MBU_EF	**Message Buffer Utilization Error Flag** — This flag is asserted if the application writes to a message buffer control field that is beyond the number of utilized message buffers programmed in the **Message Buffer Segment Size and Utilization Register (MBSSUTR).** If the application writes to a MBCCSRn register with n > LAST_MB_UTIL, the FlexRay block ignores the write attempt and asserts the message buffer utilization error flag MBU_EF in the CHI Error Flag Register (CHIERFR). 0 No such event 1 Non-utilized message buffer enabled
8 LCK_EF	**Lock Error Flag** — This flag is set if the application tries to lock a message buffer that is already locked by the FlexRay block due to internal operations. In that case, the FlexRay block does not grant the lock to the application. The application must issue the lock request again. 0 No such error 1 Lock error detected
7 DBL_EF	**Double Transmit Message Buffer Lock Error Flag** — This flag is set if the application tries to lock the transmit side of a double transmit message buffer. In this case, the FlexRay block does not grant the lock to the transmit side of a double transmit message buffer. 0 No such event 1 Double transmit buffer lock error occurred
6 SBCF_EF	**System Bus Communication Failure Error Flag** — This flag is set if the FlexRay block was not able to transmit or receive data via the system bus in time. In the case of writing, data is lost; in the case of reading, the transmission onto the FlexRay bus is stopped for the current slot and resumed in the next slot. 0 No such event 1 System bus communication failure occurred
5 FID_EF	**Frame ID Error Flag** — This flag is set if the frame ID stored in the message buffer header area differs from the frame ID stored in the message buffer control register. 0 No such error occurred 1 Frame ID error occurred
4 DPL_EF	**Dynamic Payload Length Error Flag** — This flag is set if the payload length written into the message buffer header field of a single or double transmit message buffer assigned to the dynamic segment is greater than the maximum payload length for the dynamic segment as it is configured in the corresponding protocol configuration register field max_payload_length_dynamic in the Protocol Configuration Register 24 (PCR24). 0 No such error occurred 1 Dynamic payload length error occurred

Table 13-24. CHIERFR Field Descriptions (Sheet 3 of 3)

Field	Description
3 SPL_EF	**Static Payload Length Error** Flag — This flag is set if the payload length written into the message buffer header field of a single or double transmit message buffer assigned to the static segment is different from the payload length for the static segment as it is configured in the corresponding protocol configuration register field payload_length_static in the Protocol Configuration Register 19 (PCR19). 0 No such error occurred 1 Static payload length error occurred
2 NML_EF	**Network Management Length Error Flag** — This flag is set if the payload length written into the header structure of a receive message buffer assigned to the static segment is less than the configured length of the Network Management Vector as configured in the Network Management Vector Length Register (NMVLR). In this case the received part of the Network Management Vector will be used to update the Network Management Vector. 0 No such error occurred 1 Network management length error occurred
1 NMF_EF	**Network Management Frame Error Flag** — This flag is set if a received message in the static segment with a Preamble Indicator flag PP asserted has its Null Frame indicator flag NF asserted as well. In this case, the Global Network Management Registers (see Network Management Vector Registers (NMVR0–NMVR5)) are not updated. 0 No such error occurred 1 Network management frame error occurred
0 ILSA_EF	**Illegal System Memory Access Error Flag** — This flag is set if the external system memory subsystem has detected and indicated an illegal system memory access from the FlexRay block. The exact meaning of an illegal system memory access is defined by the current implementation of the memory subsystem. 0 No such event. 1 Illegal system memory access occurred.

13.5.2.17 Message Buffer Interrupt Vector Register (MBIVEC)

Module Base + 0x0022

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	TBIVEC					0	0	0	RBIVEC				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-17. Message Buffer Interrupt Vector Register (MBIVEC)

This register indicates the lowest numbered receive message buffer and the lowest numbered transmit message buffer that have their interrupt status flag MBIF and interrupt enable MBIE bits asserted. This means that message buffers with lower message buffer numbers have higher priority.

Table 13-25. MBIVEC Field Descriptions

Field	Description
12–8 TBIVEC	**Transmit Buffer Interrupt Vector** — This field provides the number of the lowest numbered enabled transmit message buffer that has its interrupt status flag MBIF and its interrupt enable bit MBIE set. If there is no transmit message buffer with the interrupt status flag MBIF and the interrupt enable MBIE bits asserted, the value in this field is set to 0.
4–0 RBIVEC	**Receive Buffer Interrupt Vector** — This field provides the message buffer number of the lowest numbered receive message buffer which has its interrupt flag MBIF and its interrupt enable bit MBIE asserted. If there is no receive message buffer with the interrupt status flag MBIF and the interrupt enable MBIE bits asserted, the value in this field is set to 0.

13.5.2.18 Channel A Status Error Counter Register (CASERCR)

Module Base + 0x0024 Additional Reset: RUN Command

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STATUS_ERR_CNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-18. Channel A Status Error Counter Register (CASERCR)

This register provides the channel status error counter for channel A. The protocol engine generates a slot status vector for each static slot, each dynamic slot, the symbol window, and the NIT. The slot status vector contains the four protocol related error indicator bits *vSS!SyntaxError, vSS!ContentError, vSS!BViolation*, and *vSS!TxConflict*. The FlexRay block increments the status error counter by 1 if, for a slot or segment, at least one error indicator bit is set to 1. The counter wraps around after it has reached the maximum value. For more information on slot status monitoring, see Section 13.6.18, “Slot Status Monitoring”.

Table 13-26. CASERCR Field Descriptions

Field	Description
15–0 STATUS_ERR_CNT	**Channel Status Error Counter** — This field provides the current value channel status error counter. The counter value is updated within the first macrotick of the following slot or segment.

13.5.2.19 Channel B Status Error Counter Register (CBSERCR)

Module Base + 0x0026 Additional Reset: RUN Command

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STATUS_ERR_CNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-19. Channel B Status Error Counter Register (CBSERCR)

This register provides the channel status error counter for channel B. The protocol engine generates a slot status vector for each static slot, each dynamic slot, the symbol window, and the NIT. The slot status vector contains the four protocol related error indicator bits *vSS!SyntaxError*, *vSS!ContentError*, *vSS!BViolation*, and *vSS!TxConflict*. The FlexRay block increments the status error counter by 1 if, for a slot or segment, at least one error indicator bit is set to 1. The counter wraps around after it has reached the maximum value. For more information on slot status monitoring see Section 13.6.18, “Slot Status Monitoring”.

Table 13-27. CBSERCR Field Descriptions

Field	Description
15–0 STATUS_ERR_CNT	**Channel Status Error Counter** — This field provides the current channel status error count. The counter value is updated within the first macrotick of the following slot or segment.

13.5.2.20 Protocol Status Register 0 (PSR0)

Module Base + 0x0028

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ERRMODE		SLOTMODE		0	PROTSTATE			STARTUPSTATE				0	WAKEUPSTATUS		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-20. Protocol Status Register 0 (PSR0)

This register provides information about the current protocol status.

Table 13-28. PSR0 Field Descriptions (Sheet 1 of 2)

Field	Description
15–14 ERRMODE	**Error Mode** — protocol related variable: *vPOC!ErrorMode*. This field indicates the error mode of the protocol. 00 ACTIVE 01 PASSIVE 10 COMM_HALT 11 reserved
13–12 SLOTMODE	**Slot Mode** — protocol related variable: *vPOC!SlotMode*. This field indicates the slot mode of the protocol. 00 SINGLE 01 ALL_PENDING 10 ALL 11 reserved
10–8 PROTSTATE	**Protocol State** — protocol related variable: *vPOC!State.* This field indicates the state of the protocol. 000 *POC:default config* 001 *POC:config* 010 *POC:wakeup* 011 *POC:ready* 100 *POC:normal passive* 101 *POC:normal active* 110 *POC:halt* 111 *POC:startup*

Table 13-28. PSR0 Field Descriptions (Sheet 2 of 2)

Field	Description
7–4 STARTUP STATE	**Startup State** — protocol related variable: *vPOC!StartupState.* This field indicates the current sub-state of the startup procedure. 0000 reserved 0001 reserved 0010 *POC:coldstart collision resolution* 0011 *POC:coldstart listen* 0100 *POC:integration consistency check* 0101 *POC:integrationi listen* 0110 reserved 0111 *POC:initialize schedule* 1000 reserved 1001 reserved 1010 *POC:coldstart consistency check* 1011 reserved 1100 reserved 1101 *POC:integration coldstart check* 1110 *POC:coldstart gap* 1111 *POC:coldstart join*
2–0 WAKEUP STATUS	**Wakeup Status** — protocol related variable: *vPOC!WakeupStatus*. This field provides the outcome of the execution of the wakeup mechanism. 000 UNDEFINED 001 RECEIVED_HEADER 010 RECEIVED_WUP 011 COLLISION_HEADER 100 COLLISION_WUP 101 COLLISION_UNKNOWN 110 TRANSMITTED 111 reserved

13.5.2.21 Protocol Status Register 1 (PSR1)

Module Base + 0x002A Additional Reset: CSAA, CSP, CPN: RUN Command

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CSAA	CSP	0	REMCSAT					CPN	HHR	FRZ	APTAC				
W	w1c															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-21. Protocol Status Register 1 (PSR1)

Write: Normal Mode

Table 13-29. PSR1 Field Descriptions

Field	Description
15 CSAA	**Cold Start Attempt Aborted Flag** — protocol related event: 'set coldstart abort indicator in CHI' This flag is set when the FlexRay block has aborted a cold start attempt. 0 No such event 1 Cold start attempt aborted
14 CSP	**Leading Cold Start Path** — This status bit is set when the FlexRay block has reached the *POC:normal active* state via the leading cold start path. This indicates that this node has started the network 0 No such event 1 *POC:normal active* reached from *POC:startup* state via leading cold start path
12–8 REMCSAT	**Remaining Coldstart Attempts** — protocol related variable: *vRemainingColdstartAttempts* This field provides the number of remaining cold start attempts that the FlexRay block will execute.
7 CPN	**Leading Cold Start Path Noise** — protocol related variable: *vPOC!ColdstartNoise* This status bit is set if the FlexRay block has reached the *POC:normal active* state via the leading cold start path under noise conditions. This indicates there was some activity on the FlexRay bus while the FlexRay block was starting up the cluster. 0 No such event 1 *POC:normal active* state was reached from *POC:startup* state via noisy leading cold start path
6 HHR	**Host Halt Request Pending** — protocol related variable: *vPOC!CHIHaltRequest* This status bit is set when FlexRay block receives the HALT command from the application via the Protocol Operation Control Register (POCR). The FlexRay block clears this status bit after a hard reset condition or when the protocol is in the *POC:default config* state. 0 No such event 1 HALT command received
5 FRZ	**Freeze Occurred** — protocol related variable: *vPOC!Freeze* This status bit is set when the FlexRay block has reached the *POC:halt* state due to the host FREEZE command or due to an internal error condition requiring immediate halt. The FlexRay block clears this status bit after a hard reset condition or when the protocol is in the *POC:default config* state. 0 No such event 1 Immediate halt due to FREEZE or internal error condition
4–0 APTAC	**Allow Passive to Active Counter** — protocol related variable: *vPOC!vAllowPassivetoActive* This field provides the number of consecutive even/odd communication cycle pairs that have passed with valid rate and offset correction terms, but the protocol is still in the *POC:normal passive* state due to an application configured delay to enter *POC:normal active* state. This delay is defined by the allow_passive_to_active field in the Protocol Configuration Register 12 (PCR12).

13.5.2.22 Protocol Status Register 2 (PSR2)

Module Base + 0x002C — Additional Reset: RUN Command

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NBVB	NSEB	STCB	SBVB	SSEB	MTB	NBVA	NSEA	STCA	SBVA	SSEA	MTA	CLKCORRFAILCNT			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-22. Protocol Status Register 2 (PSR2)

This register provides a snapshot of status information about the Network Idle Time NIT, the Symbol Window and the clock synchronization. The NIT related status bits NBVB, NSEB, NBVA, and NSEA are updated by the FlexRay block after the end of the NIT and before the end of the first slot of the next communication cycle. The Symbol Window related status bits STCB, SBVB, SSEB, MTB, STCA, SBVA,

SSEB, and MTA are updated by the FlexRay block after the end of the symbol window and before the end of the current communication cycle. If no symbol window is configured, the symbol window related status bits remain in their reset state. The clock synchronization related CLKCORRFAILCNT is updated by the FlexRay block after the end of the static segment and before the end of the current communication cycle.

Table 13-30. PSR2 Field Descriptions (Sheet 1 of 2)

Field	Description
15 NBVB	**NIT Boundary Violation on Channel B** — protocol related variable: *vSS!BViolation* for NIT on channel B This status bit is set when there was some media activity on the FlexRay bus channel B at the end of the NIT. 0 No such event 1 Media activity at boundaries detected
14 NSEB	**NIT Syntax Error on Channel B** — protocol related variable: *vSS!SyntaxError* for NIT on channel B This status bit is set when a syntax error was detected during NIT on channel B. 0 No such event 1 Syntax error detected
13 STCB	**Symbol Window Transmit Conflict on Channel B** — protocol related variable: *vSS!TxConflict* for symbol window on channel B This status bit is set if there was a transmission conflict during the symbol window on channel B. 0 No such event 1 Transmission conflict detected
12 SBVB	**Symbol Window Boundary Violation on Channel B** — protocol related variable: *vSS!BViolation* for symbol window on channel B This status bit is set if there was some media activity on the FlexRay bus channel B at the start or at the end of the symbol window. 0 No such event 1 Media activity at boundaries detected
11 SSEB	**Symbol Window Syntax Error on Channel B** — protocol related variable: *vSS!SyntaxError* for symbol window on channel B This status bit is set when a syntax error was detected during the symbol window on channel B. 0 No such event 1 Syntax error detected
10 MTB	**Media Access Test Symbol MTS Received on Channel B** — protocol related variable: *vSS!ValidMTS* for Symbol Window on channel B This status bit is set if the Media Access Test Symbol MTS was received in the symbol window on channel B. 0 No such event 1 MTS symbol received
9 NBVA	**NIT Boundary Violation on Channel A** — protocol related variable: *vSS!BViolation* for NIT on channel A This status bit is set when there was some media activity on the FlexRay bus channel A at the end of the NIT. 0 No such event 1 Media activity at boundaries detected
8 NSEA	**NIT Syntax Error on Channel A** — protocol related variable: *vSS!SyntaxError* for NIT on channel A This status bit is set when a syntax error was detected during NIT on channel A. 0 No such event 1 Syntax error detected
7 STCA	**Symbol Window Transmit Conflict on Channel A** — protocol related variable: *vSS!TxConflict* for symbol window on channel A This status bit is set if there was a transmission conflicts during the symbol window on channel A. 0 No such event 1 Transmission conflict detected

Table 13-30. PSR2 Field Descriptions (Sheet 2 of 2)

Field	Description
6 SBVA	**Symbol Window Boundary Violation on Channel A** — protocol related variable: *vSS!BViolation* for symbol window on channel A This status bit is set if there was some media activity on the FlexRay bus channel A at the start or at the end of the symbol window. 0 No such event 1 Media activity at boundaries detected
5 SSEA	**Symbol Window Syntax Error on Channel A** — protocol related variable: *vSS!SyntaxError* for symbol window on channel A This status bit is set when a syntax error was detected during the symbol window on channel A. 0 No such event 1 Syntax error detected
4 MTA	**Media Access Test Symbol MTS Received on Channel A** — protocol related variable: *vSS!ValidMTS* for symbol window on channel A This status bit is set if the Media Access Test Symbol MTS was received in the symbol window on channel A. 1 MTS symbol received 0 No such event
3–0 CLKCORR-FAILCNT	**Clock Correction Failed Counter** — protocol related variable: *vClockCorrectionFailed* This field provides the number of consecutive even/odd communication cycle pairs that have passed without clock synchronization having performed an offset or a rate correction due to lack of synchronization frames. It is not incremented when it has reached the configured value of either max_without_clock_correction_fatal or max_without_clock_correction_passive as defined in the Protocol Configuration Register 8 (PCR8). The FlexRay block resets this counter on a hard reset condition, when the protocol enters the *POC:normal active* state, or when both the rate and offset correction terms have been calculated successfully.

13.5.2.23 Protocol Status Register 3 (PSR3)

Module Base + 0x002E — Additional Reset: RUN Command

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	WUB	ABVB	AACB	ACEB	ASEB	AVFB	0	0	WUA	ABVA	AACA	ACEA	ASEA	AVFA
W			w1c	w1c	w1c	w1c	w1c	w1c			w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-23. Protocol Status Register 3 (PSR3)

Write: Normal Mode

This register provides aggregated channel status information as an accrued status of channel activity for all communication slots, regardless of whether they are assigned for transmission or subscribed for reception. It provides accrued information for the symbol window, the NIT, and the wakeup status.

Table 13-31. PSR3 Field Descriptions (Sheet 1 of 2)

Field	Description
13 WUB	**Wakeup Symbol Received on Channel B** — This flag is set when a wakeup symbol was received on channel B. 0 No wakeup symbol received 1 Wakeup symbol received
12 ABVB	**Aggregated Boundary Violation on Channel B** — This flag is set when a boundary violation has been detected on channel B. Boundary violations are detected in the communication slots, the symbol window, and the NIT. 0 No boundary violation detected 1 Boundary violation detected
11 AACB	**Aggregated Additional Communication on Channel B** — This flag is set when at least one valid frame was received on channel B in a slot that also contained an additional communication with either syntax error, content error, or boundary violations. 0 No additional communication detected 1 Additional communication detected
10 ACEB	**Aggregated Content Error on Channel B** — This flag is set when a content error has been detected on channel B. Content errors are detected in the communication slots, the symbol window, and the NIT. 0 No content error detected 1 Content error detected
9 ASEB	**Aggregated Syntax Error on Channel B** — This flag is set when a syntax error has been detected on channel B. Syntax errors are detected in the communication slots, the symbol window and the NIT. 0 No syntax error detected 1 Syntax errors detected
8 AVFB	**Aggregated Valid Frame on Channel B** — This flag is set when a syntactically correct valid frame has been received in any static or dynamic slot through channel B. 1 At least one syntactically valid frame received 0 No syntactically valid frames received
5 WUA	**Wakeup Symbol Received on Channel A** — This flag is set when a wakeup symbol was received on channel A. 0 No wakeup symbol received 1 Wakeup symbol received
4 ABVA	**Aggregated Boundary Violation on Channel A** — This flag is set when a boundary violation has been detected on channel A. Boundary violations are detected in the communication slots, the symbol window, and the NIT. 0 No boundary violation detected 1 Boundary violation detected
3 AACA	**Aggregated Additional Communication on Channel A** — This flag is set when a valid frame was received in a slot on channel A that also contained an additional communication with either syntax error, content error, or boundary violations. 0 No additional communication detected 1 Additional communication detected
2 ACEA	**Aggregated Content Error on Channel A** — This flag is set when a content error has been detected on channel A. Content errors are detected in the communication slots, the symbol window, and the NIT. 0 No content error detected 1 Content error detected

Table 13-31. PSR3 Field Descriptions (Sheet 2 of 2)

Field	Description
1 ASEA	**Aggregated Syntax Error on Channel A** — This flag is set when a syntax error has been detected on channel A. Syntax errors are detected in the communication slots, the symbol window, and the NIT. 0 No syntax error detected 1 Syntax errors detected
0 AVFA	**Aggregated Valid Frame on Channel A** — This flag is set when a syntactically correct valid frame has been received in any static or dynamic slot through channel A. 0 No syntactically valid frames received 1 At least one syntactically valid frame received

13.5.2.24 Macrotick Counter Register (MTCTR)

Module Base + 0x0030

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	MTCT													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-24. Macrotick Counter Register (MTCTR)

This register provides the macrotick count of the current communication cycle.

Table 13-32. MTCTR Field Descriptions

Field	Description
13–0 MTCT	**Macrotick Counter** — protocol related variable: *vMacrotick* This field provides the macrotick count of the current communication cycle.

13.5.2.25 Cycle Counter Register (CYCTR)

Module Base + 0x0032

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	CYCCNT					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-25. Cycle Counter Register (CYCTR)

This register provides the number of the current communication cycle.

Table 13-33. CYCTR Field Descriptions

Field	Description
5–0 CYCCNT	**Cycle Counter** — protocol related variable: *vCycleCounter* This field provides the number of the current communication cycle. If the counter reaches the maximum value of 63, the counter wraps and starts from zero again.

13.5.2.26 Slot Counter Channel A Register (SLTCTAR)

Module Base + 0x0034

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	SLOTCNTA										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-26. Slot Counter Channel A Register (SLTCTAR)

This register provides the number of the current slot in the current communication cycle for channel A.

Table 13-34. SLTCTAR Field Descriptions

Field	Description
10–0 SLOTCNTA	**Slot Counter Value for Channel A** — protocol related variable: *vSlotCounter* for channel A This field provides the number of the current slot in the current communication cycle.

13.5.2.27 Slot Counter Channel B Register (SLTCTBR)

Module Base + 0x0036

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	SLOTCNTB										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-27. Slot Counter Channel B Register (SLTCTBR)

This register provides the number of the current slot in the current communication cycle for channel B.

Table 13-35. SLTCTBR Field Descriptions

Field	Description
10–0 SLOTCNTA	**Slot Counter Value for Channel B** — protocol related variable: *vSlotCounter* for channel B This field provides the number of the current slot in the current communication cycle.

13.5.2.28 Rate Correction Value Register (RTCORVR)

Module Base + 0x0038 Additional Reset: RUN Command

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RATECORR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-28. Rate Correction Value Register (RTCORVR)

This register provides the sign extended rate correction value in microticks as it was calculated by the clock synchronization algorithm. The FlexRay block updates this register during the NIT of each odd numbered communication cycle.

Table 13-36. RTCORVR Field Descriptions

Field	Description
15–0 RATECORR	**Rate Correction Value** — protocol related variable: *vRateCorrection* (before value limitation and external rate correction) This field provides the sign extended rate correction value in microticks as it was calculated by the clock synchronization algorithm. The value is represented in 2's complement format. This value does not include the value limitation and the application of the external rate correction. If the magnitude of the internally calculated rate correction value exceeds the limit given by rate_correction_out in the Protocol Configuration Register 13 (PCR13), the clock correction reached limit interrupt flag CCL_IF is set in the Protocol Interrupt Flag Register 0 (PIFR0). **Note:** If the FlexRay block was not able to calculate a new rate correction term due to a lack of synchronization frames, the RATECORR value is not updated.

13.5.2.29 Offset Correction Value Register (OFCORVR)

Module Base + 0x003A Additional Reset: RUN Command

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	OFFSETCORR															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-29. Offset Correction Value Register (OFCORVR)

This register provides the sign extended offset correction value in microticks as it was calculated by the clock synchronization algorithm. The FlexRay block updates this register during the NIT.

Table 13-37. OFCORVR Field Descriptions

Field	Description
15–0 OFFSET- CORR	**Offset Correction Value** — protocol related variable: *vOffsetCorrection* (before value limitation and external offset correction) This field provides the sign extended offset correction value in microticks as it was calculated by the clock synchronization algorithm. The value is represented in 2's complement format. This value does not include the value limitation and the application of the external offset correction. If the magnitude of the internally calculated rate correction value exceeds the limit given by offset_correction_out field in the Protocol Configuration Register 29 (PCR29), the clock correction reached limit interrupt flag CCL_IF is set in the Protocol Interrupt Flag Register 0 (PIFR0). **Note:** If the FlexRay block was not able to calculate an new offset correction term due to a lack of synchronization frames, the OFFSETCORR value is not updated.

13.5.2.30 Combined Interrupt Flag Register (CIFRR)

Module Base + 0x003C

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	MIF	PRIF	CHIF	WUP IF	FNEB IF	FNEA IF	RBIF	TBIF
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-30. Combined Interrupt Flag Register (CIFRR)

This register provides five combined interrupt flags and a copy of three individual interrupt flags. The combined interrupt flags are the result of a binary OR of the values of other interrupt flags regardless of the state of the interrupt enable bits. The generation scheme for the combined interrupt flags is depicted in Figure 13-146. The individual interrupt flags WUPIF, FNEBIF, and FNEAIF are copies of corresponding flags in the Global Interrupt Flag and Enable Register (GIFER) and are provided here to simplify the application interrupt flag check. To clear the individual interrupt flags, the application must use the Global Interrupt Flag and Enable Register (GIFER).

NOTE

The meanings of the combined status bits MIF, PRIF, CHIF, RBIF, and TBIF are different from those mentioned in the Global Interrupt Flag and Enable Register (GIFER).

Table 13-38. CIFRR Field Descriptions

Field	Description
7 MIF	**Module Interrupt Flag** — This flag is set if there is at least one interrupt source that has its interrupt flag asserted. 0 No interrupt source has its interrupt flag asserted 1 At least one interrupt source has its interrupt flag asserted
6 PRIF	**Protocol Interrupt Flag** — This flag is set if at least one of the individual protocol interrupt flags in the Protocol Interrupt Flag Register 0 (PIFR0) or Protocol Interrupt Flag Register 1 (PIFR1) is equal to 1. 0 All individual protocol interrupt flags are equal to 0 1 At least one of the individual protocol interrupt flags is equal to 1
5 CHIF	**CHI Interrupt Flag** — This flag is set if at least one of the individual CHI error flags in the CHI Error Flag Register (CHIERFR) is equal to 1. 0 All CHI error flags are equal to 0 1 At least one CHI error flag is equal to 1
4 WUPIF	**Wakeup Interrupt Flag** — Provides the same value as GIFER[WUPIF]
3 FNEBIF	**Receive FIFO channel B Not Empty Interrupt Flag** — Provides the same value as GIFER[FNEBI]
2 FNEAIF	**Receive FIFO channel A Not Empty Interrupt Flag** — Provides the same value as GIFER[FNEAIF]
1 RBIF	**Receive Message Buffer Interrupt Flag** — This flag is set if for at least one of the individual receive message buffers (MBCCSRn[MTD] = 0) the interrupt flag MBIF in the corresponding Message Buffer Configuration, Control, Status Registers (MBCCSRn) is equal to 1. 0 None of the individual receive message buffers has the MBIF flag asserted. 1 At least one individual receive message buffers has the MBIF flag asserted.
0 TBIF	**Transmit Message Buffer Interrupt Flag** — This flag is set if for at least one of the individual single or double transmit message buffers (MBCCSRn[MTD] = 1) the interrupt flag MBIF in the corresponding Message Buffer Configuration, Control, Status Registers (MBCCSRn) is equal to 1. 0 None of the individual transmit message buffers has the MBIF flag asserted. 1 At least one individual transmit message buffers has the MBIF flag asserted.

13.5.2.31 System Memory Access Time-Out Register (SYMATOR)

Module Base + 0x003E

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	TIMEOUT				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Figure 13-31. System Memory Access Time-Out Register (SYMATOR)

Write: Disabled Mode

Table 13-39. SYMATOR Field Descriptions

Field	Description
4–0 TIMEOUT	**Time-Out** — This value defines the maximum number of wait states on the system memory bus interface. This value must never exceeded in order to ensure no data are lost even under internal worst case conditions. If the number of wait states is greater than the TIMEOUT value, but is less than twice the TIMEOUT value, and internal worst case conditions occur, than data might be lost. If data are lost, the System Bus Communication Failure Error Flag SBCF_EF is set in the CHI Error Flag Register (CHIERFR). If the number of wait states is greater than twice the TIMEOUT value, data will be lost, and the System Bus Communication Failure Error Flag SBCF_EF is set in the CHI Error Flag Register (CHIERFR).

13.5.2.32 Sync Frame Counter Register (SFCNTR)

Module Base + 0x0040 Additional Reset: RUN Command

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SFEVB				SFEVA				SFODB				SFODA			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-32. Sync Frame Counter Register (SFCNTR)

This register provides the number of synchronization frames that are used for clock synchronization in the last even and in the last odd numbered communication cycle. This register is updated after the start of the NIT and before 10 MT after offset correction start.

NOTE

If the application has locked the even synchronization table at the end of the static segment of an even communication cycle, the FlexRay block will not update the fields SFEVB and SFEVA.

If the application has locked the odd synchronization table at the end of the static segment of an odd communication cycle, the FlexRay block will not update the values SFODB and SFODA.

Table 13-40. SFCNTR Field Descriptions

Field	Description
15–12 SFEVB	Sync Frames Channel B, even cycle — protocol related variable: size of (*vsSyncIdListB* for even cycle) This field provides the size of the internal list of frame IDs of received synchronization frames used for clock synchronization.
11–8 SFEVB	Sync Frames Channel A, even cycle — protocol related variable: size of (*vsSyncIdListA* for even cycle) This field provides the size of the internal list of frame IDs of received synchronization frames used for clock synchronization.
7–4 SFODB	Sync Frames Channel B, odd cycle — protocol related variable: size of (*vsSyncIdListB* for odd cycle) This field provides the size of the internal list of frame IDs of received synchronization frames used for clock synchronization.
3–0 SFODA	Sync Frames Channel A, odd cycle — protocol related variable: size of (*vsSyncIdListA* for odd cycle) This field provides the size of the internal list of frame IDs of received synchronization frames used for clock synchronization.

13.5.2.33 Sync Frame Table Offset Register (SFTOR)

Module Base + 0x0042

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SFT_OFFSET[15:1]															0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-33. Sync Frame Table Offset Register (SFTOR)

Write: *POC:config*

This register defines the Flexray Memory related offset for sync frame tables. For more details, see Section 13.6.12, “Sync Frame ID and Sync Frame Deviation Tables”.

Table 13-41. SFTOR Field Description

Field	Description
15–1 SFTOR	**Sync Frame Table Offset** — The offset of the Sync Frame Tables in the Flexray Memory. This offset is required to be 16-bit aligned. Thus STF_OFFSET[0] is always 0.

13.5.2.34 Sync Frame Table Configuration, Control, Status Register (SFTCCSR)

Module Base + 0x0044

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	CYCNUM						ELKS	OLKS	EVAL	OVAL	0	0	SDV EN	SID EN
W	ELKT	OLKT												OPT		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-34. Sync Frame Table Configuration, Control, Status Register (SFTCCSR)

Write: Normal Mode

This register provides configuration, control, and status information related to the generation and access of the clock sync ID tables and clock sync measurement tables. For a detailed description, see Section 13.6.12, “Sync Frame ID and Sync Frame Deviation Tables”.

Table 13-42. SFTCCSR Field Descriptions

Field	Description
15 ELKT	**Even Cycle Tables Lock/Unlock Trigger** — This trigger bit is used to lock and unlock the even cycle tables. 0 No effect 1 Triggers lock/unlock of the even cycle tables.
14 OLKT	**Odd Cycle Tables Lock/Unlock Trigger** — This trigger bit is used to lock and unlock the odd cycle tables. 0 No effect 1 Triggers lock/unlock of the odd cycle tables.
13–8 CYCNUM	**Cycle Number** — This field provides the number of the cycle in which the currently locked table was recorded. If none or both tables are locked, this value is related to the even cycle table.
7 ELKS	**Even Cycle Tables Lock Status** — This status bit indicates whether the application has locked the even cycle tables. 0 Application has not locked the even cycle tables. 1 Application has locked the even cycle tables.
6 OLKS	**Odd Cycle Tables Lock Status** — This status bit indicates whether the application has locked the odd cycle tables. 0 Application has not locked the odd cycle tables. 1 Application has locked the odd cycle tables.
5 EVAL	**Even Cycle Tables Valid** — This status bit indicates whether the Sync Frame ID and Sync Frame Deviation Tables for the even cycle are valid. The FlexRay block clears this status bit when it starts updating the tables, and sets this bit when it has finished the table update. 0 Tables are not valid (update is ongoing) 1 Tables are valid (consistent).
4 OVAL	**Odd Cycle Tables Valid** — This status bit indicates whether the Sync Frame ID and Sync Frame Deviation Tables for the odd cycle are valid. The FlexRay block clears this status bit when it starts updating the tables, and sets this bit when it has finished the table update. 0 Tables are not valid (update is ongoing) 1 Tables are valid (consistent).
2 OPT	**One Pair Trigger** — This trigger bit controls whether the FlexRay block writes continuously or only one pair of Sync Frame Tables into the FRM. If this trigger is set to 1 while SDVEN or SIDEN is set to 1, the FlexRay block writes only one pair of the enabled Sync Frame Tables corresponding to the next even-odd-cycle pair into the FRM. In this case, the FlexRay block clears the SDVEN or SIDEN bits immediately. If this trigger is set to 0 while SDVEN or SIDEN is set to 1, the FlexRay block writes continuously the enabled Sync Frame Tables into the FRM. 0 Write continuously pairs of enabled Sync Frame Tables into FRM. 1 Write only one pair of enabled Sync Frame Tables into FRM.
1 SDVEN	**Sync Frame Deviation Table Enable** — This bit controls the generation of the Sync Frame Deviation Tables. The application must set this bit to request the FlexRay block to write the Sync Frame Deviation Tables into the FRM. 0 Do not write Sync Frame Deviation Tables 1 Write Sync Frame Deviation Tables into FRM **Note:** If SDVEN is set to 1, then SIDEN must also be set to 1.
0 SIDEN	**Sync Frame ID Table Enable** — This bit controls the generation of the Sync Frame ID Tables. The application must set this bit to 1 to request the FlexRay block to write the Sync Frame ID Tables into the FRM. 0 Do not write Sync Frame ID Tables 1 Write Sync Frame ID Tables into FRM

13.5.2.35 Sync Frame ID Rejection Filter Register (SFIDRFR)

Module Base + 0x0046 — 16-bit write access required

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	SYNFRID									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-35. Sync Frame ID Rejection Filter Register (SFIDRFR)

Write: Normal Mode

This register defines the Sync Frame Rejection Filter ID. The application must update this register outside of the static segment. If the application updates this register in the static segment, it can appear that the FlexRay block accepts the sync frame in the current cycle.

Table 13-43. SFIDRFR Field Descriptions

Field	Description
9–0 SYNFRID	**Sync Frame Rejection ID** — This field defines the frame ID of a frame that must not be used for clock synchronization. For details see Section 13.6.15.2, “Sync Frame Rejection Filtering”.

13.5.2.36 Sync Frame ID Acceptance Filter Value Register (SFIDAFVR)

Module Base + 0x0048

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	FVAL									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-36. Sync Frame ID Acceptance Filter Value Register (SFIDAFVR)

Write: *POC:config*

This register defines the sync frame acceptance filter value. For details on filtering, see Section 13.6.15, “Sync Frame Filtering”.

Table 13-44. SFIDAFVR Field Descriptions

Field	Description
9–0 FVAL	**Filter Value** — This field defines the value for the sync frame acceptance filtering.

13.5.2.37 Sync Frame ID Acceptance Filter Mask Register (SFIDAFMR)

Module Base + 0x004A

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	FMSK									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-37. Sync Frame ID Acceptance Filter Mask Register (SFIDAFMR)

Write: *POC:config*

This register defines the sync frame acceptance filter mask. For details on filtering see Section 13.6.15.1, “Sync Frame Acceptance Filtering”.

Table 13-45. SFIDAFMR Field Descriptions

Field	Description
9–0 FMSK	**Filter Mask** — This field defines the mask for the sync frame acceptance filtering.

13.5.2.38 Network Management Vector Registers (NMVR0–NMVR5)

Module Base + 0x004C (NMVR0)
Module Base + 0x004E (NMVR1)
Module Base + 0x0050 (NMVR2)
Module Base + 0x0052 (NMVR3)
Module Base + 0x0054 (NMVR4)
Module Base + 0x0056 (NMVR5)

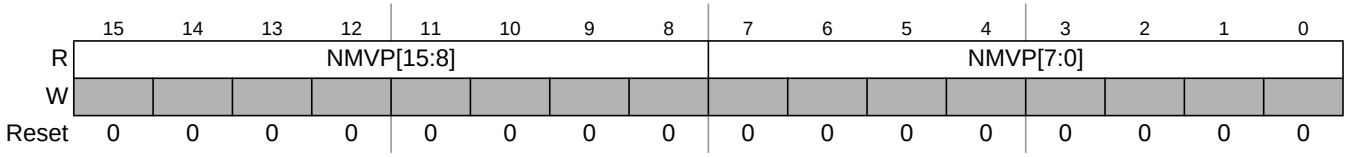


Figure 13-38. Network Management Vector Registers (NMVR0–NMVR5)

Each of these six registers holds one part of the Network Management Vector. The length of the Network Management Vector is configured in the Network Management Vector Length Register (NMVLR). If NMVLR is programmed with a value that is less than 12 bytes, the remaining bytes of the Network Management Vector Registers (NMVR0–NMVR5), which are not used for the Network Management Vector accumulating, will remain 0.

The NMVR provides accrued information over all received NMVs in the last communication cycle. All NMVs received in one cycle are ORed into the NMVR. The NMVR is updated at the end of the communication cycle.

Table 13-46. NMVR[0:5] Field Descriptions

Field	Description
15–0 NMVP	**Network Management Vector Part** — The mapping between the Network Management Vector Registers (NMVR0–NMVR5) and the receive message buffer payload bytes in NMV[0:11] is depicted in **Table 13-46**.

Table 13-47. Mapping of NMVRn to the Received Payload Bytes NMVn

NMVRn Register	NMVn Received Payload
NMVR0[NMVP[15:8]]	NMV0
NMVR0[NMVP[7:0]]	NMV1
NMVR1[NMVP[15:8]]	NMV2
NMVR1[NMVP[7:0]]	NMV3
...	

Table 13-47. Mapping of NMVRn to the Received Payload Bytes NMVn

NMVRn Register	NMVn Received Payload
NMVR5[NMVP[15:8]]	NMV10
NMVR5[NMVP[7:0]]	NMV11

13.5.2.39 Network Management Vector Length Register (NMVLR)

Module Base + 0x0058

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	NMVL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-39. Network Management Vector Length Register (NMVLR)

Write: *POC:config*

This register defines the length of the network management vector in bytes.

Table 13-48. NMVLR Field Descriptions

Field	Description
3–0 NMVL	**Network Management Vector Length** — protocol related variable: *gNetworkManagementVectorLength* This field defines the length of the Network Management Vector in bytes. Legal values are between 0 and 12.

13.5.2.40 Timer Configuration and Control Register (TICCR)

Module Base + 0x005A

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	T2_CFG	T2_REP	0	0	0	T2ST	0	0	0	T1_REP	0	0	0	T1ST
W						T2SP	T2TR							T1SP	T1TR	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-40. Timer Configuration and Control Register (TICCR)

Write: T2_CFG: *POC:config*
T2_REP, T1_REP, T1SP, T2SP, T1TR, T2TR: Normal Mode

This register is used to configure and control the two timers T1 and T2. For timer details, see Section 13.6.17, “Timer Support”. The Timer T1 is an absolute timer. The Timer T2 can be configured as an absolute or relative timer.

Table 13-49. TICCR Field Descriptions (Sheet 1 of 2)

Field	Description
13 T2_CFG	**Timer T2 Configuration** — This bit configures the timebase mode of Timer T2. 0 T2 is absolute timer. 1 T2 is relative timer.
12 T2_REP	**Timer T2 Repetitive Mode** — This bit configures the repetition mode of Timer T2. 0 T2 is non repetitive 1 T2 is repetitive

Table 13-49. TICCR Field Descriptions (Sheet 2 of 2)

Field	Description
10 T2SP	**Timer T2 Stop** — This trigger bit is used to stop timer T2. 0 no effect 1 stop timer T2
9 T2TR	**Timer T2 Trigger** — This trigger bit is used to start timer T2. 0 no effect 1 start timer T2
8 T2ST	**Timer T2 State** — This status bit provides the current state of timer T2. 0 timer T2 is idle 1 timer T2 is running
4 T1_REP	**Timer T1 Repetitive Mode** — This bit configures the repetition mode of timer T1. 0 T1 is non repetitive 1 T1 is repetitive
2 T1SP	**Timer T1 Stop** — This trigger bit is used to stop timer T1. 0 no effect 1 stop timer T1
1 T1TR	**Timer T1 Trigger** — This trigger bit is used to start timer T1. 0 no effect 1 start timer T1
0 T1ST	**Timer T1 State** — This status bit provides the current state of timer T1. 0 timer T1 is idle 1 timer T1 is running

NOTE

Both timers are deactivated immediately when the protocol enters a state different from *POC:normal active* or *POC:normal passive*.

13.5.2.41 Timer 1 Cycle Set Register (TI1CYSR)

Module Base + 0x005C

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	T1_CYC_VAL						0	0	T1_CYC_MSK					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-41. Timer 1 Cycle Set Register (TI1CYSR)

Write: Anytime

This register defines the cycle filter value and the cycle filter mask for timer T1. For a detailed description of timer T1, refer to Section 13.6.17.1, “Absolute Timer T1”.

Table 13-50. TI1CYSR Field Descriptions

Field	Description
13–8 T1_CYC_VAL	**Timer T1 Cycle Filter Value** — This field defines the cycle filter value for timer T1.
5–0 T1_CYC_MSK	**Timer T1 Cycle Filter Mask** — This field defines the cycle filter mask for timer T1.

NOTE

If the application modifies the value in this register while the timer is running, the change becomes effective immediately and timer T1 will expire according to the changed value.

13.5.2.42 Timer 1 Macrotick Offset Register (TI1MTOR)

Module Base + 0x005E

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	T1_MTOFFSET													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-42. Timer 1 Macrotick Offset Register (TI1MTOR)

Write: Anytime

This register holds the macrotick offset value for timer T1. For a detailed description of timer T1, refer to Section 13.6.17.1, “Absolute Timer T1”.

Table 13-51. TI1MTOR Field Descriptions

Field	Description
13–0 T1_MTOFFSET	**Timer 1 Macrotick Offset** — This field defines the macrotick offset value for timer 1.

NOTE

If the application modifies the value in this register while the timer is running, the change becomes effective immediately and timer T1 will expire according to the changed value.

13.5.2.43 Timer 2 Configuration Register 0 (TI2CR0)

Module Base + 0x0060

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	T2_CYC_VAL						0	0	T2_CYC_MSK					
W																
R	T2_MTCNT[31:16]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-43. Timer 2 Configuration Register 0 (TI2CR0)

Write: Anytime

The content of this register depends on the value of the T2_CFG bit in the Timer Configuration and Control Register (TICCR). For a detailed description of timer T2, refer to Section 13.6.17.2, “Absolute / Relative Timer T2”.

Table 13-52. TI2CR0 Field Descriptions

Field	Description
Fields for absolute timer T2 (TICCR[T2_CFG] = 0)	
13–8 T2_CYC_VAL	**Timer T2 Cycle Filter Value** — This field defines the cycle filter value for timer T2.
5–0 T2_CYC_MSK	**Timer T2 Cycle Filter Mask** — This field defines the cycle filter mask for timer T2.
Fields for relative timer T2 (TICCR[T2_CFG = 1)	
15–0 T2_MTCNT[31:16]	**Timer T2 Macrotick High Word** — This field defines the high word of the macrotick count for timer T2.

NOTE

If timer T2 is configured as an *absolute* timer and the application modifies the values in this register while the timer is running, the change becomes effective immediately and timer T2 will expire according to the changed values.

If timer T2 is configured as a *relative* timer and the application changes the values in this register while the timer is running, the change becomes effective when the timer has expired according to the old values.

13.5.2.44 Timer 2 Configuration Register 1 (TI2CR1)

Module Base + 0x0062

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	T2_MTOFFSET													
W																
R	T2_MTCNT[15:0]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-44. Timer 2 Configuration Register 1 (TI2CR1)

Write: Anytime

The content of this register depends on the value of the T2_CFG bit in the Timer Configuration and Control Register (TICCR). For a detailed description of timer T2, refer to Section 13.6.17.2, “Absolute / Relative Timer T2”.

Table 13-53. TI2CR1 Field Descriptions

Field	Description
Fields for absolute timer T2 (TICCR[T2_CFG] = 0)	
13–0 T2_MTOFFSET	**Timer T2 Macrotick Offset** — This field holds the macrotick offset value for timer T2.
Fields for relative timer T2 (TICCR[T2_CFG] = 1)	
15–0 T2_MTCNT[15:0]	**Timer T2 Macrotick Low Word** — This field defines the low word of the macrotick value for timer T2.

NOTE

If timer T2 is configured as an *absolute* timer and the application modifies the values in this register while the timer is running, the change becomes effective immediately and the timer T2 will expire according to the changed values.

If timer T2 is configured as a *relative* timer and the application changes the values in this register while the timer is running, the change becomes effective when the timer has expired according to the old values.

13.5.2.45 Slot Status Selection Register (SSSR)

Module Base + 0x0064 16-bit write access required

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	SEL		0	SLOTNUMBER										
W	WMD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-45. Slot Status Selection Register (SSSR)

Write: Anytime

This register is used to access the four internal non memory-mapped slot status selection registers SSSR0 to SSSR3. Each internal registers selects a slot, or symbol window/NIT, whose status vector will be saved in the corresponding Slot Status Registers (SSR0–SSR7) according to Table 13-54. For a detailed description of slot status monitoring, refer to Section 13.6.18, “Slot Status Monitoring”.

Table 13-54. SSSR Field Descriptions

Field	Description
15 WMD	**Write Mode** — This control bit defines the write mode of this register. 0 Write to all fields in this register on write access. 1 Write to SEL field only on write access.
13–12 SEL	**Selector** — This field selects one of the four internal slot status selection registers for access. 00 select SSSR0. 01 select SSSR1. 10 select SSSR2. 11 select SSSR3.
10–0 SLOTNUMBER	**Slot Number** — This field specifies the number of the slot whose status will be saved in the corresponding slot status registers. **Note:** If this value is set to 0, the related slot status register provides the status of the symbol window after the NIT start, and provides the status of the NIT after the cycle start.

Table 13-55. Mapping Between SSSRn and SSRn

Internal Slot Status Selection Register	Write the Slot Status of the Slot Selected by SSSRn for each			
	Even Communication Cycle		Odd Communication Cycle	
	For Channel B to	For Channel A to	For Channel B to	For Channel A to
SSSR0	SSR0[15:8]	SSR0[7:0]	SSR1[15:8]	SSR1[7:0]
SSSR1	SSR2[15:8]	SSR2[7:0]	SSR3[15:8]	SSR3[7:0]
SSSR2	SSR4[15:8]	SSR4[7:0]	SSR5[15:8]	SSR5[7:0]
SSSR3	SSR6[15:8]	SSR6[7:0]	SSR7[15:8]	SSR7[7:0]

13.5.2.46 Slot Status Counter Condition Register (SSCCR)

Module Base + 0x0066 16-bit write access required

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	SEL		0	CNTCFG		MCY	VFR	SYF	NUF	SUF	STATUSMASK			
W	WMD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-46. Slot Status Counter Condition Register (SSCCR)

Write: Anytime

This register is used to access and program the four internal non-memory mapped Slot Status Counter Condition Registers SSCCR0 to SSCCR3. Each of these four internal slot status counter condition registers defines the mode and the conditions for incrementing the counter in the corresponding Slot Status Counter Registers (SSCR0–SSCR3). The correspondence is given in Table 13-56. For a detailed description of slot status counters, refer to Section 13.6.18.4, “Slot Status Counter Registers”.

Table 13-56. SSCCR Field Descriptions

Field	Description
15 WMD	**Write Mode** — This control bit defines the write mode of this register. 0 Write to all fields in this register on write access. 1 Write to SEL field only on write access.
13–12 SEL	**Selector** — This field selects one of the four internal slot counter condition registers for access. 00 select SSCCR0. 01 select SSCCR1. 10 select SSCCR2. 11 select SSCCR3.
10–9 CNTCFG	**Counter Configuration** — These bit field controls the channel related incrementing of the slot status counter. 00 increment by 1 if condition is fulfilled on channel A. 01 increment by 1 if condition is fulfilled on channel B. 10 increment by 1 if condition is fulfilled on at least one channel. 11 increment by 2 if condition is fulfilled on both channels channel. increment by 1 if condition is fulfilled on only one channel.
8 MCY	**Multi Cycle Selection** — This bit defines whether the slot status counter accumulates over multiple communication cycles or provides information for the previous communication cycle only. 0 The Slot Status Counter provides information for the previous communication cycle only. 1 The Slot Status Counter accumulates over multiple communication cycles.
7 VFR	**Valid Frame Restriction** — This bit is used to restrict the counter to received valid frames. 0 The counter is not restricted to valid frames only. 1 The counter is restricted to valid frames only.
6 SYF	**Sync Frame Restriction** — This bit is used to restrict the counter to received frames with the sync frame indicator bit set to 1. 0 The counter is not restricted with respect to the sync frame indicator bit. 1 The counter is restricted to frames with the sync frame indicator bit set to 1.
5 NUF	**Null Frame Restriction** — This bit is used to restrict the counter to received frames with the null frame indicator bit set to 0. 0 The counter is not restricted with respect to the null frame indicator bit. 1 The counter is restricted to frames with the null frame indicator bit set to 0.
4 SUF	**Startup Frame Restriction** — This bit is used to restrict the counter to received frames with the startup frame indicator bit set to 1. 0 The counter is not restricted with respect to the startup frame indicator bit. 1 The counter is restricted to received frames with the startup frame indicator bit set to 1.
3–0 STATUSMASK	**Slot Status Mask** — This bit field is used to enable the counter with respect to the four slot status error indicator bits. **STATUSMASK[3]** – This bit enables the counting for slots with the syntax error indicator bit set to 1. **STATUSMASK[2]** – This bit enables the counting for slots with the content error indicator bit set to 1. **STATUSMASK[1]** – This bit enables the counting for slots with the boundary violation indicator bit set to 1. **STATUSMASK[0]** – This bit enables the counting for slots with the transmission conflict indicator bit set to 1.

Table 13-57. Mapping between internal SSCCRn and SSCRn

Condition Register	Condition Defined for Register
SSCCR0	SSCR0
SSCCR1	SSCR1
SSCCR2	SSCR2
SSCCR3	SSCR3

13.5.2.47 Slot Status Registers (SSR0–SSR7)

Module Base + 0x0068 (SSR0)
Module Base + 0x006A (SSR1)
Module Base + 0x006C (SSR2)
Module Base + 0x006E (SSR3)
Module Base + 0x0070 (SSR4)
Module Base + 0x0072 (SSR5)
Module Base + 0x0074 (SSR6)
Module Base + 0x0076 (SSR7)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VFB	SYB	NFB	SUB	SEB	CEB	BVB	TCB	VFA	SYA	NFA	SUA	SEA	CEA	BVA	TCA
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-47. Slot Status Registers (SSR0–SSR7)

Each of these eight registers holds the status vector of the slot specified in the corresponding internal slot status selection register, which can be programmed using the Slot Status Selection Register (SSSR). Each register is updated after the end of the corresponding slot as shown in Figure 13-142. The register bits are directly related to the protocol variables and described in more detail in Section 13.6.18, “Slot Status Monitoring”.

Table 13-58. SSR0–SSR7 Field Descriptions

Field	Description
15 VFB	**Valid Frame on Channel B** — protocol related variable: *vSS!ValidFrame* channel B 0 *vSS!ValidFrame* = 0 1 *vSS!ValidFrame* = 1
14 SYB	**Sync Frame Indicator Channel B** — protocol related variable: *vRF!Header!SyFIndicator* channel B 0 *vRF!Header!SyFIndicator* = 0 1 *vRF!Header!SyFIndicator* = 1
13 NFB	**Null Frame Indicator Channel B** — protocol related variable: *vRF!Header!NFIndicator* channel B 0 *vRF!Header!NFIndicator* = 0 1 *vRF!Header!NFIndicator* = 1
12 SUB	**Startup Frame Indicator Channel B** — protocol related variable: *vRF!Header!SuFIndicator* channel B 0 *vRF!Header!SuFIndicator* = 0 1 *vRF!Header!SuFIndicator* = 1
11 SEB	**Syntax Error on Channel B** — protocol related variable: *vSS!SyntaxError* channel B 0 *vSS!SyntaxError* = 0 1 *vSS!SyntaxError* = 1
10 CEB	**Content Error on Channel B** — protocol related variable: *vSS!ContentError* channel B 0 *vSS!ContentError* = 0 1 *vSS!ContentError* = 1
9 BVB	**Boundary Violation on Channel B** — protocol related variable: *vSS!BViolation* channel B 0 *vSS!BViolation* = 0 1 *vSS!BViolation* = 1
8 TCB	**Transmission Conflict on Channel B** — protocol related variable: *vSS!TxConflict* channel B 0 *vSS!TxConflict* = 0 1 *vSS!TxConflict* = 1

Table 13-58. SSR0–SSR7 Field Descriptions (continued)

Field	Description
7 VFA	**Valid Frame on Channel A** — protocol related variable: *vSS!ValidFrame* channel A 0 *vSS!ValidFrame* = 0 1 *vSS!ValidFrame* = 1
6 SYA	**Sync Frame Indicator Channel A** — protocol related variable: *vRF!Header!SyFIndicator* channel A 0 *vRF!Header!SyFIndicator* = 0 1 *vRF!Header!SyFIndicator* = 1
5 NFA	**Null Frame Indicator Channel A** — protocol related variable: *vRF!Header!NFIndicator* channel A 0 *vRF!Header!NFIndicator* = 0 1 *vRF!Header!NFIndicator* = 1
4 SUA	**Startup Frame Indicator Channel A** — protocol related variable: *vRF!Header!SuFIndicator* channel A 0 *vRF!Header!SuFIndicator* = 0 1 *vRF!Header!SuFIndicator* = 1
3 SEA	**Syntax Error on Channel A** — protocol related variable: *vSS!SyntaxError* channel A 0 *vSS!SyntaxError* = 0 1 *vSS!SyntaxError* = 1
2 CEA	**Content Error on Channel A** — protocol related variable: *vSS!ContentError* channel A 0 *vSS!ContentError* = 0 1 *vSS!ContentError* = 1
1 BVA	**Boundary Violation on Channel A** — protocol related variable: *vSS!BViolation* channel A 0 *vSS!BViolation* = 0 1 *vSS!BViolation* = 1
0 TCA	**Transmission Conflict on Channel A** — protocol related variable: *vSS!TxConflict* channel A 0 *vSS!TxConflict* = 0 1 *vSS!TxConflict* = 1

13.5.2.48 Slot Status Counter Registers (SSCR0–SSCR3)

Module Base + 0x0078 (SSCR0)
Module Base + 0x007A (SSCR1)
Module Base + 0x007C (SSCR2)
Module Base + 0x007E (SSCR3)

Additional Reset: RUN Command

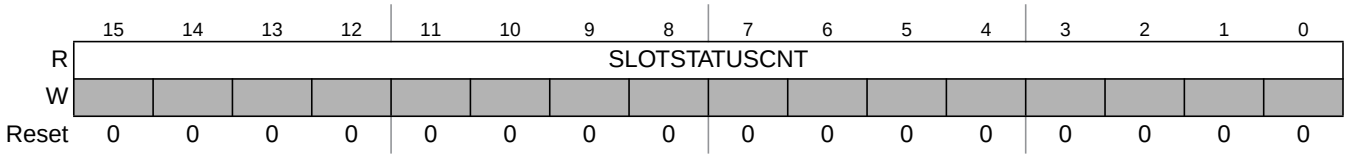


Figure 13-48. Slow Status Counter Registers (SSCR0–SSCR3)

Each of these four registers provides the slot status counter value for the previous communication cycle(s) and is updated at the cycle start. The provided value depends on the control bits and fields in the related internal slot status counter condition register SSCCRn, which can be programmed by using the Slot Status Counter Condition Register (SSCCR). For more details, see Section 13.6.18.4, “Slot Status Counter Registers”.

NOTE

If the counter has reached its maximum value 0xFFFF and is in the multicycle mode, i.e. SSCCRn[MCY] = 1, the counter is not reset to 0x0000. The application can reset the counter by clearing the SSCCRn[MCY] bit and waiting for the next cycle start, when the FlexRay block clears the counter. Subsequently, the counter can be set into the multicycle mode again.

Table 13-59. SSCR0–SSCR3 Field Descriptions

Field	Description
15–0 SLOTSTATUSCNT	**Slot Status Counter** — This field provides the current value of the Slot Status Counter.

13.5.2.49 MTS A Configuration Register (MTSACFR)

Module Base + 0x0080

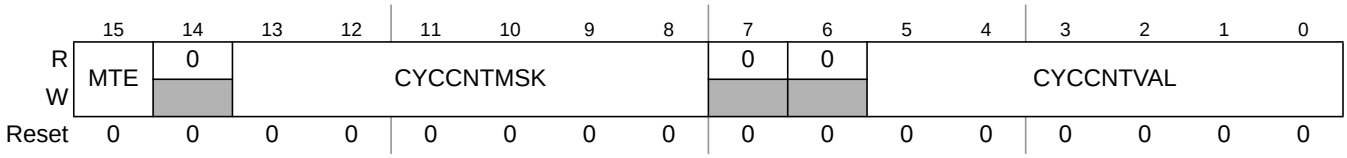


Figure 13-49. MTS A Configuration Register (MTSACFR)

Write: MTE: Anytime; CYCCNTMSK,CYCCNTVAL: *POC:config*

This register controls the transmission of the Media Access Test Symbol MTS on channel A. For more details, see Section 13.6.13, “MTS Generation”.

Table 13-60. MTSACFR Field Descriptions

Field	Description
15 MTE	**Media Access Test Symbol Transmission Enable** — This control bit is used to enable and disable the transmission of the Media Access Test Symbol in the selected set of cycles. 0 MTS transmission disabled 1 MTS transmission enabled
13–8 CYCCNTMSK	**Cycle Counter Mask** — This field provides the filter mask for the MTS cycle count filter.
5–0 CYCCNTVAL	**Cycle Counter Value** — This field provides the filter value for the MTS cycle count filter.

13.5.2.50 MTS B Configuration Register (MTSBCFR)

Module Base + 0x0082

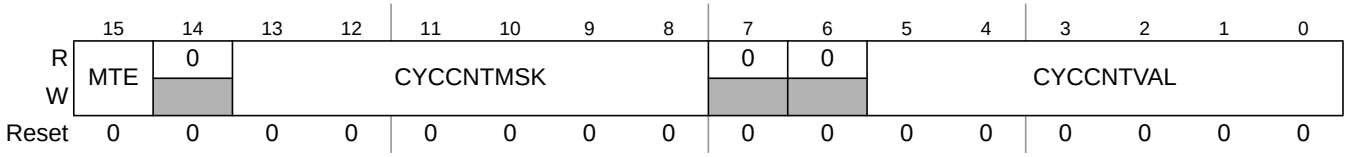


Figure 13-50. MTS B Configuration Register (MTSBCFR)

Write: MTE: Anytime; CYCCNTMSK,CYCCNTVAL: *POC:config*

This register controls the transmission of the Media Access Test Symbol MTS on channel B. For more details, see Section 13.6.13, “MTS Generation”.

Table 13-61. MTSBCFR Field Descriptions

Field	Description
15 MTE	**Media Access Test Symbol Transmission Enable** — This control bit is used to enable and disable the transmission of the Media Access Test Symbol in the selected set of cycles. 0 MTS transmission disabled 1 MTS transmission enabled
13–8 CYCCNTMSK	**Cycle Counter Mask** — This field provides the filter mask for the MTS cycle count filter.
5–0 CYCCNTVAL	**Cycle Counter Value** — This field provides the filter value for the MTS cycle count filter.

13.5.2.51 Receive Shadow Buffer Index Register (RSBIR)

Module Base + 0x0084 16-bit write access required

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	SEL		0	0	0	0	0	0	RSBIDX					
W	WMD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-51. Receive Shadow Buffer Index Register (RSBIR)

Write: WMD, SEL: Any Time; RSBIDX: *POC:config*

This register is used to provide and retrieve the indices of the message buffer header fields currently associated with the receive shadow buffers. For more details on the receive shadow buffer concept, refer to Section 13.6.6.3.5, “Receive Shadow Buffers Concept”.

Table 13-62. RSBIR Field Descriptions

Field	Description
15 WMD	**Write Mode** — This bit controls the write mode for this register. 0 update SEL and RSBIDX field on register write 1 update only SEL field on register write
13–12 SEL	**Selector** — This field is used to select the internal receive shadow buffer index register for access. 00 RSBIR_A1 — receive shadow buffer index register for channel A, segment 1 01 RSBIR_A2 — receive shadow buffer index register for channel A, segment 2 10 RSBIR_B1 — receive shadow buffer index register for channel B, segment 1 11 RSBIR_B2 — receive shadow buffer index register for channel B, segment 2
5–0 RSBIDX	**Receive Shadow Buffer Index** — This field contains the current index of the message buffer header field of the receive shadow message buffer selected by the SEL field. The FlexRay block uses this index to determine the physical location of the shadow buffer header field in the FlexRay memory. The FlexRay block will update this field during receive operation.The application provides initial message buffer header index value in the configuration phase. FlexRay block: Updates the message buffer header index after successful reception. Application: Provides initial message buffer header index.

13.5.2.52 Receive FIFO Selection Register (RFSR)

Module Base + 0x0086

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	SEL
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-52. Receive FIFO Selection Register (RFSR)

Write: Anytime

This register is used to select a receiver FIFO for subsequent access through the receiver FIFO configuration registers summarized in Table 13-62.

Table 13-63. SEL Controlled Receiver FIFO Registers

Register
Receive FIFO Start Index Register (RFSIR)
Receive FIFO Depth and Size Register (RFDSR)
Receive FIFO Message ID Acceptance Filter Value Register (RFMIDAFVR)
Receive FIFO Message ID Acceptance Filter Mask Register (RFMIAFMR)
Receive FIFO Frame ID Rejection Filter Value Register (RFFIDRFVR)
Receive FIFO Frame ID Rejection Filter Mask Register (RFFIDRFMR)
Receive FIFO Range Filter Configuration Register (RFRFCFR)
Receive FIFO Range Filter Control Register (RFRFCTR)

Table 13-64. RFSR Field Descriptions

Field	Description
0 SEL	**Select** — This control bit selects the receiver FIFO for subsequent programming. 0 Receiver FIFO for channel A selected 1 Receiver FIFO for channel B selected

13.5.2.53 Receive FIFO Start Index Register (RFSIR)

Module Base + 0x0088

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	SIDX									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-53. Receive FIFO Start Index Register (RFSIR)

Write: *POC:config*

This register defines the message buffer header index of the first message buffer of the selected FIFO.

Table 13-65. RFSIR Field Descriptions

Field	Description
9–0 SIDX	**Start Index** — This field defines the number of the message buffer header field of the first message buffer of the selected receive FIFO. The FlexRay block uses the value of the SIDX field to determine the physical location of the receiver FIFO's first message buffer header field.

13.5.2.54 Receive FIFO Depth and Size Register (RFDSR)

Module Base + 0x008A

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FIFO_DEPTH								0	ENTRY_SIZE						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-54. Receive FIFO Depth and Size Register (RFDSR)

Write: *POC:config*

This register defines the structure of the selected FIFO, i.e. the number of entries and the size of each entry.

Table 13-66. RFDSR Field Descriptions

Field	Description
15–8 FIFO_DEPTH	**FIFO Depth** — This field defines the depth of the selected receive FIFO, i.e. the number of entries.
6–0 ENTRY_SIZE	**Entry Size** — This field defines the size of the frame data sections for the selected receive FIFO in 2 byte entities.

13.5.2.55 Receive FIFO A Read Index Register (RFARIR)

Module Base + 0x008C

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	RDIDX									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-55. Receive FIFO A Read Index Register (RFARIR)

This register provides the message buffer header index of the next available receive FIFO A entry that the application can read.

Table 13-67. RFARIR Field Descriptions

Field	Description
9–0 RDIDX	**Read Index** — This field provides the message buffer header index of the next available receive FIFO message buffer that the application can read. The FlexRay block increments this index when the application writes to the FNEAIF flag in the Global Interrupt Flag and Enable Register (GIFER). The index wraps back to the first message buffer header index if the end of the FIFO was reached.

NOTE

If the receive FIFO not empty flag FNEAIF is not set, the RDIDX field points to an physical message buffer which content is not valid. Only when FNEAIF is set, the message buffer indicated by RDIDX contains valid data.

13.5.2.56 Receive FIFO B Read Index Register (RFBRIR)

Module Base + 0x008E

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	RDIDX									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-56. Receive FIFO B Read Index Register (RFBRIR)

This register provides the message buffer header index of the next available receive FIFO B entry that the application can read.

Table 13-68. RFBRIR Field Descriptions

Field	Description
9–0 RDIDX	**Read Index** — This field provides the message buffer header index of the next available receive FIFO entry that the application can read. The FlexRay block increments this index when the application writes to the FNEBIF flag in the Global Interrupt Flag and Enable Register (GIFER).The index wraps back to the first message buffer header index if the end of the FIFO was reached.

NOTE

If the receive FIFO not empty flag FNEBIF is not set, the RDIDX field points to an physical message buffer which content is not valid. Only when FNEBIF is set, the message buffer indicated by RDIDX contains valid data.

13.5.2.57 Receive FIFO Message ID Acceptance Filter Value Register (RFMIDAFVR)

Module Base + 0x0090

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MIDAFVAL															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-57. Receive FIFO Message ID Acceptance Filter Value Register (RFMIDAFVR)

Write: *POC:config*

This register defines the filter value for the message ID acceptance filter of the selected receive FIFO. For details on message ID filtering see Section 13.6.9.5, “Receive FIFO filtering”.

Table 13-69. RFMIDAFVR Field Descriptions

Field	Description
15–0 MIDAFVAL	**Message ID Acceptance Filter Value** — Filter value for the message ID acceptance filter.

13.5.2.58 Receive FIFO Message ID Acceptance Filter Mask Register (RFMIAFMR)

Module Base + 0x0092

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MIDAFMSK															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-58. Receive FIFO Message ID Acceptance Filter Mask Register (RFMIAFMR)

Write: *POC:config*

This register defines the filter mask for the message ID acceptance filter of the selected receive FIFO. For details on message ID filtering see Section 13.6.9.5, “Receive FIFO filtering”.

Table 13-70. RFMIAFMR Field Descriptions

Field	Description
15–0 MIDAFMSK	**Message ID Acceptance Filter Mask** — Filter mask for the message ID acceptance filter.

13.5.2.59 Receive FIFO Frame ID Rejection Filter Value Register (RFFIDRFVR)

Module Base + 0x0094

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	FIDRFVAL										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-59. Receive FIFO Frame ID Rejection Filter Value Register (RFFIDRFVR)

Write: *POC:config*

This register defines the filter value for the frame ID rejection filter of the selected receive FIFO. For details on frame ID filtering see Section 13.6.9.5, “Receive FIFO filtering”.

Table 13-71. RFFIDRFVR Field Descriptions

Field	Description
10–0 FIDRFVAL	**Frame ID Rejection Filter Value** — Filter value for the frame ID rejection filter.

13.5.2.60 Receive FIFO Frame ID Rejection Filter Mask Register (RFFIDRFMR)

Module Base + 0x0096

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	FIDRFMSK										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-60. Receive FIFO Frame ID Rejection Filter Mask Register (RFFIDRFMR)

Write: *POC:config*

This register defines the filter mask for the frame ID rejection filter of the selected receive FIFO. For details on frame ID filtering see Section 13.6.9.5, “Receive FIFO filtering”.

Table 13-72. RFFIDRFMR Field Descriptions

Field	Description
10–0 FIDRFMSK	**Frame ID Rejection Filter Mask** — Filter mask for the frame ID rejection filter.

13.5.2.61 Receive FIFO Range Filter Configuration Register (RFRFCFR)

Module Base + 0x0098 16-bit write access required

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	IBD	SEL		0	SID										
W	WMD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-61. Receive FIFO Range Filter Configuration Register (RFRFCFR)

Write: WMD, IBD, SEL: Any Time; SID: *POC:config*

This register provides access to the four internal frame ID range filter boundary registers of the selected receive FIFO. For details on frame ID range filter see Section 13.6.9.5, “Receive FIFO filtering”.

Table 13-73. RFRFCFR Field Descriptions

Field	Description
15 WMD	**Write Mode** — This control bit defines the write mode of this register. 0 Write to all fields in this register on write access. 1 Write to SEL and IBD field only on write access.
14 IBD	**Interval Boundary** — This control bit selects the interval boundary to be programmed with the SID value. 0 program lower interval boundary 1 program upper interval boundary

Table 13-73. RFRFCFR Field Descriptions (continued)

Field	Description
13–12 SEL	**Filter Selector** — This control field selects the frame ID range filter to be accessed. 00 select frame ID range filter 0. 01 select frame ID range filter 1. 10 select frame ID range filter 2. 11 select frame ID range filter 3.
10–0 SID	**Slot ID —** Defines the IBD-selected frame ID boundary value for the SEL-selected range filter.

13.5.2.62 Receive FIFO Range Filter Control Register (RFRFCTR)

Module Base + 0x009A

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	F3MD	F2MD	F1MD	F0MD	0	0	0	0	F3EN	F2EN	F1EN	F0EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-62. Receive FIFO Range Filter Control Register (RFRFCTR)

Write: Anytime

This register is used to enable and disable each frame ID range filter and to define whether it is running as acceptance or rejection filter.

Table 13-74. RFRFCTR Field Descriptions (Sheet 1 of 2)

Field	Description
11 F3MD	**Range Filter 3 Mode** — This control bit defines the filter mode of the frame ID range filter 3. 0 range filter 3 runs as acceptance filter 1 range filter 3 runs as rejection filter
10 F2MD	**Range Filter 2 Mode** — This control bit defines the filter mode of the frame ID range filter 2. 0 range filter 2 runs as acceptance filter 1 range filter 2 runs as rejection filter
9 F1MD	**Range Filter 1 Mode** — This control bit defines the filter mode of the frame ID range filter 1. 0 range filter 1 runs as acceptance filter 1 range filter 1 runs as rejection filter
8 F0MD	**Range Filter 0 Mode** — This control bit defines the filter mode of the frame ID range filter 0. 0 range filter 0 runs as acceptance filter 1 range filter 0 runs as rejection filter
3 F3EN	**Range Filter 3 Enable** — This control bit is used to enable and disable the frame ID range filter 3. 0 range filter 3 disabled 1 range filter 3 enabled
2 F2EN	**Range Filter 2 Enable** — This control bit is used to enable and disable the frame ID range filter 2. 0 range filter 2 disabled 1 range filter 2 enabled

Table 13-74. RFRFCTR Field Descriptions (Sheet 2 of 2)

Field	Description
1 F1EN	**Range Filter 1 Enable** — This control bit is used to enable and disable the frame ID range filter 1. 0 range filter 1 disabled 1 range filter 1 enabled
0 F0EN	**Range Filter 0 Enable** — This control bit is used to enable and disable the frame ID range filter 0. 0 range filter 0 disabled 1 range filter 0 enabled

13.5.2.63 Last Dynamic Transmit Slot Channel A Register (LDTXSLAR)

Module Base + 0x009C

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	LASTDYNTXSLOTA										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-63. Last Dynamic Slot Channel A Register (LDTXSLAR)

This register provides the number of the last transmission slot in the dynamic segment for channel A. This register is updated after the end of the dynamic segment and before the start of the next communication cycle.

Table 13-75. LDTXSLAR Field Descriptions

Field	Description
10–0 LASTDYNTX SLOTA	**Last Dynamic Transmission Slot Channel A** — protocol related variable: *zLastDynTxSlot* channel A Number of the last transmission slot in the dynamic segment for channel A. If no frame was transmitted during the dynamic segment on channel A, the value of this field is set to 0.

13.5.2.64 Last Dynamic Transmit Slot Channel B Register (LDTXSLBR)

Module Base + 0x009E

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	LASTDYNTXSLOTB										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-64. Last Dynamic Slot Channel B Register (LDTXSLBR)

This register provides the number of the last transmission slot in the dynamic segment for channel B. This register is updated after the end of the dynamic segment and before the start of the next communication cycle.

Table 13-76. LDTXSLBR Field Descriptions

Field	Description
10–0 LASTDYNTX SLOTB	**Last Dynamic Transmission Slot Channel B** — protocol related variable: *zLastDynTxSlot* channel B Number of the last transmission slot in the dynamic segment for channel B. If no frame was transmitted during the dynamic segment on channel B the value of this field is set to 0.

13.5.2.65 Protocol Configuration Registers

The following configuration registers provide the necessary configuration information to the protocol engine. The individual values in the registers are described in Table 13-76. For more details about the FlexRay related configuration parameters and the allowed parameter ranges, see *FlexRay Communications System Protocol Specification, Version 2.1 Rev A*.

Table 13-77. Protocol Configuration Register Fields (Sheet 1 of 2)

Name	Description[1]	Min	Max	Unit	PCR
coldstart_attempts	*gColdstartAttempts*			number	3
action_point_offset	*gdActionPointOffset* - 1			MT	0
cas_rx_low_max	*gdCASRxLowMax* - 1			*gdBit*	4
dynamic_slot_idle_phase	*gdDynamicSlotIdlePhase*			minislot	28
minislot_action_point_offset	*gdMinislotActionPointOffset* - 1			MT	3
minislot_after_action_point	*gdMinislot* - *gdMinislotActionPointOffset* - 1			MT	2
static_slot_length	*gdStaticSlot*			MT	0
static_slot_after_action_point	*gdStaticSlot* - *gdActionPointOffset* - 1			MT	13
symbol_window_exists	*gdSymbolWindow*!=0	0	1	bool	9
symbol_window_after_action_point	*gdSymbolWindow* - *gdActionPointOffset* - 1			MT	6
tss_transmitter	*gdTSSTransmitter*			*gdBit*	5
wakeup_symbol_rx_idle	*gdWakeupSymbolRxIdle*			*gdBit*	5
wakeup_symbol_rx_low	*gdWakeupSymbolRxLow*			*gdBit*	3
wakeup_symbol_rx_window	*gdWakeupSymbolRxWindow*			*gdBit*	4
wakeup_symbol_tx_idle	*gdWakeupSymbolTxIdle*			*gdBit*	8
wakeup_symbol_tx_low	*gdWakeupSymbolTxLow*			*gdBit*	5
noise_listen_timeout	(*gListenNoise* * *pdListenTimeout*) - 1			µT	16/17
macro_initial_offset_a	*pMacroInitialOffset[A]*			MT	6
macro_initial_offset_b	*pMacroInitialOffset[B]*			MT	16
macro_per_cycle	*gMacroPerCycle*			MT	10
macro_after_first_static_slot	*gMacroPerCycle* - *gdStaticSlot*			MT	1
macro_after_offset_correction	*gMacroPerCycle* - *gOffsetCorrectionStart*			MT	28
max_without_clock_correction_fatal	*gMaxWithoutClockCorrectionFatal*			cyclepairs	8
max_without_clock_correction_passive	*gMaxWithoutClockCorrectionPassive*			cyclepairs	8
minislot_exists	*gNumberOfMinislots*!=0	0	1	bool	9
minislots_max	*gNumberOfMinislots* - 1			minislot	29
number_of_static_slots	*gNumberOfStaticSlots*			static slot	2
offset_correction_start	*gOffsetCorrectionStart*			MT	11
payload_length_static	*gPayloadLengthStatic*			2-bytes	19
max_payload_length_dynamic	*pPayloadLengthDynMax*			2-bytes	24
first_minislot_action_point_offset	max(*gdActionPointOffset*, *gdMinislotActionPointOffset*) - 1			MT	13
allow_halt_due_to_clock	*pAllowHaltDueToClock*			bool	26
allow_passive_to_active	*pAllowPassiveToActive*			cyclepairs	12

Table 13-77. Protocol Configuration Register Fields (Sheet 2 of 2)

Name	Description[1]	Min	Max	Unit	PCR
cluster_drift_damping	*pClusterDriftDamping*			µT	24
comp_accepted_startup_range_a	*pdAcceptedStartupRange* - *pDelayCompensationChA*			µT	22
comp_accepted_startup_range_b	*pdAcceptedStartupRange* - *pDelayCompensationChB*			µT	26
listen_timeout	*pdListenTimeout* - 1			µT	14/15
key_slot_id	*pKeySlotId*			number	18
key_slot_used_for_startup	*pKeySlotUsedForStartup*			bool	11
key_slot_used_for_sync	*pKeySlotUsedForSync*			bool	11
latest_tx	*gNumberOfMinislots* - *pLatestTx*			minislot	21
sync_node_max	*gSyncNodeMax*			number	30
micro_initial_offset_a	*pMicroInitialOffset[A]*			µT	20
micro_initial_offset_b	*pMicroInitialOffset[B]*			µT	20
micro_per_cycle	*pMicroPerCycle*			µT	22/23
micro_per_cycle_min	*pMicroPerCycle - pdMaxDrift*			µT	24/25
micro_per_cycle_max	*pMicroPerCycle + pdMaxDrift*			µT	26/27
micro_per_macro_nom_half	round(*pMicroPerMacroNom* / 2)			µT	7
offset_correction_out	*pOffsetCorrectionOut*			µT	9
rate_correction_out	*pRateCorrectionOut*			µT	14
single_slot_enabled	*pSingleSlotEnabled*			bool	10
wakeup_channel	*pWakeupChannel*	see Table 13-77			10
wakeup_pattern	*pWakeupPattern*			number	18
decoding_correction_a	*pDecodingCorrection* + *pDelayCompensation[A]* + 2			µT	19
decoding_correction_b	*pDecodingCorrection* + *pDelayCompensation[B]* + 2			µT	7
key_slot_header_crc	header CRC for key slot	0x000	0x7FF	number	12
extern_offset_correction	*pExternOffsetCorrection*			µT	29
extern_rate_correction	*pExternRateCorrection*			µT	21

1. See *FlexRay Communications System Protocol Specification, Version 2.1 Rev A* for detailed protocol parameter definitions

Table 13-78. Wakeup Channel Selection

wakeup_channel	Wakeup Channel
0	A
1	B

13.5.2.65.1 Protocol Configuration Register 0 (PCR0)

Module Base + 0x00A0

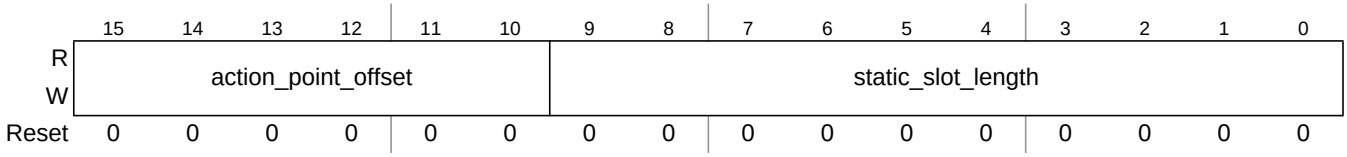


Figure 13-65. Protocol Configuration Register 0 (PCR0)

Write: *POC:config*

13.5.2.65.2 Protocol Configuration Register 1 (PCR1)

Module Base + 0x00A2

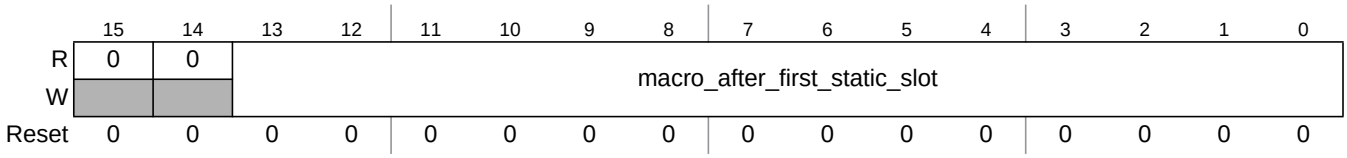


Figure 13-66. Protocol Configuration Register 1 (PCR1)

Write: *POC:config*

13.5.2.65.3 Protocol Configuration Register 2 (PCR2)

Module Base + 0x00A4

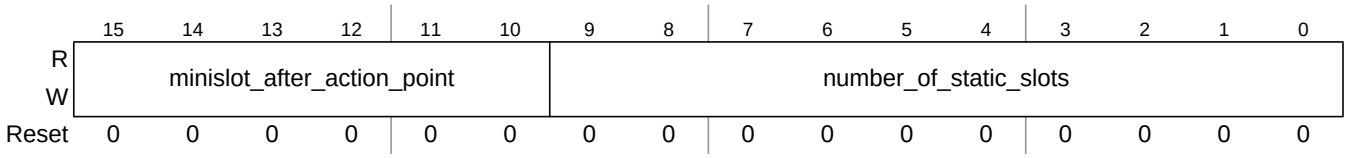


Figure 13-67. Protocol Configuration Register 2 (PCR2)

Write: *POC:config*

13.5.2.65.4 Protocol Configuration Register 3 (PCR3)

Module Base + 0x00A6

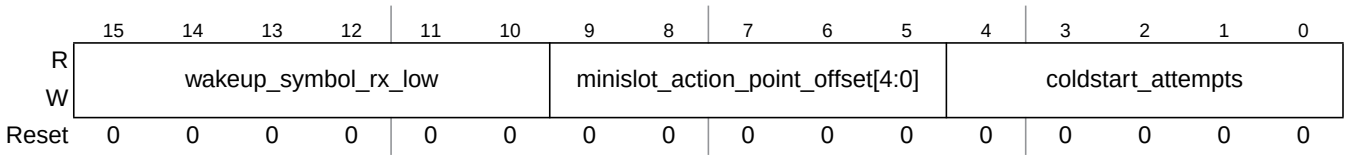


Figure 13-68. Protocol Configuration Register 3 (PCR3)

Write: *POC:config*

13.5.2.65.5 Protocol Configuration Register 4 (PCR4)

Module Base + 0x00A8

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	cas_rx_low_max								wakeup_symbol_rx_window							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-69. Protocol Configuration Register 4 (PCR4)

Write: *POC:config*

13.5.2.65.6 Protocol Configuration Register 5 (PCR5)

Module Base + 0x00AA

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	tss_transmitter				wakeup_symbol_tx_low						wakeup_symbol_rx_idle					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-70. Protocol Configuration Register 5 (PCR5)

Write: *POC:config*

13.5.2.65.7 Protocol Configuration Register 6 (PCR6)

Module Base + 0x00AC

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	symbol_window_after_action_point							macro_initial_offset_a							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-71. Protocol Configuration Register 6 (PCR6)

Write: *POC:config*

13.5.2.65.8 Protocol Configuration Register 7 (PCR7)

Module Base + 0x00AE

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	decoding_correction_b								micro_per_macro_nom_half							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-72. Protocol Configuration Register 7 (PCR7)

Write: *POC:config*

13.5.2.65.9 Protocol Configuration Register 8 (PCR8)

Module Base + 0x00B0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	max_without_clock_correction_fatal				max_without_clock_correction_passive				wakeup_symbol_tx_idle							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-73. Protocol Configuration Register 8 (PCR8)

Write: *POC:config*

13.5.2.65.10 Protocol Configuration Register 9 (PCR9)

Module Base + 0x00B2

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	mini slot_ exists	sym bol_ win dow_ exists	offset_correction_out													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-74. Protocol Configuration Register 9 (PCR9)

Write: *POC:config*

13.5.2.65.11 Protocol Configuration Register 10 (PCR10)

Module Base + 0x00B4

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	single _slot _en abled	wake up_ chan nel	macro_per_cycle													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-75. Protocol Configuration Register 10 (PCR10)

Write: *POC:config*

13.5.2.65.12 Protocol Configuration Register 11 (PCR11)

Module Base + 0x00B6

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	key_slot_used_for_start up	key_slot_used_for_sync	offset_correction_start													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-76. Protocol Configuration Register 11 (PCR11)

Write: *POC:config*

13.5.2.65.13 Protocol Configuration Register 12 (PCR12)

Module Base + 0x00B8

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	allow_passive_to_active				key_slot_header_crc											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-77. Protocol Configuration Register 12 (PCR12)

Write: *POC:config*

13.5.2.65.14 Protocol Configuration Register 13 (PCR13)

Module Base + 0x00BA

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	first_minislot_action_point_offset						static_slot_after_action_point									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-78. Protocol Configuration Register 13 (PCR13)

Write: *POC:config*

13.5.2.65.15 Protocol Configuration Register 14 (PCR14)

Module Base + 0x00BC

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	rate_correction_out												listen_timeout[20:16]			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-79. Protocol Configuration Register 14 (PCR14)

Write: *POC:config*

13.5.2.65.16 Protocol Configuration Register 15 (PCR15)

Module Base + 0x00BE

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	listen_timeout[15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-80. Protocol Configuration Register 15 (PCR15)

Write: *POC:config*

13.5.2.65.17 Protocol Configuration Register 16 (PCR16)

Module Base + 0x00C0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	macro_initial_offset_b								noise_listen_timeout[24:16]							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-81. Protocol Configuration Register 16 (PCR16)

Write: *POC:config*

13.5.2.65.18 Protocol Configuration Register 17 (PCR17)

Module Base + 0x00C2

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	noise_listen_timeout[15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-82. Protocol Configuration Register 17 (PCR17)

Write: *POC:config*

13.5.2.65.19 Protocol Configuration Register 18 (PCR18)

Module Base + 0x00C4

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	wakeup_pattern						key_slot_id									
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-83. Protocol Configuration Register 18 (PCR18)

Write: *POC:config*

13.5.2.65.20 Protocol Configuration Register 19 (PCR19)

Module Base + 0x00C6

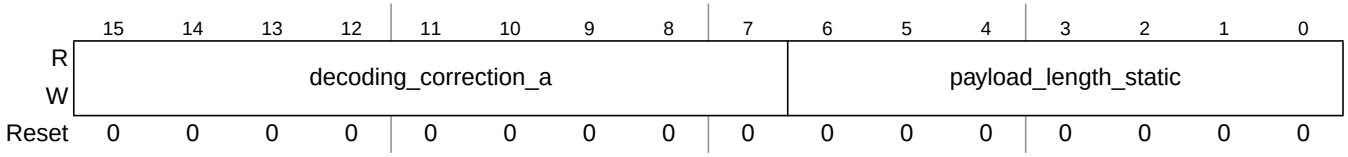


Figure 13-84. Protocol Configuration Register 19 (PCR19)

Write: *POC:config*

13.5.2.65.21 Protocol Configuration Register 20 (PCR20)

Module Base + 0x00C8

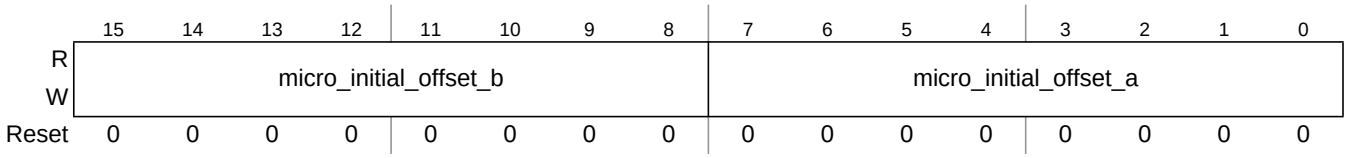


Figure 13-85. Protocol Configuration Register 20 (PCR20)

Write: *POC:config*

13.5.2.65.22 Protocol Configuration Register 21 (PCR21)

Module Base + 0x00CA

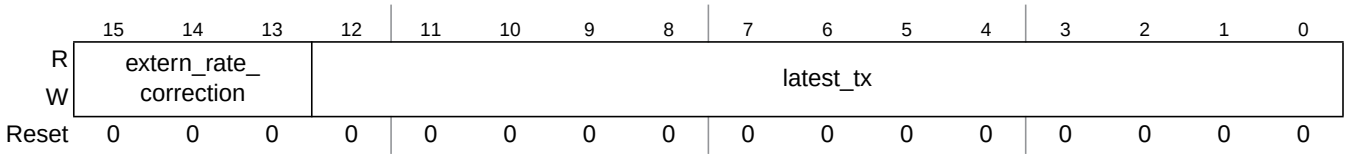


Figure 13-86. Protocol Configuration Register 21 (PCR21)

Write: *POC:config*

13.5.2.65.23 Protocol Configuration Register 22 (PCR22)

Module Base + 0x00CC

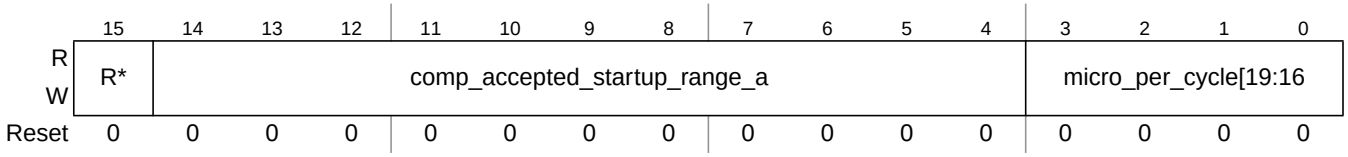


Figure 13-87. Protocol Configuration Register 22 (PCR22)

Write: *POC:config*

13.5.2.65.24 Protocol Configuration Register 23 (PCR23)

Module Base + 0x00CE

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	micro_per_cycle[15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-88. Protocol Configuration Register 23 (PCR23)

Write: *POC:config*

13.5.2.65.25 Protocol Configuration Register 24 (PCR24)

Module Base + 0x00D0

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	cluster_drift_damping				max_payload_length_dynamic								micro_per_cycle_min [19:16]			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-89. Protocol Configuration Register 24 (PCR24)

Write: *POC:config*

13.5.2.65.26 Protocol Configuration Register 25 (PCR25)

Module Base + 0x00D2

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	micro_per_cycle_min[15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-90. Protocol Configuration Register 25 (PCR25)

Write: *POC:config*

13.5.2.65.27 Protocol Configuration Register 26 (PCR26)

Module Base + 0x00D4

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	allow_halt_due_to_clock	comp_accepted_startup_range_b											micro_per_cycle_max [19:16]			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-91. Protocol Configuration Register 26 (PCR26)

Write: *POC:config*

13.5.2.65.28 Protocol Configuration Register 27 (PCR27)

Module Base + 0x00D6

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	micro_per_cycle_max[15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-92. Protocol Configuration Register 27 (PCR27)

Write: *POC:config*

13.5.2.65.29 Protocol Configuration Register 28 (PCR28)

Module Base + 0x00D8

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	dynamic_slot_idle_phase				macro_after_offset_correction											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-93. Protocol Configuration Register 28 (PCR28)

Write: *POC:config*

13.5.2.65.30 Protocol Configuration Register 29 (PCR29)

Module Base + 0x00DA

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	extern_offset_correction				minislots_max											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-94. Protocol Configuration Register 29 (PCR29)

Write: *POC:config*

13.5.2.65.31 Protocol Configuration Register 30 (PCR30)

Module Base + 0x00DC

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	0	0	sync_node_max			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-95. Protocol Configuration Register 30 (PCR30)

Write: *POC:config*

13.5.2.66 Message Buffer Configuration, Control, Status Registers (MBCCSRn)

Module Base + 0x0100 (MBCCSR0)
Module Base + 0x0108 (MBCCSR1)
...
Module Base + 0x01F8 (MBCCSR31)

Additional Reset: CMT, DUP, DVAL, MBIF: Message Buffer Disable

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	MCM	MBT	MTD	CMT	0	0	MBIE	0	0	0	DUP	DVAL	EDS	LCKS	MBIF
W					rwm	EDT	LCKT									w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13-96. Message Buffer Configuration, Control, Status Registers (MBCCSRn)

Write: MCM, MBT, MTD: *POC:config* or MB_DIS
CMT: MB_LCK
EDT, LCKT, MBIE, MBIF: Normal Mode

The content of these registers comprises message buffer configuration data, message buffer control data, message buffer status information, and message buffer interrupt flags.

Table 13-79. MBCCSRn Field Descriptions (Sheet 1 of 3)

Field	Description
Message Buffer Configuration	
14 MCM	**Message Buffer Commit Mode** — This bit applies only to double buffered transmit message buffers and defines the commit mode. 0 Streaming commit mode 1 Immediate commit mode
13 MBT	**Message Buffer Type** — This bit applies only to transmit message buffers and defines the buffering type. 0 Single buffered transmit message buffer 1 Double buffered transmit message buffer
12 MTD	**Message Buffer Transfer Direction** — This bit defines the transfer direction of the message buffer. 0 Receive message buffer 1 Transmit message buffer
Message Buffer Control	
11 CMT	**Commit for Transmission** — This bit applies only to transmit message buffers and indicates whether the message buffer contains valid data that are ready for transmission. Both the application and the FlexRay block can modify this bit. • Application: The application sets this bit to indicate that the transmit message buffer contains valid data ready for transmission. The application clears this bit to indicate that the message buffer data are no longer valid for transmission. • FlexRay block: The FlexRay block clears this bit when the message buffer data are no longer valid for transmission. 0 Message buffer does not contain valid data. 1 Message buffer contains valid data.

Table 13-79. MBCCSRn Field Descriptions (Sheet 2 of 3)

Field	Description
10 EDT	**Enable/Disable Trigger** — This trigger bit is used to enable and disable a message buffer. The message buffer enable is triggered when the application writes 1 to this bit and the message buffer is disabled, i.e. the EDS status bit is 0. The message buffer disable is triggered when the application writes 1 to this bit and the message buffer is enabled, i.e. the EDS status bit is 1. 0 No effect 1 message buffer enable/disable triggered **Note:** If the application writes 1 to this bit, the write access to all other bits is ignored.
9 LCKT	**Lock/Unlock Trigger** — This trigger bit is used to lock and unlock a message buffer. The message buffer lock is triggered when the application writes 1 to this bit and the message buffer is not locked, i.e. the LCKS status bit is 0. The message buffer unlock is triggered when the application writes 1 to this bit and the message buffer is locked, i.e. the LCKS status bit is 1. 0 No effect 1 Trigger message buffer lock/unlock **Note:** If the application writes 1 to this bit and 0 to the EDT bit, the write access to all other bits is ignored.
8 MBIE	**Message Buffer Interrupt Enable** — This control bit defines whether the message buffer will generate an interrupt request when its MBIF flag is set. 0 Interrupt request generation disabled 1 Interrupt request generation enabled
Message Buffer Status	
4 DUP	**Data Updated** — This status bit applies only to receive message buffers. It is always 0 for transmit message buffers. This bit provides information whether the frame header in the message buffer header field and the message buffer data field were updated. See Section 13.6.6.3.3, “Message Buffer Status Update” for a detailed description of the update condtions. 0 Frame Header and Message buffer data field not updated. 1 Frame Header and Message buffer data field updated.
3 DVAL	**Data Valid** — The semantic of this status bit depends on the message buffer type and transfer direction. • *Receive Message Buffer:* Indicates whether the message buffer data field contains valid frame data. See Section 13.6.6.3.3, “Message Buffer Status Update” for a detailed update description of the update conditions. 0 message buffer data field contains no valid frame data 1 message buffer data field contains valid frame data • *Single Transmit Message Buffer*: Indicates whether the message is transferred again due to the state transmission mode of the message buffer. 0 Message transferred for the first time. 1 Message will be transferred again. • *Double Transmit Message Buffer*: For the commit side it is always 0. For the transmit side it indicates whether the message is transferred again due to the state transmission mode of the message buffer. 0 Message transferred for the first time. 1 Message will be transferred again.
2 EDS	**Enable/Disable Status** — This status bit indicates whether the message buffer is enabled or disabled. 0 Message buffer is disabled. 1 Message buffer is enabled.

Table 13-79. MBCCSRn Field Descriptions (Sheet 3 of 3)

Field	Description
1 LCKS	**Lock Status** — This status bit indicates the current lock status of the message buffer. 0 Message buffer is not locked by the application. 1 Message buffer is locked by the application.
0 MBIF	**Message Buffer Interrupt Flag** — The semantic of this flag depends on the message buffer transfer direction. • *Receive Message Buffer:* This flag is set when the slot status in the message buffer header field was updated and this slot was not an empty dynamic slot. See Section 13.6.6.3.3, “Message Buffer Status Update” for a detailed description of the update conditions. 0 slot status not updated 1 slot status updated and slot was not an empty dynamic slot • *Transmit Message Buffer:* This flag is set when the slot status in the message buffer header field was updated. Additionally this flag is set immediately when a transmit message buffer was enabled. 0 slot status not updated 1 slot status updated / message buffer just enabled

13.5.2.67 Message Buffer Cycle Counter Filter Registers (MBCCFRn)

Module Base + 0x0102 (MBCCFR0)
Module Base + 0x010A (MBCCFR1)
...
Module Base + 0x01FA (MBCCFR31)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	MTM	CHA	CHB	CCFE	CCFMSK						CCFVAL					
Reset	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Figure 13-97. Message Buffer Cycle Counter Filter Registers (MBCCFRn)

Write: *POC:config* or MB_DIS

This register contains message buffer configuration data for the transmission mode, the channel assignment, and for the cycle counter filtering. For detailed information on cycle counter filtering, refer to Section 13.6.7.1, “Message Buffer Cycle Counter Filtering”.

Table 13-80. MBCCFRn Field Descriptions

Field	Description
15 MTM	**Message Buffer Transmission Mode** — This control bit applies only to transmit message buffers and defines the transmission mode. 0 Event transmission mode 1 State transmission mode
14–13 CHA CHB	**Channel Assignment** — These control bits define the channel assignment and control the receive and transmit behavior of the message buffer according to **Table 13-80**.
12 CCFE	**Cycle Counter Filtering Enable** — This control bit is used to enable and disable the cycle counter filtering. 0 Cycle counter filtering disabled 1 Cycle counter filtering enabled

Table 13-80. MBCCFRn Field Descriptions

Field	Description
11–6 CCFMSK	**Cycle Counter Filtering Mask** — This field defines the filter mask for the cycle counter filtering.
5–0 CCFVAL	**Cycle Counter Filtering Value** — This field defines the filter value for the cycle counter filtering.

Table 13-81. Channel Assignment Description

CHA	CHB	Transmit Message Buffer		Receive Message Buffer	
		static segment	dynamic segment	static segment	dynamic segment
1	1	transmit on both channel A and channel B	transmit on channel A only	store first valid frame received on either channel A or channel B	store first valid frame received on channel A, ignore channel B
0	1	transmit on channel B	transmit on channel B	store first valid frame received on channel B	store first valid frame received on channel B
1	0	transmit on channel A	transmit on channel A	store first valid frame received on channel A	store first valid frame received on channel A
0	0	no frame transmission	no frame transmission	no frame stored	no frame stored

NOTE

If at least one message buffer assigned to a certain slot is assigned to both channels, then all message buffers assigned to this slot have to be assigned to both channels. Otherwise, the message buffer configuration is illegal and the result of the message buffer search is not defined.

13.5.2.68 Message Buffer Frame ID Registers (MBFIDRn)

Module Base + 0x0104 (MBFIDR0)
Module Base + 0x010C (MBFIDR1)
...
Module Base + 0x01FC (MBFIDR31)

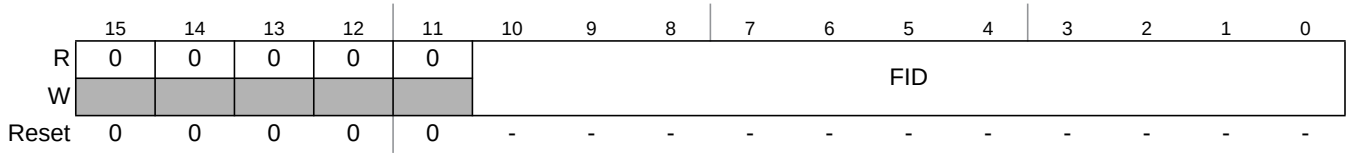


Figure 13-98. Message Buffer Frame ID Registers (MBFIDRn)

Write: *POC:config* or MB_DIS

Table 13-82. MBFIDRn Field Descriptions

Field	Description
10–0 FID	**Frame ID** — The semantic of this field depends on the message buffer transfer type. • *Receive Message Buffer:* This field is used as a filter value to determine if the message buffer is used for reception of a message received in a slot with the slot ID equal to FID. • *Transmit Message Buffer:* This field is used to determine the slot in which the message in this message buffer should be transmitted.

13.5.2.69 Message Buffer Index Registers (MBIDXRn)

Module Base + 0x0106 (MBIDXR0)
Module Base + 0x010E (MBIDXR1)
...
Module Base + 0x01FE (MBIDXR31)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	MBIDX					
W																
Reset	0	0	0	0	0	0	0	0	0	0	-	-	-	-	-	-

Figure 13-99. Message Buffer Index Registers (MBIDXRn)

Write: *POC:config* or MB_DIS

Table 13-83. MBIDXRn Field Descriptions

Field	Description
5–0 MBIDX	**Message Buffer Index** — This field provides the index of the message buffer header field of the physical message buffer that is currently associated with this message buffer. The application writes the index of the initially associated message buffer header field into this register. The FlexRay block updates this register after frame reception or transmission.

13.6 Functional Description

This section provides a detailed description of the functionality implemented in the FlexRay block.

13.6.1 Message Buffer Concept

The FlexRay block uses a data structure called *message buffer* to store frame data, configuration, control, and status data. Each message buffer consists of two parts, the *message buffer control data* and the *physical message buffer.* The message buffer control data are located in dedicated registers. The structure of the message buffer control data depends on the message buffer type and is described in Section 13.6.3, “Message Buffer Types”. The physical message buffer is located in the FRM and is described in Section 13.6.2, “Physical Message Buffer”.

13.6.2 Physical Message Buffer

All FlexRay messages and related frame and slot status information of received frames and of frames to be transmitted to the FlexRay bus are stored in data structures called *physical message buffers*. The physical message buffers are located in the FRM.The structure of a physical message buffer is depicted in Figure 13-100.

A physical message buffer consists of two fields, the *message buffer header field* and the *message buffer data field*. The message buffer header field contains the *frame header*, the *data field offset*, and the *slot status*.The message buffer data field contains the *frame data*.

The connection between the two fields is established by the *data field offset*.

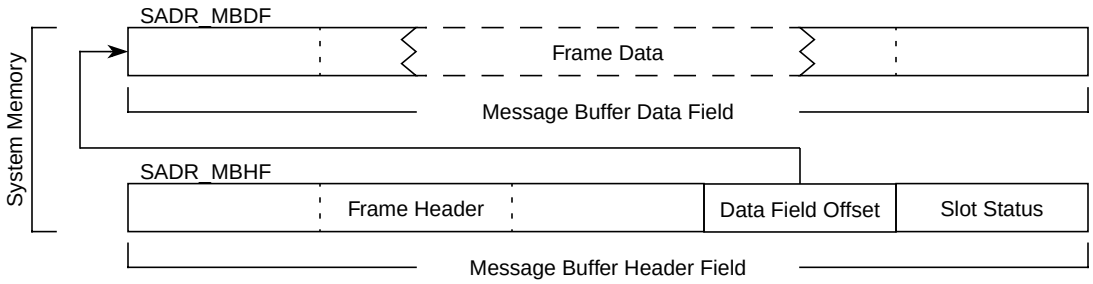


Figure 13-100. Physical Message Buffer Structure

13.6.2.1 Message Buffer Header Field

The message buffer header field is a contiguous region in the FRM and occupies ten bytes. It contains the frame header, the data field offset, and the slot status. Its structure is shown in Figure 13-100. The physical start address *SADR_MBHF* of the message buffer header field must be 16-bit aligned.

13.6.2.1.1 Frame Header

The frame header occupies the first six bytes in the message buffer header field. It contains all FlexRay frame header related information according to the *FlexRay Communications System Protocol*

Specification, Version 2.1 Rev A. A detailed description of the usage and the content of the frame header is provided in Section 13.6.5.2.1, “Frame Header Section Description”.

13.6.2.1.2 Data Field Offset

The data field offset follows the frame header in the message buffer data field and occupies two bytes. It contains the offset of the corresponding message buffer data field with respect to the FlexRay block FRM base address as provided by SYS_MEM_BASE_ADDR field in the System Memory Base Address High Register (SYMBADHR) and System Memory Base Address Low Register (SYMBADLR)”. The data field offset is used to determine the start address *SADR_MBDF* of the corresponding message buffer data field in the FRM according to Equation 13-2.

$$SADR\_MBDF = [Data\ Field\ Offset] + SYS\_MEM\_BASE\_ADDR \quad \textit{Eqn. 13-2}$$

13.6.2.1.3 Slot Status

The slot status occupies the last two bytes of the message buffer header field. It provides the slot and frame status related information according to the *FlexRay Communications System Protocol Specification, Version 2.1 Rev A*. A detailed description of the content and usage of the slot status is provided in Section 13.6.5.2.3, “Slot Status Description”.

13.6.2.2 Message Buffer Data Field

The message buffer data field is a contiguous area of 2-byte entities. This field contains the frame payload data, or a part of it, of the frame to be transmitted to or received from the FlexRay bus. The minimum length of this field depends on the specific message buffer configuration and is specified in the message buffer descriptions given in Section 13.6.3, “Message Buffer Types”.

13.6.3 Message Buffer Types

The FlexRay block provides three different types of message buffers.

- Individual Message Buffers
- Receive Shadow Buffers
- Receive FIFO Buffers

For each message buffer type the structure of the physical message buffer is identical. The message buffer types differ only in the structure and content of message buffer control data, which control the related physical message buffer. The message buffer control data are described in the following sections.

13.6.3.1 Individual Message Buffers

The individual message buffers are used for all types of frame transmission and for dedicated frame reception based on individual filter settings for each message buffer. The FlexRay block supports three types of individual message buffers, which are described in Section 13.6.6, “Individual Message Buffer Functional Description”.

Each individual message buffer consists of two parts, the physical message buffer, which is located in the FRM, and the message buffer control data, which are located in dedicated registers. The structure of an individual message buffer is given in Figure 13-101.

Each individual message buffer has a message buffer number n assigned, which determines the set of message buffer control registers associated to this individual message buffer. The individual message buffer with message buffer number *n* is controlled by the registers MBCCSRn, MBCCFRn, MBFIDRn, and MBIDXRn.

The connection between the message buffer control registers and the physical message buffer is established by the message buffer index field MBIDX in the Message Buffer Index Registers (MBIDXRn). The start address SADR_MBHF of the related message buffer header field in the FRM is determined according to Equation 13-3.

$$SADR\_MBHF = (MBIDXRn[MBIDX] * 10) + SYS\_MEM\_BASE\_ADDR \qquad \textit{Eqn. 13-3}$$

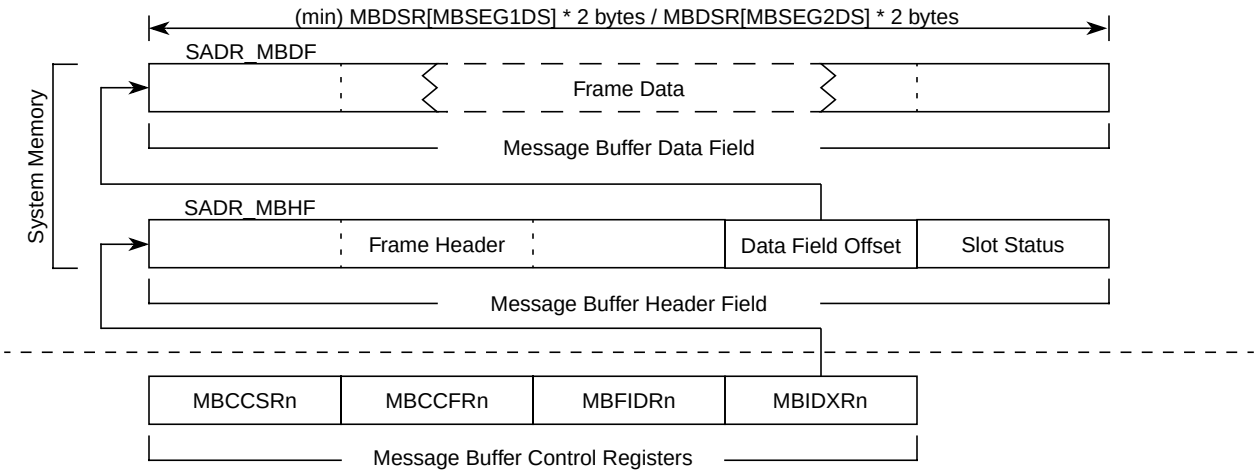


Figure 13-101. Individual Message Buffer Structure

13.6.3.1.1 Individual Message Buffer Segments

The set of the individual message buffers can be split up into two message buffer segments using the Message Buffer Segment Size and Utilization Register (MBSSUTR). All individual message buffers with a message buffer number n <= MBSSUTR.LAST_MB_SEG1 belong to the first message buffer segment. All individual message buffers with a message buffer number n > MBSSUTR.LAST_MB_SEG1 belong to the second message buffer segment. The following rules apply to the length of the message buffer data field:

- all physical message buffers associated to individual message buffers that belong to the same message buffer segment must have message buffer data fields of the same length
- the minimum length of the message buffer data field for individual message buffers in the first message buffer segment is 2 * MBDSR.MBSEG1DS bytes
- the minimum length of the message buffer data field for individual message buffers assigned to the second segment is 2 * MBDSR.MBSEG2DS bytes.

13.6.3.2 Receive Shadow Buffers

The receive shadow buffers are required for the frame reception process for individual message buffers. The FlexRay block provides four receive shadow buffers, one receive shadow buffer per channel and per message buffer segment.

Each receive shadow buffer consists of two parts, the physical message buffer located in the FRM and the receive shadow buffer control registers located in dedicated registers. The structure of a receive shadow buffer is shown in Figure 13-102. The four internal shadow buffer control registers can be accessed by the Receive Shadow Buffer Index Register (RSBIR).

The connection between the receive shadow buffer control register and the physical message buffer for the selected receive shadow buffer is established by the receive shadow buffer index field RSBIDX in the Receive Shadow Buffer Index Register (RSBIR). The start address SADR_MBHF of the related message buffer header field in the FRM is determined according to Equation 13-4.

$$SADR\_MBHF = (RSBIR[RSBIDX] * 10) + SYS\_MEM\_BASE\_ADDR \qquad \textit{Eqn. 13-4}$$

The length required for the message buffer data field depends on the message buffer segment that the receive shadow buffer is assigned to. For the receive shadow buffers assigned to the first message buffer segment, the length must be the same as for the individual message buffers assigned to the first message buffer segment. For the receive shadow buffers assigned to the second message buffer segment, the length must be the same as for the individual message buffers assigned to the second message buffer segment. The receive shadow buffer assignment is described in Receive Shadow Buffer Index Register (RSBIR).

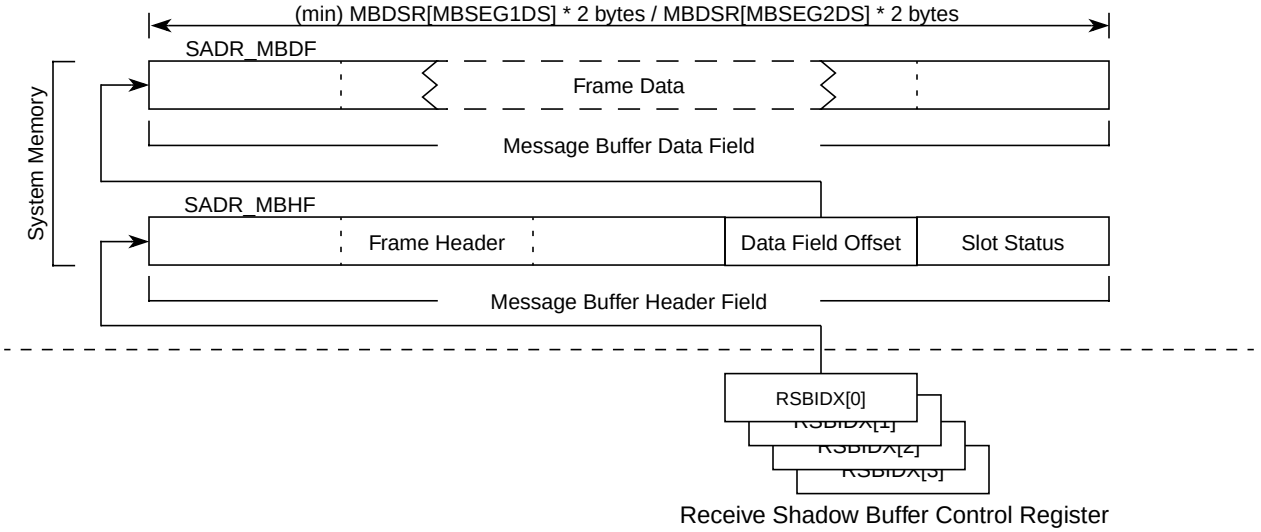


Figure 13-102. Receive Shadow Buffer Structure

13.6.3.3 Receive FIFO

The receive FIFO implements a frame reception system based on the FIFO concept. The FlexRay block provides two independent receive FIFOs, one per channel.

A receive FIFO consists of a set of physical message buffers in the FRM and a set of receive FIFO control registers located in dedicated registers. The structure of a receive FIFO is given in Figure 13-103.

The connection between the receive FIFO control registers and the set of physical message buffers is established by the start index field SIDX in the Receive FIFO Start Index Register (RFSIR), the FIFO depth field FIFO_DEPTH in the Receive FIFO Depth and Size Register (RFDSR), and the read index field RDIDX Receive FIFO A Read Index Register (RFARIR) / Receive FIFO B Read Index Register (RFBRIR). The start address SADR_MBHF_1 of the first message buffer header field that belongs to the receive FIFO in the FRM is determined according to Equation 13-5.

$$SADR\_MBHF[1] = (RFSIR[SIDX] * 10) + SYS\_MEM\_BASE\_ADDR \quad \textit{Eqn. 13-5}$$

The start address SADR_MBHF[n] of the last message buffer header field that belongs to the receive FIFO in the FRM is determined according to Equation 13-6.

$$SADR\_MBHF[n] = ((RFSIR[SIDX] + RFDSR[FIFO\_DEPTH]) * 10) + SYS\_MEM\_BASE\_ADDR \quad \textit{Eqn. 13-6}$$

NOTE

All message buffer header fields assigned to a receive FIFO must be a contiguous region.

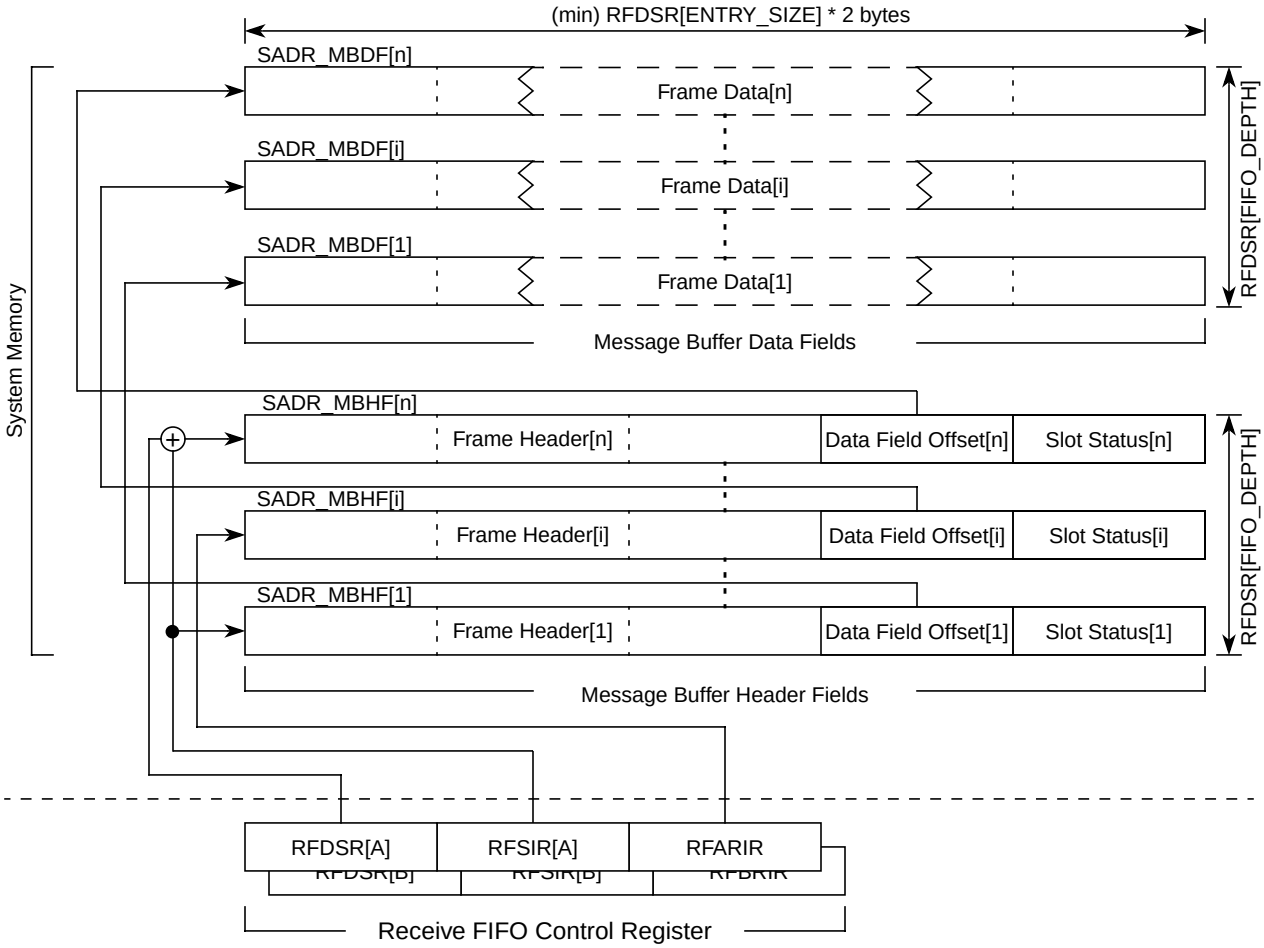


Figure 13-103. Receive FIFO Structure

13.6.3.4 Message Buffer Configuration and Control Data

This section describes the configuration and control data for each message buffer type.

13.6.3.4.1 Individual Message Buffer Configuration Data

Before an individual message buffer can be used for transmission or reception, it must be configured. There is a set of common configuration parameters that applies to all individual message buffers and a set of configuration parameters that applies to each message buffer individually.

Common Configuration Data

The set of common configuration data for individual message buffers is located in the following registers.

- Message Buffer Data Size Register (MBDSR)
  The MBSEG2DS and MBSEG1DS fields define the minimum length of the message buffer data field with respect to the message buffer segment.
- Message Buffer Segment Size and Utilization Register (MBSSUTR)
  The LAST_MB_SEG1 and LAST_MB_UTIL fields define the segmentation of the individual message buffers and the number of individual message buffers that are used. For more details, see Section 13.6.3.1.1, “Individual Message Buffer Segments”

Specific Configuration Data

The set of message buffer specific configuration data for individual message buffers is located in the following registers.

- Message Buffer Configuration, Control, Status Registers (MBCCSRn)
  The MCM, MBT, MTD bits configure the message buffer type.
- Message Buffer Cycle Counter Filter Registers (MBCCFRn)
  The MTM, CHA, CHB bits configure the transmission mode and the channel assignment. The CCFE, CCFMSK, and CCFVAL bits and fields configure the cycle counter filter.
- Message Buffer Frame ID Registers (MBFIDRn)
  For a transmit message buffer, the FID field is used to determine the slot in which the message in this message buffer will be transmitted.
- Message Buffer Index Registers (MBIDXRn)
  This MBIDX field provides the index of the message buffer header field of the physical message buffer that is currently associated with this message buffer.

13.6.3.5 Individual Message Buffer Control Data

During normal operation, each individual message buffer can be controlled by the control and trigger bits CMT, LCKT, EDT, and MBIE in the Message Buffer Configuration, Control, Status Registers (MBCCSRn).

13.6.3.6 Receive Shadow Buffer Configuration Data

Before frame reception into the individual message buffers can be performed, the receive shadow buffers must be configured. The configuration data are provided by the Receive Shadow Buffer Index Register (RSBIR). For each receive shadow buffer, the application provides the message buffer header index. When the protocol is in the *POC:normal active* or *POC:normal passive* state, the receive shadow buffers are under full FlexRay block control.

13.6.3.7 Receive FIFO Control and Configuration Data

This section describes the configuration and control data for the two receive FIFOs.

13.6.3.7.1 Receive FIFO Configuration Data

The FlexRay block provides two completely independent receive FIFOs, one per channel. Each FIFO has its own set of configuration data. The configuration data are located in the following registers:

- Receive FIFO Start Index Register (RFSIR)
- Receive FIFO Depth and Size Register (RFDSR)
- Receive FIFO Message ID Acceptance Filter Value Register (RFMIDAFVR)
- Receive FIFO Message ID Acceptance Filter Mask Register (RFMIAFMR)
- Receive FIFO Frame ID Rejection Filter Value Register (RFFIDRFVR)
- Receive FIFO Frame ID Rejection Filter Mask Register (RFFIDRFMR)
- Receive FIFO Range Filter Configuration Register (RFRFCFR)

13.6.3.7.2 Receive FIFO Control Data

The application can access the receive FIFO at any time using the values provided in the Receive FIFO A Read Index Register (RFARIR) and Receive FIFO B Read Index Register (RFBRIR). To update the Receive FIFO A Read Index Register (RFARIR), the application must write 1 to the FIFO A Not Empty Interrupt Flag FNEAIF in the Global Interrupt Flag and Enable Register (GIFER). To update the Receive FIFO B Read Index Register (RFBRIR) the application must write 1 to the FIFO B Not Empty Interrupt Flag FNEBIF in the Global Interrupt Flag and Enable Register (GIFER). As long as the FIFO is not empty, each update increments the related read index. If the read index has reached the last FIFO entry, it wraps back to the FIFO start index.

13.6.4 FlexRay Memory Layout

The FlexRay block supports a wide range of possible layouts for the FRM. Figure 13-104 shows an example layout. The following set of rules applies to the layout of the FRM:

- The FRM is a contiguous region.
- The FRM size is maximum 64 Kbytes.
- The FRM starts at a 16 byte boundary.

The FRM contains three areas: the *message buffer header area*, the *message buffer data area*, and the *sync frame table area*. The areas are described in this section.

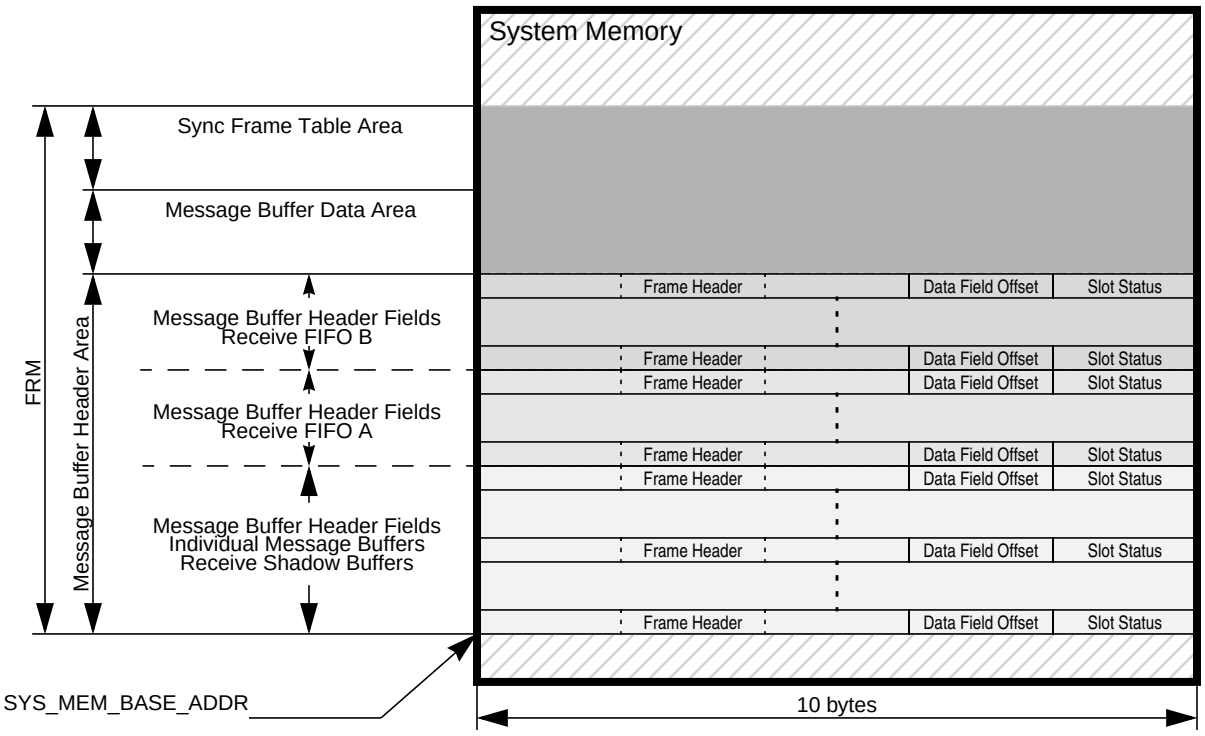


Figure 13-104. Example of FRM Layout

13.6.4.1 Message Buffer Header Area

The message buffer header area contains all message buffer header fields of the physical message buffers for all message buffer types. The following rules apply to the message buffer header fields for the three type of message buffers.

1. The start address SADR_MBHF of each message buffer header field for *individual message buffers* and *receive shadow buffers* must fulfill Equation 13-7.

$$SADR\_MBHF = (i * 10) + SYS\_MEM\_BASE\_ADDR;\ (0 <= i < 64) \qquad \textit{Eqn. 13-7}$$

2. The start address SADR_MBHF of each message buffer header field for the *receive FIFO* must fulfill Equation 13-8.

$$SADR\_MBHF = (i * 10) + SYS\_MEM\_BASE\_ADDR;\ (0 <= i < 1024) \qquad \textit{Eqn. 13-8}$$

3. The message buffer header fields for a receive FIFO have to be a contiguous area.

13.6.4.2 Message Buffer Data Area

The message buffer data area contains all the message buffer data fields of the physical message buffers. Each message buffer data field must start at a 16-bit boundary.

13.6.4.3 Sync Frame Table Area

The sync frame table area is used to provide a copy of the internal sync frame tables for application access. Refer to Section 13.6.12, “Sync Frame ID and Sync Frame Deviation Tables” for the description of the sync frame table area.

13.6.5 Physical Message Buffer Description

This section provides a detailed description of the usage and the content of the two parts of a physical message buffer, the message buffer header field and the message buffer data field.

13.6.5.1 Message Buffer Protection and Data Consistency

The physical message buffers are located in the FRM. The FlexRay block provides no means to protect the FRM from uncontrolled or illegal host or other client write access. To ensure data consistency of the physical message buffers, the application must follow the write access scheme that is given in the description of each of the physical message buffer fields.

13.6.5.2 Message Buffer Header Field Description

This section provides a detailed description of the usage and content of the message buffer header field. A description of the structure of the message buffer header fields is given in Section 13.6.2.1, “Message Buffer Header Field”. Each message buffer header field consists of three sections: the frame header section, the data field offset, and the slot status section. For a detailed description of the Data Field Offset, see Section 13.6.2.1.2, “Data Field Offset”.

13.6.5.2.1 Frame Header Section Description

Frame Header Section Content

The semantic and content of the frame header section depends on the message buffer type.

For individual receive message buffers and receive FIFOs, the frame header receives the frame header data of the *first valid frame* received on the assigned channels. If a receive message buffer is assigned to both channels, the first valid frame received on either channel A or channel B is stored.

For receive shadow buffers, the frame header receives the frame header data of the current frame received regardless of whether the frame is valid or not.

For single and double transmit message buffers, the application writes the frame header of the frame to be transmitted into this location. The frame header will be read out when the frame is transferred to the FlexRay bus.

The structure of the frame header in the message buffer header field is given in Figure 13-105. A detailed description of the frame header fields is given in Table 13-84.

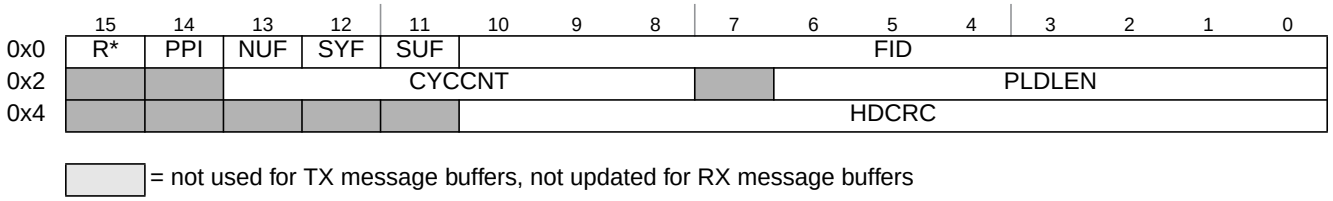


Figure 13-105. Frame Header Structure

Frame Header Section Access

The frame header is located in the FRM. To ensure data consistency, the application must follow the write access scheme described below.

For receive message buffers, receive shadow buffers, and receive FIFOs, the application must not write to the frame header field.

For transmit message buffers, the application must follow the write access restrictions given in Table 13-83. This table shows the condition under which the application can write to the frame header entries. In general, the application can modify all frame header entries when the protocol is in the *POC:config* state or when the message buffer is disabled. For message buffers assigned to the dynamic segment, the application can modify all frame header entries except the frame ID when the message buffer is locked.

Table 13-84. Frame Header Write Access Constraints

Field	TX					
	Single Buffered		Double Buffered			
	Static Segment	Dynamic Segment	Static Segment		Dynamic Segment	
			Commit Side	Transmit Side	Commit Side	Transmit Side
FID	*POC:config* or MB_DIS					
R*, PPI NUF, SYF SUF CYCCNT PLDLEN HDCRD	*POC:config* or MB_DIS	*POC:config* or MB_DIS or MB_LCK	*POC:config* or MB_DIS		*POC:config* or MB_DIS or MB_LCK	*POC:config* or MB_DIS

The frame header entries NUF, SYF, SUF, and CYCCNT are not used for frame transmission. These values are generated internally before frame transmission depending on the current transmission state and configuration.

For transmit message buffers assigned to the *static* segment, the PLDLEN value must be equal to the value of the payload_length_static field in the Protocol Configuration Register 19 (PCR19). If this is not fulfilled, the static payload length error flag SPL_EF in the CHI Error Flag Register (CHIERFR) is set when the message buffer is under transmission. The PE generates a syntactically and semantically correct frame with payload_length_static payload words and the payload length field in the frame header set to payload_length_static.

For transmit message buffers assigned to the *dynamic* segment, the PLDLEN value must be less than or equal to the value of the max_payload_length_dynamic field in the Protocol Configuration Register 24 (PCR24). If this is not fulfilled, the dynamic payload length error flag DPL_EF in the CHI Error Flag Register (CHIERFR) is set when the message buffer is under transmission. The PE generates a syntactically and semantically correct dynamic frame with PLDLEN payload words and the payload length field in the frame header set to PLDLEN.

Table 13-85. Frame Header Field Descriptions

Field	Description
R*	**Reserved Bit** — This bit corresponds to the *Reserved bit* in the FlexRay frame header. • For receive and FIFO message buffers, this is a status bit and represents the value of the Reserved bit in the frame received on the FlexRay bus in the corresponding slot. • For transmit message buffers, this is a control bit. The FlexRay block transmits this within the frame header. **Note:** For protocol compliant operation, this control bit must be set to 0 for transmit message buffers.
PPI	**Payload Preamble Indicator** — This bit corresponds to the *Payload Preamble Indicator* in the FlexRay frame header. • For receive and FIFO message buffers, this is a status bit and represents the value of the *Payload Preamble Indicator* of the first valid frame received on the FlexRay in the slot indicated by the CYCCNT field. • For transmit message buffers, this is a control bit. The FlexRay block uses this value to set the *Payload Preamble Indicator* in the frame header of the frame to transmit. 0 No network management vector or message ID in frame payload data 1 Static Segment: Frame payload data contains network management vector Dynamic Segment: Frame payload data contains message ID
NUF	**Null Frame Indicator** — This bit corresponds to the *Null Frame Indicator* in the FlexRay frame header. • For receive message buffers and receive FIFOs, this is a status bit and represents the value of the *Null Frame Indicator* of the first valid frame received on the FlexRay bus in the slot indicated by the CYCCNT field. • For transmit message buffers, the value of this bit is ignored. The FlexRay block determines internally whether a null frame or non-null frame must be transmitted and sets the *Null Frame Indicator* accordingly. 0 Null frame received 1 Normal frame received
SYF	**Sync Frame Indicator** — This bit corresponds to the *Sync Frame Indicator* in the FlexRay frame header. • For receive message buffers and receive FIFOs, this is a status bit and represents the value of the *Sync Frame Indicator* of the first valid frame received on the FlexRay bus in the slot indicated by the CYCCNT field. • For transmit message buffers, the value of this bit is ignored. The FlexRay block determines internally whether a sync frame must be transmitted and sets the *Sync Frame Indicator* accordingly.
SUF	**Startup Frame Indicator** — This bit corresponds to the *Startup Frame Indicator* in the FlexRay frame header. • For receive message buffers and receive FIFOs, this is a status bit and represents the value of the *Startup Frame Indicator* of the first valid frame received on the FlexRay bus in the slot indicated by the CYCCNT field • For transmit message buffers, the value of this bit is ignored. The FlexRay block determines internally whether a startup frame must be transmitted and sets the *Startup Frame Indicator* accordingly.
FID	**Frame ID** • For receive message buffers and receive FIFOs, this field provides the value of the *Frame ID* field of the first valid frame received on the FlexRay bus in the slot indicated by the CYCCNT field. • For transmit message buffers, this field provides the value that will be transmitted in the *Frame ID* field of the FlexRay frame. **Note:** For transmit message buffers, the application must program this field to the same value as in the corresponding Message Buffer Frame ID Registers (MBFIDRn). If the FlexRay block detects a mismatch while transmitting the frame header, it will set the frame ID error flag FID_EF in the CHI Error Flag Register (CHIERFR). The value of the FID field will be ignored and replaced by the value provided in the Message Buffer Frame ID Registers (MBFIDRn).

Table 13-85. Frame Header Field Descriptions (continued)

Field	Description
CYCCNT	**Cycle Count** • For receive message buffer and receive FIFOs, this field provides the number of the communication cycle in which the frame stored in this message buffer was received. • For transmit message buffers, the value of this field is ignored. The FlexRay block will overwrite this value with the current cycle count value when it transmits the frame.
PLDLEN	**Payload Length in 16-Bit Units** • For receive message buffers and receive FIFOs, this field provides the value of the payload length field of the first valid frame received on the FlexRay bus in the slot indicated by the FID field. • For transmit message buffers assigned to the static segment, this value is ignored for the frame generation. The FlexRay block uses the value in the PCR19.paylaod_length_static to set the value of the *Payload length* field in the transmitted frame. • For transmit message buffers assigned to the dynamic segment, this value is used to set the value of the *Payload length* field in the transmitted frame. **Note:** The value of this field is given in numbers of 16-bit units
HDCRC	**Header CRC** • For receive and FIFO message buffers, this field provides the value of the *Header CRC* of the received frame. • For transmit message buffers, this field provides the *Header CRC* value as it was given by the application.The FlexRay block transmits this value in the *Header CRC* field of the transmitted frame.

13.6.5.2.2 Data Field Offset Description

Data Field Offset Content

For a detailed description of the Data Field Offset, see Section 13.6.2.1.2, “Data Field Offset”.

Data Field Offset Access

The application shall program the Data Field Offset when configuring the message buffers either in the *POC:config* state or when the message buffer is disabled.

13.6.5.2.3 Slot Status Description

The slot status is a read-only structure for the application and a write-only structure for the FlexRay block. The meaning and content of the slot status in the message buffer header field depends on the message buffer type.

Receive Message Buffer and Receive FIFO Slot Status Description

This section describes the slot status structure for the individual receive message buffers and receive FIFOs. The content of the slot status structure for receive message buffers depends on the message buffer type and on the channel assignment for individual receive message buffers as given by Table 13-85.

Table 13-86. Receive Message Buffer Slot Status Content

Receive Message Buffer Type	Slot Status Content
Individual Receive Message Buffer assigned to both channels MBCCSRn.CHA=1 and MBCCSRn.CHB=1	see Figure 13-106
Individual Receive Message Buffer assigned to channel A MBCCSRn.CHA=1 and MBCCSRn.CHB=0	see Figure 13-107
Individual Receive Message Buffer assigned to channel B MBCCSRn.CHA=0 and MBCCSRn.CHB=1	see Figure 13-108
Receive FIFO Channel A Message Buffer	see Figure 13-107
Receive FIFO Channel B Message Buffer	see Figure 13-108

The meaning of the bits in the slot status structure is explained in Table 13-86.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VFB	SYB	NFB	SUB	SEB	CEB	BVB	CH	VFA	SYA	NFA	SUA	SEA	CEA	BVA	0
Reset	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Figure 13-106. Receive Message Buffer Slot Status Structure (ChAB)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	VFA	SYA	NFA	SUA	SEA	CEA	BVA	0
Reset	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Figure 13-107. Receive Message Buffer Slot Status Structure (ChA)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VFB	SYB	NFB	SUB	SEB	CEB	BVB	1	0	0	0	0	0	0	0	0
Reset	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Figure 13-108. Receive Message Buffer Slot Status Structure (ChB)

Table 13-87. Receive Message Buffer Slot Status Field Descriptions

Field	Description
Common Message Buffer Status Bits	
15 VFB	**Valid Frame on Channel B** — protocol related variable: *vSS!ValidFrame* channel B 0 *vSS!ValidFrame* = 0 1 *vSS!ValidFrame* = 1
14 SYB	**Sync Frame Indicator Channel B** — protocol related variable: *vRF!Header!SyFIndicator* channel B 0 *vRF!Header!SyFIndicator* = 0 1 *vRF!Header!SyFIndicator* = 1
13 NFB	**Null Frame Indicator Channel B** — protocol related variable: *vRF!Header!NFIndicator* channel B 0 *vRF!Header!NFIndicator* = 0 1 *vRF!Header!NFIndicator* = 1
12 SUB	**Startup Frame Indicator Channel B** — protocol related variable: *vRF!Header!SuFIndicator* channel B 0 *vRF!Header!SuFIndicator* = 0 1 *vRF!Header!SuFIndicator* = 1

Table 13-87. Receive Message Buffer Slot Status Field Descriptions (continued)

Field	Description
11 SEB	**Syntax Error on Channel B** — protocol related variable: *vSS!SyntaxError* channel B 0 *vSS!SyntaxError* = 0 1 *vSS!SyntaxError* = 1
10 CEB	**Content Error on Channel B** — protocol related variable: *vSS!ContentError* channel B 0 *vSS!ContentError* = 0 1 *vSS!ContentError* = 1
9 BVB	**Boundary Violation on Channel B** — protocol related variable: *vSS!BViolation* channel B 0 *vSS!BViolation* = 0 1 *vSS!BViolation* = 1
8 CH	**Channel first valid received** — This status bit applies only to receive message buffers assigned to the static segment and to both channels. It indicates the channel that has received the *first valid* frame in the slot. This flag is set to 0 if no valid frame was received at all in the subscribed slot. 0 first valid frame received on channel A, or no valid frame received at all 0 first valid frame received on channel B
7 VFA	**Valid Frame on Channel A** — protocol related variable: *vSS!ValidFrame* channel A 0 *vSS!ValidFrame* = 0 1 *vSS!ValidFrame* = 1
6 SYA	**Sync Frame Indicator Channel A** — protocol related variable: *vRF!Header!SyFIndicator* channel A 0 *vRF!Header!SyFIndicator* = 0 1 *vRF!Header!SyFIndicator* = 1
5 NFA	**Null Frame Indicator Channel A** — protocol related variable: *vRF!Header!NFIndicator* channel A 0 *vRF!Header!NFIndicator* = 0 1 *vRF!Header!NFIndicator* = 1
4 SUA	**Startup Frame Indicator Channel A** — protocol related variable: *vRF!Header!SuFIndicator* channel A 0 *vRF!Header!SuFIndicator* = 0 1 *vRF!Header!SuFIndicator* = 1
3 SEA	**Syntax Error on Channel A** — protocol related variable: *vSS!SyntaxError* channel A 0 *vSS!SyntaxError* = 0 1 *vSS!SyntaxError* = 1
2 CEA	**Content Error on Channel A** — protocol related variable: *vSS!ContentError* channel A 0 *vSS!ContentError* = 0 1 *vSS!ContentError* = 1
1 BVA	**Boundary Violation on Channel A** — protocol related variable: *vSS!BViolation* channel A 0 *vSS!BViolation* = 0 1 *vSS!BViolation* = 1

Transmit Message Buffer Slot Status Description

This section describes the slot status structure for transmit message buffers. Only the TCA and TCB status bits are directly related to the transmission process. All other status bits in this structure are related to a receive process that may have occurred. The content of the slot status structure for transmit message buffers depends on the channel assignment as given by Table 13-87.

Table 13-88. Transmit Message Buffer Slot Status Content

Transmit Message Buffer Type	Slot Status Content
Individual Transmit Message Buffer assigned to both channels MBCCSRn.CHA=1 and MBCCSRn.CHB=1	see Figure 13-109
Individual Transmit Message Buffer assigned to channel A MBCCSRn.CHA=1 and MBCCSRn.CHB=0	see Figure 13-110
Individual Transmit Message Buffer assigned to channel B MBCCSRn.CHA=0 and MBCCSRn.CHB=1	see Figure 13-111

The meaning of the bits in the slot status structure is described in Table 13-86.

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VFB	SYB	NFB	SUB	SEB	CEB	BVB	TCB	VFA	SYA	NFA	SUA	SEA	CEA	BVA	TCA
Reset	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Figure 13-109. Transmit Message Buffer Slot Status Structure (ChAB)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	VFA	SYA	NFA	SUA	SEA	CEA	BVA	TCA
Reset	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Figure 13-110. Transmit Message Buffer Slot Status Structure (ChA)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VFB	SYB	NFB	SUB	SEB	CEB	BVB	TCB	0	0	0	0	0	0	0	0
Reset	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–

Figure 13-111. Transmit Message Buffer Slot Status Structure (ChB)

Table 13-89. Transmit Message Buffer Slot Status Structure Field Descriptions

Field	Description
15 VFB	**Valid Frame on Channel B** — protocol related variable: *vSS!ValidFrame* channel B 0 *vSS!ValidFrame* = 0 1 *vSS!ValidFrame* = 1
14 SYB	**Sync Frame Indicator Channel B** — protocol related variable: *vRF!Header!SyFIndicator* channel B 0 *vRF!Header!SyFIndicator* = 0 1 *vRF!Header!SyFIndicator* = 1
13 NFB	**Null Frame Indicator Channel B** — protocol related variable: *vRF!Header!NFIndicator* channel B 0 *vRF!Header!NFIndicator* = 0 1 *vRF!Header!NFIndicator* = 1
12 SUB	**Startup Frame Indicator Channel B** — protocol related variable: *vRF!Header!SuFIndicator* channel B 0 *vRF!Header!SuFIndicator* = 0 1 *vRF!Header!SuFIndicator* = 1
11 SEB	**Syntax Error on Channel B** — protocol related variable: *vSS!SyntaxError* channel B 0 *vSS!SyntaxError* = 0 1 *vSS!SyntaxError* = 1
10 CEB	**Content Error on Channel B** — protocol related variable: *vSS!ContentError* channel B 0 *vSS!ContentError* = 0 1 *vSS!ContentError* = 1

Table 13-89. Transmit Message Buffer Slot Status Structure Field Descriptions (continued)

Field	Description
9 BVB	**Boundary Violation on Channel B** — protocol related variable: *vSS!BViolation* channel B 0 *vSS!BViolation* = 0 1 *vSS!BViolation* = 1
8 TCB	**Transmission Conflict on Channel B** — protocol related variable: *vSS!TxConflict* channel B 0 *vSS!TxConflict* = 0 1 *vSS!TxConflict* = 1
7 VFA	**Valid Frame on Channel A** — protocol related variable: *vSS!ValidFrame* channel A 0 *vSS!ValidFrame* = 0 1 *vSS!ValidFrame* = 1
6 SYA	**Sync Frame Indicator Channel A** — protocol related variable: *vRF!Header!SyFIndicator* channel A 0 *vRF!Header!SyFIndicator* = 0 1 *vRF!Header!SyFIndicator* = 1
5 NFA	**Null Frame Indicator Channel A** — protocol related variable: *vRF!Header!NFIndicator* channel A 0 *vRF!Header!NFIndicator* = 0 1 *vRF!Header!NFIndicator* = 1
4 SUA	**Startup Frame Indicator Channel A** — protocol related variable: *vRF!Header!SuFIndicator* channel A 0 *vRF!Header!SuFIndicator* = 0 1 *vRF!Header!SuFIndicator* = 1
3 SEA	**Syntax Error on Channel A** — protocol related variable: *vSS!SyntaxError* channel A 0 *vSS!SyntaxError* = 0 1 *vSS!SyntaxError* = 1
2 CEA	**Content Error on Channel A** — protocol related variable: *vSS!ContentError* channel A 0 *vSS!ContentError* = 0 1 *vSS!ContentError* = 1
1 BVA	**Boundary Violation on Channel A** — protocol related variable: *vSS!BViolation* channel A 0 *vSS!BViolation* = 0 1 *vSS!BViolation* = 1
0 TCA	**Transmission Conflict on Channel A** — protocol related variable: *vSS!TxConflict* channel A 0 *vSS!TxConflict* = 0 1 *vSS!TxConflict* = 1

13.6.5.3 Message Buffer Data Field Description

The message buffer data field is used to store the frame payload data, or a part of it, of the frame to be transmitted to or received from the FlexRay bus. The minimum required length of this field depends on the message buffer type that the physical message buffer is assigned to and is given in Table 13-89. The structure of the message buffer data field is given in Figure 13-112.

Table 13-90. Message Buffer Data Field Minimum Length

physical message buffer assigned to	minimum length defined by
Individual Message Buffer in Segment 1	MBDSR.MBSEG1DS
Receive Shadow Buffer in Segment 1	MBDSR.MBSEG1DS
Individual Message Buffer in Segment 2	MBDSR.MBSEG2DS
Receive Shadow Buffer in Segment 2	MBDSR.MBSEG2DS
Receive FIFO for channel A	RFDSR.ENTRY_SIZE (RFSR.SEL = 0)

Table 13-90. Message Buffer Data Field Minimum Length

physical message buffer assigned to	minimum length defined by
Receive FIFO for channel B	RFDSR.ENTRY_SIZE (RFSR.SEL = 1)

NOTE

The FlexRay block will not access any locations outside the message buffer data field boundaries given by Table 13-89.

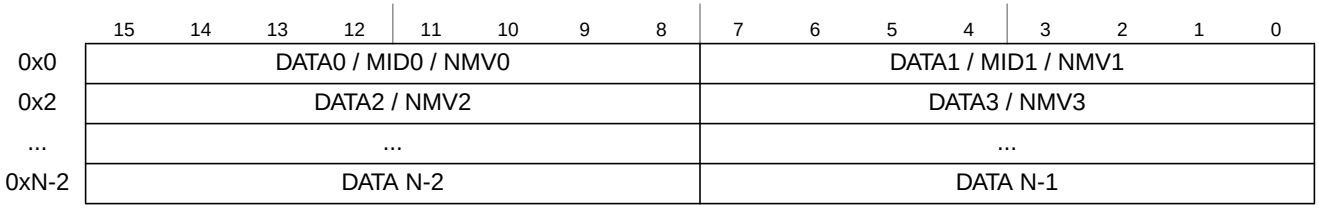


Figure 13-112. Message Buffer Data Field Structure

The message buffer data field is located in the FRM; thus, the FlexRay block has no means to control application write access to the field. To ensure data consistency, the application must follow a write and read access scheme.

13.6.5.3.1 Message Buffer Data Field Read Access

For transmit message buffers, the FlexRay block will not modify the content of the Message Buffer Data Field. Thus the application can read back the data at any time without any impact on data consistency.

For receive message buffers the application must lock the related receive message buffer and retrieve the message buffer header index from the Message Buffer Index Registers (MBIDXRn). While the message buffer is locked, the FlexRay block will not update the Message Buffer Data Field.

For receive FIFOs, the application can read the message buffer indicated by the Receive FIFO A Read Index Register (RFARIR) or the Receive FIFO B Read Index Register (RFBRIR) when the related receive FIFO non-empty interrupt flag FNEAIF or FNEBIF is set in the Global Interrupt Flag and Enable Register (GIFER). While the non-empty interrupt flag is set, the FlexRay block will not update the Message Buffer Data Field related to message buffer indicated by Receive FIFO A Read Index Register (RFARIR) or the Receive FIFO B Read Index Register (RFBRIR).

13.6.5.3.2 Message Buffer Data Field Write Access

For receive message buffers, receive shadow buffers, and receive FIFOs, the application must not write to the message buffer data field.

For transmit message buffers, the application must follow the write access restrictions given in Table 13-90.

Table 13-91. Frame Data Write Access Constraints

Field	single buffered	double buffered	
		commit side	transmit side
DATA, MID, NMV	*POC:config* or MB_DIS or MB_LCK	*POC:config* or MB_DIS or MB_LCK	*POC:config* or MB_DIS

Table 13-92. Frame Data Field Descriptions

Field	Description
DATA[0:N-1]	**Message Data** — Provides the message data received or to be transmitted. For receive message buffer and receive FIFOs, this field provides the message data received for this message buffer. For transmit message buffers, the field provides the message data to be transmitted.
MID[0:1]	**Message Identifier** — If the payload preamble bit PPI is set in the message buffer frame header, the MID field holds the message ID of a dynamic frame located in the message buffer. The receive FIFO filter uses the received message ID for message ID filtering.
NMV[0:11]	**Network Management Vector** — If the payload preamble bit PPI is set in the message buffer frame header, the network management vector field holds the network management vector of a static frame located in the message buffer. **Note:** The MID and NMV bytes replace the corresponding DATA bytes.

13.6.6 Individual Message Buffer Functional Description

The FlexRay block provides three basic types of individual message buffers:

1. Single Transmit Message Buffers
2. Double Transmit Message Buffers
3. Receive Message Buffers

Before an individual message buffer can be used, it must be configured by the application. After the initial configuration, the message buffer can be reconfigured later. The set of the configuration data for individual message buffers is given in Section 13.6.3.4.1, “Individual Message Buffer Configuration Data”.

13.6.6.1 Individual Message Buffer Configuration

The individual message buffer configuration consists of two steps. The first step is the allocation of the required amount of memory for the FRM. The second step is the programming of the message buffer configuration registers, which is described in this section.

13.6.6.1.1 Common Configuration Data

One part of the message buffer configuration data is common to all individual message buffers and the receive shadow buffers. These data can only be set when the protocol is in the *POC:config* state.

The application configures the number of utilized individual message buffers by writing the message buffer number of the last utilized message buffer into the LAST_MB_UTIL field in the Message Buffer Segment Size and Utilization Register (MBSSUTR).

The application configures the size of the two segments of individual message buffers by writing the message buffer number of the last message buffer in the first segment into the LAST_MB_SEG1 field in the Message Buffer Segment Size and Utilization Register (MBSSUTR)

The application configures the length of the message buffer data fields for both of the message buffer segments by writing to the MBSEG2DS and MBSEG1DS fields in the Message Buffer Data Size Register (MBDSR).

Depending on the current receive functionality of the FlexRay block, the application must configure the receive shadow buffers. For each segment and for each channel with at least one individual receive message buffer assigned, the application must configure the related receive shadow buffer using the Receive Shadow Buffer Index Register (RSBIR).

13.6.6.1.2 Specific Configuration Data

The second part of the message buffer configuration data is specific for each message buffer.

These data can be changed only when either

- the protocol is in the *POC:config* state or
- the message buffer is disabled, i.e. MBCCSRn.EDS = 0

The individual message buffer type is defined by the MTD and MBT bits in the Message Buffer Configuration, Control, Status Registers (MBCCSRn) as given in Table 13-92.

Table 13-93. Individual Message Buffer Types

MBCCSRn.MTD	MBCCSRn.MBT	Individual Message Buffer Description
0	0	Receive Message Buffer
0	1	Reserved
1	0	Single Transmit Message Buffer
1	1	Double Transmit Message Buffer

The message buffer specific configuration data are

1. MCM, MBT, MTD bits in Message Buffer Configuration, Control, Status Registers (MBCCSRn)
2. all fields and bits in Message Buffer Cycle Counter Filter Registers (MBCCFRn)
3. all fields and bits in Message Buffer Frame ID Registers (MBFIDRn)
4. all fields and bits in Message Buffer Index Registers (MBIDXRn)

The meaning of the specific configuration data depends on the message buffer type, as given in the detailed message buffer type descriptions Section 13.6.6.2, “Single Transmit Message Buffers”, Section 13.6.6.3, “Receive Message Buffers”, and Section 13.6.6.4, “Double Transmit Message Buffer”.

13.6.6.2 Single Transmit Message Buffers

The section provides a detailed description of the functionality of single buffered transmit message buffers.

A single transmit message buffer is used by the application to provide message data to the FlexRay block that will be transmitted over the FlexRay Bus. The FlexRay block uses the transmit message buffers to

provide information about the transmission process and status information about the slot in which message was transmitted.

The individual message buffer with message buffer number *n* is configured to be a single transmit message buffer by the following settings:

- MBCCSRn.MBT = 0 (single buffered message buffer)
- MBCCSRn.MTD = 1 (transmit message buffer)

13.6.6.2.1 Access Regions

To certain message buffer fields, both the application and the FlexRay block have access. To ensure data consistency, a message buffer locking scheme is implemented, which is used to control the access to the data, control, and status bits of a message buffer. The access regions for single transmit message buffers are depicted in Figure 13-113. A description of the regions is given in Table 13-93. If an region is active as indicated in Table 13-94, the access scheme given for that region applies to the message buffer.

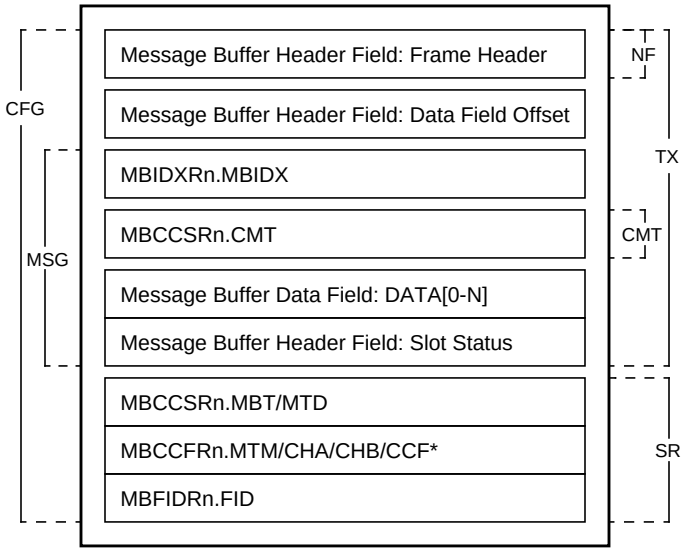


Figure 13-113. Single Transmit Message Buffer Access Regions

Table 13-94. Single Transmit Message Buffer Access Regions Description

Region	Access from		Region used for
	Application	Module	
CFG	read/write	-	Message Buffer Configuration
MSG	read/write	-	Message Data and Slot Status Access
NF	-	read-only	Message Header Access for Null Frame Transmission
TX	-	read/write	Message Transmission and Slot Status Update
CM	-	read-only	Message Buffer Validation
SR	-	read-only	Message Buffer Search

The trigger bits MBCCSRn.EDT and MBCCSRn.LCKT, and the interrupt enable bit MBCCSRn.MBIE are not under access control and can be accessed from the application at any time. The status bits

MBCCSRn.EDS and MBCCSRn.LCKS are not under access control and can be accessed from the FlexRay block at any time.

The interrupt flag MBCCSnR.MBIF is not under access control and can be accessed from the application and the FlexRay block at any time. FlexRay block clear access has higher priority.

The FlexRay block restricts its access to the regions depending on the current state of the message buffer. The application must adhere to these restrictions in order to ensure data consistency. The transmit message buffer states are given in Figure 13-114. A description of the states is given in Table 13-94, which also provides the access scheme for the access regions.

The status bits MBCCSRn.EDS and MBCCSRn.LCKS provide the application with the required message buffer status information. The internal status information is not visible to the application.

13.6.6.2.2 Message Buffer States

This section describes the transmit message buffer states and provides a state diagram.

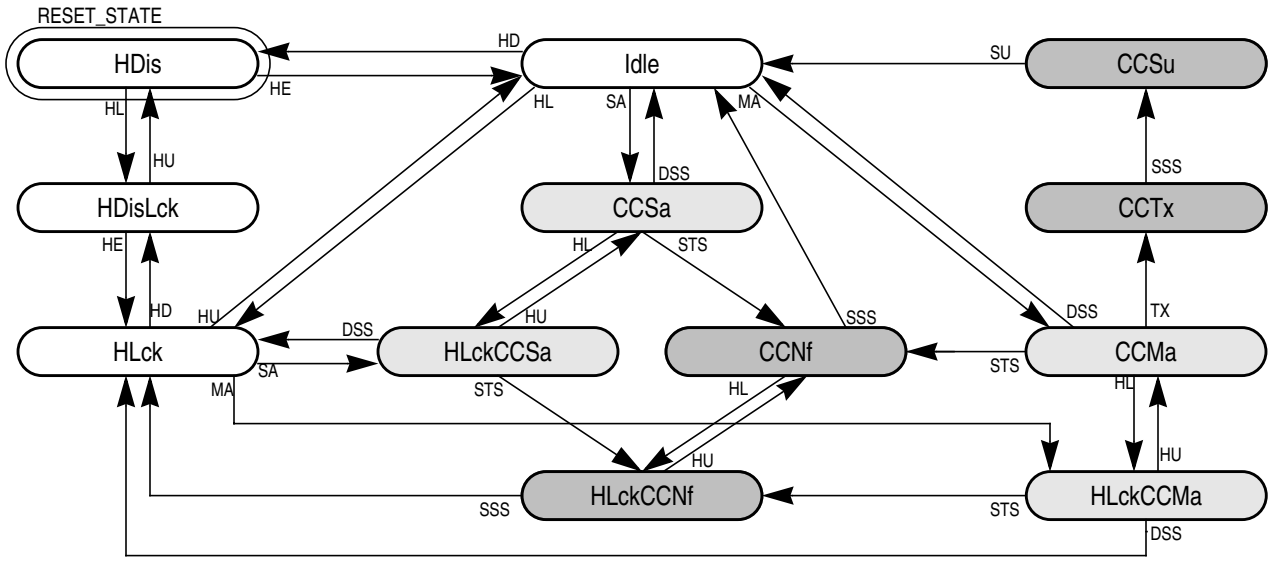


Figure 13-114. Single Transmit Message Buffer States

Table 13-95. Single Transmit Message Buffer State Description (Sheet 1 of 2)

State	MBCCSRn		Access Region		Description
	EDS	LCKS	Appl.	Module	
Idle	1	0	–	CM, SR	**Idle** - Message Buffer is idle. Included in message buffer search.
HDis	0	0	CFG	–	**Disabled** - Message Buffer under configuration. Excluded from message buffer search.
HDisLck	0	1	CFG	–	**Disabled and Locked** - Message Buffer under configuration. Excluded from message buffer search.
HLck	1	1	MSG	SR	**Locked** - Applications access to data, control, and status. Included in message buffer search.

Table 13-95. Single Transmit Message Buffer State Description (Sheet 2 of 2)

State	MBCCSRn		Access Region		Description
	EDS	LCKS	Appl.	Module	
CCSa	1	0	–	–	**Slot Assigned** - Message buffer assigned to next static slot. Ready for Null Frame transmission.
HLckCCSa	1	1	MSG	–	**Locked and Slot Assigned** - Applications access to data, control, and status.Message buffer assigned to next static slot
CCNf	1	0	–	NF	**Null Frame Transmission** Header is used for null frame transmission.
HLckCCNf	1	1	MSG	NF	**Locked and Null Frame Transmission** - Applications access to data, control, and status. Header is used for null frame transmission.
CCMa	1	0	–	CM	**Message Available** - Message buffer is assigned to next slot and cycle counter filter matches.
HLckCCMa	1	1	MSG	–	**Locked and Message Available** - Applications access to data, control, and status. Message buffer is assigned to next slot and cycle counter filter matches.
CCTx	1	0	–	TX	**Message Transmission** - Message buffer data transmit. Payload data from buffer transmitted
CCSu	1	0	–	TX	**Status Update** - Message buffer status update. Update of status flags, the slot status field, and the header index.

13.6.6.2.3 Message Buffer Transitions

Application Transitions

The application transitions can be triggered by the application using the commands described in Table 13-95. The application issues the commands by writing to the Message Buffer Configuration, Control, Status Registers (MBCCSRn). Only one command can be issued with one write access. Each command is executed immediately. If the command is ignored, it must be issued again.

The enable and disable commands issued by writing 1 to the trigger bit MBCCSRn.EDT. The transition that will be triggered by each of these command depends on the current value of the status bit MBCCSRn.EDS. If the command triggers the disable transition HD and the message buffer is in one of the states CCSa, HLckCCSa, CCMa, HLckCCMa, CCNf, HLckCCNf, or CCTx, the disable transition has no effect (command is ignored) and the message buffer state is not changed. No notification is given to the application.

The lock and unlock commands issued by writing 1 to the trigger bit MBCCSRn.LCKT. The transition that will be triggered by each of these commands depends on the current value of the status bit MBCCSRn.LCKS. If the command triggers the lock transition HL and the message buffer is in the state CCTx, the lock transition has no effect (command is ignored) and message buffer state is not changed. In this case, the message buffer lock error flag LCK_EF in the CHI Error Flag Register (CHIERFR) is set.

Table 13-96. Single Transmit Message Buffer Application Transitions

Transition	Command	Condition	Description
HE	MBCCSRn.EDT:= 1	MBCCSRn.EDS = 0	Application triggers message buffer enable.
HD		MBCCSRn.EDS = 1	Application triggers message buffer disable.

Table 13-96. Single Transmit Message Buffer Application Transitions

Transition	Command	Condition	Description
HL	MBCCSRn.LCKT:= 1	MBCCSRn.LCKS = 0	Application triggers message buffer lock.
HU		MBCCSRn.LCKS = 1	Application triggers message buffer unlock.

Module Transitions

The module transitions that can be triggered by the FlexRay block are described in Table 13-96. Each transition will be triggered for certain message buffers when the related condition is fulfilled.

Table 13-97. Single Transmit Message Buffer Module Transitions

Transition	Condition	Description
SA	slot match and static slot	**Slot Assigned** - Message buffer is assigned to next static slot.
MA	slot match and CycleCounter match	**Message Available** - Message buffer is assigned to next slot and cycle counter filter matches.
TX	slot start and MBCCSRn.CMT = 1	**Transmission Slot Start** - Slot Start and commit bit CMT is set. In case of a dynamic slot, pLatestTx is not exceeded.
SU	status updated	**Status Updated** - Slot Status field and message buffer status flags updated. Interrupt flag set.
STS	static slot start	**Static Slot Start** - Start of static slot.
DSS	dynamic slot start or symbol window start or NIT start	**Dynamic Slot or Segment Start.** - Start of dynamic slot or symbol window or NIT.
SSS	slot start or symbol window start or NIT start	**Slot or Segment Start** - Start of static slot or dynamic slot or symbol window or NIT.

Transition Priorities

The application can trigger only one transition at a time. There is no need to specify priorities among them.

As shown in the first part of Table 13-97, the module transitions have a higher priority than the application transitions. For all states except the CCMa state, both a lock/unlock transition HL/HD and a module transition can be executed at the same time. The result state is reached by first applying the application transition and subsequently the module transition to the intermediately reached state. For example, if the message buffer is in the HLck state and the application unlocks the message buffer by the HU transition and the module triggers the slot assigned transition SA, the intermediate state is Idle and the resulting state is CCSa.

The priorities among the module transitions is given in the second part of Table 13-97.

Table 13-98. Single Transmit Message Buffer Transition Priorities

State	Priorities	Description
module vs. application		
Idle, HLck	SA > HD MA > HD	Slot Assigned > Message Buffer Disable Message Available > Message Buffer Disable
CCMa	TX > HL	Transmission Start > Message Buffer Lock

Table 13-98. Single Transmit Message Buffer Transition Priorities

State	Priorities	Description
module internal		
Idle, HLck	MA > SA	Message Available > Slot Assigned
CCMa	TX > STS TX > DSS	Transmission Slot Start > Static Slot Start Transmission Slot Start > Dynamic Slot Start

13.6.6.2.4 Transmit Message Setup

To transmit a message over the FlexRay bus, the application writes the message data into the message buffer data field and sets the commit bit CMT in the Message Buffer Configuration, Control, Status Registers (MBCCSRn). The physical access to the message buffer data field is described in Section 13.6.3.1, “Individual Message Buffers”.

As indicated by Table 13-94, the application shall write to the message buffer data field and change the commit bit CMT only if the transmit message buffer is in one of the states HDis, HDisLck, HLck, HLckCCSa, HLckCCMa, or HLckCCMa. The application can change the state of a message buffer if it issues the appropriate commands shown in Table 13-95. The state change is indicated through the MBCCSRn.EDS and MBCCSRn.LCKS status bits.

If the transmit message buffer enters one of the states HDis, HDisLck, HLck, HLckCCSa, HLckCCMa, or HLckCCMa the MBCCSRn.DVAL flag is negated.

13.6.6.2.5 Message Transmission

As a result of the message buffer search described in Section 13.6.7, “Individual Message Buffer Search”, the FlexRay block triggers the message available transition MA for up to two transmit message buffers. This changes the message buffer state from Idle to CCMa and the message buffers can be used for message transmission in the next slot.

The FlexRay block transmits a message from a message buffer if both of the following two conditions are fulfilled at the start of the transmission slot:

1. the message buffer is in the message available state CCMa
2. the message data are still valid, i.e. MBCCSRn.CMT = 1

In this case, the FlexRay block triggers the TX transition and changes the message buffer state to CCTx. A transmit message buffer timing and state change diagram for message transmission is given in Figure 13-115. In this example, the message buffer with message buffer number n is Idle at the start of the search slot, matches the slot and cycle number of the next slot, and message buffer data are valid, i.e. MBCCSRn.CMT = 1.

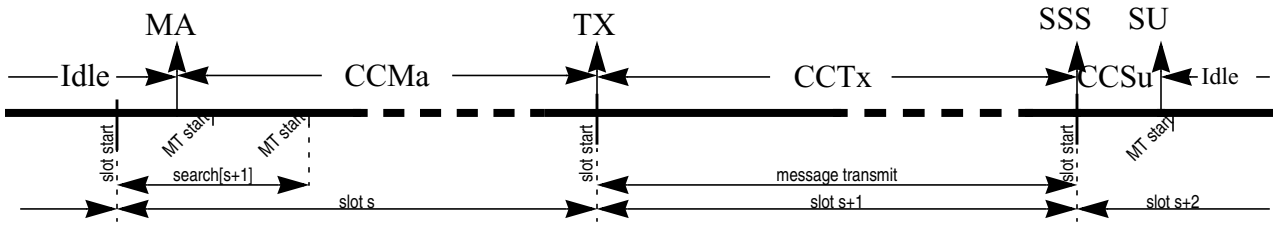


Figure 13-115. Message Transmission Timing

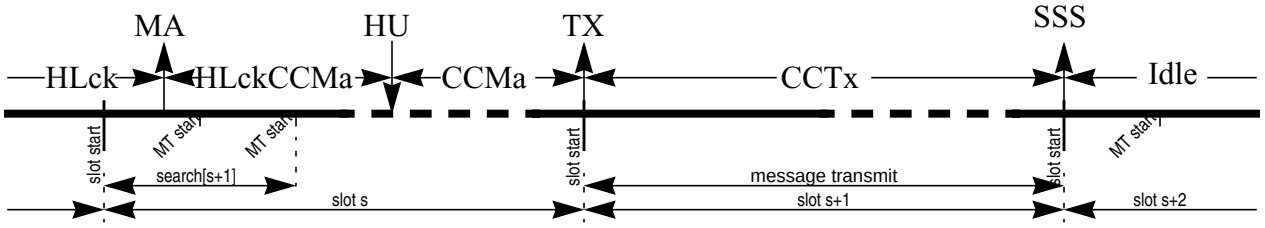


Figure 13-116. Message Transmission from HLck state with unlock

The amount of message data read from the FRM and transferred to the FlexRay bus is determined by the following three items

1. the message buffer segment that the message buffer is assigned to, as defined by the Message Buffer Segment Size and Utilization Register (MBSSUTR).
2. the message buffer data field size, as defined by the related field of the Message Buffer Data Size Register (MBDSR)
3. the value of the PLDLEN field in the message buffer header field, as described in Section 13.6.5.2.1, “Frame Header Section Description”

If a message buffer is assigned to message buffer segment 1, and PLDLEN > MBSEG1DS, then 2 * MBSEG1DS bytes will be read from the message buffer data field and zero padding is used for the remaining bytes for the FlexRay bus transfer. If PLDLEN <= MBSEG1DS, the FlexRay block reads and transfers 2*PLDLEN bytes. The same holds for segment 2 and MBSEG2DS.

13.6.6.2.6 Null Frame Transmission

A static slot with slot number S is assigned to the FlexRay block for channel A, if at least one transmit message buffer is configured with the MBFIDRn.FID set to S and MBCCFRn.CHA set to 1. A Null Frame is transmitted in the static slot S on channel A, if this slot is assigned to the FlexRay block for channel A, and all transmit message buffers with MBFIDRn.FID = s and MBCCFRn.CHA = 1 are either not committed, i.e MBCCSRn.CMT = 0, or locked by the application, i.e. MBCCSRn.LCKS = 1, or the cycle counter filter is enabled and does not match.

Additionally, the application can clear the commit bit of a message buffer that is in the CCMa state, which is called *uncommit* or *transmit abort*. This message buffer will be used for null frame transmission.

As a result of the message buffer search described in Section 13.6.7, “Individual Message Buffer Search”, the FlexRay block triggers the slot assigned transition SA for up to two transmit message buffers if at least

one of the conditions mentioned above is fulfilled for these message buffers. The transition SA changes the message buffer states from either Idle to CCSa or from HLck to HLckCCSa. In each case, these message buffers will be used for null frame transmission in the next slot. A message buffer timing and state change diagram for null frame transmission from Idle state is given in Figure 13-117.

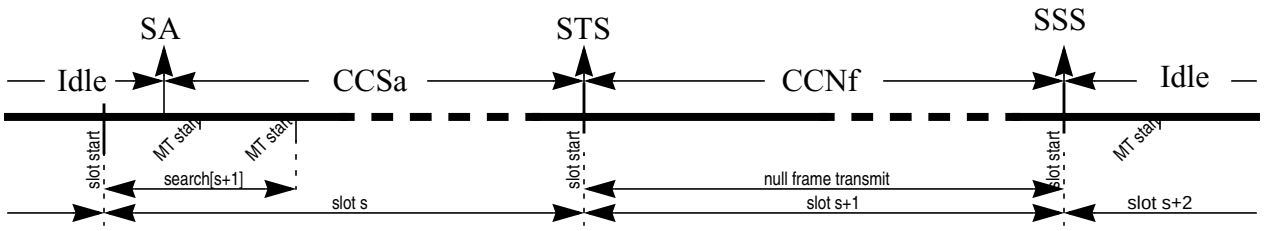


Figure 13-117. Null Frame Transmission from Idle state

A message buffer timing and state change diagram for null frame transmission from HLck state is given in Figure 13-118.

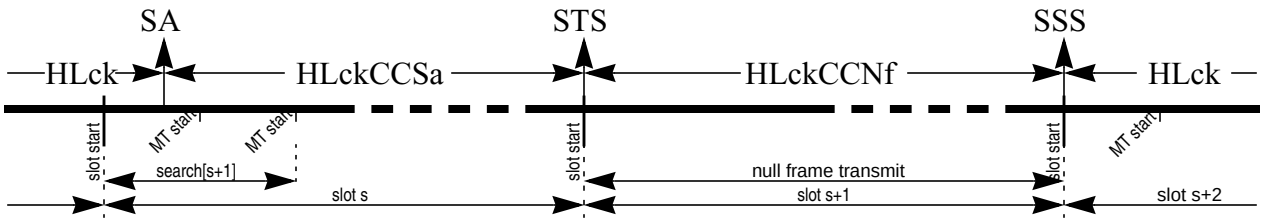


Figure 13-118. Null Frame Transmission from HLck state

If a transmit message buffer is in the CCSa or HLckCCSa state at the start of the transmission slot, a null frame is transmitted in any case, even if the message buffer is unlocked or committed before the transmission slot starts. A transmit message buffer timing and state change diagram for null frame transmission for this case is given in Figure 13-119.

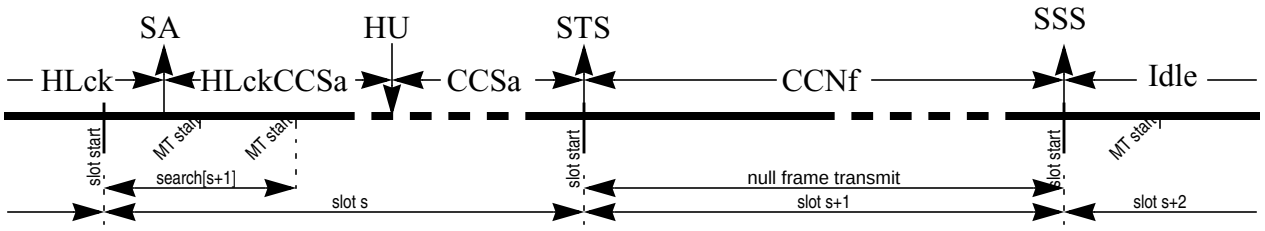


Figure 13-119. Null Frame Transmission from HLck state with unlock

Since the null frame transmission will not use the message buffer data, the application can lock/unlock the message buffer during null frame transmission. A transmit message buffer timing and state change diagram for null frame transmission for this case is given in Figure 13-120.

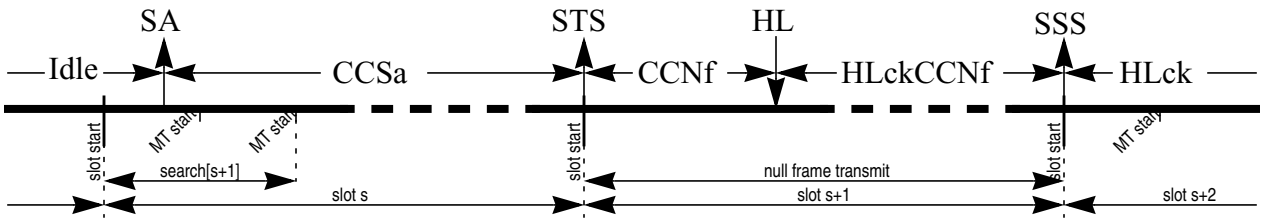


Figure 13-120. Null Frame Transmission from Idle state with locking

13.6.6.2.7 Message Buffer Status Update

After the end of each slot, the PE generates the slot status vector. Depending on the this status, the transmitted frame type, and the amount of transmitted data, the message buffer status is updated.

Message Buffer Status Update after Complete Message Transmission

The term complete message transmission refers to the fact that all payload data stored in the message buffer were send to FlexRay bus. In this case, the FlexRay block updates the slot status field of the message buffer and triggers the status updated transition SU. With the SU transition, the FlexRay block sets the message buffer interrupt flag MBCCSn.MBIF to indicate the successful message transmission.

Depending on the transmission mode flag MBCCFRn.MTM, the FlexRay block changes the commit flag MBCCSRn.CMT and the valid flag MBCCSRn.DVAL. If the MBCCFRn.MTM flag is negated, the message buffer is in the *event transmission mode.* In this case, each committed message is transmitted only once. The commit flag MBCCSRn.CMT is cleared with the SU transition. If the MBCCFRn.MTM flag is asserted, the message buffer is in the *state transmission mode*. In this case, each committed message is transmitted as long as the application provides new data or locks the message buffers. The FlexRay block will not clear the MBCCSRn.CMT flag at the end of transmission and will set the valid flag MBCCSRn.DVAL to indicate that the message will be transmitted again.

Message Buffer Status Update after Incomplete Message Transmission

The term incomplete message transmission refers to the fact that not all payload data that should be transmitted were send to FlexRay bus. This may be caused by the following regular conditions in the dynamic segment:

1. The transmission slot starts in a minislot with a minislot number greater than *pLatestTx*.
2. The transmission slot did not exist in the dynamic segment at all.

Additionally, an incomplete message transmission can be caused by internal communication errors. If those error occur, the Protocol Engine Communication Failure Interrupt Flag PECF_IF is set in the Protocol Interrupt Flag Register 1 (PIFR1).

In any of these two cases, the status of the message buffer is not changed at all with the SU transition. The slot status field is not updated, the status and control flags are not changed, and the interrupt flag is not set.

Message Buffer Status Update after Null Frame Transmission

After the transmission of a null frame, the status of the message buffer that was used for the null frame transmission is not changed at all. The slot status field is not updated, the status and control flags are not changed, and the interrupt flag is not set.

13.6.6.3 Receive Message Buffers

The section provides a detailed description of the functionality of the receive message buffers.

A receive message buffer is used to receive a message from the FlexRay Bus based on individual filter criteria. The FlexRay block uses the receive message buffer to provide the following data to the application

1. message data received
2. information about the reception process
3. status information about the slot in which the message was received

A individual message buffer with message buffer number *n* is configured as a receive message buffer by the following configuration settings

- MBCCSRn.MBT = 0 (single buffered message buffer)
- MBCCSRn.MTD = 0 (receive message buffer)

To certain message buffer fields, both the application and the FlexRay block have access. To ensure data consistency, a message buffer locking scheme is implemented that is used to control the access to the data, control, and status bits of a message buffer. The access regions for receive message buffers are depicted in Figure 13-121. A description of the regions is given in Table 13-98. If an region is active as indicated in Table 13-99, the access scheme given for that region applies to the message buffer.

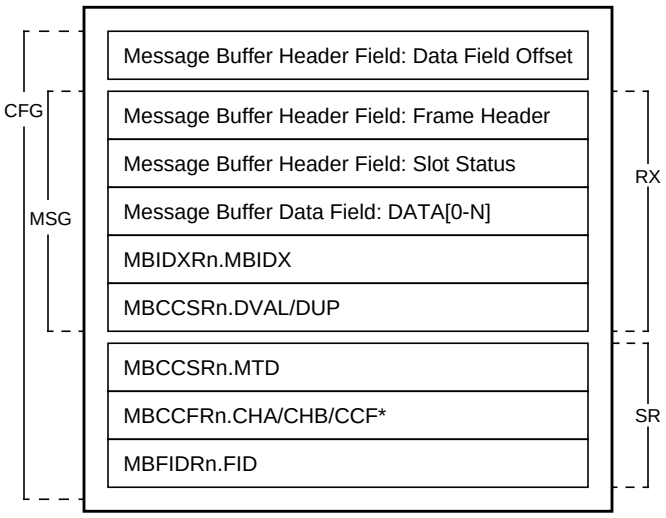


Figure 13-121. Receive Message Buffer Access Regions

Table 13-99. Receive Message Buffer Access Region Description

Region	Access from		Region used for
	Application	Module	
CFG	read/write	-	Message Buffer Configuration, Message Data and Status Access
MSG	read/write	-	Message Data, Header, and Status Access
RX	-	write-only	Message Reception and Status Update
SR	-	read-only	Message Buffer Search Data

The trigger bits MBCCSRn.EDT and MBCCSRn.LCKT and the interrupt enable bit MBCCSRn.MBIE are not under access control and can be accessed from the application at any time. The status bits MBCCSRn.EDS and MBCCSRn.LCKS are not under access control and can be accessed from the FlexRay block at any time.

The interrupt flag MBCCSRn.MBIF is not under access control and can be accessed from the application and the FlexRay block at any time. FlexRay block set access has higher priority.

The FlexRay block restricts its access to the regions depending on the current state of the message buffer. The application must adhere to these restrictions in order to ensure data consistency. The receive message buffer states are given in Figure 13-122. A description of the message buffer states is given in Table 13-94, which also provides the access scheme for the access regions.

The status bits MBCCSRn.EDS and MBCCSRn.LCKS provide the application with the required status information. The internal status information is not visible to the application.

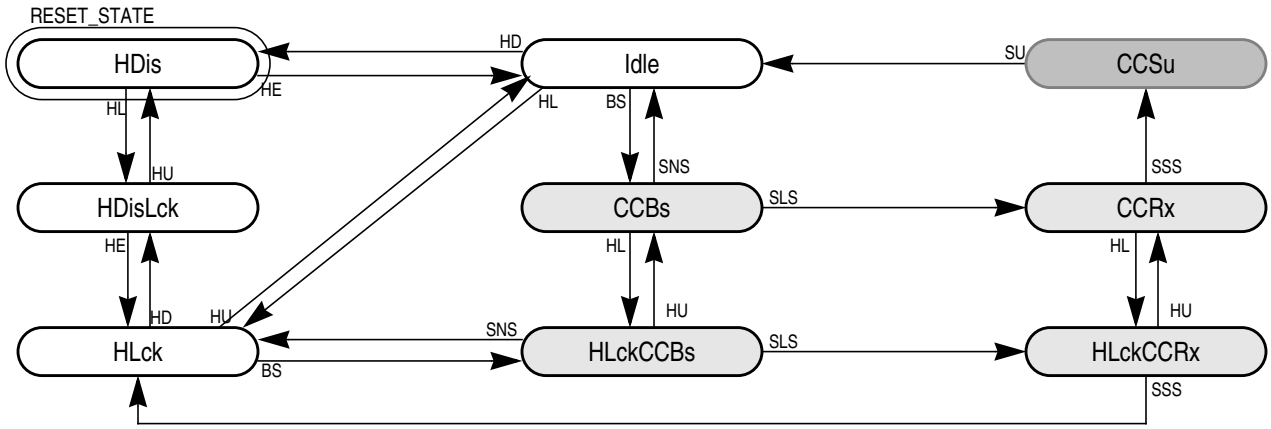


Figure 13-122. Receive Message Buffer States

Table 13-100. Receive Message Buffer States and Access (Sheet 1 of 2)

State	MBCCSRn		Access from		Description
	EDS	LCKS	Appl.	Module	
Idle	1	0	–	SR	**Idle** - Message Buffer is idle. Included in message buffer search.
HDis	0	0	CFG	–	**Disabled** - Message Buffer under configuration. Excluded from message buffer search.

Table 13-100. Receive Message Buffer States and Access (Sheet 2 of 2)

State	MBCCSRn		Access from		Description
	EDS	LCKS	Appl.	Module	
HDisLck	0	1	CFG	–	**Disabled and Locked** - Message Buffer under configuration. Excluded from message buffer search.
HLck	1	1	MSG	–	**Locked** - Applications access to data, control, and status. Included in message buffer search.
CCBs	1	0	–	–	**Buffer Subscribed** - Message buffer subscribed for reception. Filter matches next (slot, cycle, channel) tuple.
HLckCCBs	1	1	MSG	–	**Locked and Buffer Subscribed** - Applications access to data, control, and status. Message buffer subscribed for reception.
CCRx	1	0	–	–	**Message Receive** - Message data received into related shadow buffer.
HLckCCRx	1	1	MSG	–	**Locked and Message Receive** - Applications access to data, control, and status. Message data received into related shadow buffer.
CCSu	1	0	–	RX	**Status Update** - Message buffer status update. Update of status flags, the slot status field, and the header index.

13.6.6.3.1 Message Buffer Transitions

Application Transitions

The application transitions that can be triggered by the application using the commands described in Table 13-100. The application issues the commands by writing to the Message Buffer Configuration, Control, Status Registers (MBCCSRn). Only one command can be issued with one write access. Each command is executed immediately. If the command is ignored, it must be issued again.

The enable and disable commands issued by writing 1 to the trigger bit MBCCSRn.EDT. The transition that will be triggered by each of these command depends on the current value of the status bit MBCCSRn.EDS. If the command triggers the disable transition HD and the message buffer is in one of the states CCBs, HLckCCBs, or CCRx, the disable transition has no effect (command is ignored) and the message buffer state is not changed. No notification is given to the application.

The lock and unlock commands issued by writing 1 to the trigger bit MBCCSRn.LCKT. The transition that will be triggered by each of these commands depends on the current value of the status bit MBCCSRn.LCKS. If the command triggers the lock transition HL while the message buffer is in the state CCRx, the lock transition has no effect (command is ignored) and message buffer state is not changed. In this case, the message buffer lock error flag LCK_EF in the CHI Error Flag Register (CHIERFR) is set.

Table 13-101. Receive Message Buffer Application Transitions

Transition	Host Command	Condition	Description
HE	MBCCSRn.EDT:= 1	MBCCSRn.EDS = 0	Application triggers message buffer enable.
HD		MBCCSRn.EDS = 1	Application triggers message buffer disable.
HL	MBCCSRn.LCKT:= 1	MBCCSRn.LCKS = 0	Application triggers message buffer lock.
HU		MBCCSRn.LCKS = 1	Application triggers message buffer unlock.

Module Transitions

The module transitions that can be triggered by the FlexRay block are described in Table 13-101. Each transition will be triggered for certain message buffers when the related condition is fulfilled.

Table 13-102. Receive Message Buffer Module Transitions

Transition	Condition	Description
BS	slot match and CycleCounter match	**Buffer Subscribed** - The message buffer filter matches next slot and cycle.
SLS	slot start	**Slot Start** - Start of either Static Slot or Dynamic Slot.
SNS	symbol window start or NIT start	**Symbol Window or NIT Start** - Start of either Symbol Window or NIT.
SSS	slot start or symbol window start or NIT start	**Slot or Segment Start** - Start of either Static Slot, Dynamic Slot, Symbol Window, or NIT.
SU	status updated	**Status Updated** - Slot Status field, message buffer status flags, header index updated. Interrupt flag set.

Transition Priorities

The application can trigger only one transition at a time. There is no need to specify priorities among them.

As shown in Table 13-102, the module transitions have a higher priority than the application transitions. For all states except the CCRx state, a module transition and the application lock/unlock transition HL/HU and can be executed at the same time. The result state is reached by first applying the module transition and subsequently the application transition to the intermediately reached state. For example, if the message buffer is in the buffer subscribed state CCBs and the module triggers the slot start transition SLS at the same time as the application locks the message buffer by the HL transition, the intermediate state is CCRx and the resulting state is locked buffer subscribed state HLckCCRx.

Table 13-103. Receive Message Buffer Transition Priorities

State	Priorities	Description
module vs. application		
Idle	BS > HD	Buffer Subscribed > Message Buffer Disable
HLck	BS > HD	Buffer Subscribed > Message Buffer Disable
CCRx	SSS > HL	Slot or Segment Start > Message Buffer Lock

13.6.6.3.2 Message Reception

As a result of the message buffer search, the FlexRay block changes the state of up to two enabled receive message buffers from either idle state Idle or locked state HLck to the either subscribed state CCBs or locked buffer subscribed state HLckCCBs by triggering the buffer subscribed transition BS.

If the receive message buffers for the next slot are assigned to both channels, then at most one receive message buffer is changed to a buffer subscribed state.

If more than one matching message buffers assigned to a certain channel, then only the message buffer with the lowest message buffer number is in one of the states mentioned above.

With the start of the next static or dynamic slot the module trigger the slot start transition SLS. This changes the state of the subscribed receive message buffers from either CCBs to CCRx or from HLckCCBs to HLckCCRx, respectively.

During the reception slot, the received frame data are written into the shadow buffers. For details on receive shadow buffers, see Section 13.6.6.3.5, “Receive Shadow Buffers Concept”. The data and status of the receive message buffers that are the CCRx or HLckCCRx are not modified in the reception slot.

13.6.6.3.3 Message Buffer Status Update

With the start of the next static or dynamic slot or with the start of the symbol window or NIT, the module trigger the slot or segment start transition SSS. This transition changes the state of the receiving receive message buffers from either CCRx to CCSu or from HLckCCRx to HLck, respectively.

If a message buffer was in the locked state HLckCCRx, no update will be performed. The received data are lost. This is indicated by setting the Frame Lost Channel A/B Error Flag FRLA_EF/FRLB_EF in the CHI Error Flag Register (CHIERFR).

If a message buffer was in the CCRx state it is now in the CCSu state. After the evaluation of the slot status provided by the PE the message buffer is updated. The message buffer update depends on the slot status bits and the segment the message buffer is assigned to. This is described in Table 13-103.

Table 13-104. Receive Message Buffer Update

vSS!ValidFrame	*vRF!Header!NFIndicator*	**Update description**
1	1	**Valid non-null frame received.** - Message Buffer Data Field updated. - Frame Header Field updated. - Slot Status Field updated. - DUP:= 1 - DVAL:= 1 - MBIF:= 1
1	0	**Valid null frame received.** - Message Buffer Data Field *not* updated. - Frame Header Field *not* updated. - Slot Status Field updated. - DUP:= 0 - DVAL *not* changed - MBIF:= 1
0	x	**No valid frame received.** - Message Buffer Data Field not updated. - Frame Header Field not updated. - Slot Status Field updated. - DUP:= 0 - DVAL *not* changed. - MBIF:= 1, if the slot was not an empty dynamic slot. **Note:** An empty dynamic slot is indicated by the following frame and slot status bit values: *vSS!ValidFrame* = 0 and *vSS!SyntaxError* = 0 and *vSS!ContentError* = 0 and *vSS!BViolation* = 0.

NOTE

If the number of the last slot in the current communication cycle on a given channel is *n*, then all receive message buffers assigned to this channel with MBFIDRn.FID > *n* will not be updated at all.

When the receive message buffer update has finished the status updated transition SU is triggered, which changes the buffer state from CCSu to Idle. An example receive message buffer timing and state change diagram for a normal frame reception is given in Figure 13-123.

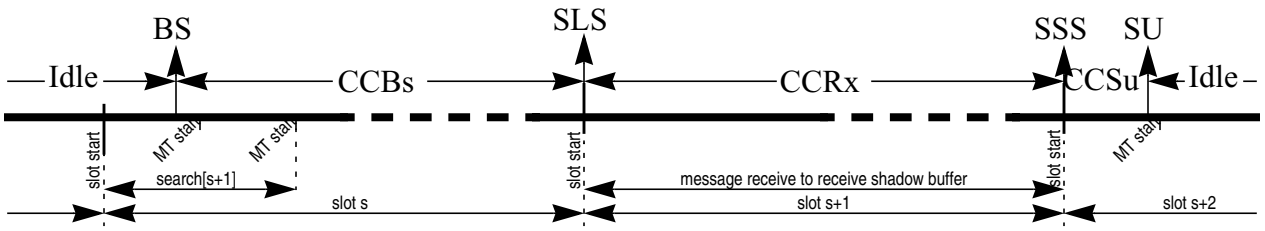


Figure 13-123. Message Reception Timing

The amount of message data written into the message buffer data field of the receive shadow buffer is determined by the following two items:

1. the message buffer segment that the message buffer is assigned to, as defined by the Message Buffer Segment Size and Utilization Register (MBSSUTR).
2. the message buffer data field size, as defined by the related field of the Message Buffer Data Size Register (MBDSR)
3. the number of bytes received over the FlexRay bus

If the message buffer is assigned to the message buffer segment 1, and the number of received bytes is greater than 2*MBDSR.MBSEG1DS, the FlexRay block writes only 2*MBDSR.MBSEG1DS bytes into the message buffer data field of the receive shadow buffer. If the number of received bytes is less than 2*MBDSR.MBSEG1DS, the FlexRay block writes only the received number of bytes and will not change the trailing bytes in the message buffer data field of the receive shadow buffer. The same holds for the message buffer segment 2 with MBDSR.MBSEG2DS.

13.6.6.3.4 Received Message Access

To access the message data received over the FlexRay bus, the application reads the message data stored in the message buffer data field of the corresponding receive message buffer. The access to the message buffer data field is described in Section 13.6.3.1, “Individual Message Buffers”.

The application can read the message buffer data field if the receive message buffer is one of the states HDis, HDisLck, or HLck. If the message buffer is in one of these states, the FlexRay block will not change the content of the message buffer.

13.6.6.3.5 Receive Shadow Buffers Concept

The receive shadow buffer concept applies only to individual receive message buffers. The intention of this concept is to ensure that only syntactically and semantically valid received non-null frames are presented

to the application in a receive message buffer. The basic structure of a receive shadow buffer is described in Section 13.6.3.2, “Receive Shadow Buffers”.

The receive shadow buffers temporarily store the received frame header and message data. After the slot boundary the slot status information is generated. If the slot status information indicates the reception of the valid non-null frame (see Table 13-103), the FlexRay block writes the slot status into the slot status field of the receive shadow buffer and exchanges the content of the Message Buffer Index Registers (MBIDXRn) with the content of the corresponding internal shadow buffer index register. In all other cases, the FlexRay block writes the slot status into the identified receive message buffer, depending on the slot status and the FlexRay segment the message buffer is assigned to.

The shadow buffer concept, with its index exchange, results in the fact that the FRM located message buffer associated to an individual receive message buffer changes after successful reception of a valid frame. This means that the message buffer area in the FRM accessed by the application for reading the received message is different from the initial setting of the message buffer. Therefore, the application must not rely on the index information written initially into the Message Buffer Index Registers (MBIDXRn). Instead, the index of the message buffer header field must be fetched from the Message Buffer Index Registers (MBIDXRn).

13.6.6.4 Double Transmit Message Buffer

The section provides a detailed description of the functionality of the double transmit message buffers.

Double transmit message buffers are used by the application to provide the FlexRay block with the message data to be transmitted over the FlexRay Bus. The FlexRay block uses this message buffer to provide information to the application about the transmission process, and status information about the slot in which message data was transmitted.

In contrast to the single transmit message buffers, the application can provide new transmission data while the transmission of the previously provided message data is running. This scheme is called double buffering and can be considered as a FIFO of depth 2.

Double transmit message buffers are implemented by combining two individual message buffers that form the two sides of an double transmit message buffer. One side is called the *commit side* and will be accessed by the application to provide the message data. The other side is called the *transmit side* and is used by the FlexRay block to transmit the message data to the FlexRay bus. The two sides are located in adjacent individual message buffers. The message buffer that implements the commit side has an even message buffer number 2n. The transmit side message buffer follows the commit side message buffer and has the message buffer number 2n+1. The basic structure and data flow of a double transmit message buffer is given in Figure 13-124.

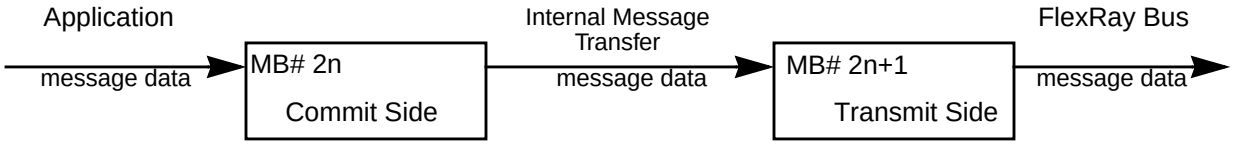


Figure 13-124. Double Transmit Buffer Structure and Data Flow

NOTE

Both the commit and the transmit side must be configured with identical values except for the Message Buffer Index Registers (MBIDXRn).

13.6.6.4.1 Access Regions

To certain message buffer fields, both the application and the FlexRay block have access. To ensure data consistency, a message buffer locking scheme is implemented, which controls the exclusive access to the data, control, and status bits of the message buffer.

The access scheme for double transmit message buffers is depicted in Figure 13-125. The given regions represent fields that can be accessed from both the application and the FlexRay block and, thus, require access restrictions. A description of the regions is given in Table 13-104.

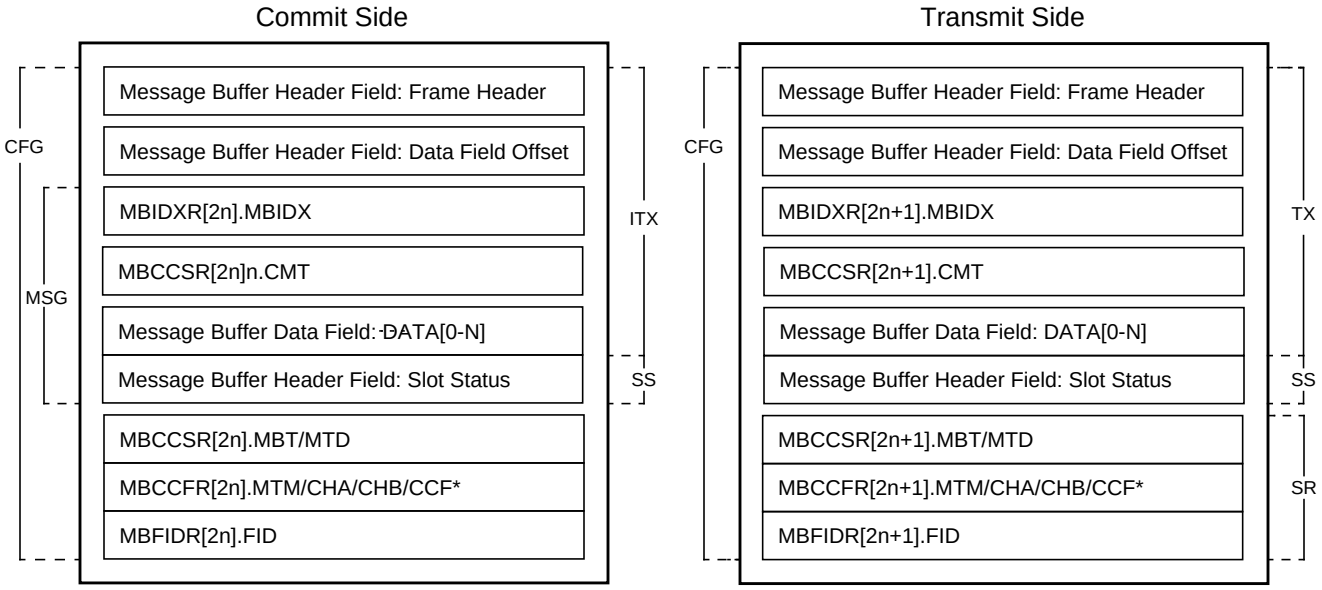


Figure 13-125. Double Transmit Message Buffer Access Regions Layout

Table 13-105. Double Transmit Message Buffer Access Regions Description

Access			Description
Region	Type		
	Application	Module	
Commit Side			
CFG	read/write	-	Message Buffer Configuration
MSG	read/write	-	Message Buffer Data and Control access
ITX	-	read/write	Internal Message Transfer.
SS	-	write-only	Slot Status Update
Transmit Side			
CFG	read/write	-	Message Buffer Configuration
SR	-	read-only	Message Buffer Search
TX	-	read-only	Internal Message Transfer, Message Transmission

Table 13-105. Double Transmit Message Buffer Access Regions Description

Access			Description
Region	Type		
	Application	Module	
SS	-	write-only	Slot Status Update

The trigger bits MBCCSRn.EDT and MBCCSRn.LCKT, and the interrupt enable bit MBCCSRn.MBIE are not under access control and can be accessed from the application at any time. The status bits MBCCSRn.EDS and MBCCSRn.LCKS are not under access control and can be accessed from the FlexRay block at any time.

The interrupt flag MBCCSnR.MBIF is not under access control and can be accessed from the application and the FlexRay block at any time. FlexRay block set access has higher priority.

The FlexRay block restricts its access to the regions, depending on the current state of the corresponding part of the double transmit message buffer. The application must adhere to these restrictions in order to ensure data consistency. The states for the commit side of a double transmit message buffer are given in Figure 13-126. A description of the states is given in Table 13-106. The states for the transmit side of a double transmit message buffer are given in Figure 13-127. A description of the states is given in Table 13-106. The description tables also provide the access scheme for the access regions.

The status bits MBCCSRn.EDS and MBCCSRn.LCKS provide the application with the required message buffer status information. The internal status information is not visible to the application.

13.6.6.4.2 Message Buffer States

This section describes the transmit message buffer states and provides a state diagram.

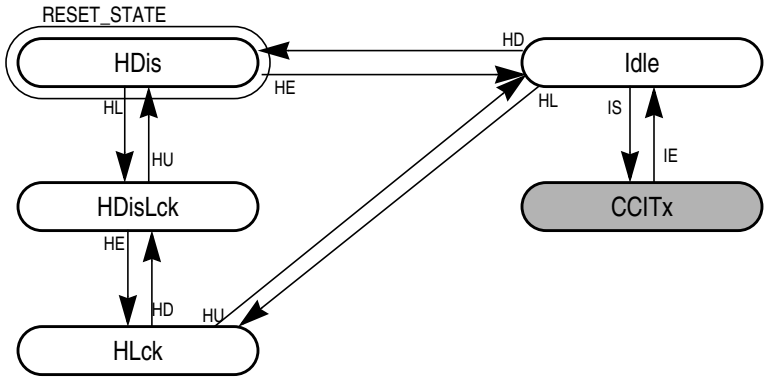


Figure 13-126. Double Transmit Message Buffer State Diagram (Commit Side)

A description of the states of the commit side of a double transmit message buffer is given in Table 13-105.

Table 13-106. Double Transmit Message Buffer State Description (Sheet 1 of 2)(Commit Side)

State	MBCCSR[2n]		Access Region		Description
	EDS	LCKS	Appl.	Module	
common states					

Table 13-106. Double Transmit Message Buffer State Description (Sheet 2 of 2)(Commit Side)

State	MBCCSR[2n]		Access Region		Description
	EDS	LCKS	Appl.	Module	
HDis	0	0	CFG	–	**Disabled** - Message Buffer under configuration. Commit Side can *not* be used for internal message transfer.
CCITx	1	0	–	ITX	**Internal Message Transfer** - Message Buffer Data transferred from commit side to transmit side.
commit side specific states					
Idle	1	0	–	ITX, SS	**Idle** - Message Buffer Commit Side is idle. Commit Side can be used for internal message transfer.
HDisLck	0	1	CFG	SS	**Disabled and Locked** - Message Buffer under configuration. Commit Side can *not* be used for internal message transfer.
HLck	1	1	MSG	SS	**Locked** - Applications access to data, control, and status. Commit Side can *not* be used for internal message transfer.

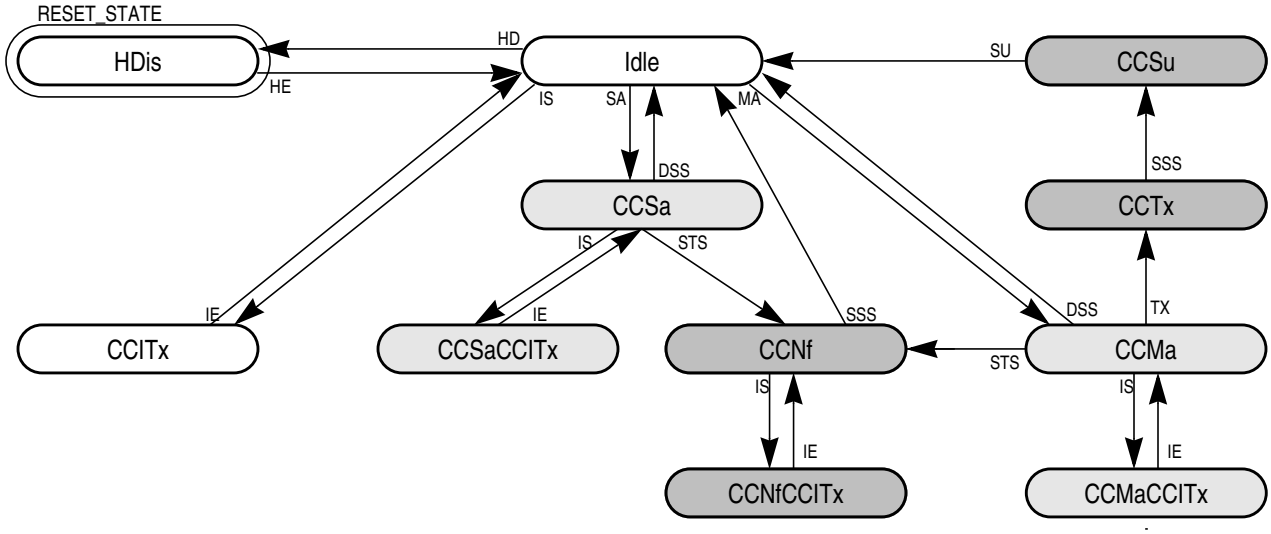


Figure 13-127. Double Transmit Message Buffer State Diagram (Transmit Side)

A description of the states of the transmit side of a double transmit message buffer is given in Table 13-106.

Table 13-107. Double Transmit Message Buffer State Description (Transmit Side) (Sheet 1 of 2)

State	MBCCSRn		Access Region		Description
	EDS	LCKS	Appl.	Module	
common states					
HDis	0	0	CFG	–	**Disabled** - Message Buffer under configuration. Excluded from message buffer search.
CCITx	1	0	–	TX	**Internal Message Transfer** - Message Buffer Data transferred from commit side to transmit side.
transmit side specific states					
Idle	1	0	–	SR	**Idle** - Message Buffer Transmit Side is idle. Transmit Side is included in message buffer search.

Table 13-107. Double Transmit Message Buffer State Description (Transmit Side) (Sheet 2 of 2)

State	MBCCSRn		Access Region		Description
	EDS	LCKS	Appl.	Module	
CCSa	1	0	–	–	**Slot Assigned** - Message buffer assigned to next static slot. Ready for Null Frame transmission.
CCSaCCITx	1	0	–	TX	**Slot Assigned and Internal Message Transfer** - Message buffer assigned to next static slot and Message Buffer Data transferred from commit side to transmit side.
CCNf	1	0	–	TX	**Null Frame Transmission** Header is used for null frame transmission.
CCNfCCITx	1	0	–	TX	**Null Frame Transmission and Internal Message Transfer** - Header is used for null frame transmission and Message Buffer Data transferred from commit side to transmit side.
CCMa	1	0	–	–	**Message Available** - Message buffer is assigned to next slot and cycle counter filter matches.
CCMaCCITx	1	0	–	–	**Message Available and Internal Message Transfer** - Message buffer is assigned to next slot and cycle counter filter matches and Message Buffer Data transferred from commit side to transmit side.
CCTx	1	0	–	TX	**Message Transmission** - Message buffer data transmit. Payload data from buffer transmitted
CCSu	1	0	–	SS	**Status Update** - Message buffer status update. Update of status flags, the slot status field, and the header index. **Note:** The slot status field of the commit side is updated too, even if the application has locked the commit side.

13.6.6.4.3 Message Buffer Transitions

Application Transitions

The application transitions that can be triggered by the application using the commands described in Table 13-107. The application issues the commands by writing to the Message Buffer Configuration, Control, Status Registers (MBCCSRn). Only one command can be issued with one write access. Each command is executed immediately. If the command is ignored, it must be issued again.

The enable and disable commands can be issued on the transmit side only. Any enable or disable command issued on the commit side will be ignored without notification. The transitions that will be triggered depends on the value of the EDS bit. The enable and disable commands will affect both the commit side and the transmit side at the same time. If the application triggers the disable transition HD while the transmit side is in one of the states CCSa, CCSaCCITx, CCNf, CCNfCCITx, CCMa, CCMaCCITx, CCTx, or CCSu, the disable transition has no effect (command is ignored) and the message buffer state is not changed. No notification is given to the application.

The lock and unlock commands can be issued on the commit side only. Any lock or unlock command issued on the transmit side will be ignored and the double transmit buffer lock error flag DBL_EF in the CHI Error Flag Register (CHIERFR) will be set. The transitions that will be triggered depends on the current value of the LCKS bit. The lock and unlock commands will only affect the commit side. If the application triggers the lock transition HL while the commit side is in the state CCITx, the message buffer state will not be changed and the message buffer lock error flag LCK_EF in the CHI Error Flag Register (CHIERFR) will be set.

Table 13-108. Double Transmit Message Buffer Host Transitions

Transition	Host Command	Condition	Description
HE	MBCCSR[2n+1].EDT:= 1	MBCCSR[2n+1].EDS = 0	Application triggers message buffer enable.
HD		MBCCSR[2n+1].EDS = 1	Application triggers message buffer disable.
HL	MBCCSR[2n].LCKT:= 1	MBCCSR[2n].LCKS = 0	Application triggers message buffer lock.
HU		MBCCSR[2n].LCKS = 1	Application triggers message buffer unlock.

Module Transitions

The module transitions that can be triggered by the FlexRay block are described in Table 13-108. The transitions C1 and C2 apply to both sides of the message buffer and are applied at the same time. All other FlexRay block transitions apply to the transmit side only.

Table 13-109. Double Transmit Message Buffer Module Transitions

Transition	Condition	Description
common transitions		
IS	see Section 13.6.6.4.5, “Internal Message Transfer	**Internal Message Transfer Start** - Start transfer of message data from commit side to transmit side.
IE		**Internal Message Transfer End** - Stop transfer of message data from commit side to transmit side. **Note:** The internal message transfer is stopped before the slot or segment start.
transmit side specific transitions		
SA	slot match and static slot	**Slot Assigned** - Message buffer is assigned to next static slot.
MA	slot match and CycleCounter match	**Message Available** - Message buffer is assigned to next slot and cycle counter filter matches.
TX	slot start and MBCCSR[2n+1].CMT = 1	**Transmission Slot Start** - Slot Start and commit bit CMT is set. In case of a dynamic slot, pLatestTx is not exceeded.
SU	status updated	**Status Updated** - Slot Status field and message buffer status flags updated. Interrupt flag set.
STS	static slot start	**Static Slot Start** - Start of static slot.
DSS	dynamic slot start or symbol window start or NIT start	**Dynamic Slot or Segment Start.** - Start of dynamic slot or symbol window or NIT.
SSS	slot start or symbol window start or NIT start	**Slot or Segment Start** - Start of static slot or dynamic slot or symbol window or NIT.

Transition Priorities

The application can trigger only one transition at a time. There is no need to specify priorities among them.

As shown in the first part of Table 13-109, the module transitions have a higher priority than the application transitions. The priorities among the FlexRay block transitions and the related states are given in the second part of Table 13-109. These priorities apply only to the transmit side. The internal message transmit start transition IS has tho lowest priority.

Table 13-110. Double Transmit Message Buffer Transition Priorities

State	Priority	Description
module vs. application		
Idle	IS > HD IS > HL	Internal Message Transfer Start > Message Buffer Disable Internal Message Transfer Start > Message Buffer Lock
module internal		
Idle	MA > SA	Message Available > Slot Assigned
CCMa	TX > STS TX > DSS	Transmission Slot Start > Static Slot Start Transmission Slot Start > Dynamic Slot Start

13.6.6.4.4 Message Preparation

The application provides the message data through the commit side. The transmission itself is executed from the transmit side. The transfer of the message data from the commit side to the transmit side is done by the *Internal Message Transfer*, which is described in Section 13.6.6.4.5, “Internal Message Transfer

To transmit a message over the FlexRay bus, the application writes the message data into the message buffer data field of the commit side and sets the commit bit CMT in the Message Buffer Configuration, Control, Status Registers (MBCCSRn). The physical access to the message buffer data field is described in Section 13.6.3.1, “Individual Message Buffers”.

As indicated by Table 13-105, the application shall write to the message buffer data field and change the commit bit CMT only if the transmit message buffer is in one of the states HDis, HDisLck, or HLck. The application can change the state of a message buffer if it issues the appropriate commands shown in Table 13-107. The state change is indicated through the MBCCSRn.EDS and MBCCSRn.LCKS status bits.

13.6.6.4.5 Internal Message Transfer

The internal message transfer transfers the message data from the commit side to the transmit side. The internal message transfer is implemented as the swapping of the content of the Message Buffer Index Registers (MBIDXRn) of the commit side and the transmit side. After the swapping, the commit side CMT bit is cleared, the commit side interrupt flag MBIF is set, the transmit side CMT bit is set, and the transmit side DVAL bit is cleared.

The conditions and the point in time when the internal message transfer is started are controlled by the message buffer commit mode bit MCM in the Message Buffer Configuration, Control, Status Registers (MBCCSRn). The MCM bit configures the message buffer for either the streaming commit mode or the immediate commit mode. A detailed description is given in Streaming Commit Mode and Immediate Commit Mode. The Internal Message Transfer is triggered with the transition IS. Both sides of the message buffer enter one of the CCITx states. The internal message transfer is finished with the transition IE.

Streaming Commit Mode

The intention of the streaming commit mode is to ensure that each committed message is transmitted *at least once*. The FlexRay block will not start the Internal Message Transfer for a message buffer as long as the message data on the transmit side is not transmitted at least once.

The streaming commit mode is configured by clearing the message buffer commit mode bit MCM in the Message Buffer Configuration, Control, Status Registers (MBCCSRn).

In this mode, the internal message transfer from the commit side to the transmit side is started for a double transmit message buffer when all of the following conditions are fulfilled

1. the commit side is in the Idle state
2. the commit site message data are valid, i.e. MBCCSR[2n].CMT = 1
3. the transmit side is in one of the states Idle, CCSa, or CCMa
4. the transmit side contains either no valid message data, i.e. MBCCSR[2n+1].CMT = 0 or the message data were transmitted at least once, i.e. MBCCSR[2n+1].DVAL = 1

An example of a streaming commit mode state change diagram is given in Figure 13-128. In this example, both the commit and the transmit side do not contain valid message data and the application provides two messages. The message buffer does not match the next slot.

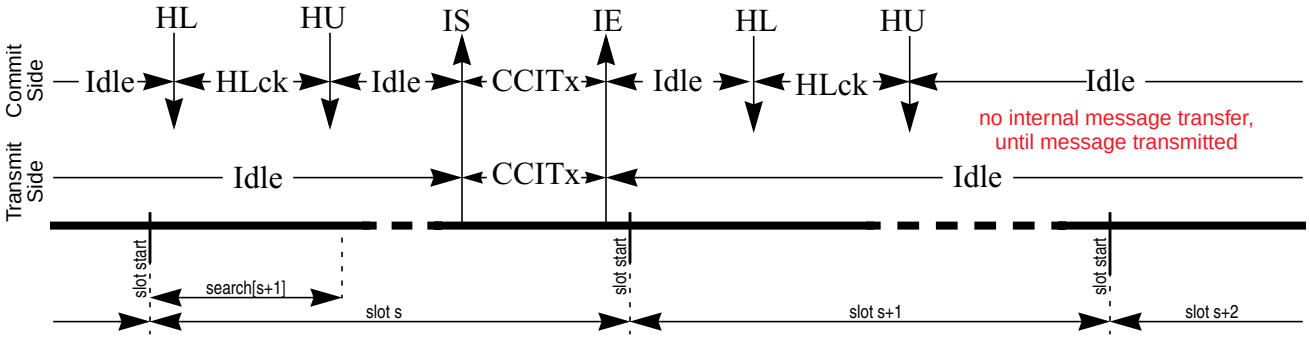


Figure 13-128. Internal Message Transfer in Streaming Commit Mode

Immediate Commit Mode

The intention of the immediate commit mode is to transmit the *latest* data provided by the application. This implies that it is not guaranteed that each provided message will be transmitted at least once.

The immediate commit mode is configured by setting the message buffer commit mode bit MCM in the Message Buffer Configuration, Control, Status Registers (MBCCSRn).

In this mode, the internal message transfer from the commit side to the transmit side is started for one double transmit message buffer when all of the following conditions are fulfilled

1. the commit side is in the Idle state
2. the commit site message data are valid, i.e. MBCCSR[2n].CMT = 1
3. the transmit side is in one of the states Idle, CCSa, or CCMa

It is not checked whether the transmit side contains no valid message data or valid message data were transmitted at least once. If message data are valid and not transmitted, they may be overwritten.

An example of a streaming commit mode state change diagram is given in Figure 13-129. In this example, both the commit and the transmit side do not contain valid message data, and the application provides two messages and the first message is gets overwritten. The message buffer does not match the next slot.

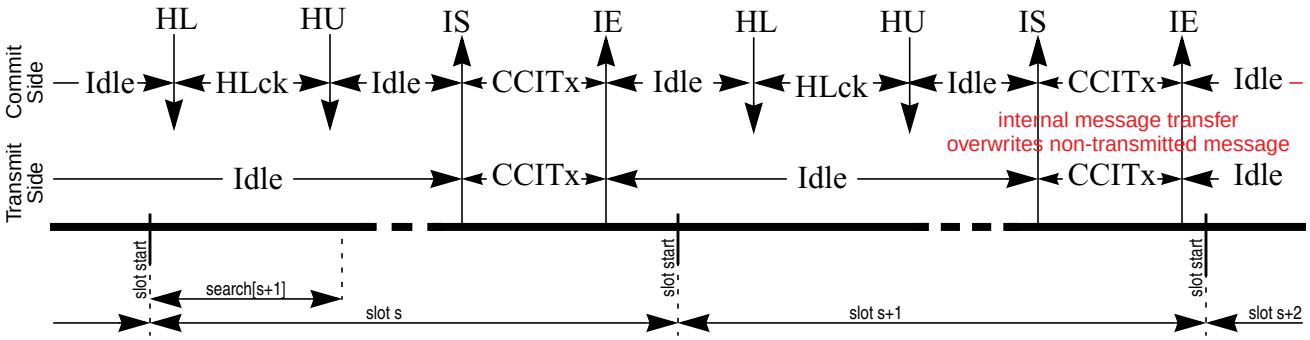


Figure 13-129. Internal Message Transfer in Immediate Commit Mode

13.6.6.4.6 Message Transmission

For double transmit message buffers, the message buffer search checks only the transmit side part. The internal scheduling ensures, that the internal message transfer is stopped on the message buffer search start. Thus, the transmit side of message buffer, that is not in its transmission or status update slot, is always in the Idle state.

The message transmit behavior and transmission state changes of the transmit side of a double transmit message buffer are the same as for single buffered transmit buffers, except that the transmit side of double buffers can not be locked by the application, i.e. the HU and HL transition do not exist. Therefore, refer to Section 13.6.6.2.5, “Message Transmission”

13.6.6.4.7 Message Buffer Status Update

The message buffer status update behavior of the transmit side of a double transmit message buffer is the same as for single transmit message buffers which is described in Section 13.6.6.2.7, “Message Buffer Status Update”.

Additionally, the slot status field of the commit side is update after the update of the slot status field of the transmit side, even if the commit side is locked by the application. This is implemented to provide the slot status of the most recent transmission slot.

13.6.7 Individual Message Buffer Search

This section provides a detailed description of the message buffer search algorithm.

The message buffer search determines for each enabled channel if a slot *s* in a communication cycle *c* is assigned for frame or null frame transmission or if it is subscribed for frame reception on that channel.

The message buffer search is a sequential algorithm which is invoked at the following protocol related events:

1. NIT start
2. slot start in the static segment
3. minislot start in the dynamic segment

The message buffer search within the NIT searches for message buffers assigned or subscribed to slot 1. The message buffer search within slot *n* searches for message buffers assigned or subscribed to slot *n+1*.

In general, the message buffer search for the next slot *n* considers only message buffers which are

1. enabled, i.e. MBCCSRn.EDS = 1, and
2. matches the next slot *n*, i.e. MBFIDRn.FID = *n*, and
3. are the transmit side buffer in case of a double transmit message buffer.

On top of that, for the static segment only those message buffers are considered, that match the condition of at least one row of Table 13-110. For the dynamic segment only those message buffers are considered, that match the condition of at least one row of Table 13-111. These message buffers are called *matching* message buffers.

For each enabled channel the message buffer search may identify multiple *matching* message buffers. Among all matching message buffers the message buffers with highest priority according to Table 13-110 for the static segment and according to Table 13-111 for the dynamic segment are selected.

Table 13-111. Message Buffer Search Priority (static segment)

Priority	MTD	LCKS	CMT	CCFM (1)	Description	Transition
(highest) 0	1	0	1	1	transmit buffer, matches cycle count, not locked and committed	MA
1	1	-	0	1	transmit buffer, matches cycle count, not committed	SA
	1	1	-	1	transmit buffer, matches cycle count, locked	SA
2	1	-	-	-	transmit buffer	SA
3	0	0	n/a	1	receive buffer, matches cycle count, not locked	SB
(lowest) 4	0	1	n/a	1	receive buffer, matches cycle count, locked	SB

1. Cycle Counter Filter Match, see Section 13.6.7.1, "Message Buffer Cycle Counter Filtering"

Table 13-112. Message Buffer Search Priority (dynamic segment)

Priority	MTD	LCKS	CMT	CCFM (1)	Description	Transition
(highest) 0	1	0	1	1	transmit buffer, matches cycle count, not locked and committed	MA
1	0	0	n/a	1	receive buffer, matches cycle count, not locked	SB
(lowest) 2	0	1	n/a	1	receive buffer, matches cycle count, locked	SB

1. Cycle Counter Filter Match, see Section 13.6.7.1, "Message Buffer Cycle Counter Filtering"

If there are multiple message buffer with highest priority, the message buffer with the lowest message buffer number is selected. All message buffer which have the highest priority must have a consistent channel assignment as specified in Section 13.6.7.2, "Message Buffer Channel Assignment Consistency".

Depending on the message buffer channel assignment the same message buffer can be found for both channel A and channel B. In this case, this message buffer is used as described in Section 13.6.3.1, "Individual Message Buffers".

13.6.7.1 Message Buffer Cycle Counter Filtering

The message buffer cycle counter filter is a value-mask filter defined by the CCFE, CCFMSK, and CCFVAL fields in the Message Buffer Cycle Counter Filter Registers (MBCCFRn). This filter determines a set of communication cycles in which the message buffer is considered for message reception or message transmission. If the cycle counter filter is disabled, i.e. CCFE = 0, this set of cycles consists of all communication cycles.

If the cycle counter filter of a message buffer does not match a certain communication cycle number, this message buffer is not considered for message transmission or reception in that communication cycle. In case of a transmit message buffer assigned to a slot in the static segment, though, this buffer is added to the matching message buffers to indicate the slot assignment and to trigger the null frame transmission.

The cycle counter filter of a message buffer matches the communication cycle with the number CYCCNT if at least one of the following conditions evaluates to true:

$$\mathrm{MBCCFRn[CCFE]} = 0 \qquad \textit{Eqn. 13-9}$$

$$\mathrm{CYCCNT} \wedge \mathrm{MBCCFRn[CCFMSK]} = \mathrm{MBCCFRn[CCFVAL]} \wedge \mathrm{MBCCFRn[CCFMSK]} \qquad \textit{Eqn. 13-10}$$

13.6.7.2 Message Buffer Channel Assignment Consistency

The message buffer channel assignment given by the CHA and CHB bits in the Message Buffer Cycle Counter Filter Registers (MBCCFRn) defines the channels on which the message buffer will receive or transmit. The message buffer with number *n* transmits or receives on channel A if MBCCFRn[CHA] = 1 and transmits or receives on channel B if MBCCFRn[CHB] = 1.

To ensure correct message buffer operation, all message buffers assigned to the same slot and with the same priority must have a *consistent* channel assignment. That means they must be either assigned to one channel only, or must be assigned to *both* channels. The behavior of the message buffer search is not

defined, if both types of channel assignments occur for one slot and priority. An inconsistent channel assignment for message buffer 0 and message buffer 1 is depicted in Figure 13-130.

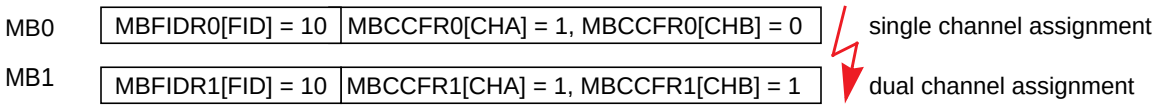


Figure 13-130. Inconsistent Channel Assignment

13.6.7.3 Node Related Slot Multiplexing

The term *Node Related Slot Multiplexing* applies to the dynamic segment only and refers to the functionality if there are transmit as well as receive message buffers are configured for the same slot.

According to Table 13-111 the transmit buffer is only found if the cycle counter filter matches, and the buffer is not locked and committed. In all other cases, the receive buffer will be found. Thus, if the block has no data to transmit in a dynamic slot, it is able to receive frames on that slot.

13.6.7.4 Message Buffer Search Error

If the message buffer search is running while the next message buffer search start event appears, the message buffer search is stopped and the Message Buffer Search Error Flag MSB_EF is set in the CHI Error Flag Register (CHIERFR). This appears only if the CHI frequency is too low to search through all message buffers within the NIT or a minislot. The message buffer result is not defined in this case. For more details see Section 13.7.3, “Number of Usable Message Buffers”.

13.6.8 Individual Message Buffer Reconfiguration

The initial configuration of each individual message buffer can be changed even when the protocol is not in the *POC:config* state. This is referred to as individual message buffer *reconfiguration*. The configuration bits and fields that can be changed are given in the section on Specific Configuration Data. The common configuration data given in the section on Specific Configuration Data can not be reconfigured when the protocol is out of the *POC:config* state.

13.6.8.1 Reconfiguration Schemes

Depending on the target and destination basic state of the message buffer that is to be reconfigured, there are three reconfiguration schemes.

13.6.8.1.1 Basic Type Not Changed (RC1)

A reconfiguration will not change the basic type of the individual message buffer, if both the message buffer transfer direction bit MBCCSn.MTD and the message buffer type bit MBCCSn.MBT are not changed. This type of reconfiguration is denoted by RC1 in Figure 13-131. Single transmit and receive message buffers can be RC1-reconfigured when in the HDis or HDisLck state. Double transmit message buffers can be RC1-reconfigured if both the transmit side and the commit side are in the HDis state.

13.6.8.1.2 Buffer Type Not Changed (RC2)

A reconfiguration will not change the buffer type of the individual message buffer if the message buffer buffer type bit MBCCSRn.MBT is not changed. This type of reconfiguration is denoted by RC2 in Figure 13-131. It applies only to single transmit and receive message buffers. Single transmit and receive message buffers can be RC2-reconfigured when in the HDis or HDisLck state.

13.6.8.1.3 Buffer Type Changed (RC3)

A reconfiguration will change the buffer type of the individual message buffer if the message buffer type bit MBCCSRn.MBT is changed. This type of reconfiguration is denoted by RC3 in Figure 13-131. The RC3 reconfiguration splits one double buffer into two single buffers or combines two single buffer into one double buffer. In the later case, the two single message buffers must have consecutive message buffer numbers and the smaller one must be even. Message Buffers can be RC3 reconfigured if they are in the HDis state.

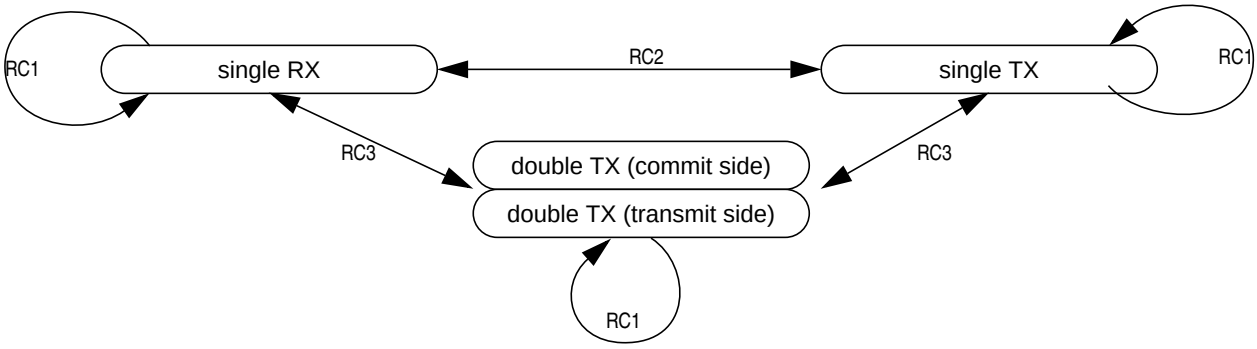


Figure 13-131. Message Buffer Reconfiguration Scheme

13.6.9 Receive FIFO

This section provides a detailed description of the two receive FIFOs.

13.6.9.1 Overview

The receive FIFOs implement the queued receive buffer defined by the *FlexRay Communications System Protocol Specification, Version 2.1 Rev A.* One receive FIFO is assigned to channel A, the other receive FIFO is assigned to channel B. Both FIFOs work completely independent from each other.

The message buffer structure of each FIFO is described in Section 13.6.3.3, “Receive FIFO”. The area in the FRM for each of the two receive FIFOs is characterized by:

- The index of the first FIFO entry given by Receive FIFO Start Index Register (RFSIR)
- The number of FIFO entries and the length of each FIFO entry as given by Receive FIFO Depth and Size Register (RFDSR)

13.6.9.2 Receive FIFO Configuration

The receive FIFO control and configuration data are given in Section 13.6.3.7, “Receive FIFO Control and Configuration Data”. The configuration of the receive FIFOs consists of two steps.

The first step is the allocation of the required amount of FRM for the FlexRay window. This includes the allocation of the message buffer header area and the allocation of the message buffer data fields. For more details see Section 13.6.4, “FlexRay Memory Layout”.

The second step is the programming of the configuration data register while the PE is in *POC:config.*

The following steps configure the layout of the FIFO.

- The number of the first message buffer header index that belongs to the FIFO is written into the Receive FIFO Start Index Register (RFSIR).
- The depth of the FIFO is written into the FIFO_DEPTH field in the Receive FIFO Depth and Size Register (RFDSR).
- The length of the message buffer data field for the FIFO is written into the ENTRY_SIZE field in the Receive FIFO Depth and Size Register (RFDSR).

NOTE

To ensure, that the read index RDIDX always points to a message buffer that contains valid data, the receive FIFO must have at least 2 entries.

The FIFO filters are configured through the fifo filter registers.

13.6.9.3 Receive FIFO Reception

The frame reception to the receive FIFO is enabled, if for a certain slots no message buffer is assigned or subscribed. In this case the FIFO filter path shown in Figure 13-132 is activated.

When the receive FIFO filter path indicates that the received frame must be appended to the FIFO, the FlexRay block writes the received frame header and slot status into the message buffer header field

indicated by the internal FIFO header write index. The payload data are written in the message buffer data field. If the status of the received frame indicates a valid frame, the internal FIFO header write index is updated and the fifo not-empty interrupt flag FNEAIF/FNEBIF in the Global Interrupt Flag and Enable Register (GIFER) is set.

13.6.9.4 Receive FIFO Message Access

If the fifo not-empty interrupt flag FNEAIF/FNEBIF in the Global Interrupt Flag and Enable Register (GIFER) is set, the receive FIFO contains valid received messages, which can be accessed by the application.

The receive FIFO does not require locking to access the message buffers. To access the message the application first reads the receive FIFO read index RDIDX from the Receive FIFO A Read Index Register (RFARIR) or Receive FIFO B Read Index Register (RFBRIR), respectively. This index points to the message buffer header field of the next message buffer that contains valid data. The application can access the message data as described in Section 13.6.3.3, “Receive FIFO”. When the application has read all message buffer data and status information, it writes 1 to the fifo not-empty interrupt flags FNEAIF or FNEBIF. This clears the interrupt flag and updates the RDIDX field in the Receive FIFO A Read Index Register (RFARIR) or Receive FIFO B Read Index Register (RFBRIR), respectively.When the RDIDX value has reached the last message buffer header field that belongs to the fifo, it wraps around to the index of the first message buffer header field that belongs to the fifo. This value is provided by the SIDX field in the Receive FIFO Start Index Register (RFSIR).

13.6.9.5 Receive FIFO filtering

The receive FIFO filtering is activated after all enabled individual receive message buffers have been searched without success for a message buffer to receive the current frame.

The FlexRay block provides three sets of FIFO filters. The FIFO filters are applied to valid non-null frames only. The FIFO will not receive invalid or null-frames. For each FIFO filter, the pass criteria is specified in the related section given below. Only frames that have passed all filters will be appended to the FIFO. The FIFO filter path is depicted in Figure 13-132.

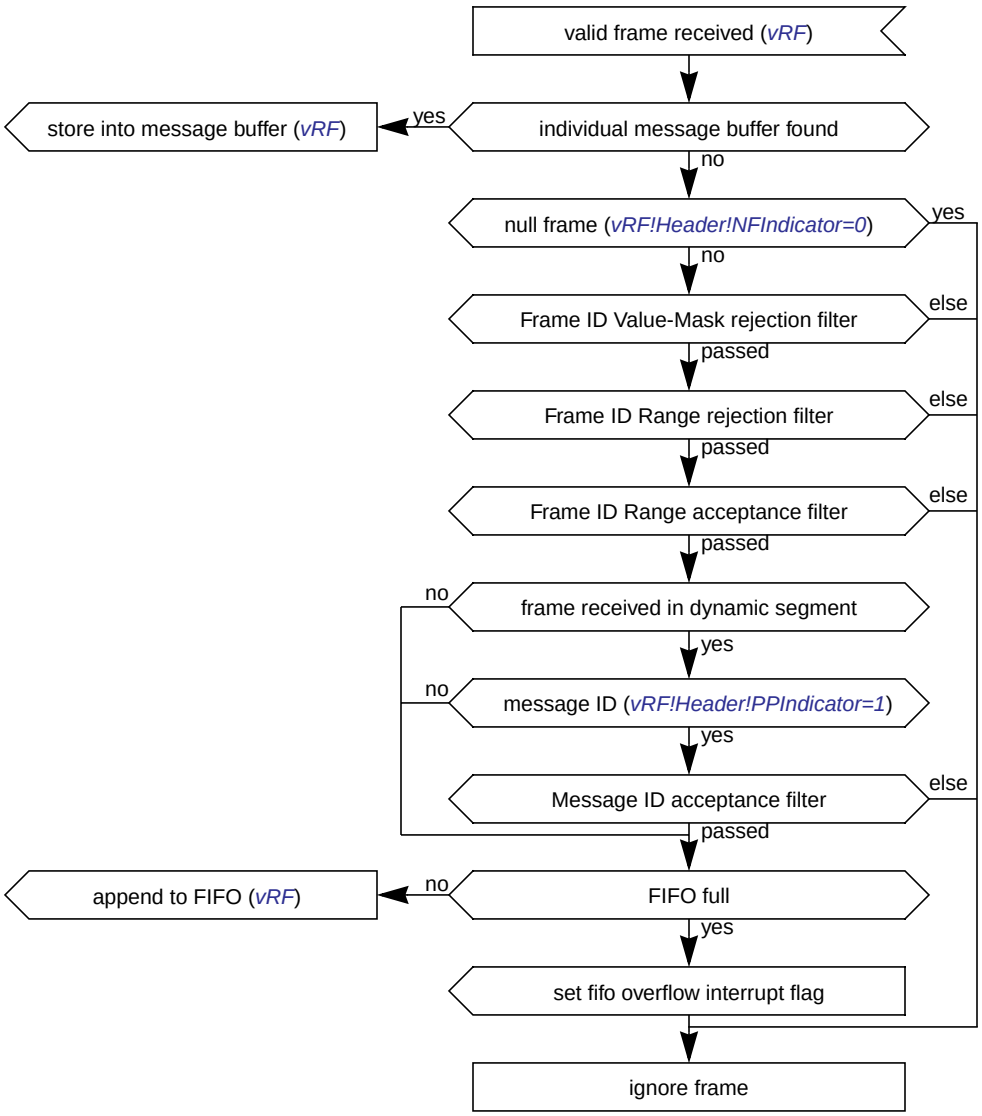


Figure 13-132. Received Frame FIFO Filter Path

A received frame passes the FIFO filtering if it has passed all three type of filter.

13.6.9.5.1 RX FIFO Frame ID Value-Mask Rejection Filter

The frame ID value-mask rejection filter is a value-mask filter and is defined by the fields in the Receive FIFO Frame ID Rejection Filter Value Register (RFFIDRFVR) and the Receive FIFO Frame ID Rejection Filter Mask Register (RFFIDRFMR). Each received frame with a frame ID FID that does not match the value-mask filter value passes the filter, i.e. is not rejected.

Consequently, a received valid frame with the frame ID FID passes the RX FIFO Frame ID Value-Mask Rejection Filter if Equation 13-11 is fulfilled.

$$FID \wedge RFFIDRFMR[FIDRFMSK] \neq RFFIDRFVR[FIDRFVAL] \wedge RFFIDRFMR[FIDRFMSK] \qquad \textit{Eqn. 13-11}$$

The RX FIFO Frame ID Value-Mask Rejection Filter can be configured to pass all frames by the following settings.

- RFFIDRFVR.FIDRFVAL:= 0x000 and RFFIDRFMR.FIDRFMSK:= 0x7FF

Using the settings above, only the frame with frame ID 0 will be rejected, which is an invalid frame. All other frames will pass.

The RX FIFO Frame ID Value-Mask Rejection Filter can be configured to reject all frames by the following settings.

- RFFIDRFMR.FIDRFMSK:= 0x000

Using the settings above, Equation 13-11 can never be fulfilled (0!= 0) and thus all frames are rejected; no frame will pass. This is the reset value for the RX FIFO.

13.6.9.5.2 RX FIFO Frame ID Range Rejection Filter

Each of the four RX FIFO Frame ID Range filters can be configured as a rejection filter. The filters are configured by the Receive FIFO Range Filter Configuration Register (RFRFCFR) and controlled by the Receive FIFO Range Filter Control Register (RFRFCTR). The RX FIFO Frame ID range filters apply to all received valid frames. A received frame with the frame ID FID passes the RX FIFO Frame ID Range rejection filters if either no rejection filter is enabled, or, for all of the enabled RX FIFO Frame ID Range rejection filters, i.e. RFRFCTR.FiMD = 1 and RFRFCTR.FiEN = 1, Equation 13-12 is fulfilled.

$$(FID < RFRFCFR_{SEL}[SID_{IBD=0}]) \text{ or } (RFRFCFR_{SEL}[SID_{IBD=1}] < FID) \qquad \textit{Eqn. 13-12}$$

Consequently, all frames with a frame ID that fulfills Equation 13-13 for at least one of the enabled rejection filters will be rejected and thus not pass.

$$RFRFCFR_{SEL}[SID_{IBD=0}] \leq FID \leq RFRFCFR_{SEL}[SID_{IBD=1}] \qquad \textit{Eqn. 13-13}$$

13.6.9.5.3 RX FIFO Frame ID Range Acceptance filter

Each of the four RX FIFO Frame ID Range filters can be configured as an acceptance filter. The filters are configured by the Receive FIFO Range Filter Configuration Register (RFRFCFR) and controlled by the Receive FIFO Range Filter Control Register (RFRFCTR). The RX FIFO Frame ID range filters apply to all received valid frames. A received frame with the frame ID FID passes the RX FIFO Frame ID Range acceptance filters if either no acceptance filter is enabled, or, for at least one of the enabled RX FIFO Frame ID Range acceptance filters, i.e. RFRFCTR.FiMD = 0 and RFRFCTR.FiEN = 1, Equation 13-14 is fulfilled.

$$RFRFCFR_{SEL}[SID_{IBD=0}] \leq FID \leq RFRFCFR_{SEL}[SID_{IBD=1}] \qquad \textit{Eqn. 13-14}$$

13.6.9.5.4 RX FIFO Message ID Acceptance Filter

The RX FIFO Message ID Acceptance Filter is a value-mask filter and is defined by the Receive FIFO Message ID Acceptance Filter Value Register (RFMIDAFVR) and the Receive FIFO Message ID Acceptance Filter Mask Register (RFMIAFMR). This filter applies only to valid frames received in the dynamic segment with the payload preamble indicator bit PPI set to 1. All other frames will pass this filter.

A received valid frame in the dynamic segment with the payload preamble indicator bit PPI set to 1 and with the message ID MID (the first two bytes of the payload) will pass the RX FIFO Message ID Acceptance Filter if Equation 13-15 is fulfilled.

$$MID \wedge RFMIDAFMR[MIDAFMSK] = RFMIDAFMR[MIDAFVAL] \wedge RFMIDAFMR[MIDAFMSK] \quad \textit{Eqn. 13-15}$$

The RX FIFO Message ID Acceptance Filter can be configured to accept all frames by setting

- RFMIDAFMR.MIDAFMSK:= 0x000

Using the settings above, Equation 13-15 is always fulfilled and all frames will pass.

13.6.10 Channel Device Modes

This section describes the two FlexRay channel device modes that are supported by the FlexRay block.

13.6.10.1 Dual Channel Device Mode

In the dual channel device mode, both FlexRay ports are connected to physical FlexRay bus lines. The FlexRay port consisting of RXD_A, TXD_A, and $\overline{\text{TXE\_A}}$ is connected to the physical bus channel A and the FlexRay port consisting of RXD_B, TXD_B, and $\overline{\text{TXE\_A}}$ is connected to the physical bus channel B. The dual channel system is shown in Figure 13-133.

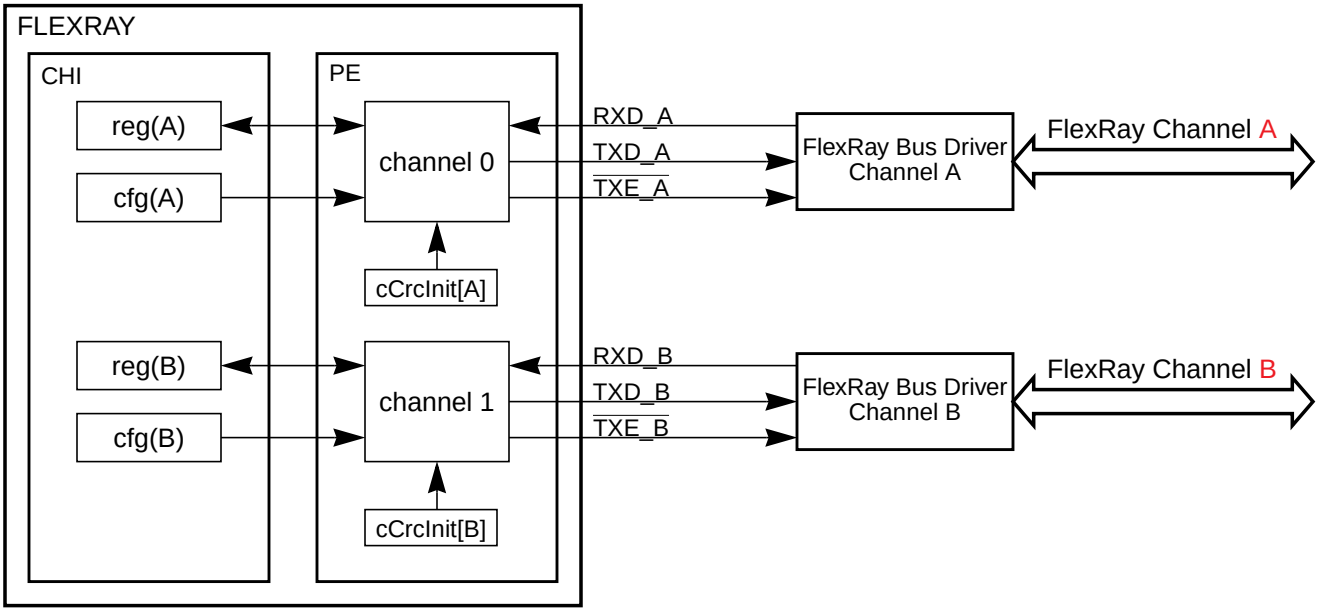


Figure 13-133. Dual Channel Device Mode

13.6.10.2 Single Channel Device Mode

The single channel device mode supports devices that have only one FlexRay port available. This FlexRay port consists of the signals RXD_A, TXD_A, and $\overline{\text{TXE\_A}}$ and can be connected to either the physical bus channel A (shown in Figure 13-134) or the physical bus channel B (shown in Figure 13-135).

If the device is configured as a single channel device by setting MCR.SCD to 1, only the internal channel A and the FlexRay Port A is used. Depending on the setting of MCR.CHA and MCR.CHB, the internal channel A behaves either as a FlexRay Channel A or FlexRay Channel B. The bit MCR.CHA must be set, if the FlexRay Port A is connected to a FlexRay Channel A. The bit MCR.CHB must be set if the FlexRay Port A is connected to a FlexRay Channel B. The two FlexRay channels differ only in the initial value for the frame CRC *cCrcInit*. For a single channel device, the application can access and configure only the registers related to internal channel A.

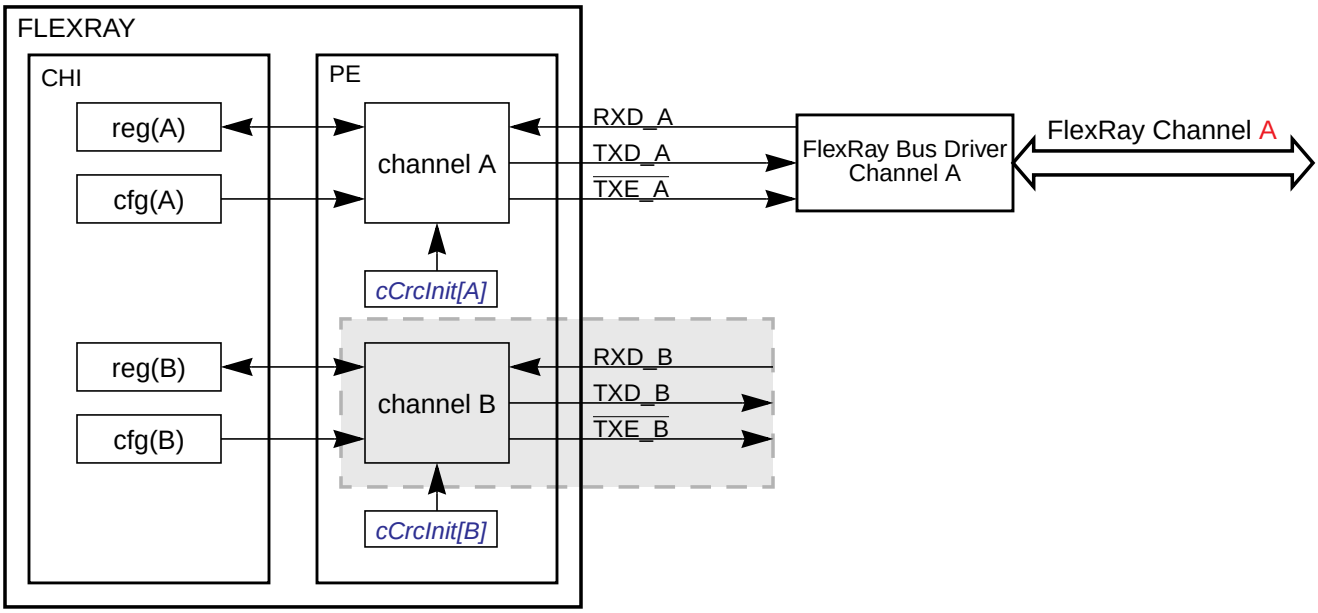


Figure 13-134. Single Channel Device Mode (Channel A)

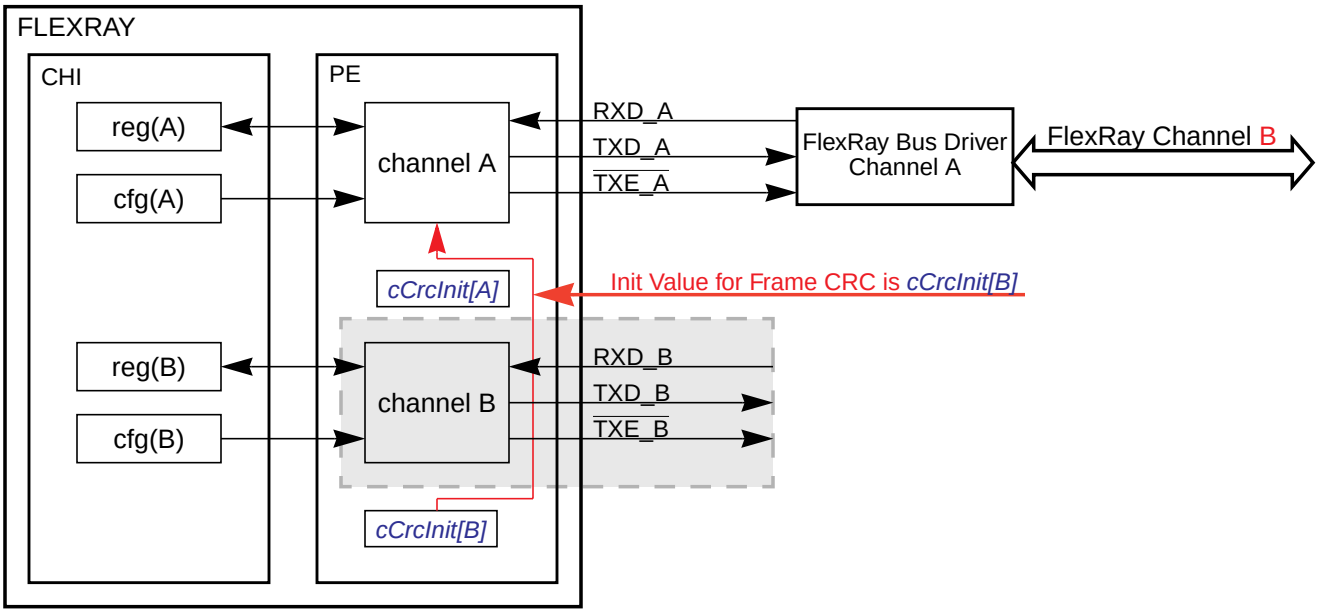


Figure 13-135. Single Channel Device Mode (Channel B)

13.6.11 External Clock Synchronization

The application of the external rate and offset correction is triggered when the application writes to the EOC_AP and ERC_AP fields in the Protocol Operation Control Register (POCR). The PE applies the external correction values in the next even-odd cycle pair as shown in Figure 13-136 and Figure 13-137.

NOTE

The values provided in the EOC_AP and ERC_AP fields are the values that were written from the application most recently. If these value were already applied, they will not be applied in the current cycle pair again.

If the offset correction applied in the NIT of cycle 2n+1 shall be affect by the external offset correction, the EOC_AP field must be written to after the start of cycle 2n and before the end of the static segment of cycle 2n+1. If this field is written to after the end of the static segment of cycle 2n+1, it is not guaranteed that the external correction value is applied in cycle 2n+1. If the value is not applied in cycle 2n+1, then the value will be applied in the cycle 2n+3. Refer to Figure 13-136 for timing details.

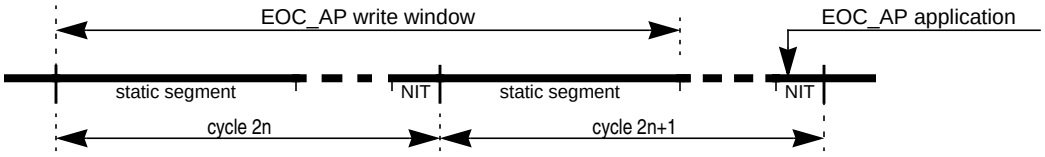


Figure 13-136. External Offset Correction Write and Application Timing

If the rate correction for the cycle pair [2n+2, 2n+3] shall be affect by the external offset correction, the ERC_AP field must be written to after the start of cycle 2n and before the end of the static segment start of cycle 2n+1. If this field is written to after the end of the static segment of cycle 2n+1, it is not guaranteed that the external correction value is applied in cycle pair [2n+2, 2n+3]. If the value is not applied for cycle pair [2n+2, 2n+3], then the value will be applied for cycle pair [2n+4, 2n+5]. Refer to Figure 13-137 for details.

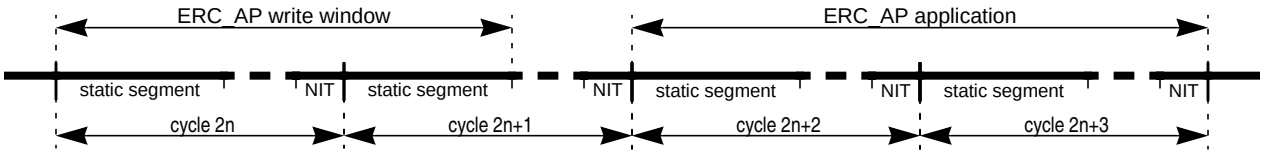


Figure 13-137. External Rate Correction Write and Application Timing

13.6.12 Sync Frame ID and Sync Frame Deviation Tables

The FlexRay protocol requires the provision of a snapshot of the Synchronization Frame ID tables for the even and odd communication cycle for both channels. The FlexRay block provides the means to write a copy of these internal tables into the FRM and ensures application access to consistent tables by means of table locking. Once the application has locked the table successfully, the FlexRay block will not overwrite these tables and the application can read a consistent snapshot.

NOTE

Only synchronization frames that have passed the synchronization frame filters are considered for clock synchronization and appear in the sync frame tables.

13.6.12.1 Sync Frame ID Table Content

The Sync Frame ID Table is a snapshot of the protocol related variables *vsSyncIdListA* and *vsSyncIdListB* for each even and odd communication cycle. This table provides a list of the frame IDs of the synchronization frames received on the corresponding channel and cycle that are used for the clock synchronization.

13.6.12.2 Sync Frame Deviation Table Content

The Sync Frame Deviation Table is a snapshot of the protocol related variable zsDev(id)(oe)(ch)!Value. Each Sync Frame Deviation Table entry provides the deviation value for the sync frame, with the frame ID presented in the corresponding entry in the Sync Frame ID Table.

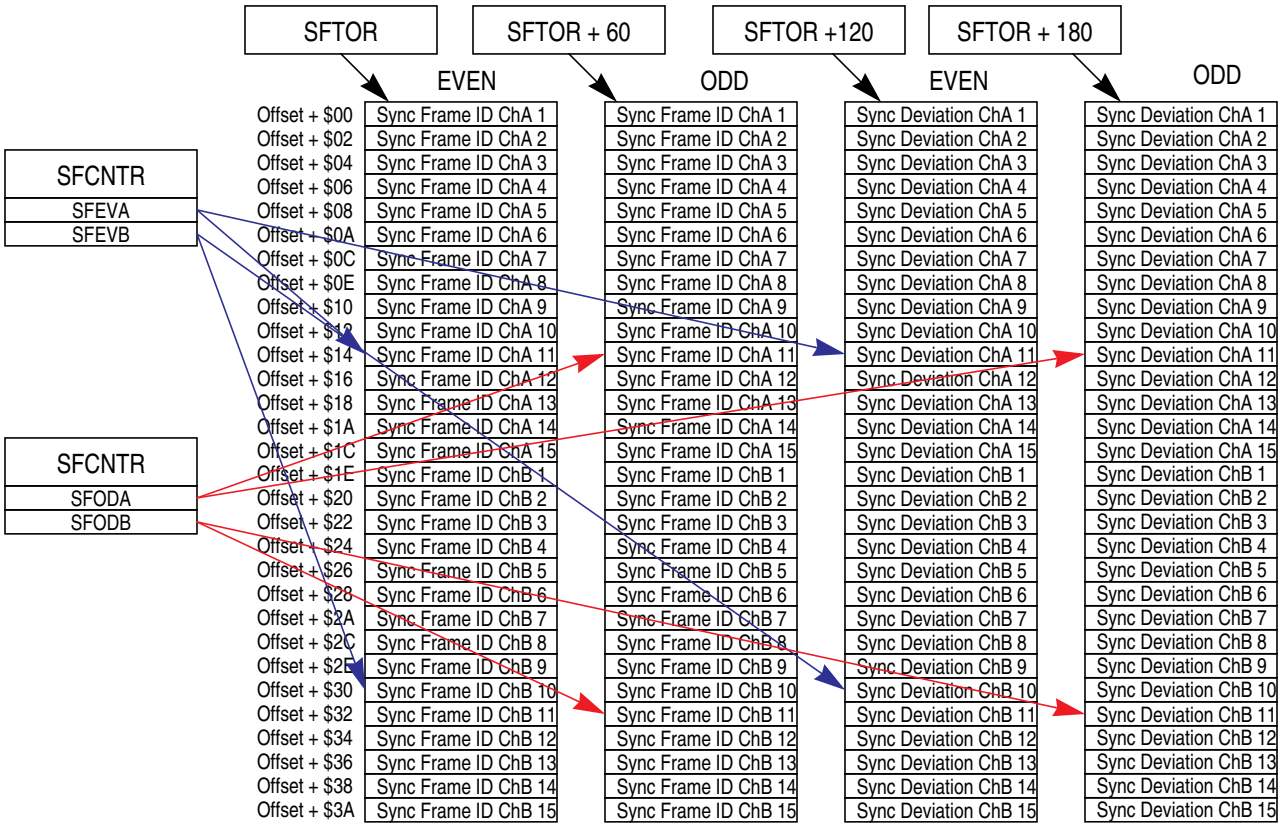


Figure 13-138. Sync Table Memory Layout

13.6.12.3 Sync Frame ID and Sync Frame Deviation Table Setup

The FlexRay block writes a copy of the internal synchronization frame ID and deviation tables into the FRM if requested by the application. The application must provide the appropriate amount of FRM for the tables. The memory layout of the tables is given in Figure 13-138. Each table occupies 120 16-bit entries.

While the protocol is in *POC:config* state, the application must program the offsets for the tables into the Sync Frame Table Offset Register (SFTOR).

13.6.12.4 Sync Frame ID and Sync Frame Deviation Table Generation

The application controls the generation process of the Sync Frame ID and Sync Frame Deviation Tables into the FRM using the Sync Frame Table Configuration, Control, Status Register (SFTCCSR). A summary of the copy modes is given in Table 13-112.

Table 13-113. Sync Frame Table Generation Modes

SFTCCSR			Description
OPT	SDVEN	SIDEN	
0	0	0	No Sync Frame Table copy
0	0	1	Sync Frame ID Tables will be copied continuously
0	1	0	Reserved
0	1	1	Sync Frame ID Tables and Sync Frame Deviation Tables will be copied continuously
1	0	0	No Sync Frame Table copy
1	0	1	Sync Frame ID Tables for next even-odd-cycle pair will be copied
0	1	0	Reserved
1	1	1	Sync Frame ID Tables and Sync Frame Deviation Tables for next even-odd-cycle pair will be copied

The Sync Frame Table generation process is described in the following for the even cycle. The same sequence applies to the odd cycle.

If the application has enabled the sync frame table generation by setting SFTCCSR.SIDEN to 1, the FlexRay block starts the update of the even cycle related tables after the start of the NIT of the next even cycle. The FlexRay block checks if the application has locked the tables by reading the SFTCCSR.ELKS lock status bit. If this bit is set, the FlexRay block will not update the table in this cycle. If this bit is cleared, the FlexRay block locks this table and starts the table update. To indicate that these tables are currently updated and may contain inconsistent data, the FlexRay block clears the even table valid status bit SFTCCSR.EVAL. Once all table entries related to the even cycle have been transferred into the FRM, the FlexRay block sets the even table valid bit SFTCCSR.EVAL and the Even Cycle Table Written Interrupt Flag EVT_IF in the Protocol Interrupt Flag Register 1 (PIFR1). If the interrupt enable flag EVT_IE is set, an interrupt request is generated.

To read the generated tables, the application must lock the tables to prevent the FlexRay block from updating these tables. The locking is initiated by writing a 1 to the even table lock trigger SFTCCSR.ELKT. When the even table is not currently updated by the FlexRay block, the lock is granted and the even table lock status bit SFTCCSR.ELKS is set. This indicates that the application has successfully locked the even sync tables and the corresponding status information fields SFRA, SFRB in

the Sync Frame Counter Register (SFCNTR). The value in the SFTCCSR.CYCNUM field provides the number of the cycle that this table is related to.

The number of available table entries per channel is provided in the SFCNTR.SFEVA and SFCNTR.SFEVB fields. The application can now start to read the sync table data from the locations given in Figure 13-138.

After reading all the data from the locked tables, the application must unlock the table by writing to the even table lock trigger SFTCCSR.ELKT again. The even table lock status bit SFTCCSR.ELKS is reset immediately.

If the sync frame table generation is disabled, the table valid bits SFTCCSR.EVAL and SFTCCSR.EVAL are reset when the counter values in the Sync Frame Counter Register (SFCNTR) are updated. This is done because the tables stored in the FRM are no longer related to the values in the Sync Frame Counter Register (SFCNTR).

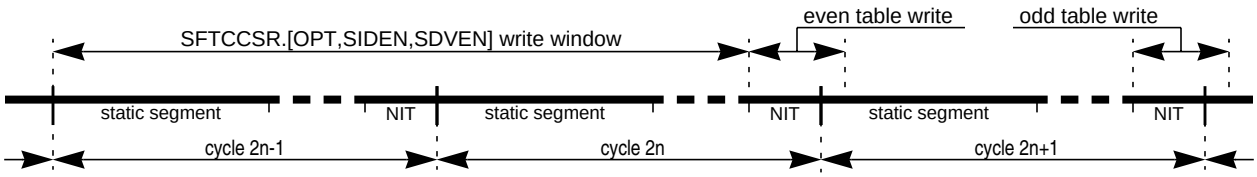


Figure 13-139. Sync Frame Table Trigger and Generation Timing

13.6.12.5 Sync Frame Table Access

The sync frame tables will be transferred into the FRM during the table write windows shown in Figure 13-139. During the table write, the application can not lock the table that is currently written. If the application locks the table outside of the table write window, the lock is granted immediately.

13.6.12.5.1 Sync Frame Table Locking and Unlocking

The application locks the even/odd sync frame table by writing 1 to the lock trigger bit ELKT/OLKT in the Sync Frame Table Configuration, Control, Status Register (SFTCCSR). If the affected table is not currently written to the FRM, the lock is granted immediately, and the lock status bit ELKS/OLKS is set. If the affected table is currently written to the FRM, the lock is not granted. In this case, the application must issue the lock request again until the lock is granted.

The application unlocks the even/odd sync frame table by writing 1 to the lock trigger bit ELKT/OLKT. The lock status bit ELKS/OLKS is cleared immediately.

13.6.13 MTS Generation

The FlexRay block provides a flexible means to request the transmission of the Media Access Test Symbol MTS in the symbol window on channel A or channel B.

The application can configure the set of communication cycles in which the MTS will be transmitted over the FlexRay bus by programming the CYCCNTMSK and CYCCNTVAL fields in the MTS A Configuration Register (MTSACFR) and MTS B Configuration Register (MTSBCFR).

The application enables or disables the generation of the MTS on either channel by setting or clearing the MTE control bit in the MTS A Configuration Register (MTSACFR) or MTS B Configuration Register (MTSBCFR). If an MTS is to be transmitted in a certain communication cycle, the application must set the MTE control bit during the static segment of the preceding communication cycle.

The MTS is transmitted over channel A in the communication cycle with number CYCCNT, if Equation 13-17, Equation 13-18, and Equation 13-18 are fulfilled.

$$\text{PSR0[PROTSTATE]} = \textit{POC:normal active} \qquad \textit{Eqn. 13-16}$$

$$\text{MTSACRF[MTE]} = 1 \qquad \textit{Eqn. 13-17}$$

$$\text{CYCCNT} \wedge \text{MTSACFR[CYCCNTMSK]} = \text{MTSACFR[CYCCNTVAL]} \wedge \text{MTSACFR[CYCCNTMSK]} \qquad \textit{Eqn. 13-18}$$

The MTS is transmitted over channel B in the communication cycle with number CYCCNT, if Equation 13-16, Equation 13-19, and Equation 13-20 are fulfilled.

$$\text{MTSBCRF[MTE]} = 1 \qquad \textit{Eqn. 13-19}$$

$$\text{CYCCNT} \wedge \text{MTSBCFR[CYCCNTMSK]} = \text{MTSBCFR[CYCCNTVAL]} \wedge \text{MTSBCFR[CYCCNTMSK]} \qquad \textit{Eqn. 13-20}$$

13.6.14 Sync Frame and Startup Frame Transmission

The transmission of sync frames and startup frames is controlled by the following register fields:

- PCR18.key_slot_id: provides the number of the slot for sync or startup frame transmission
- PCR11.key_slot_used_for_sync: indicates sync frame transmission
- PCR11.key_slot_used_for_startup: indicates startup frame transmission
- PCR12.key_slot_header_crc: provides header crc for sync frame or startup frame
- Message Buffer with message buffer number n=PCR18.key_slot_id

The generation of the sync or startup frames depends on the current protocol state. In the *POC:startup* state, the generation is independent of the message buffer setup; in the *POC:normal active* state, the generation is affected by the current message buffer setup.

13.6.14.1 Sync Frame and Startup Frame Transmission in *POC:startup*

In the *POC:startup* state, the sync and startup frame transmission is independent of the message buffer setup. If at least one of the indication bits PCR11.key_slot_used_for_sync or PCR11.key_slot_used_for_startup is set, a Null Frame will be transmitted in the slot with slot number PCR18.key_slot_id. The header CRC for this Null Frame is taken from PCR12.key_slot_header_crc. The settings of the sync and startup frame indicators are taken from PCR11.key_slot_used_for_sync and PCR11.key_slot_used_for_startup.

13.6.14.2 Sync Frame and Startup Frame Transmission in *POC:normal active*

In the *POC:normal active* state, the sync and startup frame transmission depends on the message buffer setup. If at least one of the indication bits PCR11.key_slot_used_for_sync or

PCR11.key_slot_used_for_startup is set, or if a transmit message buffer with MBFIDRn.FID == PCR18.key_slot_id is configured and enabled, a Null Frame or Data Frame will be transmitted in the slot with slot number PCR18.key_slot_id. The header CRC for this frame is taken from PCR12.key_slot_header_crc, the settings of the sync and startup frame indicators are taken from PCR11.key_slot_used_for_sync and PCR11.key_slot_used_for_startup. A data frame will be transmitted if the message buffer is unlocked and committed and the cycle counter filter matches the current cycle.

13.6.15 Sync Frame Filtering

Each received synchronization frame must pass the Sync Frame Acceptance Filter and the Sync Frame Rejection Filter before it is considered for clock synchronization. If the synchronization frame filtering is globally disabled, i.e. the SFFE control bit in the Module Configuration Register (MCR) is cleared, all received synchronization frames are considered for clock synchronization. If a received synchronization frame did not pass at least one of the two filters, this frame is processed as a normal frame and is not considered for clock synchronization.

13.6.15.1 Sync Frame Acceptance Filtering

The synchronization frame acceptance filter is implemented as a value-mask filter. The value is configured in the Sync Frame ID Acceptance Filter Value Register (SFIDAFVR) and the mask is configured in the Sync Frame ID Acceptance Filter Mask Register (SFIDAFMR). A received synchronization frame with the frame ID FID passes the sync frame acceptance filter, if Equation 13-21 or Equation 13-22evaluates to true.

$$MCR[SFFE] = 0 \qquad \textit{Eqn. 13-21}$$

$$FID[9:0] \wedge SFIDAFMR[FMSK[9:0]] = SFIDAFVR[FVAL[9:0]] \wedge SFIDAFMR[FMSK[9:0]] \qquad \textit{Eqn. 13-22}$$

NOTE

Sync frames are transmitted in the static segment only. Thus FID <= 1023.

13.6.15.2 Sync Frame Rejection Filtering

The synchronization frame rejection filter is a comparator. The compare value is defined by the Sync Frame ID Rejection Filter Register (SFIDRFR). A received synchronization frame with the frame ID FID passes the sync frame rejection filter if Equation 13-23 or Equation 13-24 evaluates to true.

$$MCR[SFFE] = 0 \qquad \textit{Eqn. 13-23}$$

$$FID[9:0] \neq SFIDRFR[SYNFRID[9:0]] \qquad \textit{Eqn. 13-24}$$

NOTE

Sync frames are transmitted in the static segment only. Thus FID <= 1023.

13.6.16 Strobe Signal Support

The FlexRay block provides a number of strobe signals for observing internal protocol timing related signals in the protocol engine. The signals are listed and described in Table 13-13.

13.6.16.1 Strobe Signal Assignment

Each of the strobe signals listed in Table 13-13 can be assigned to one of the four strobe ports using the Strobe Signal Control Register (STBSCR). To assign multiple strobe signals, the application must write multiple times to the Strobe Signal Control Register (STBSCR) with appropriate settings.

To read out the current settings for a strobe signal with number N, the application must execute the following sequence.

1. Write to STBSCR with WMD = 1 and SEL = N. (updates SEL field only)
2. Read STBCSR.
   The SEL field provides N and the ENB and STBPSEL fields provides the settings for signal N.

13.6.16.2 Strobe Signal Timing

This section provides detailed timing information of the strobe signals with respect to the protocol engine clock.

The strobe signals display internal PE signals. Due to the internal architecture of the PE, some signals are generated several PE clock cycles before the actual action is performed on the FlexRay Bus. These signals are listed in Table 13-13 with a negative clock offset. An example waveform is given in Figure 13-140.

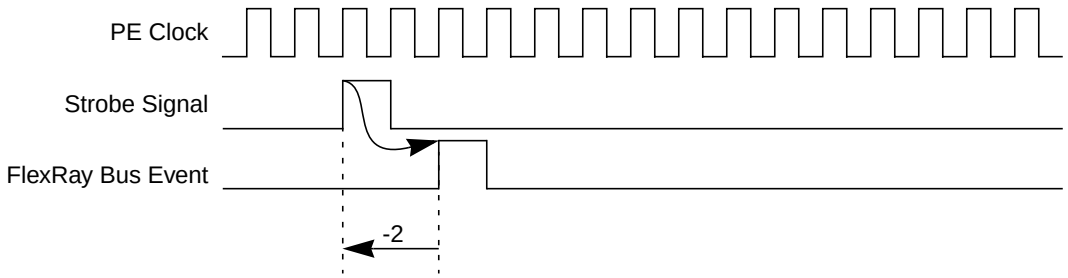


Figure 13-140. Strobe Signal Timing (type = pulse, clk_offset = -2)

Other signals refer to events that occurred on the FlexRay Bus some cycles before the strobe signal is changed. These signals are listed in Table 13-13 with a positive clock offset. An example waveform is given in Figure 13-141.

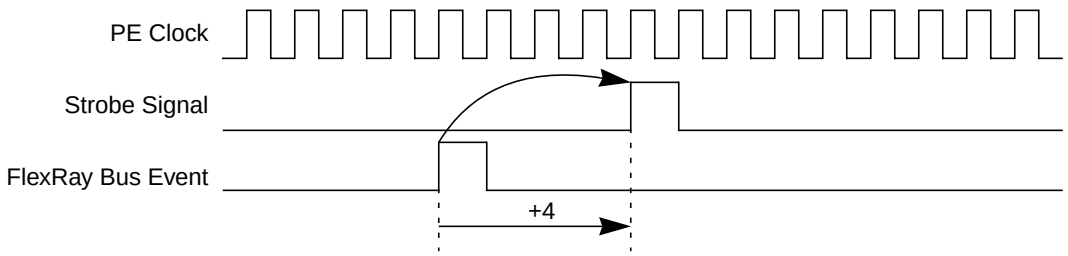


Figure 13-141. Strobe Signal Timing (type = pulse, clk_offset = +4)

13.6.17 Timer Support

The FlexRay block provides two timers, which run on the FlexRay time base. Each timer generates a maskable interrupt when it reaches a configured point in time. Timer T1 is an absolute timer. Timer T2 can be configured to be an absolute or a relative timer. Both timers can be configured to be repetitive. In the non-repetitive mode, timer stops if it expires. In repetitive mode, timer is restarted when it expires.

Both timers are active only when the protocol is in *POC:normal active* or *POC:normal passive* state. If the protocol is not in one of these modes, the timers are stopped. The application must restart the timers when the protocol has reached the *POC:normal active* or *POC:normal passive* state.

13.6.17.1 Absolute Timer T1

The absolute timer T1 has the protocol cycle count and the macrotick count as the time base. The timer 1 interrupt flag TI1_IF in the Protocol Interrupt Flag Register 0 (PIFR0) is set at the macrotick start event, if Equation 13-25 and Equation 13-26 are fulfilled

$$\text{CYCCTR.CYCCNT \& T1CYSR.T1\_CYC\_MSK} == \text{T1CYSR.T1\_CYC\_VAL \& T1CYSR.T1\_CYC\_MSK} \quad \textit{Eqn. 13-25}$$

$$\text{MTCTR.MTCT} == \text{TI1MTOR.T1\_MTOFFSET} \quad \textit{Eqn. 13-26}$$

If the timer 1 interrupt enable bit TI1_IE in the Protocol Interrupt Enable Register 0 (PIER0) is asserted, an interrupt request is generated.

The status bit T1ST is set when the timer is triggered, and is cleared when the timer expires and is non-repetitive. If the timer expires but is repetitive, the T1ST bit is not cleared and the timer is restarted immediately. The T1ST is cleared when the timer is stopped.

13.6.17.2 Absolute / Relative Timer T2

The timer T2 can be configured to be an absolute or relative timer by setting the T2_CFG control bit in the Timer Configuration and Control Register (TICCR). The status bit T2ST is set when the timer is triggered, and is cleared when the timer expires and is non-repetitive. If the timer expires but is repetitive, the T2ST bit is not cleared and the timer is restarted immediately. The T2ST is cleared when the timer is stopped.

13.6.17.2.1 Absolute Timer T2

If timer T2 is configured as an absolute timer, it has the same functionality timer T1 but the configuration from Timer 2 Configuration Register 0 (TI2CR0) and Timer 2 Configuration Register 1 (TI2CR1) is used.

On expiration of timer T2, the interrupt flag TI2_IF in the Protocol Interrupt Flag Register 0 (PIFR0) is set. If the timer 1 interrupt enable bit TI1_IE in the Protocol Interrupt Enable Register 0 (PIER0) is asserted, an interrupt request is generated.

13.6.17.2.2 Relative Timer T2

If the timer T2 is configured as a relative timer, the interrupt flag TI2_IF in the Protocol Interrupt Flag Register 0 (PIFR0) is set, when the programmed amount of macroticks MT[31:0], defined by Timer 2 Configuration Register 0 (TI2CR0) and Timer 2 Configuration Register 1 (TI2CR1), has expired since the trigger or restart of timer 2. The relative timer is implemented as a down counter and expires when it has reached 0. At the macrotick start event, the value of MT[31:0] is checked and then decremented. Thus, if the timer is started with MT[31:0] == 0, it expires at the next macrotick start.

13.6.18 Slot Status Monitoring

The FlexRay block provides several means for slot status monitoring. All slot status monitors use the same slot status vector provided by the PE. The PE provides a slot status vector for each static slot, for each dynamic slot, for the symbol window, and for the NIT, on a per channel base. The content of the slot status vector is described in Table 13-113. The PE provides the slot status vector within the first macrotick after the end of the related slot/window/NIT, as shown in Figure 13-142.

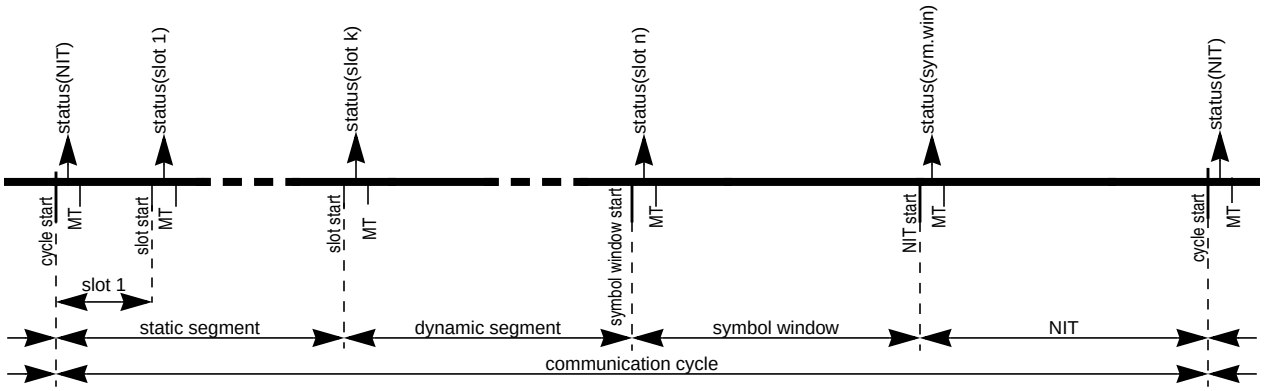


Figure 13-142. Slot Status Vector Update

NOTE

The slot status for the NIT of cycle n is provided after the start of cycle n+1.

Table 13-114. Slot Status Content

	Status Content
static / dynamic Slot	**slot related status** *vSS!ValidFrame* - valid frame received *vSS!SyntaxError* - syntax error occurred while receiving *vSS!ContentError* - content error occurred while receiving *vSS!BViolation* - boundary violation while receiving *for slots in which the module transmits:* *vSS!TxConflict* - reception ongoing while transmission starts *for slots in which the module does not transmit:* *vSS!TxConflict* - reception ongoing while transmission starts first valid - channel that has received the first valid frame **received frame related status** *extracted from* *a) header of valid frame, if vSS!ValidFrame = 1* *b) last received header, if vSS!ValidFrame = 0* *c) set to 0, if nothing was received* *vRF!Header!NFIndicator* - Null Frame Indicator (0 for null frame) *vRF!Header!SuFIndicator* - Startup Frame Indicator *vRF!Header!SyFIndicator* - Sync Frame Indicator
Symbol Window	**window related status** *vSS!ValidFrame* - always 0 *vSS!ContentError* - content error occurred while receiving *vSS!SyntaxError* - syntax error occurred while receiving *vSS!BViolation* - boundary violation while receiving *vSS!TxConflict* - reception ongoing while transmission starts **received symbol related status** *vSS!ValidMTS* - valid Media Test Access Symbol received **received frame related status** *see static/dynamic slot*
NIT	**NIT related status** *vSS!ValidFrame* - always 0 *vSS!ContentError* - content error occurred while receiving *vSS!SyntaxError* - syntax error occurred while receiving *vSS!BViolation* - boundary violation while receiving *vSS!TxConflict* - always 0 **received frame related status** *see static/dynamic slot*

13.6.18.1 Channel Status Error Counter Registers

The two channel status error counter registers, Channel A Status Error Counter Register (CASERCR) and Channel B Status Error Counter Register (CBSERCR), incremented by one, if at least one of four slot status error bits, *vSS!SyntaxError*, *vSS!ContentError*, *vSS!BViolation*, or *vSS!TxConflict* is set to 1. The status vectors for all slots in the static and dynamic segment, in the symbol window, and in the NIT are taken into account. The counters wrap round after they have reached the maximum value.

13.6.18.2 Protocol Status Registers

The Protocol Status Register 2 (PSR2) provides slot status information about the Network Idle Time NIT and the Symbol Window. The Protocol Status Register 3 (PSR3) provides aggregated slot status information.

13.6.18.3 Slot Status Registers

The eight slot status registers, Slot Status Registers (SSR0–SSR7), can be used to observe the status of static slots, dynamic slots, the symbol window, or the NIT without individual message buffers. These registers provide all slot status related and received frame / symbol related status information, as given in Table 13-113, except of the *first valid* indicator for non-transmission slots.

13.6.18.4 Slot Status Counter Registers

The FlexRay block provides four slot status error counter registers, Slot Status Counter Registers (SSCR0–SSCR3). Each of these slot status counter registers is updated with the value of an internal slot status counter at the start of a communication cycle. The internal slot status counter is incremented if its increment condition, defined by the Slot Status Counter Condition Register (SSCCR), matches the status vector provided by the PE. All static slots, the symbol window, and the NIT status are taken into account. *Dynamic* slots are *excluded*. The internal slot status counting and update timing is shown in Figure 13-143.

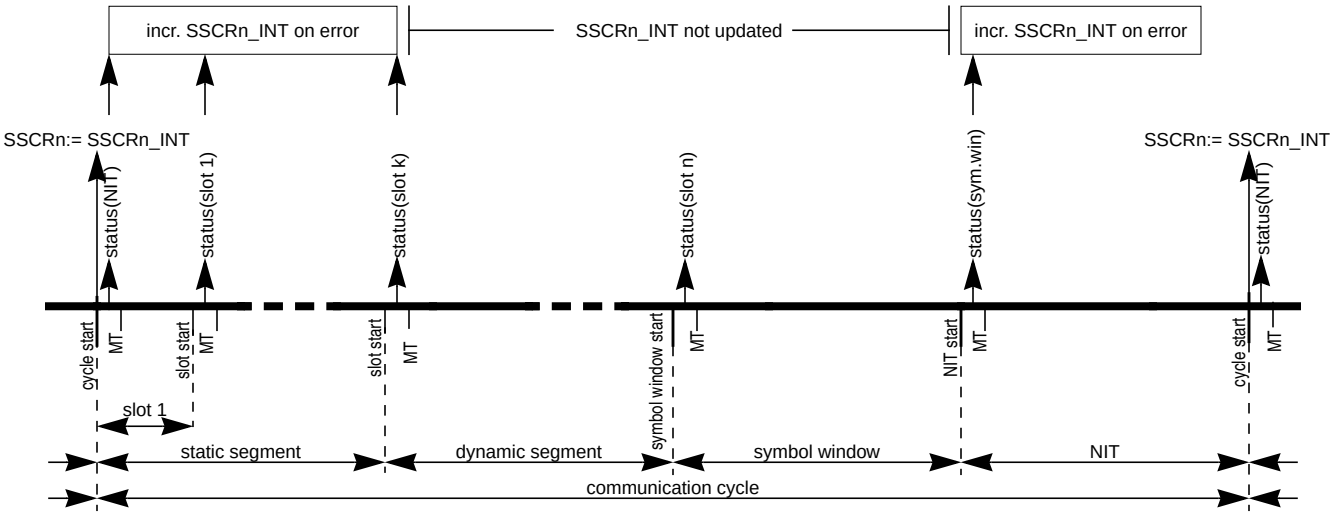


Figure 13-143. Slot Status Counting and SSCRn Update

The PE provides the status of the NIT in the first slot of the next cycle. Due to these facts, the SSCRn register reflects, in cycle n, the status of the NIT of cycle n-2, and the status of all static slots and the symbol window of cycle n-1.

The increment condition for each slot status counter consists of two parts, the frame related condition part and the slot related condition part. The internal slot status counter SSCRn_INT is incremented if at least one of the conditions is fulfilled:

1. frame related condition:

- (SSCCRn.VFR | SSCCRn.SYF | SSCCRn.NUF | SSCCRn.SUF) // count on frame condition = 1;

and

- ((~SSCCRn.VFR | *vSS!ValidFrame*) & // valid frame restriction
  (~SSCCRn.SYF | *vRF!Header!SyFIndicator*) & // sync frame indicator restriction
  (~SSCCRn.NUF | ~*vRF!Header!NFIndicator*) & // null frame indicator restriction
  (~SSCCRn.SUF | *vRF!Header!SuFIndicator*)) // startup frame indicator restriction
  = 1;

NOTE

The indicator bits SYF, NUF, and SUF are valid only when a valid frame was received. Thus it is required to set the VFR always, whenever count on frame condition is used.

2. slot related condition:

- ((SSCCRn.STATUSMASK[3] & *vSS!ContentError*) | // increment on content error
  (SSCCRn.STATUSMASK[2] & *vSS!SyntaxError*) | // increment on syntax error
  (SSCCRn.STATUSMASK[1] & *vSS!BViolation*) | // increment on boundary violation
  (SSCCRn.STATUSMASK[0] & *vSS!TxConflict*)) // increment on transmission conflict
  = 1;

If the slot status counter is in single cycle mode, i.e. SSCCRn.MCY = 0, the internal slot status counter SSCRn_INT is reset at each cycle start. If the slot status counter is in the multicycle mode, i.e. SSCCRn.MCY = 1, the counter is not reset and incremented, until the maximum value is reached.

13.6.18.5 Message Buffer Slot Status Field

Each individual message buffer and each FIFO message buffer provides a slot status field, which provides the information shown in Table 13-113 for the static/dynamic slot. The update conditions for the slot status field depend on the message buffer type. Refer to the Message Buffer Update Sections in Section 13.6.6, “Individual Message Buffer Functional Description”.

13.6.19 Interrupt Support

The FlexRay block provides 76 individual interrupt sources and five combined interrupt sources.

13.6.19.1 Individual Interrupt Sources

13.6.19.1.1 Message Buffer Interrupts

The FlexRay block provides 32 message buffer interrupt sources.

Each individual message buffer provides an interrupt flag MBCCSn.MBIF and an interrupt enable bit MBCCSn.MBIE. The FlexRay block sets the interrupt flag when the slot status of the message buffer was updated. If the interrupt enable bit is asserted, an interrupt request is generated.

13.6.19.1.2 Receive FIFO Interrupts

The FlexRay block provides 2 Receive FIFO interrupt sources.

Each of the 2 Receive FIFO provides a Receive FIFO Not Empty Interrupt Flag. The FlexRay block sets the Receive FIFO Not Empty Interrupt Flags (GIFER.FNEBIF, GIFER.FNEAIF) in the Global Interrupt Flag and Enable Register (GIFER) if the corresponding Receive FIFO is not empty.

13.6.19.1.3 Wakeup Interrupt

The FlexRay block provides one interrupt source related to the wakeup.

The FlexRay block sets the Wakeup Interrupt Flag GIFER.WUPIF when it has received a wakeup symbol on the FlexRay bus. The FlexRay block generates an interrupt request if the interrupt enable bit GIFER.WUPIE is asserted.

13.6.19.1.4 Protocol Interrupts

The FlexRay block provides 25 interrupt sources for protocol related events. For details, see Protocol Interrupt Flag Register 0 (PIFR0) and Protocol Interrupt Flag Register 1 (PIFR1). Each interrupt source has its own interrupt enable bit.

13.6.19.1.5 CHI Error Interrupts

The FlexRay block provides 16 interrupt sources for CHI related error events. For details, see CHI Error Flag Register (CHIERFR). There is one common interrupt enable bit GIFER.CHIIE for all CHI error interrupt sources.

13.6.19.2 Combined Interrupt Sources

Each combined interrupt source generates an interrupt request only when at least one of the interrupt sources that is combined generates an interrupt request.

13.6.19.2.1 Receive Message Buffer Interrupt

The combined receive message buffer interrupt request RBIRQ is generated when at least one of the individual receive message buffers generates an interrupt request MBXIRQ[n] and the interrupt enable bit GIFER.RBIE is set.

13.6.19.2.2 Transmit Message Buffer Interrupt

The combined transmit message buffer interrupt request TBIRQ is generated when at least one of the individual transmit message buffers generates an interrupt request MBXIRQ[n] and the interrupt enable bit GIFER.TBIE is asserted.

13.6.19.2.3 Protocol Interrupt

The combined protocol interrupt request PRTIRQ is generated when at least one of the individual protocol interrupt sources generates an interrupt request and the interrupt enable bit GIFER.PRIE is set.

13.6.19.2.4 CHI Error Interrupt

The combined CHI error interrupt request CHIIRQ is generated when at least one of the individual chi error interrupt sources generates an interrupt request and the interrupt enable bit GIFER.CHIE is set.

13.6.19.2.5 Module Interrupt

The combined module interrupt request MIRQ is generated if at least one of the combined interrupt sources generates an interrupt request and the interrupt enable bit GIFER.MIE is set.

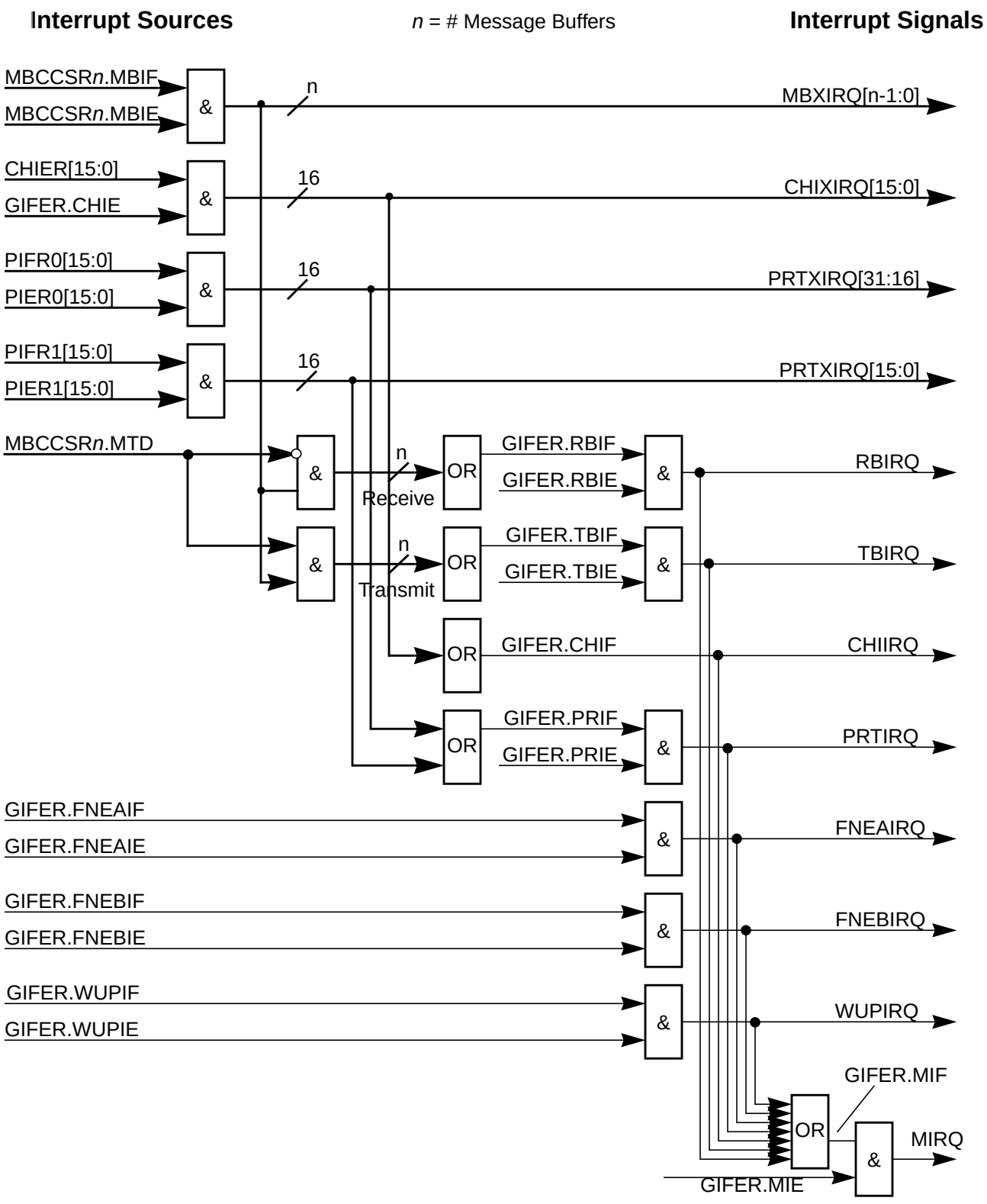


Figure 13-144. Scheme of cascaded interrupt request

Figure 13-145.

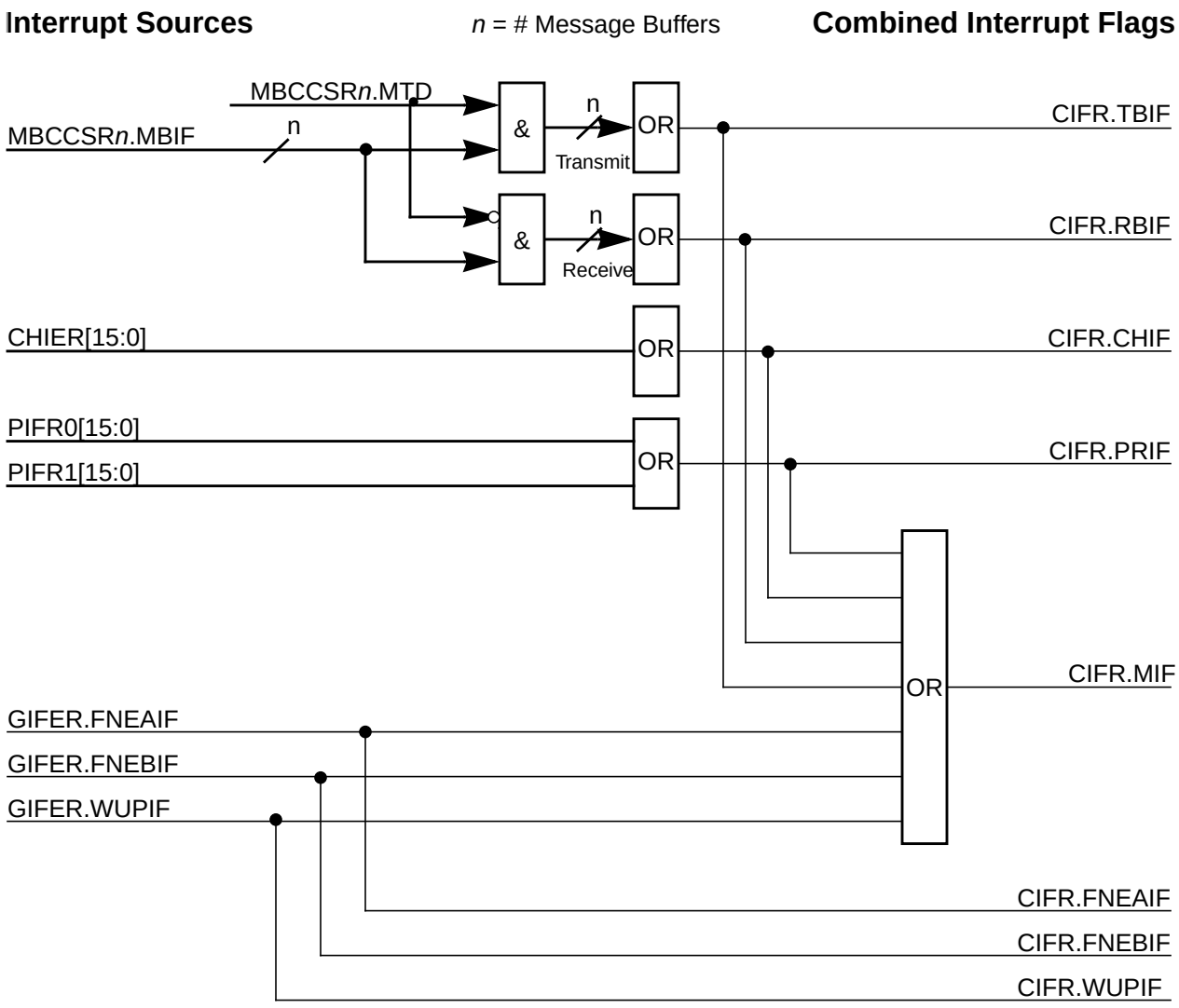


Figure 13-146. Scheme of combined interrupt flags

13.6.20 Lower Bit Rate Support

The FlexRay block supports a number of lower bit rates on the FlexRay bus channels. The lower bit rates are implemented by modifying the duration of the microtick *pdMicrotick*, the number of samples per microtick *pSamplesPerMicrotick*, the number of samples per bit *cSamplesPerBit*, and the strobe offset *cStrobeOffset.* The application configures the FlexRay channel bit rate by setting the BITRATE field in the Module Configuration Register (MCR). The protocol values are set internally. The available bit rates, the related BITRATE field configuration settings and related protocol parameter values are shown in Table 13-114.

Table 13-115. FlexRay Channel Bit Rate Control

FlexRay Channel Bit Rate [Mbit/s]	MCR.BITRATE	*pdMicrotick* [ns]	*gdSampleClockPeriod* [ns]	*pSamplesPerMicrotick*	*cSamplesPerBit*	*cStrobeOffset*
10.0	000	25.0	12.5	2	8	5
8.0	011	25.0	12.5	2	10	6
5.0	001	25.0	25.0	1	8	5
2.5	010	50.0	50.0	1	8	5

NOTE

The bit rate of 8 Mbit/s is not defined by the *FlexRay Communications System Protocol Specification, Version 2.1 Rev A.*

13.7 Application Information

13.7.1 Initialization Sequence

This section describes the required steps to initialize the FlexRay block. The first subsection describes the steps required after a system reset, the second section describes the steps required after preceding shutdown of the FlexRay block.

13.7.1.1 Module Initialization

This section describes the module related initialization steps after a system reset.

1. Configure FlexRay block.
   a) configure the control bits in the Module Configuration Register (MCR)
   b) configure the system memory base address in System Memory Base Address High Register (SYMBADHR) and System Memory Base Address Low Register (SYMBADLR)
2. Enable the FlexRay block.
   a) write 1 to the module enable bit MEN in the Module Configuration Register (MCR)

The FlexRay block now enters the Normal Mode. The application can commence with the protocol initialization described in Section 13.7.1.2, “Protocol Initialization”.

13.7.1.2 Protocol Initialization

This section describes the protocol related initialization steps.

1. Configure the Protocol Engine.
   a) issue CONFIG command via Protocol Operation Control Register (POCR)
   b) wait for *POC:config* in Protocol Status Register 0 (PSR0)
   c) configure the PCR0,..., PCR30 registers to set all protocol parameters
2. Configure the Message Buffers and FIFOs.
   a) set the number of message buffers used and the message buffer segmentation in the Message Buffer Segment Size and Utilization Register (MBSSUTR)
   b) define the message buffer data size in the Message Buffer Data Size Register (MBDSR)
   c) configure each message buffer by setting the configuration values in the Message Buffer Configuration, Control, Status Registers (MBCCSRn), Message Buffer Cycle Counter Filter Registers (MBCCFRn), Message Buffer Frame ID Registers (MBFIDRn), Message Buffer Index Registers (MBIDXRn)
   d) configure the receive FIFOs
   e) issue CONFIG_COMPLETE command via Protocol Operation Control Register (POCR)
   f) wait for *POC:ready* in Protocol Status Register 0 (PSR0)

After this sequence, the FlexRay block is configured as a FlexRay node and is ready to integrate into the FlexRay cluster.

13.7.2 Shut Down Sequence

This section describes a secure shut down sequence to stop the FlexRay block gracefully. The main targets of this sequence are

- finish all ongoing reception and transmission
- do not corrupt FlexRay bus and do not disturb ongoing FlexRay bus communication

For a graceful shutdown the application shall perform the following tasks:

1. Disable all enabled message buffers.
   a) repeatedly write 1 to MBCCSRn[EDT] until MBCCSRn[EDS] == 0.
2. Stop Protocol Engine.
   a) issue HALT command via Protocol Operation Control Register (POCR)
   b) wait for *POC:halt* in Protocol Status Register 0 (PSR0)

13.7.3 Number of Usable Message Buffers

This section describes the relationship between the number of message buffers that can be utilized and the required minimum CHI clock frequency. Additional constraints for the minimum CHI clock frequency are given in Section 13.3, “Controller Host Interface Clocking”.

The FlexRay block uses a sequential search algorithm to determine the individual message buffer assigned or subscribed to the next slot. This search must be finished within one FlexRay slot. The shortest FlexRay slot is an empty dynamic slot. An empty dynamic slot is a minislot and consists of *gdMinislot* macroticks with a nominal duration of *gdMacrotick*. The minimum duration of a corrected macrotick is $gdMacrotick_{min}$ = 39 µT. This results in a minimum slot length of

$$\Delta_{slotmin} = 39 \cdot pdMicrotick \cdot gdMinislot \qquad \textit{Eqn. 13-27}$$

The search engine is located in the CHI and runs on the CHI clock. It evaluates one individual message buffer per CHI clock cycle. For internal status update and double buffer commit operations, and as a result of the clock domain crossing jitter, an additional amount of 10 CHI clock cycles is required to ensure correct operation. For a given number of message buffers and for a given CHI clock frequency f_{chi}, this results in a search duration of

$$\Delta_{search} = \frac{1}{f_{chi}} \cdot (\#\ \text{MessageBuffers} + 10) \qquad \textit{Eqn. 13-28}$$

The message buffer search must be finished within one slot which requires that Equation 13-29 must be fulfilled

$$\Delta_{search} \leq \Delta_{slotmin} \qquad \textit{Eqn. 13-29}$$

This results in the formula given in Equation 13-30 which determines the required minimum CHI frequency for a given number of message buffers that are utilized.

$$f_{chi} \geq \frac{\#\ \text{MessageBuffers} + 10}{39 \cdot pdMicrotick \cdot gdMinislot} \qquad \textit{Eqn. 13-30}$$

The minimum CHI frequency for a selected set of relevant protocol parameters is given in Table 13-115.

Table 13-116. Minimum f_{chi} [MHz] examples (32 message buffers)

pdMicrotick [ns]	*gdMinislot*					
	2	3	4	5	6	7
25.0	21.54	14.35	10.77	8.61	7.18	6.16
50.0	10.73	7.18	5.39	4.31	3.59	3.08

13.7.4 Protocol Control Command Execution

This section considers the issues of the protocol control command execution.

The application issues any of the protocol control commands listed in the POCCMD field of Table 13-17 by writing the command to the POCCMD field of the Protocol Operation Control Register (POCR). As a result the FlexRay block sets the BSY bit while the command is transferred to the PE. When the PE has accepted the command, the BSY flag is cleared. All commands are accepted by the PE.

The PE maintains a protocol command vector. For each command that was accepted by the PE, the PE sets the corresponding command bit in the protocol command vector. If a command is issued while the corresponding command bit is set, the command is not queued and is lost.

If the command execution block of the PE is idle, it selects the next accepted protocol command with the highest priority from the current protocol command vector according to the protocol control command priorities given in Table 13-116. If the current protocol state does not allow the execution of this protocol command (see POC state changes in *FlexRay Communications System Protocol Specification, Version 2.1 Rev A*) the FlexRay block asserts the illegal protocol command interrupt flag IPC_IF in the Protocol Interrupt Flag Register 1 (PIFR1). The protocol command is not executed in this case.

Some protocol commands may be interrupted by other commands or the detection of a fatal protocol error as indicated by Table 13-116. If the application issues the RESET, FREEZE, or READY command, or if the the PE detects a fatal protocol error, some commands already stored in the command vector will be removed from this vector.

Table 13-117. Protocol Control Command Priorities

Protocol Command	Priority	Interrupted By	Cleared and Terminated By
RESET	(highest) 1	none	
FREEZE	2		RESET
READY	3		RESET
CONFIG_COMPLETE	3		RESET
ALL_SLOTS	4	RESET, FREEZE, READY, CONFIG_COMPLET, fatal protocol error	RESET, FREEZE, READY, CONFIG_COMPLETE, fatal protocol error
ALLOW_COLDSTART	5		RESET
RUN	6		RESET, FREEZE, fatal protocol error
WAKEUP	7		RESET, FREEZE, fatal protocol error
DEFAULT_CONFIG	8		RESET, FREEZE, fatal protocol error
CONFIG	9		RESET
HALT	(lowest) 10		RESET, FREEZE, READY, CONFIG_COMPLETE, fatal protocol error

13.7.5 Protocol Reset Command

The section considers the issues of the protocol RESET command.

The application issues the protocol reset command by writing the RESET command code to the POCCMD field of the Protocol Operation Control Register (POCR). As a result, the PE stops its operation immediately, the FlexRay bus ports put into their idle state, and no more data or status information is sent to the CHI. The lack of PE signals stops all message buffer operations in the CHI. In particular, the message buffers that are currently under internal use remain internally locked. To overcome this message buffer internal lock situation, the application must put the protocol into the *POC:default config* state. This will release all internal message buffer locks.

The following sequence must be executed by the application to put the protocol into the *POC:default config* state.

1. Repeatedly send Protocol Command FREEZE via Protocol Operation Control Register (POCR), until the freeze bit FRZ in Protocol Status Register 1 (PSR1) is set.
2. Repeatedly send Protocol Command DEFAULT_CONFIG via Protocol Operation Control Register (POCR), until the freeze bit FRZ bit in Protocol Status Register 1 (PSR1) is cleared and the PROTSTATE field in Protocol Status Register 0 (PSR0) is set to *POC:default config*.

13.7.6 Message Buffer Search on Simple Message Buffer Configuration

This sections describes the message buffer search behavior for a simplified message buffer configuration. The receive FIFO behavior is not considered in this section.

13.7.6.1 Simple Message Buffer Configuration

A simple message buffer configuration is a configuration that has at most one transmit message buffer and at most one receive message buffer assigned to a slot *S*. The simple configuration used in this section utilizes two message buffers, one single buffered transmit message buffer and one receive message buffer.

The transmit message buffer has the message buffer number *t* and has following configuration

Table 13-118. Transmit Buffer Configuration

Register	Field	Value	Description
MBCCSR*t*	MCM	-	used only for double buffers
	MBT	0	single transmit buffer
	MTD	1	transmit buffer
MBCCFR*t*	MTM	0	event transition mode
	CHA	1	assigned to channel A
	CHB	0	not assigned to channel B
	CCFE	1	cycle counter filter enabled
	CCFMSK	000011	cycle set = {4n} = {0,4,8,12,...}
	CCFVAL	000000	
MBFIDR*t*	FID	*S*	assigned to slot S

The availability of data in the transmit buffer is indicated by the commit bit MBCCSRt[CMT] and the lock bit MBCCSRt[LCKS].

The receive message buffer has the message buffer number *r* and has following configuration

Table 13-119. Receive Buffer Configuration

Register	Field	Value	Description
MBCCSR*r*	MCM	-	n/a
	MBT	-	n/a
	MTD	0	receive buffer
MBCCF*r*	MTM	-	n/a
	CHA	1	assigned to channel A
	CHB	0	not assigned to channel B
	CCFE	1	cycle counter filter enabled
	CCFMSK	000001	cycle set = {2n} = {0,2,4,6, ... }
	CCFVAL	000000	
MBFIDR*r*	FID	*S*	subscribed slot

Furthermore the assumption is that both message buffers are enabled (MBCCSR*t*[EDS] = 1 and MBCCSR*r*[EDS] = 1)

NOTE

The cycle set {4n+2} = {2,6,10, ...} is assigned to the receive buffer only.

The cycle set {4n} = {0,4,8,12, ...} is assigned to both buffers.

13.7.6.2 Behavior in static segment

In this case, both message buffers are assigned to a slot *S* in the *static* segment.

The configuration of a transmit buffer for a static slot *S* assigns this slot to the node as a transmit slot. The FlexRay protocol requires:

- When a slot occurs, if the slot is assigned to a node on a channel that node must transmit either a normal frame or a null frame on that channel. Specifically, a null frame will be sent if there is no data ready, or if there is no match on a transmit filter (cycle counter filtering, for example).

Regardless of the availability of data and the cycle counter filter, the node will transmit a frame in the static slot *S*. In any case, the result of the message buffer search will be the transmit message buffer *t*. The receive message buffer *r* will not be found, no reception is possible.

13.7.6.3 Behavior in dynamic segment

In this case, both message buffers are assigned to a slot *S* in the *dynamic* segment. The FlexRay protocol requires:

- When a slot occurs, if a slot is assigned to a node on a channel that node only transmits a frame on that channel if there is data ready and there is a match on relevant transmit filters (no null frames are sent).

The transmission of a frame in the dynamic segment is determined by the availability of data and the match of the cycle counter filter of the transmit message buffer.

13.7.6.3.1 Transmit Data Not Available

If transmit data are *not available*, i.e. the transmit buffer is not committed MBCCSR*t*[CMT]=0 and/or locked MBCCSR*t*[LCKS]=1,

c) for the cycles in the set {4n}, which is assigned to both buffers, the receive buffer will be found and the node can receive data, and

d) for the cycles in the set {4n+2}, which is assigned to the receive buffer only, the receive buffer will be found and the node can receive data.

The receive cycles are shown in Figure 13-147

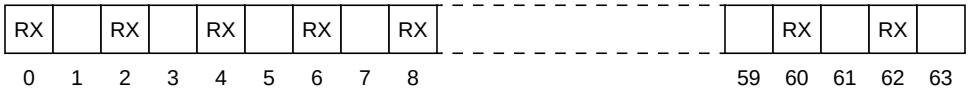


Figure 13-147. Transmit Data Not Available

13.7.6.3.2 Transmit Data Available

If transmit data are *available,* i.e. the transmit buffer is committed MBCCSR*t*[CMT]=1 and not locked MBCCSR*t*[LCKS]=0,

e) for the cycles in the set {4n}, which is assigned to both buffers, the transmit buffer will be found and the node transmits data.

f) for the cycles in the set {4n+2}, which is assigned to the receive buffer only, the receive buffer will be found and the node can receive data.

The receive and transmit cycles are shown in Figure 13-147

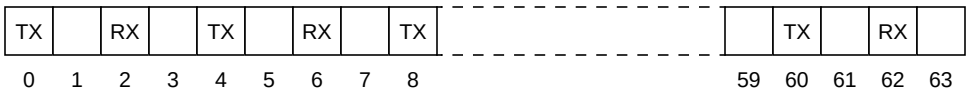


Figure 13-148. Transmit Data Not Available

Chapter 14
XGATE (S12XGATEV3)

Table 14-1. Revision History

Revision Number	Revision Date	Sections Affected	Description of Changes
V03.22	06 Oct 2005		- Internal updates
V03.23	14 Dec 2005	14.9.2/14-725	- Updated code example
V03.24	17 Jan 2006		- Internal updates

14.1 Introduction

The XGATE module is a peripheral co-processor that allows autonomous data transfers between the MCU's peripherals and the internal memories. It has a built in RISC core that is able to pre-process the transferred data and perform complex communication protocols.

The XGATE module is intended to increase the MCU's data throughput by lowering the S12X_CPU's interrupt load.

Figure 14-1 gives an overview on the XGATE architecture.

This document describes the functionality of the XGATE module, including:

- XGATE registers (Section 14.3, "Memory Map and Register Definition")
- XGATE RISC core (**Section 14.4.1, "XGATE RISC Core**")
- Hardware semaphores (Section 14.4.4, "Semaphores")
- Interrupt handling (Section 14.5, "Interrupts")
- Debug features (Section 14.6, "Debug Mode")
- Security (Section 14.7, "Security")
- Instruction set (Section 14.8, "Instruction Set")

14.1.1 Glossary of Terms

XGATE Request

A service request from a peripheral module which is directed to the XGATE by the S12X_INT module (see Figure 14-1). Each XGATE request attempts to activate a XGATE channel at a certain priority level.

XGATE Channel

The resources in the XGATE module (i.e. Channel ID number, Priority level, Service Request Vector, Interrupt Flag) which are associated with a particular XGATE Request.

XGATE Channel ID

A 7-bit identifier associated with an XGATE channel. In S12XE designs valid Channel IDs range from $0D to $78.

XGATE Priority Level

A priority ranging from 1 to 7 which is associated with an XGATE channel. The priority level of an XGATE channel is selected in the S12X_INT module.

XGATE Register Bank

A register bank consists of registers R1-R7, CCR and the PC. Each interrupt level is associated with one register bank.

XGATE Channel Interrupt

An S12X_CPU interrupt that is triggered by a code sequence running on the XGATE module.

XGATE Software Channel

Special XGATE channel that is not associated with any peripheral service request. A Software Channel is triggered by its Software Trigger Bit which is implemented in the XGATE module.

XGATE Semaphore

A set of hardware flip-flops that can be exclusively set by either the S12X_CPU or the XGATE. (see Section 14.4.4, “Semaphores”)

XGATE Thread

A code sequence which is executed by the XGATE’s RISC core after receiving an XGATE request.

XGATE Debug Mode

A special mode in which the XGATE’s RISC core is halted for debug purposes. This mode enables the XGATE’s debug features (see Section 14.6, “Debug Mode”).

XGATE Software Error

The XGATE is able to detect a number of error conditions caused by erratic software (see Section 14.4.5, “Software Error Detection”). These error conditions will cause the XGATE to seize program execution and flag an Interrupt to the S12X_CPU.

Word

A 16 bit entity.

Byte

An 8 bit entity.

14.1.2 Features

The XGATE module includes these features:

- Data movement between various targets (i.e. Flash, RAM, and peripheral modules)
- Data manipulation through built in RISC core

- Provides up to 108 XGATE channels, including 8 software triggered channels
- Interruptible thread execution
- Two register banks to support fast context switching between threads
- Hardware semaphores which are shared between the S12X_CPU and the XGATE module
- Able to trigger S12X_CPU interrupts upon completion of an XGATE transfer
- Software error detection to catch erratic application code

14.1.3 Modes of Operation

There are four run modes on S12XE devices.

- Run mode, wait mode, stop mode

  The XGATE is able to operate in all of these three system modes. Clock activity will be automatically stopped when the XGATE module is idle.
- Freeze mode (BDM active)

  In freeze mode all clocks of the XGATE module may be stopped, depending on the module configuration (see Section 14.3.1.1, “XGATE Control Register (XGMCTL)”).

14.1.4 Block Diagram

Figure 14-1 shows a block diagram of the XGATE.

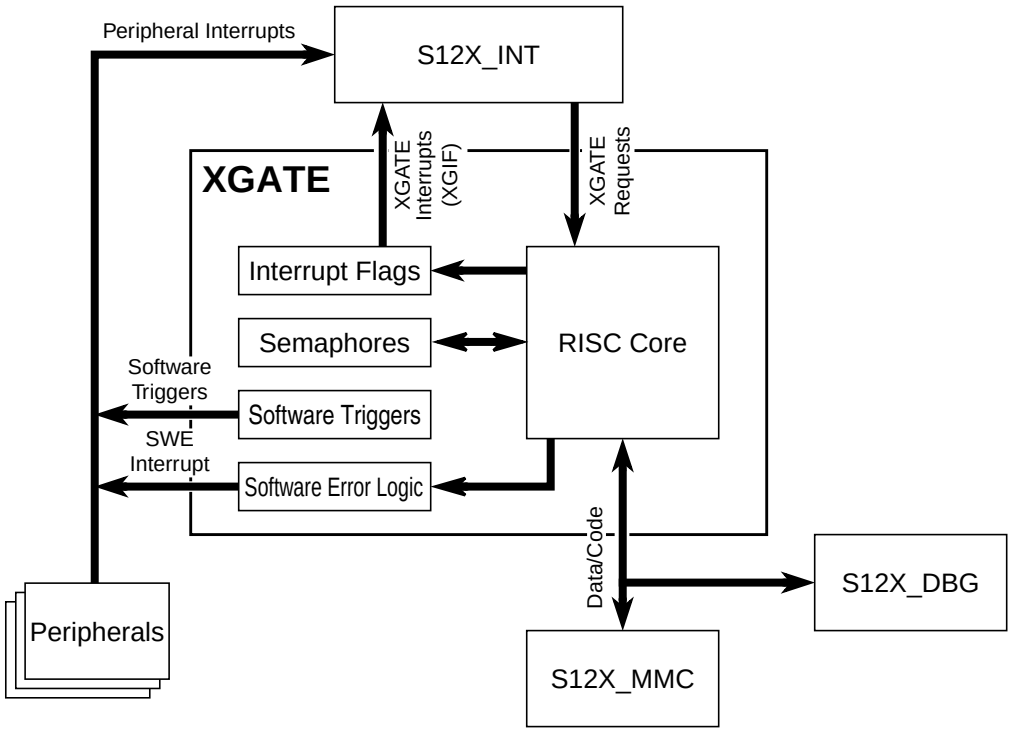


Figure 14-1. XGATE Block Diagram

14.2 External Signal Description

The XGATE module has no external pins.

14.3 Memory Map and Register Definition

This section provides a detailed description of address space and registers used by the XGATE module.

The memory map for the XGATE module is given below in Figure 14-2.The address listed for each register is the sum of a base address and an address offset. The base address is defined at the SoC level and the address offset is defined at the module level. Reserved registers read zero. Write accesses to the reserved registers have no effect.

14.3.1 Register Descriptions

This section consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number. Details of register bits and field functions follow the register diagrams, in bit order.

Register Name		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0000 XGMCTL	R	0	0	0	0	0	0	0	0	XGE	XGFRZ	XGDBG	XGSS	XG FACT	0	XG SWEF	XGIE
	W	XGEM	XG FRZM	XG DBGM	XGSSM	XG FACTM		XG SWEFM	XGIEM								
0x0002 XGCHID	R									0	XGCHID[6:0]						
	W																
0x0003 XGCHPL	R									0	0	0	0	0	XGCHPL[2:0]		
	W																
0x0004 Reserved	R																
	W																
0x0005 XGISPSEL	R									0	0	0	0	0	0	XGISPSEL[1:0]	
	W																
0x0006 XGISP74	R	XGISP74[15:1]															0
	W																
0x0006 XGISP31	R	XGISP31[15:1]															0
	W																
0x0006 XGVBR	R	XGVBR[15:1]															0
	W																

= Unimplemented or Reserved

Figure 14-2. XGATE Register Summary (Sheet 1 of 3)

		127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112
0x0008 XGIF	R	0	0	0	0	0	0	0	XGIF_78	XGF_77	XGIF_76	XGIF_75	XGIF_74	XGIF_73	XGIF_72	XGIF_71	XGIF_70
	W																

		111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96
0x000A XGIF	R	XGIF_6F	XGIF_6E	XGIF_6D	XGIF_6C	XGIF_6B	XGIF_6A	XGIF_69	XGIF_68	XGF_67	XGIF_66	XGIF_65	XGIF_64	XGIF_63	XGIF_62	XGIF_61	XGIF_60
	W																

		95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80
0x000C XGIF	R	XGIF_5F	XGIF_5E	XGIF_5D	XGIF_5C	XGIF_5B	XGIF_5A	XGIF_59	XGIF_58	XGF_57	XGIF_56	XGIF_55	XGIF_54	XGIF_53	XGIF_52	XGIF_51	XGIF_50
	W																

		79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64
0x000E XGIF	R	XGIF_4F	XGIF_4E	XGIF_4D	XGIF_4C	XGIF_4B	XGIF_4A	XGIF_49	XGIF_48	XGF_47	XGIF_46	XGIF_45	XGIF_44	XGIF_43	XGIF_42	XGIF_41	XGIF_40
	W																

		63	62	61	60	59	58	57	56	55	54	53	52	51	50	49	48
0x0010 XGIF	R	XGIF_3F	XGIF_3E	XGIF_3D	XGIF_3C	XGIF_3B	XGIF_3A	XGIF_39	XGIF_38	XGF_37	XGIF_36	XGIF_35	XGIF_34	XGIF_33	XGIF_32	XGIF_31	XGIF_30
	W																

		47	46	45	44	43	42	41	40	39	38	37	36	35	34	33	32
0x0012 XGIF	R	XGIF_2F	XGIF_2E	XGIF_2D	XGIF_2C	XGIF_2B	XGIF_2A	XGIF_29	XGIF_28	XGF_27	XGIF_26	XGIF_25	XGIF_24	XGIF_23	XGIF_22	XGIF_21	XGIF_20
	W																

		31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0x0014 XGIF	R	XGIF_1F	XGIF_1E	XGIF_1D	XGIF_1C	XGIF_1B	XGIF_1A	XGIF_19	XGIF_18	XGF_17	XGIF_16	XGIF_15	XGIF_14	XGIF_13	XGIF_12	XGIF_11	XGIF_10
	W																

		15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0016 XGIF	R	XGIF_0F	XGIF_0E	XGIF_0D	0	0	0	0	0	0	0	0	0	0	0	0	0
	W																

= Unimplemented or Reserved

Figure 14-2. XGATE Register Summary (Sheet 2 of 3)

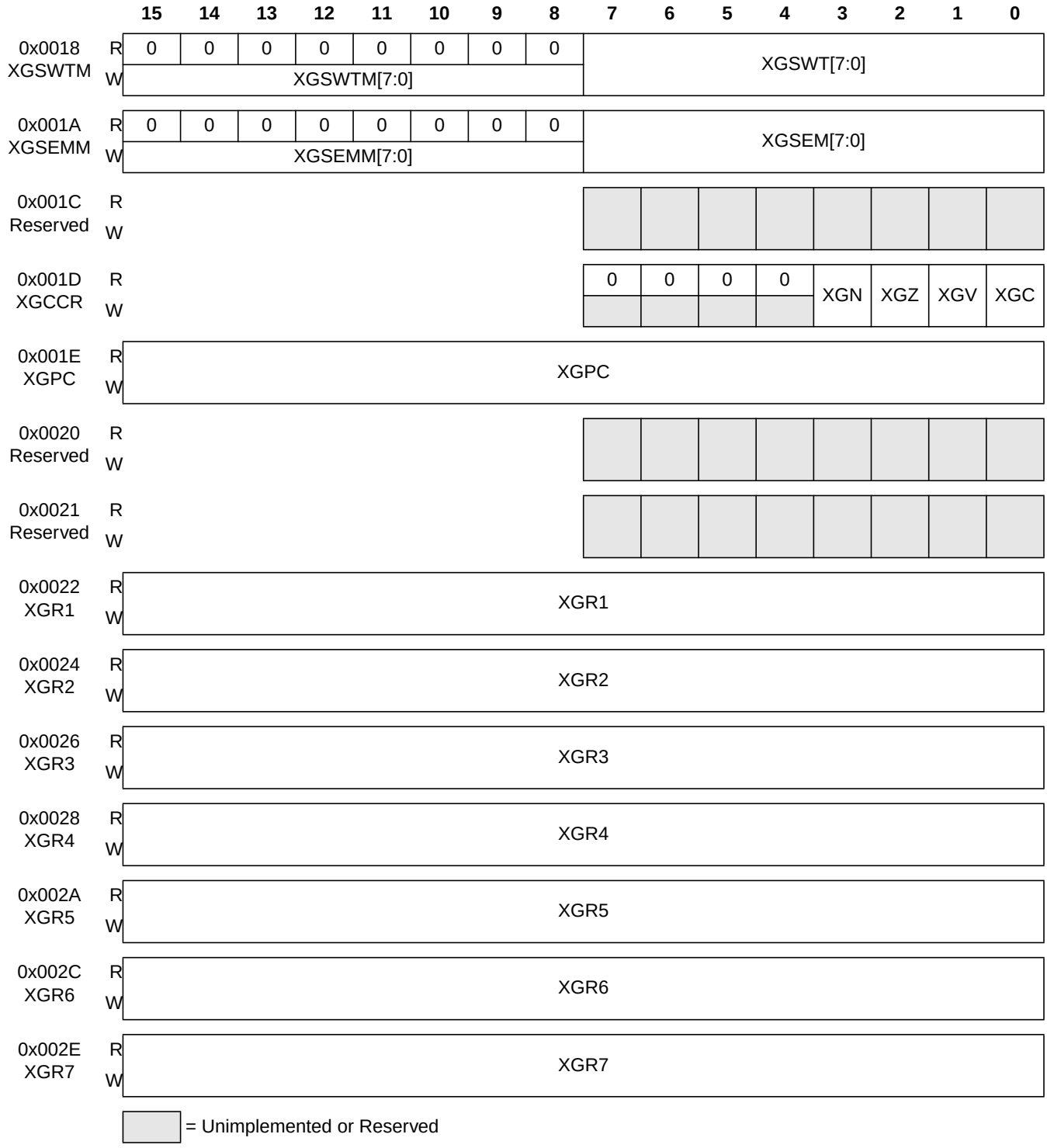


Figure 14-2. XGATE Register Summary (Sheet 3 of 3)

14.3.1.1 XGATE Control Register (XGMCTL)

All module level switches and flags are located in the XGATE Module Control Register Figure 14-3.

Module Base +0x00000

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	XGE	XGFRZ	XGDBG	XGSS	XGFACT	0	XG SWEF	XGIE
W	XGEM	XG FRZM	XG DBGM	XG SSM	XG FACTM		XG SWEFM	XGIEM								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 14-3. XGATE Control Register (XGMCTL)

Read: Anytime

Write: Anytime

Table 14-2. XGMCTL Field Descriptions (Sheet 1 of 3)

Field	Description
15 XGEM	**XGE Mask** — This bit controls the write access to the XGE bit. The XGE bit can only be set or cleared if a "1" is written to the XGEM bit in the same register access. Read: This bit will always read "0". Write: 0 Disable write access to the XGE in the same bus cycle 1 Enable write access to the XGE in the same bus cycle
14 XGFRZM	**XGFRZ Mask** — This bit controls the write access to the XGFRZ bit. The XGFRZ bit can only be set or cleared if a "1" is written to the XGFRZM bit in the same register access. Read: This bit will always read "0". Write: 0 Disable write access to the XGFRZ in the same bus cycle 1 Enable write access to the XGFRZ in the same bus cycle
13 XGDBGM	**XGDBG Mask** — This bit controls the write access to the XGDBG bit. The XGDBG bit can only be set or cleared if a "1" is written to the XGDBGM bit in the same register access. Read: This bit will always read "0". Write: 0 Disable write access to the XGDBG in the same bus cycle 1 Enable write access to the XGDBG in the same bus cycle
12 XGSSM	**XGSS Mask** — This bit controls the write access to the XGSS bit. The XGSS bit can only be set or cleared if a "1" is written to the XGSSM bit in the same register access. Read: This bit will always read "0". Write: 0 Disable write access to the XGSS in the same bus cycle 1 Enable write access to the XGSS in the same bus cycle

Table 14-2. XGMCTL Field Descriptions (Sheet 2 of 3)

Field	Description
11 XGFACTM	**XGFACT Mask** — This bit controls the write access to the XGFACT bit. The XGFACT bit can only be set or cleared if a "1" is written to the XGFACTM bit in the same register access. Read: This bit will always read "0". Write: 0 Disable write access to the XGFACT in the same bus cycle 1 Enable write access to the XGFACT in the same bus cycle
9 XGSWEFM	**XGSWEF Mask** — This bit controls the write access to the XGSWEF bit. The XGSWEF bit can only be cleared if a "1" is written to the XGSWEFM bit in the same register access. Read: This bit will always read "0". Write: 0 Disable write access to the XGSWEF in the same bus cycle 1 Enable write access to the XGSWEF in the same bus cycle
8 XGIEM	**XGIE Mask** — This bit controls the write access to the XGIE bit. The XGIE bit can only be set or cleared if a "1" is written to the XGIEM bit in the same register access. Read: This bit will always read "0". Write: 0 Disable write access to the XGIE in the same bus cycle 1 Enable write access to the XGIE in the same bus cycle
7 XGE	**XGATE Module Enable (Request Enable)**— This bit enables incoming XGATE requests from the S12X_INT module. If the XGE bit is cleared, pending XGATE requests will be ignored. The thread that is executed by the RISC core while the XGE bit is cleared will continue to run. Read: 0 Incoming requests are disabled 1 Incoming requests are enabled Write: 0 Disable incoming requests 1 Enable incoming requests
6 XGFRZ	**Halt XGATE in Freeze Mode** — The XGFRZ bit controls the XGATE operation in Freeze Mode (BDM active). Read: 0 RISC core operates normally in Freeze (BDM active) 1 RISC core stops in Freeze Mode (BDM active) Write: 0 Don’t stop RISC core in Freeze Mode (BDM active) 1 Stop RISC core in Freeze Mode (BDM active)
5 XGDBG	**XGATE Debug Mode** — This bit indicates that the XGATE is in Debug Mode (see Section 14.6, “Debug Mode”). Debug Mode can be entered by Software Breakpoints (BRK instruction), Tagged or Forced Breakpoints (see **S12X_DBG Section**), or by writing a "1" to this bit. Read: 0 RISC core is not in Debug Mode 1 RISC core is in Debug Mode Write: 0 Leave Debug Mode 1 Enter Debug Mode **Note:** Freeze Mode and Software Error Interrupts have no effect on the XGDBG bit.

Table 14-2. XGMCTL Field Descriptions (Sheet 3 of 3)

Field	Description
4 XGSS	**XGATE Single Step** — This bit forces the execution of a single instruction.(1) Read: 0 No single step in progress 1 Single step in progress Write 0 No effect 1 Execute a single RISC instruction **Note:** Invoking a Single Step will cause the XGATE to temporarily leave Debug Mode until the instruction has been executed.
3 XGFACT	**Fake XGATE Activity** — This bit forces the XGATE to flag activity to the MCU even when it is idle. When it is set the MCU will never enter system stop mode which assures that peripheral modules will be clocked during XGATE idle periods Read: 0 XGATE will only flag activity if it is not idle or in debug mode. 1 XGATE will always signal activity to the MCU. Write: 0 Only flag activity if not idle or in debug mode. 1 Always signal XGATE activity.
1 XGSWEF	**XGATE Software Error Flag** — This bit signals a software error. It is set whenever the RISC core detects an error condition(2). The RISC core is stopped while this bit is set. Clearing this bit will terminate the current thread and cause the XGATE to become idle. Read: 0 No software error detected 1 Software error detected Write: 0 No effect 1 Clears the XGSWEF bit
0 XGIE	**XGATE Interrupt Enable** — This bit acts as a global interrupt enable for the XGATE module Read: 0 All outgoing XGATE interrupts disabled (except software error interrupts) 1 All outgoing XGATE interrupts enabled Write: 0 Disable all outgoing XGATE interrupts (except software error interrupts) 1 Enable all outgoing XGATE interrupts

1. Refer to Section 14.6.1, “Debug Features”
2. Refer to Section 14.4.5, “Software Error Detection”

14.3.1.2 XGATE Channel ID Register (XGCHID)

The XGATE Channel ID Register (Figure 14-4) shows the identifier of the XGATE channel that is currently active. This register will read “$00” if the XGATE module is idle. In debug mode this register can be used to start and terminate threads. Refer to Section 14.6.1, “Debug Features” for further information.

Module Base +0x0002

Figure 14-4. XGATE Channel ID Register (XGCHID)

Read: Anytime

Write: In Debug Mode[1]

Table 14-3. XGCHID Field Descriptions

Field	Description
6–0 XGCHID[6:0]	**Request Identifier** — ID of the currently active channel

14.3.1.3 XGATE Channel Priority Level (XGCHPL)

The XGATE Channel Priority Level Register (Figure 14-5) shows the priority level of the current thread. In debug mode this register can be used to select a priority level when launching a thread (see Section 14.6.1, “Debug Features”).

Module Base +0x0003

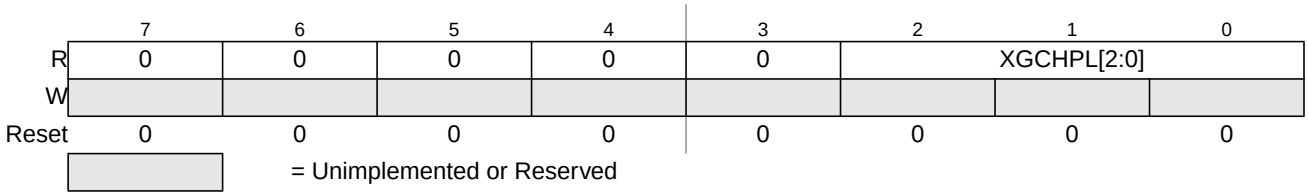


Figure 14-5. XGATE Channel Priority Level Register (XGCHPL)

Read: Anytime

Write: In Debug Mode[1]

Table 14-4. XGCHPL Field Descriptions

Field	Description
2-0 XGCHPL[2:0]	**Priority Level**— Priority level of the currently active channel

14.3.1.4 XGATE Initial Stack Pointer Select Register (XGISPSEL)

The XGATE Initial Stack Pointer Select Register (Figure 14-6) determines the register which is mapped to address “Module Base +0x0006”. A value of zero selects the Vector Base Register (XGVBR). Setting

1. Refer to Section 14.6.1, “Debug Features”

this register to a channel priority level (non-zero value) selects the corresponding Initial Stack Pointer Registers XGISP74 or XGISP31 (see Table 14-6).

Module Base +0x0005

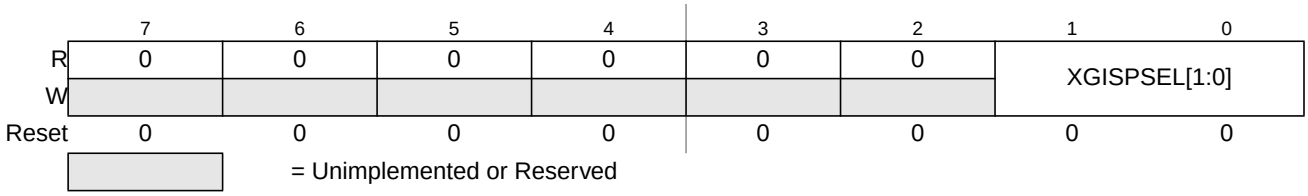


Figure 14-6. XGATE Initial Stack Pointer Select Register (XGISPSEL)

Read: Anytime

Write: Anytime

Table 14-5. XGISPSEL Field Descriptions

Field	Description
1-0 XGISPSEL[1:0]	**Register select**— Determines whether XGISP74, XGISP31, or XGVBR is mapped to “Module Base +0x0006”. See Table 14-6.

Table 14-6. XGISP74, XGISP31, XGVBR Mapping

XGISPSEL[1:0]	Register Mapped to “Module Base +0x0006“
3	Reserved
2	XGISP74
1	XGISP31
0	XGVBR

14.3.1.5 XGATE Initial Stack Pointer for Interrupt Priorities 7 to 4 (XGISP74)

The XGISP74 register is intended to point to the stack region that is used by XGATE channels of priority 7 to 4. Every time a thread of such priority is started, RISC core register R7 will be initialized with the content of XGISP74.

Module Base +0x0006

Figure 14-7. XGATE Initial Stack Pointer for Interrupt Priorities 7 to 4 (XGISP74)

Read: Anytime

Write: Only if XGATE requests are disabled (XGE = 0) and idle (XGCHID = $00))

Table 14-7. XGISP74 Field Descriptions

Field	Description
15–1 XBISP74[15:1]	**Initial Stack Pointer**— The XGISP74 register holds the initial value of RISC core register R7, for threads of priority 7 to 4.

14.3.1.6 XGATE Initial Stack Pointer for Interrupt Priorities 3 to 1 (XGISP31)

The XGISP31 register is intended to point to the stack region that is used by XGATE channels of priority 3 to 1. Every time a thread of such priority is started, RISC core register R7 will be initialized with the content of XGISP31.

Module Base +0x0006

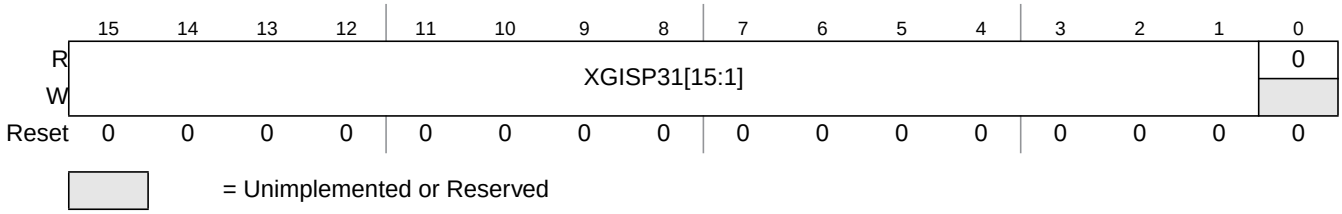


Figure 14-8. XGATE Initial Stack Pointer for Interrupt Priorities 3 to 1 (XGISP31)

Read: Anytime

Write: Only if XGATE requests are disabled (XGE = 0) and idle (XGCHID = $00))

Table 14-8. XGISP31 Field Descriptions

Field	Description
15–1 XBISP31[15:1]	**Initial Stack Pointer**— The XGISP31 register holds the initial value of RISC core register R7, for threads of priority 3 to 1.

14.3.1.7 XGATE Vector Base Address Register (XGVBR)

The Vector Base Address Register (Figure 14-9) determines the location of the XGATE vector block (see Section Figure 14-23., “XGATE Vector Block).

Module Base +0x0006

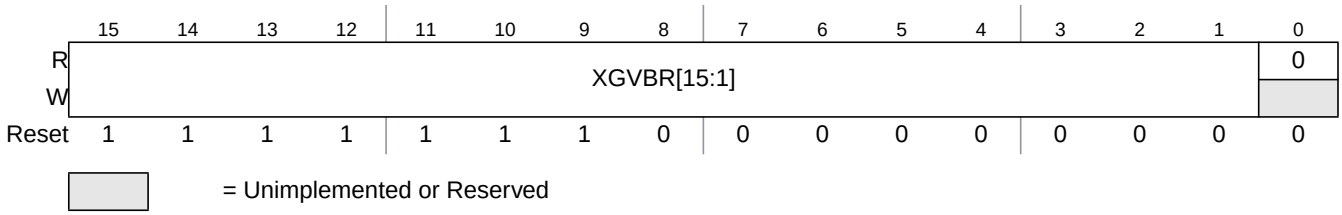


Figure 14-9. XGATE Vector Base Address Register (XGVBR)

Read: Anytime

Write: Only if XGATE requests are disabled (XGE = 0) and idle (XGCHID = $00))

Table 14-9. XGVBR Field Descriptions

Field	Description
15–1 XBVBR[15:1]	**Vector Base Address** — The XGVBR register holds the start address of the vector block in the XGATE memory map.

14.3.1.8 XGATE Channel Interrupt Flag Vector (XGIF)

The XGATE Channel Interrupt Flag Vector (Figure 14-10) provides access to the interrupt flags of all channels. Each flag may be cleared by writing a "1" to its bit location. Refer to Section 14.5.2, “Outgoing Interrupt Requests” for further information.

Module Base +0x0008

	127	126	125	124	123	122	121	120	119	118	117	116	115	114	113	112
R	0	0	0	0	0	0	0	XGIF_78	XGF_77	XGIF_76	XGIF_75	XGIF_74	XGIF_73	XGIF_72	XGIF_71	XGIF_70
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	111	110	109	108	107	106	105	104	103	102	101	100	99	98	97	96
R	XGIF_6F	XGIF_6E	XGIF_6D	XGIF_6C	XGIF_6B	XGIF_6A	XGIF_69	XGIF_68	XGF_67	XGIF_66	XGIF_65	XGIF_64	XGIF_63	XGIF_62	XGIF_61	XGIF_60
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	95	94	93	92	91	90	89	88	87	86	85	84	83	82	81	80
R	XGIF_5F	XGIF_5E	XGIF_5D	XGIF_5C	XGIF_5B	XGIF_5A	XGIF_59	XGIF_58	XGF_57	XGIF_56	XGIF_55	XGIF_54	XGIF_53	XGIF_52	XGIF_51	XGIF_50
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

	79	78	77	76	75	74	73	72	71	70	69	68	67	66	65	64
R	XGIF_4F	XGIF_4E	XGIF_4D	XGIF_4C	XGIF_4B	XGIF_4A	XGIF_49	XGIF_48	XGF _47	XGIF_46	XGIF_45	XGIF_44	XGIF_43	XGIF_42	XGIF_41	XGIF_40
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 14-10. XGATE Channel Interrupt Flag Vector (XGIF)

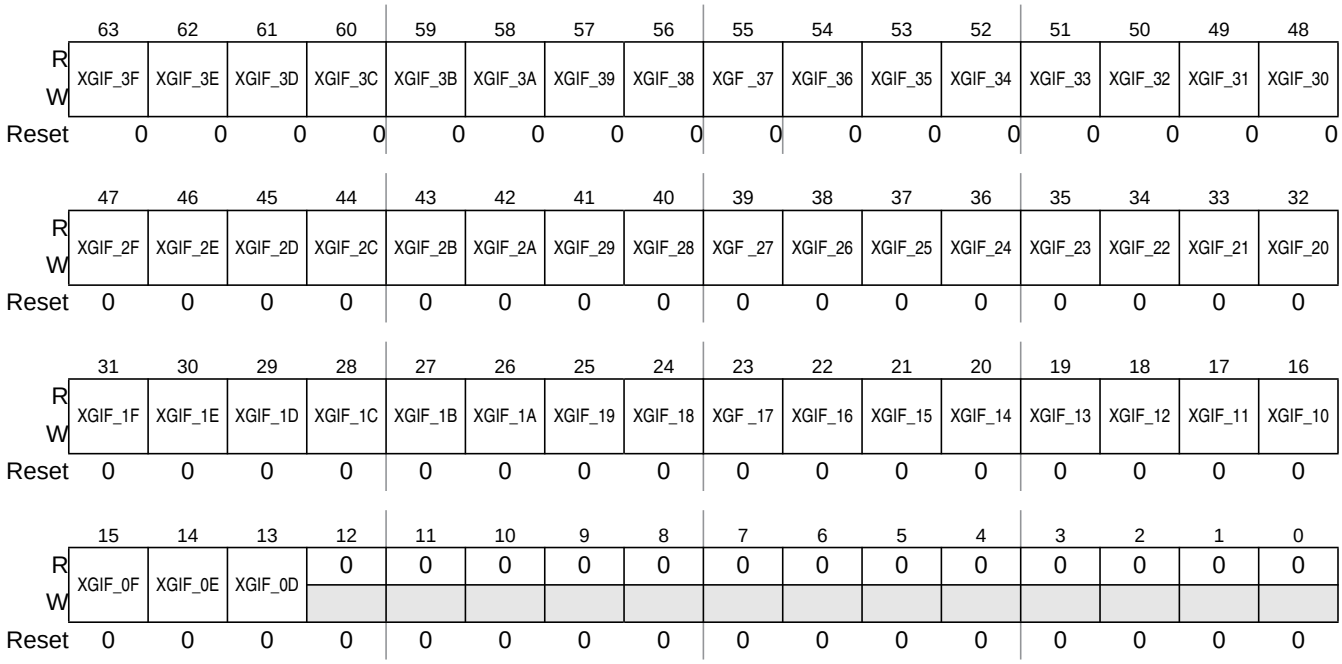


Figure 14-10. XGATE Channel Interrupt Flag Vector (XGIF) (continued)

Read: Anytime

Write: Anytime

Table 14-10. XGIV Field Descriptions

Field	Description
127–9 XGIF[78:9]	**Channel Interrupt Flags** — These bits signal pending channel interrupts. They can only be set by the RISC core (see SIF instruction on page 14-711). Each flag can be cleared by writing a "1" to its bit location. Unimplemented interrupt flags will always read "0". Section “Interrupts” of the **SoC Guide** for a list of implemented Interrupts. Read: 0 Channel interrupt is not pending 1 Channel interrupt is pending if XGIE is set Write: 0 No effect 1 Clears the interrupt flag

NOTE

Suggested Mnemonics for accessing the interrupt flag vector on a word basis are:

XGIF_7F_70 (XGIF[127:112]),
XGIF_6F_60 (XGIF[111:96]),
XGIF_5F_50 (XGIF[95:80]),
XGIF_4F_40 (XGIF[79:64]),
XGIF_3F_30 (XGIF[63:48]),
XGIF_2F_20 (XGIF[47:32]),
XGIF_1F_10 (XGIF[31:16]),
XGIF_0F_00 (XGIF[15:0])

14.3.1.9 XGATE Software Trigger Register (XGSWT)

The eight software triggers of the XGATE module can be set and cleared through the XGATE Software Trigger Register (Figure 14-11). The upper byte of this register, the software trigger mask, controls the write access to the lower byte, the software trigger bits. These bits can be set or cleared if a "1" is written to the associated mask in the same bus cycle. Refer to Section 14.5.2, “Outgoing Interrupt Requests” for further information.

Module Base +0x00018

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	XGSWT[7:0]							
W	XGSWTM[7:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 14-11. XGATE Software Trigger Register (XGSWT)

Read: Anytime

Write: Anytime

Table 14-11. XGSWT Field Descriptions

Field	Description
15–8 XGSWTM[7:0]	**Software Trigger Mask** — These bits control the write access to the XGSWT bits. Each XGSWT bit can only be written if a "1" is written to the corresponding XGSWTM bit in the same access. Read: These bits will always read "0". Write: 0 Disable write access to the XGSWT in the same bus cycle 1 Enable write access to the corresponding XGSWT bit in the same bus cycle
7–0 XGSWT[7:0]	**Software Trigger Bits** — These bits act as interrupt flags that are able to trigger XGATE software channels. They can only be set and cleared by software. Read: 0 No software trigger pending 1 Software trigger pending if the XGIE bit is set Write: 0 Clear Software Trigger 1 Set Software Trigger

NOTE

The XGATE channel IDs that are associated with the eight software triggers are determined on chip integration level. (see Section “Interrupts“ of the **Soc Guide**)

XGATE software triggers work like any peripheral interrupt. They can be used as XGATE requests as well as S12X_CPU interrupts. The target of the software trigger must be selected in the S12X_INT module.

14.3.1.10 XGATE Semaphore Register (XGSEM)

The XGATE provides a set of eight hardware semaphores that can be shared between the S12X_CPU and the XGATE RISC core. Each semaphore can either be unlocked, locked by the S12X_CPU or locked by the RISC core. The RISC core is able to lock and unlock a semaphore through its SSEM and CSEM instructions. The S12X_CPU has access to the semaphores through the XGATE Semaphore Register (Figure 14-12). Refer to section Section 14.4.4, “Semaphores” for details.

Module Base +0x0001A

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	XGSEM[7:0]							
W	XGSEMM[7:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 14-12. XGATE Semaphore Register (XGSEM)

Read: Anytime

Write: Anytime (see Section 14.4.4, “Semaphores”)

Table 14-12. XGSEM Field Descriptions

Field	Description
15–8 XGSEMM[7:0]	**Semaphore Mask** — These bits control the write access to the XGSEM bits. Read: These bits will always read "0". Write: 0 Disable write access to the XGSEM in the same bus cycle 1 Enable write access to the XGSEM in the same bus cycle
7–0 XGSEM[7:0]	**Semaphore Bits** — These bits indicate whether a semaphore is locked by the S12X_CPU. A semaphore can be attempted to be set by writing a "1" to the XGSEM bit and to the corresponding XGSEMM bit in the same write access. Only unlocked semaphores can be set. A semaphore can be cleared by writing a "0" to the XGSEM bit and a "1" to the corresponding XGSEMM bit in the same write access. Read: 0 Semaphore is unlocked or locked by the RISC core 1 Semaphore is locked by the S12X_CPU Write: 0 Clear semaphore if it was locked by the S12X_CPU 1 Attempt to lock semaphore by the S12X_CPU

14.3.1.11 XGATE Condition Code Register (XGCCR)

The XGCCR register (Figure 14-13) provides access to the RISC core's condition code register.

Module Base +0x001D

	7	6	5	4	3	2	1	0
R	0	0	0	0	XGN	XGZ	XGV	XGC
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 14-13. XGATE Condition Code Register (XGCCR)

Read: In debug mode if unsecured and not idle (XGCHID ≠ 0x00)

Write: In debug mode if unsecured and not idle (XGCHID ≠ 0x00)

Table 14-13. XGCCR Field Descriptions

Field	Description
3 XGN	**Sign Flag** — The RISC core's Sign flag
2 XGZ	**Zero Flag** — The RISC core's Zero flag
1 XGV	**Overflow Flag** — The RISC core's Overflow flag
0 XGC	**Carry Flag** — The RISC core's Carry flag

14.3.1.12 XGATE Program Counter Register (XGPC)

The XGPC register (Figure 14-14) provides access to the RISC core's program counter.

Module Base +0x0001E

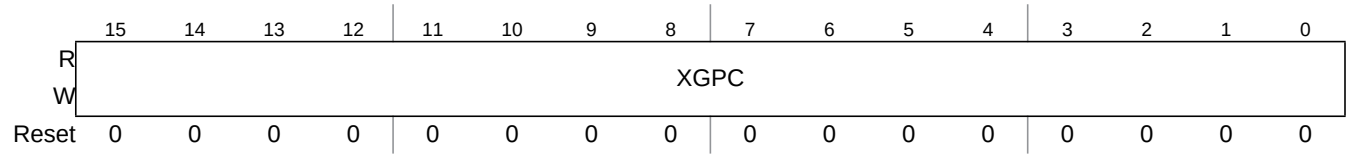


Figure 14-14. XGATE Program Counter Register (XGPC)

Read: In debug mode if unsecured and not idle (XGCHID ≠ 0x00)

Write: In debug mode if unsecured and not idle (XGCHID ≠ 0x00)

Table 14-14. XGPC Field Descriptions

Field	Description
15–0 XGPC[15:0]	**Program Counter** — The RISC core's program counter

14.3.1.13 XGATE Register 1 (XGR1)

The XGR1 register (Figure 14-15) provides access to the RISC core's register 1.

Module Base +0x00022

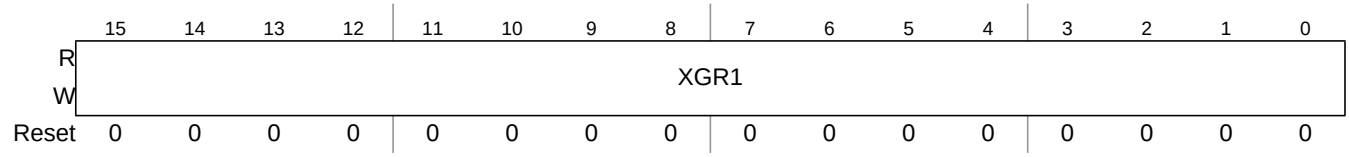


Figure 14-15. XGATE Register 1 (XGR1)

Read: In debug mode if unsecured and not idle (XGCHID ≠ 0x00)

Write: In debug mode if unsecured and not idle (XGCHID ≠ 0x00)

Table 14-15. XGR1 Field Descriptions

Field	Description
15–0 XGR1[15:0]	**XGATE Register 1** — The RISC core's register 1

14.3.1.14 XGATE Register 2 (XGR2)

The XGR2 register (Figure 14-16) provides access to the RISC core's register 2.

Module Base +0x00024

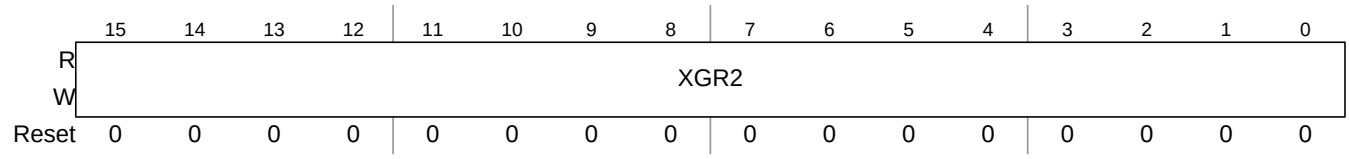


Figure 14-16. XGATE Register 2 (XGR2)

Read: In debug mode if unsecured and not idle (XGCHID ≠ 0x00)

Write: In debug mode if unsecured and not idle (XGCHID ≠ 0x00)

Table 14-16. XGR2 Field Descriptions

Field	Description
15–0 XGR2[15:0]	**XGATE Register 2** — The RISC core's register 2

14.3.1.15 XGATE Register 3 (XGR3)

The XGR3 register (Figure 14-17) provides access to the RISC core's register 3.

Module Base +0x00026

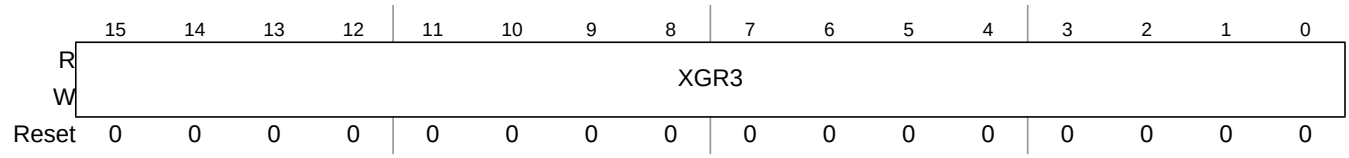


Figure 14-17. XGATE Register 3 (XGR3)

Read: In debug mode if unsecured and not idle (XGCHID ≠ 0x00)

Write: In debug mode if unsecured and not idle (XGCHID ≠ 0x00)

Table 14-17. XGR3 Field Descriptions

Field	Description
15–0 XGR3[15:0]	**XGATE Register 3** — The RISC core's register 3

14.3.1.16 XGATE Register 4 (XGR4)

The XGR4 register (Figure 14-18) provides access to the RISC core's register 4.

Module Base +0x00028

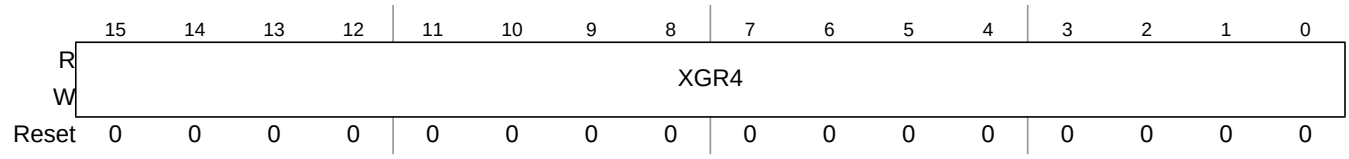


Figure 14-18. XGATE Register 4 (XGR4)

Read: In debug mode if unsecured and not idle (XGCHID ≠ 0x00)

Write: In debug mode if unsecured and not idle (XGCHID ≠ 0x00)

Table 14-18. XGR4 Field Descriptions

Field	Description
15–0 XGR4[15:0]	**XGATE Register 4** — The RISC core's register 4

14.3.1.17 XGATE Register 5 (XGR5)

The XGR5 register (Figure 14-19) provides access to the RISC core's register 5.

Module Base +0x0002A

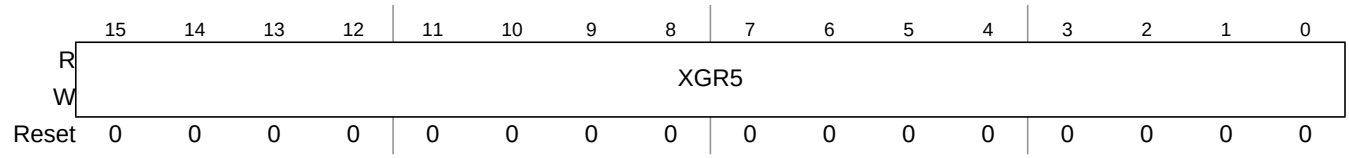


Figure 14-19. XGATE Register 5 (XGR5)

Read: In debug mode if unsecured and not idle (XGCHID ≠ 0x00)

Write: In debug mode if unsecured and not idle (XGCHID ≠ 0x00)

Table 14-19. XGR5 Field Descriptions

Field	Description
15–0 XGR5[15:0]	**XGATE Register 5** — The RISC core's register 5

14.3.1.18 XGATE Register 6 (XGR6)

The XGR6 register (Figure 14-20) provides access to the RISC core's register 6.

Module Base +0x0002C

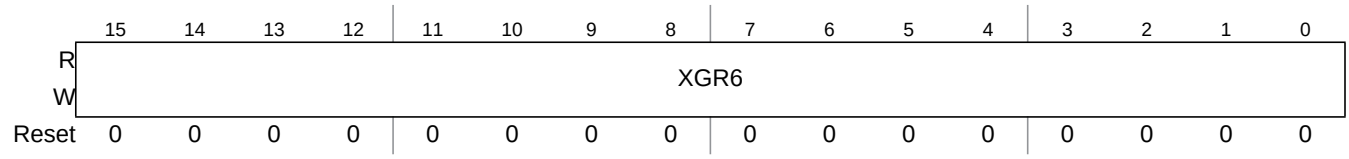


Figure 14-20. XGATE Register 6 (XGR6)

Read: In debug mode if unsecured and not idle (XGCHID ≠ 0x00)

Write: In debug mode if unsecured and not idle (XGCHID ≠ 0x00)

Table 14-20. XGR6 Field Descriptions

Field	Description
15–0 XGR6[15:0]	**XGATE Register 6** — The RISC core's register 6

14.3.1.19 XGATE Register 7 (XGR7)

The XGR7 register (Figure 14-21) provides access to the RISC core's register 7.

Module Base +0x0002E

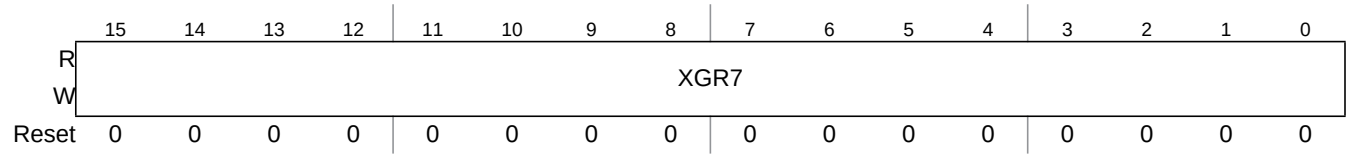


Figure 14-21. XGATE Register 7 (XGR7)

Read: In debug mode if unsecured and not idle (XGCHID ≠ 0x00)

Write: In debug mode if unsecured and not idle (XGCHID ≠ 0x00)

Table 14-21. XGR7 Field Descriptions

Field	Description
15–0 XGR7[15:0]	**XGATE Register 7** — The RISC core's register 7

14.4 Functional Description

The core of the XGATE module is a RISC processor which is able to access the MCU's internal memories and peripherals (see Figure 14-1). The RISC processor always remains in an idle state until it is triggered by an XGATE request. Then it executes a code sequence (thread) that is associated with the requested XGATE channel. Each thread can run on a priority level ranging from 1 to 7. Refer to the **S12X_INT**

Section for information on how to select priority levels for XGATE threads. Low priority threads (interrupt levels 1 to 3) can be interrupted by high priority threads (interrupt levels 4 to 7). High priority threads are not interruptible. The register content of an interrupted thread is maintained and restored by the XGATE hardware.

To signal the completion of a task the XGATE is able to send interrupts to the S12X_CPU. Each XGATE channel has its own interrupt vector. Refer to the **S12X_INT Section** for detailed information.

The XGATE module also provides a set of hardware semaphores which are necessary to ensure data consistency whenever RAM locations or peripherals are shared with the S12X_CPU.

The following sections describe the components of the XGATE module in further detail.

14.4.1 XGATE RISC Core

The RISC core is a 16 bit processor with an instruction set that is well suited for data transfers, bit manipulations, and simple arithmetic operations (see Section 14.8, “Instruction Set”).

It is able to access the MCU’s internal memories and peripherals without blocking these resources from the S12X_CPU[1]. Whenever the S12X_CPU and the RISC core access the same resource, the RISC core will be stalled until the resource becomes available again.[1]

The XGATE offers a high access rate to the MCU’s internal RAM. Depending on the bus load, the RISC core can perform up to two RAM accesses per S12X_CPU bus cycle.

Bus accesses to peripheral registers or flash are slower. A transfer rate of one bus access per S12X_CPU cycle can not be exceeded.

The XGATE module is intended to execute short interrupt service routines that are triggered by peripheral modules or by software.

14.4.2 Programmer’s Model

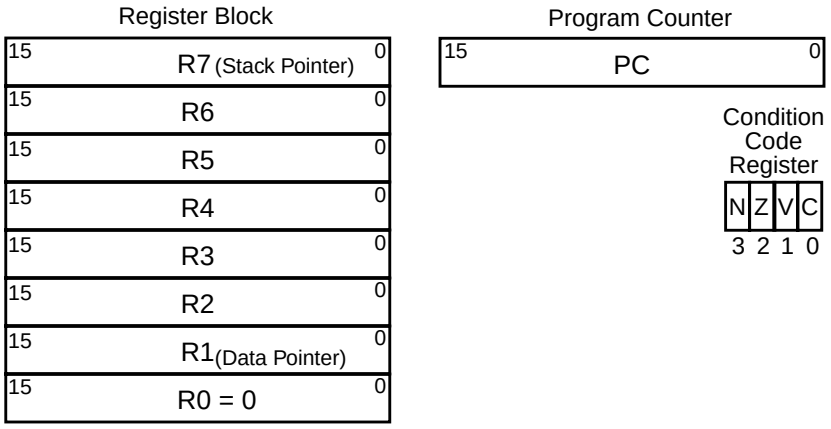


Figure 14-22. Programmer’s Model

1. With the exception of PRR registers (see Section “S12X_MMC”).

The programmer's model of the XGATE RISC core is shown in Figure 14-22. The processor offers a set of seven general purpose registers (R1 - R7), which serve as accumulators and index registers. An additional eighth register (R0) is tied to the value "$0000". Registers R1 and R7 have additional functionality. R1 is preloaded with the initial data pointer of the channel's service request vector (see Figure 14-23). R7 is either preloaded with the content of XGISP74 if the interrupt priority of the current channel is in the range 7 to 4, or it is with preloaded the content of XGISP31 if the interrupt priority of the current channel is in the range 3 to 1. The remaining general purpose registers will be reset to an unspecified value at the beginning of each thread.

The 16 bit program counter allows the addressing of a 64 kbyte address space.

The condition code register contains four bits: the sign bit (S), the zero flag (Z), the overflow flag (V), and the carry bit (C). The initial content of the condition code register is undefined.

14.4.3 Memory Map

The XGATE's RISC core is able to access an address space of 64K bytes. The allocation of memory blocks within this address space is determined on chip level. Refer to the **S12X_MMC Section** for a detailed information.

The XGATE vector block assigns a start address and a data pointer to each XGATE channel. Its position in the XGATE memory map can be adjusted through the XGVBR register (see Section 14.3.1.7, "XGATE Vector Base Address Register (XGVBR)"). Figure 14-23 shows the layout of the vector block. Each vector consists of two 16 bit words. The first contains the start address of the service routine. This value will be loaded into the program counter before a service routine is executed. The second word is a pointer to the service routine's data space. This value will be loaded into register R1 before a service routine is executed.

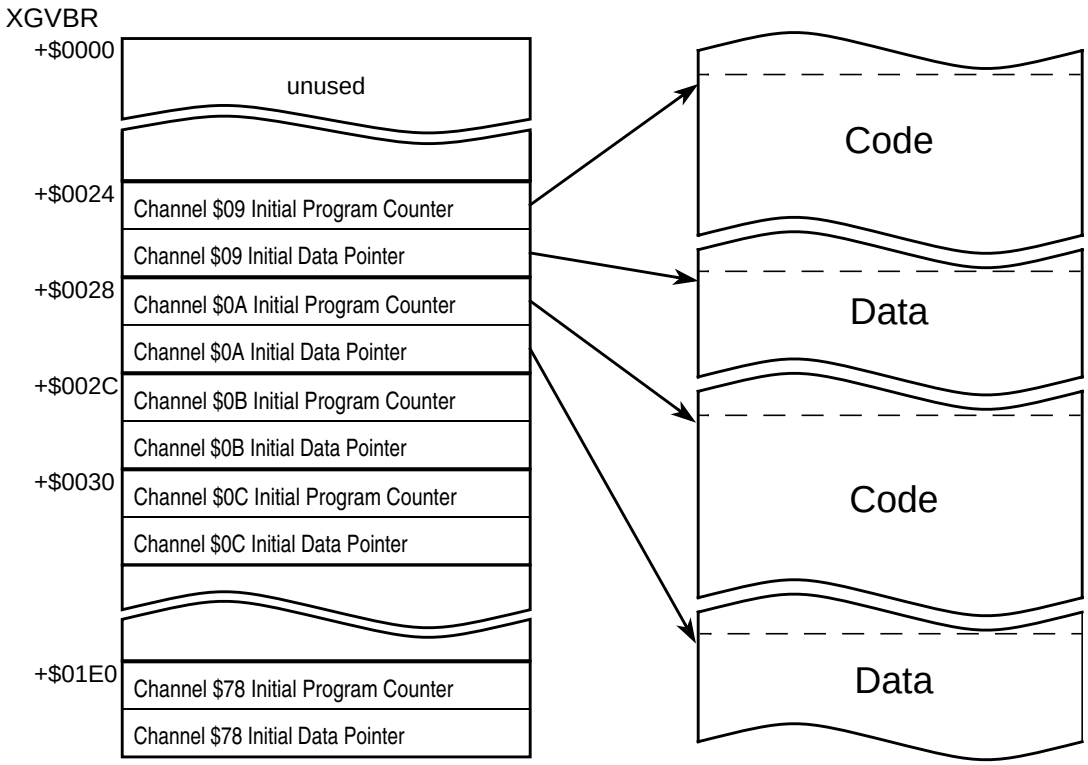


Figure 14-23. XGATE Vector Block

14.4.4 Semaphores

The XGATE module offers a set of eight hardware semaphores. These semaphores provide a mechanism to protect system resources that are shared between two concurrent threads of program execution; one thread running on the S12X_CPU and one running on the XGATE RISC core.

Each semaphore can only be in one of the three states: “Unlocked”, “Locked by S12X_CPU”, and “Locked by XGATE”. The S12X_CPU can check and change a semaphore’s state through the XGATE semaphore register (XGSEM, see Section 14.3.1.10, “XGATE Semaphore Register (XGSEM)”). The RISC core does this through its SSEM and CSEM instructions.

IFigure 14-24 illustrates the valid state transitions.

set_xgsem: 1 is written to XGSEM[*n*] (and 1 is written to XGSEMM[*n*])
clr_xgsem: 0 is written to XGSEM[*n*] (and 1 is written to XGSEMM[*n*])
ssem: Executing SSEM instruction (on semaphore *n*)
csem: Executing CSEM instruction (on semaphore *n*)

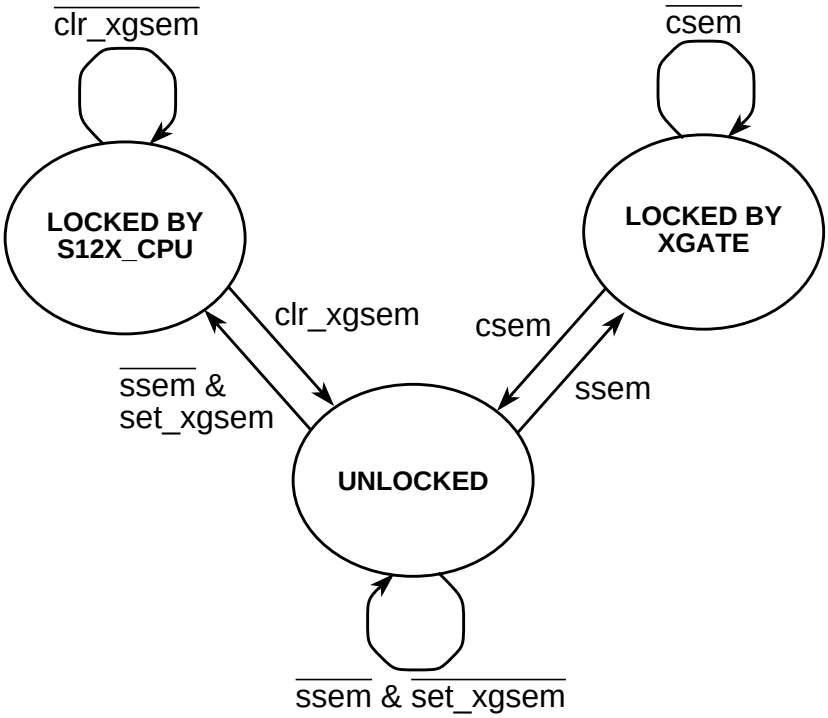


Figure 14-24. Semaphore State Transitions

Figure 14-25 gives an example of the typical usage of the XGATE hardware semaphores.

Two concurrent threads are running on the system. One is running on the S12X_CPU and the other is running on the RISC core. They both have a critical section of code that accesses the same system resource. To guarantee that the system resource is only accessed by one thread at a time, the critical code sequence must be embedded in a semaphore lock/release sequence as shown.

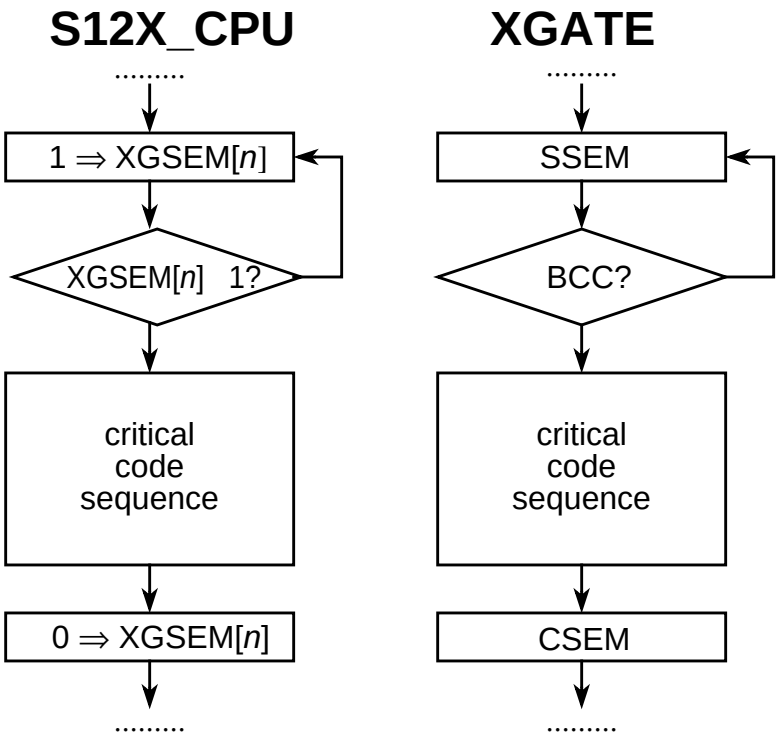


Figure 14-25. Algorithm for Locking and Releasing Semaphores

14.4.5 Software Error Detection

Upon detecting an error condition caused by erratic application code, the XGATE module will immediately terminate program execution and trigger a non-maskable interrupt to the S12X_CPU. There are three error conditions:

- Execution of an illegal opcode
- Illegal opcode fetches
- Illegal load or store accesses

All opcodes which are not listed in section Section 14.8, “Instruction Set” are illegal opcodes. Illegal opcode fetches as well as illegal load and store accesses are defined on chip level. Refer to the **S12X_MMC Section** for a detailed information.

NOTE

When executing a branch (BCC, BCS,...), a jump (JAL) or an RTS instruction, the XGATE prefetches and discards the opcode of the following instruction. The XGATE will perform its software error handling actions (see above) if this opcode fetch is illegal.

14.5 Interrupts

14.5.1 Incoming Interrupt Requests

XGATE threads are triggered by interrupt requests which are routed to the XGATE module (see S12X_INT Section). Only a subset of the MCU's interrupt requests can be routed to the XGATE. Which specific interrupt requests these are and which channel ID they are assigned to is documented in Section "Interrupts" of the **SoC Guide**.

14.5.2 Outgoing Interrupt Requests

There are three types of interrupt requests which can be triggered by the XGATE module:

3. Channel interrupts

   For each XGATE channel there is an associated interrupt flag in the XGATE interrupt flag vector (XGIF, see Section 14.3.1.8, "XGATE Channel Interrupt Flag Vector (XGIF)"). These flags can be set through the "SIF" instruction by the RISC core. They are typically used to flag an interrupt to the S12X_CPU when the XGATE has completed one of its task.

4. Software triggers

   Software triggers are interrupt flags, which can be set and cleared by software (see Section 14.3.1.9, "XGATE Software Trigger Register (XGSWT)"). They are typically used to trigger XGATE tasks by the S12X_CPU software. However these interrupts can also be routed to the S12X_CPU (see **S12X_INT Section**) and triggered by the XGATE software.

5. Software error interrupt

   The software error interrupt signals to the S12X_CPU the detection of an error condition in the XGATE application code (see Section 14.4.5, "Software Error Detection"). This is a non-maskable interrupt. Executing the interrupt service routine will automatically reset the interrupt line.

All outgoing XGATE interrupts, except software error interrupts, can be disabled by the XGIE bit in the XGATE module control register (XGMCTL, see Section 14.3.1.1, "XGATE Control Register (XGMCTL)").

14.6 Debug Mode

The XGATE debug mode is a feature to allow debugging of application code.

14.6.1 Debug Features

In debug mode the RISC core will be halted and the following debug features will be enabled:

- Read and Write accesses to RISC core registers (XGCCR, XGPC, XGR1–XGR7)[1]

  All RISC core registers can be modified. Leaving debug mode will cause the RISC core to continue program execution with the modified register values.

1. Only possible if MCU is unsecured

- Single Stepping

  Writing a "1" to the XGSS bit will call the RISC core to execute a single instruction. All RISC core registers will be updated accordingly.
- Write accesses to the XGCHID register and the XGCHPL register

  XGATE threads can be initiated and terminated through a 16 write access to the XGCHID and the XGCHPL register or through a 8 bit write access to the XGCHID register. Detailed operation is shown in Table 14-22. Once a thread has been initiated, the thread code can be either single stepped or it can be executed by leaving debug mode.

Table 14-22. Initiating and Terminating Threads in Debug Mode

Register Content		Single Cycle Write Access to...		Action
XGCHID	XGCHPL	XGCHID	XGCHPL	
0	0	1..127	-(1)	Set new XGCHID Set XGCHPL to 0x01 Initiate new thread
0	0	1..127	0..7	Set new XGCHID Set new XGCHPL Initiate new thread
1..127	0..3	1..127	4..7	Interrupt current thread Set new XGCHID Set new XGCHPL Initiate new thread
1..127	0..7	0	0..7	Terminate current thread. Resume interrupted thread or become idle if no interrupted thread is pending
			-1	
All other combinations				No action

1. 8 bit write access to XGCHID

NOTE

Even though zero is not a valid interrupt priority level of the S12X_INT module, a thread of priority level 0 can be initiated in debug mode. The XGATE handles requests of priority level 0 in the same way as it handles requests of priority levels 1 to 3.

NOTE

All channels 1 to 127 can be initiated by writing to the XGCHID register, even if they are not assigned to any peripheral module.

NOTE

In Debug Mode the XGATE will ignore all requests from peripheral modules.

14.6.1.0.1 Entering Debug Mode

Debug mode can be entered in four ways:

1. Setting XGDBG to "1"

Writing a "1" to XGDBG and XGDBGM in the same write access causes the XGATE to enter debug mode upon completion of the current instruction.

NOTE

After writing to the XGDBG bit the XGATE will not immediately enter debug mode. Depending on the instruction that is executed at this time there may be a delay of several clock cycles. The XGDBG will read "0" until debug mode is entered.

2. Software breakpoints

   XGATE programs which are stored in the internal RAM allow the use of software breakpoints. A software breakpoint is set by replacing an instruction of the program code with the "BRK" instruction.

   As soon as the program execution reaches the "BRK" instruction, the XGATE enters debug mode. Additionally a software breakpoint request is sent to the S12X_DBG module (see section 4.9 of the **S12X_DBG Section**).

   Upon entering debug mode, the program counter will point to the "BRK" instruction. The other RISC core registers will hold the result of the previous instruction.

   To resume program execution, the "BRK" instruction must be replaced by the original instruction before leaving debug mode.

3. Tagged Breakpoints

   The S12X_DBG module is able to place tags on fetched opcodes. The XGATE is able to enter debug mode right before a tagged opcode is executed (see section 4.9 of the **S12X_DBG Section**). Upon entering debug mode, the program counter will point to the tagged instruction. The other RISC core registers will hold the result of the previous instruction.

4. Forced Breakpoints

   Forced breakpoints are triggered by the S12X_DBG module (see section 4.9 of the **S12X_DBG Section**). When a forced breakpoint occurs, the XGATE will enter debug mode upon completion of the current instruction.

14.6.2 Leaving Debug Mode

Debug mode can only be left by setting the XGDBG bit to "0". If a thread is active (XGCHID has not been cleared in debug mode), program execution will resume at the value of XGPC.

14.7 Security

In order to protect XGATE application code on secured S12X devices, a few restrictions in the debug features have been made. These are:

- Registers XGCCR, XGPC, and XGR1–XGR7 will read zero on a secured device
- Registers XGCCR, XGPC, and XGR1–XGR7 can not be written on a secured device
- Single stepping is not possible on a secured device

14.8 Instruction Set

14.8.1 Addressing Modes

For the ease of implementation the architecture is a strict Load/Store RISC machine, which means all operations must have one of the eight general purpose registers R0 … R7 as their source as well their destination.

All word accesses must work with a word aligned address, that is A[0] = 0!

14.8.1.1 Naming Conventions

RD	Destination register, allowed range is R0–R7
RD.L	Low byte of the destination register, bits [7:0]
RD.H	High byte of the destination register, bits [15:8]
RS, RS1, RS2	Source register, allowed range is R0–R7
RS.L, RS1.L, RS2.L	Low byte of the source register, bits [7:0]
RS.H, RS1.H, RS2.H	High byte of the source register, bits[15:8]
RB	Base register for indexed addressing modes, allowed range is R0–R7
RI	Offset register for indexed addressing modes with register offset, allowed range is R0–R7
RI+	Offset register for indexed addressing modes with register offset and post-increment, Allowed range is R0–R7 (R0+ is equivalent to R0)
–RI	Offset register for indexed addressing modes with register offset and pre-decrement, Allowed range is R0–R7 (–R0 is equivalent to R0)

NOTE

Even though register R1 is intended to be used as a pointer to the data segment, it may be used as a general purpose data register as well.

Selecting R0 as destination register will discard the result of the instruction. Only the condition code register will be updated

14.8.1.2 Inherent Addressing Mode (INH)

Instructions that use this addressing mode either have no operands or all operands are in internal XGATE registers.

Examples:

```
BRK
RTS
```

14.8.1.3 Immediate 3-Bit Wide (IMM3)

Operands for immediate mode instructions are included in the instruction stream and are fetched into the instruction queue along with the rest of the 16 bit instruction. The '#' symbol is used to indicate an immediate addressing mode operand. This address mode is used for semaphore instructions.

Examples:

```
CSEM    #1      ; Unlock semaphore 1
SSEM    #3      ; Lock Semaphore 3
```

14.8.1.4 Immediate 4 Bit Wide (IMM4)

The 4 bit wide immediate addressing mode is supported by all shift instructions.

RD = RD ∗ IMM4

Examples:

```
LSL     R4,#1   ; R4 = R4 << 1; shift register R4 by 1 bit to the left
LSR     R4,#3   ; R4 = R4 >> 3; shift register R4 by 3 bits to the right
```

14.8.1.5 Immediate 8 Bit Wide (IMM8)

The 8 bit wide immediate addressing mode is supported by four major commands (ADD, SUB, LD, CMP).

RD = RD ∗ imm8

Examples:

```
ADDL    R1,#1   ; adds an 8 bit value to register R1
SUBL    R2,#2   ; subtracts an 8 bit value from register R2
LDH     R3,#3   ; loads an 8 bit immediate into the high byte of Register R3
CMPL    R4,#4   ; compares the low byte of register R4 with an immediate value
```

14.8.1.6 Immediate 16 Bit Wide (IMM16)

The 16 bit wide immediate addressing mode is a construct to simplify assembler code. Instructions which offer this mode are translated into two opcodes using the eight bit wide immediate addressing mode.

RD = RD ∗ IMM16

Examples:

```
LDW     R4,#$1234       ; translated to LDL R4,#$34; LDH R4,#$12
ADD     R4,#$5678       ; translated to ADDL R4,#$78; ADDH R4,#$56
```

14.8.1.7 Monadic Addressing (MON)

In this addressing mode only one operand is explicitly given. This operand can either be the source (*f*(RD)), the target (RD = *f*()), or both source and target of the operation (RD = *f*(RD)).

Examples:

```
JAL     R1      ; PC = R1, R1 = PC+2
SIF     R2      ; Trigger IRQ associated with the channel number in R2.L
```

14.8.1.8 Dyadic Addressing (DYA)

In this mode the result of an operation between two registers is stored in one of the registers used as operands.

RD = RD ∗ RS is the general register to register format, with register RD being the first operand and RS the second. RD and RS can be any of the 8 general purpose registers R0 … R7. If R0 is used as the destination register, only the condition code flags are updated. This addressing mode is used only for shift operations with a variable shift value

Examples:

```
LSL     R4,R5    ; R4 = R4 << R5
LSR     R4,R5    ; R4 = R4 >> R5
```

14.8.1.9 Triadic Addressing (TRI)

In this mode the result of an operation between two or three registers is stored into a third one.
RD = RS1 ∗ RS2 is the general format used in the order RD, RS1, RS1. RD, RS1, RS2 can be any of the 8 general purpose registers R0 … R7. If R0 is used as the destination register RD, only the condition code flags are updated. This addressing mode is used for all arithmetic and logical operations.

Examples:

```
ADC     R5,R6,R7          ; R5 = R6 + R7 + Carry
SUB     R5,R6,R7          ; R5 = R6 - R7
```

14.8.1.10 Relative Addressing 9-Bit Wide (REL9)

A 9-bit signed word address offset is included in the instruction word. This addressing mode is used for conditional branch instructions.

Examples:

```
BCC     REL9              ; PC = PC + 2 + (REL9 << 1)
BEQ     REL9              ; PC = PC + 2 + (REL9 << 1)
```

14.8.1.11 Relative Addressing 10-Bit Wide (REL10)

An 10-bit signed word address offset is included in the instruction word. This addressing mode is used for the unconditional branch instruction.

Examples:

```
BRA     REL10             ; PC = PC + 2 + (REL10 << 1)
```

14.8.1.12 Index Register plus Immediate Offset (IDO5)

(RS, #OFFS5) provides an unsigned offset from the base register.

Examples:

```
LDB     R4,(R1,#OFFS5)    ; loads a byte from (R1+OFFS5) into R4
STW     R4,(R1,#OFFS5)    ; stores R4 as a word to (R1+OFFS5)
```

14.8.1.13 Index Register plus Register Offset (IDR)

For load and store instructions (RS, RI) provides a variable offset in a register.

Examples:

```
LDB     R4,(R1,R2)      ; loads a byte from (R1+R2) into R4
STW     R4,(R1,R2)      ; stores R4 as a word to (R1+R2)
```

14.8.1.14 Index Register plus Register Offset with Post-increment (IDR+)

[RS, RI+] provides a variable offset in a register, which is incremented after accessing the memory. In case of a byte access the index register will be incremented by one. In case of a word access it will be incremented by two.

Examples:

```
LDB     R4,(R1,R2+)     ; loads a byte from (R1+R2) into R4, R2+=1
STW     R4,(R1,R2+)     ; stores R4 as a word to (R1+R2), R2+=2
```

14.8.1.15 Index Register plus Register Offset with Pre-decrement (–IDR)

[RS, -RI] provides a variable offset in a register, which is decremented before accessing the memory. In case of a byte access the index register will be decremented by one. In case of a word access it will be decremented by two.

Examples:

```
LDB     R4,(R1,-R2)     ; R2 -=1, loads a byte from (R1+R2) into R4
STW     R4,(R1,-R2)     ; R2 -=2, stores R4 as a word to (R1+R2)
```

14.8.2 Instruction Summary and Usage

14.8.2.1 Load & Store Instructions

Any register can be loaded either with an immediate or from the address space using indexed addressing modes.

```
LDL     RD,#IMM8        ; loads an immediate 8 bit value to the lower byte of RD
LDW     RD,(RB,RI)      ; loads data using RB+RI as effective address

LDB     RD,(RB, RI+)    ; loads data using RB+RI as effective address
                        ; followed by an increment of RI depending on
                        ; the size of the operation
```

The same set of modes is available for the store instructions

```
STB     RS,(RB, RI)     ; stores data using RB+RI as effective address

STW     RS,(RB, RI+)    ; stores data using RB+RI as effective address
                        ; followed by an increment of RI depending on
                        ; the size of the operation.
```

14.8.2.2 Logic and Arithmetic Instructions

All logic and arithmetic instructions support the 8 bit immediate addressing mode (IMM8: RD = RD ∗ #IMM8) and the triadic addressing mode (TRI: RD = RS1 ∗ RS2).

All arithmetic is considered as signed, sign, overflow, zero and carry flag will be updated. The carry will not be affected for logical operations.

```
        ADDL    R2,#1           ; increment R2
        ANDH    R4,#$FE         ; R4.H = R4.H & $FE, clear lower bit of higher byte

        ADD     R3,R4,R5        ; R3 = R4 + R5
        SUB     R3,R4,R5        ; R3 = R4 - R5

        AND     R3,R4,R5        ; R3 = R4 & R5 logical AND on the whole word
        OR      R3,R4,R5        ; R3 = R4 | R5
```

14.8.2.3 Register – Register Transfers

This group comprises transfers from and to some special registers

```
        TFR     R3,CCR          ; transfers the condition code register to the low byte of
                                ; register R3
Branch Instructions
```

The branch offset is +255 words or -256 words counted from the beginning of the next instruction. Since instructions have a fixed 16 bit width, the branch offsets are word aligned by shifting the offset value by 2.

```
        BEQ     label           ; if Z flag = 1 branch to label
```

An unconditional branch allows a +511 words or -512 words branch distance.

```
        BRA     label
```

14.8.2.4 Shift Instructions

Shift operations allow the use of a 4 bit wide immediate value to identify a shift width within a 16 bit word. For shift operations a value of 0 does not shift at all, while a value of 15 shifts the register RD by 15 bits. In a second form the shift value is contained in the bits 3:0 of the register RS.

Examples:

```
        LSL     R4,#1   ; R4 = R4 << 1; shift register R4 by 1 bit to the left
        LSR     R4,#3   ; R4 = R4 >> 3; shift register R4 by 3 bits to the right
        ASR     R4,R2   ; R4 = R4 >> R2;arithmetic shift register R4 right by the amount
                        ;            of bits contained in R2[3:0].
```

14.8.2.5 Bit Field Operations

This addressing mode is used to identify the position and size of a bit field for insertion or extraction. The width and offset are coded in the lower byte of the source register 2, RS2. The content of the upper byte is ignored. An offset of 0 denotes the right most position and a width of 0 denotes 1 bit. These instructions are very useful to extract, insert, clear, set or toggle portions of a 16 bit word

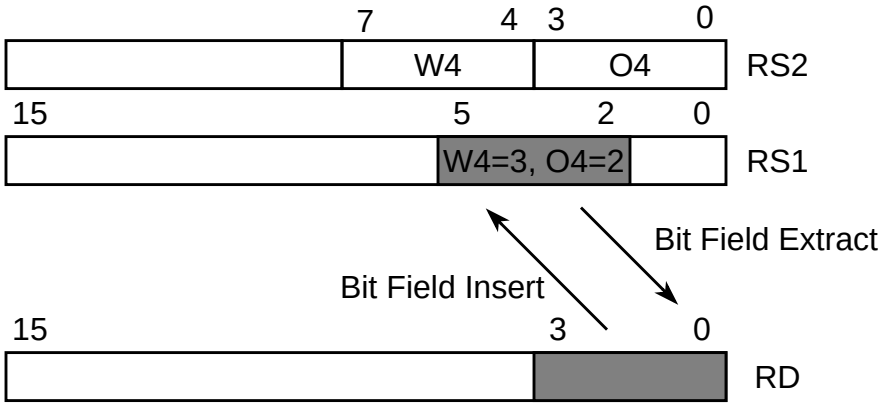


Figure 14-26. Bit Field Addressing

```
BFEXT    R3,R4,R5 ; R5: W4+1 bits with offset O4, will be extracted from R4 into R3
```

14.8.2.6 Special Instructions for DMA Usage

The XGATE offers a number of additional instructions for flag manipulation, program flow control and debugging:

1. SIF: Set a channel interrupt flag
2. SSEM: Test and set a hardware semaphore
3. CSEM: Clear a hardware semaphore
4. BRK: Software breakpoint
5. NOP: No Operation
6. RTS: Terminate the current thread

14.8.3 Cycle Notation

Table 14-23 show the XGATE access detail notation. Each code letter equals one XGATE cycle. Each letter implies additional wait cycles if memories or peripherals are not accessible. Memories or peripherals are not accessible if they are blocked by the S12X_CPU. In addition to this Peripherals are only accessible every other XGATE cycle. Uppercase letters denote 16 bit operations. Lowercase letters denote 8 bit operations. The XGATE is able to perform two bus or wait cycles per S12X_CPU cycle.

Table 14-23. Access Detail Notation

V — Vector fetch: always an aligned word read, lasts for at least one RISC core cycle
P — Program word fetch: always an aligned word read, lasts for at least one RISC core cycle
r — 8 bit data read: lasts for at least one RISC core cycle
R — 16 bit data read: lasts for at least one RISC core cycle
w — 8 bit data write: lasts for at least one RISC core cycle
W — 16 bit data write: lasts for at least one RISC core cycle
A — Alignment cycle: no read or write, lasts for zero or one RISC core cycles
f — Free cycle: no read or write, lasts for one RISC core cycles
Special Cases
PP/P — Branch: PP if branch taken, P if not

14.8.4 Thread Execution

When the RISC core is triggered by an interrupt request (see Figure 14-1) it first executes a vector fetch sequence which performs three bus accesses:

1. A V-cycle to fetch the initial content of the program counter.
2. A V-cycle to fetch the initial content of the data segment pointer (R1).
3. A P-cycle to load the initial opcode.

Afterwards a sequence of instructions (thread) is executed which is terminated by an "RTS" instruction. If further interrupt requests are pending after a thread has been terminated, a new vector fetch will be performed. Otherwise the RISC core will either resume a previous thread (beginning with a P-cycle to refetch the interrupted opcode) or it will become idle until a new interrupt request is received. A thread can only be interrupted by an interrupt request of higher priority.

14.8.5 Instruction Glossary

This section describes the XGATE instruction set in alphabetical order.

ADC — Add with Carry — ADC

Operation

RS1 + RS2 + C ⇒ RD

Adds the content of register RS1, the content of register RS2 and the value of the Carry bit using binary addition and stores the result in the destination register RD. The Zero Flag is also carried forward from the previous operation allowing 32 and more bit additions.

Example:

```
ADD     R6,R2,R2
ADC     R7,R3,R3 ; R7:R6 = R5:R4 + R3:R2
BCC              ; conditional branch on 32 bit addition
```

CCR Effects

N	Z	V	C
Δ	Δ	Δ	Δ

N: Set if bit 15 of the result is set; cleared otherwise.

Z: Set if the result is $0000 and Z was set before this operation; cleared otherwise.

V: Set if a two´s complement overflow resulted from the operation; cleared otherwise.
RS1[15] & RS2[15] & $\overline{RD[15]}_{new}$ | $\overline{RS1[15]}$ & $\overline{RS2[15]}$ & $RD[15]_{new}$

C: Set if there is a carry from bit 15 of the result; cleared otherwise.
RS1[15] & RS2[15] | RS1[15] & $\overline{RD[15]}_{new}$ | RS2[15] & $\overline{RD[15]}_{new}$

Code and CPU Cycles

Source Form	Address Mode	Machine Code										Cycles
ADC RD, RS1, RS2	TRI	0	0	0	1	1	RD	RS1	RS2	1	1	P

ADD — Add without Carry — ADD

Operation

RS1 + RS2 ⇒ RD
RD + IMM16 ⇒ RD (translates to ADDL RD, #IMM16[7:0]; ADDH RD, #IMM16[15:8])

Performs a 16 bit addition and stores the result in the destination register RD.

NOTE

When using immediate addressing mode (ADD RD, #IMM16), the V-flag and the C-Flag of the first instruction (ADDL RD, #IMM16[7:0]) are not considered by the second instruction (ADDH RD, #IMM16[15:8]).
⇒ Don't rely on the V-Flag if RD + IMM16[7:0] ≥ 2^{15}.
⇒ Don't rely on the C-Flag if RD + IMM16[7:0] ≥ 2^{16}.

CCR Effects

N	Z	V	C
Δ	Δ	Δ	Δ

N: Set if bit 15 of the result is set; cleared otherwise.

Z: Set if the result is $0000; cleared otherwise.

V: Set if a two´s complement overflow resulted from the operation; cleared otherwise.
$RS1[15] \& RS2[15] \& \overline{RD[15]}_{new} | \overline{RS1[15]} \& \overline{RS2[15]} \& RD[15]_{new}$
Refer to ADDH instruction for #IMM16 operations.

C: Set if there is a carry from bit 15 of the result; cleared otherwise.
$RS1[15] \& RS2[15] | RS1[15] \& \overline{RD[15]}_{new} | RS2[15] \& \overline{RD[15]}_{new}$
Refer to ADDH instruction for #IMM16 operations.

Code and CPU Cycles

Source Form	Address Mode	Machine Code										Cycles
ADD RD, RS1, RS2	TRI	0	0	0	1	1	RD	RS1	RS2	1	0	P
ADD RD, #IMM16	IMM8	1	1	1	0	0	RD	IMM16[7:0]				P
	IMM8	1	1	1	0	1	RD	IMM16[15:8]				P

ADDH — Add Immediate 8 bit Constant (High Byte) — ADDH

Operation

RD + IMM8:$00 ⇒ RD

Adds the content of high byte of register RD and a signed immediate 8 bit constant using binary addition and stores the result in the high byte of the destination register RD. This instruction can be used after an ADDL for a 16 bit immediate addition.

Example:

```
ADDL    R2,#LOWBYTE
ADDH    R2,#HIGHBYTE      ; R2 = R2 + 16 bit immediate
```

CCR Effects

N	Z	V	C
Δ	Δ	Δ	Δ

N: Set if bit 15 of the result is set; cleared otherwise.

Z: Set if the result is $0000; cleared otherwise.

V: Set if a two´s complement overflow resulted from the operation; cleared otherwise.
$RD[15]_{old}$ & IMM8[7] & $\overline{RD[15]}_{new}$ | $\overline{RD[15]}_{old}$ & $\overline{IMM8[7]}$ & $RD[15]_{new}$

C: Set if there is a carry from the bit 15 of the result; cleared otherwise.
$RD[15]_{old}$ & IMM8[7] | $RD[15]_{old}$ & $\overline{RD[15]}_{new}$ | IMM8[7] & $\overline{RD[15]}_{new}$

Code and CPU Cycles

Source Form	Address Mode	Machine Code							Cycles
ADDH RD, #IMM8	IMM8	1	1	1	0	1	RD	IMM8	P

ADDL — Add Immediate 8 bit Constant (Low Byte) — ADDL

Operation

RD + $00:IMM8 ⇒ RD

Adds the content of register RD and an unsigned immediate 8 bit constant using binary addition and stores the result in the destination register RD. This instruction must be used first for a 16 bit immediate addition in conjunction with the ADDH instruction.

CCR Effects

N	Z	V	C
Δ	Δ	Δ	Δ

N: Set if bit 15 of the result is set; cleared otherwise.

Z: Set if the result is $0000; cleared otherwise.

V: Set if a two´s complement overflow resulted from the 8 bit operation; cleared otherwise.
$\overline{RD[15]}_{old}$ & $RD[15]_{new}$

C: Set if there is a carry from the bit 15 of the result; cleared otherwise.
$RD[15]_{old}$ & $\overline{RD[15]}_{new}$

Code and CPU Cycles

Source Form	Address Mode	Machine Code							Cycles
ADDL RD, #IMM8	IMM8	1	1	1	0	0	RD	IMM8	P

AND — Logical AND — AND

Operation

RS1 & RS2 ⇒ RD
RD & IMM16 ⇒ RD (translates to ANDL RD, #IMM16[7:0]; ANDH RD, #IMM16[15:8])

Performs a bit wise logical AND of two 16 bit values and stores the result in the destination register RD.

NOTE

When using immediate addressing mode (AND RD, #IMM16), the Z-flag of the first instruction (ANDL RD, #IMM16[7:0]) is not considered by the second instruction (ANDH RD, #IMM16[15:8]).
⇒ Don't rely on the Z-Flag.

CCR Effects

N	Z	V	C
Δ	Δ	0	—

N: Set if bit 15 of the result is set; cleared otherwise.
Z: Set if the result is $0000; cleared otherwise.
Refer to ANDH instruction for #IMM16 operations.
V: 0; cleared.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code										Cycles
AND RD, RS1, RS2	TRI	0	0	0	1	0	RD	RS1	RS2	0	0	P
AND RD, #IMM16	IMM8	1	0	0	0	0	RD	IMM16[7:0]				P
	IMM8	1	0	0	0	1	RD	IMM16[15:8]				P

ANDH — Logical AND Immediate 8 bit Constant (High Byte) — ANDH

Operation

RD.H & IMM8 ⇒ RD.H

Performs a bit wise logical AND between the high byte of register RD and an immediate 8 bit constant and stores the result in the destination register RD.H. The low byte of RD is not affected.

CCR Effects

N	Z	V	C
Δ	Δ	0	—

N: Set if bit 15 of the result is set; cleared otherwise.
Z: Set if the 8 bit result is $00; cleared otherwise.
V: 0; cleared.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code							Cycles
ANDH RD, #IMM8	IMM8	1	0	0	0	1	RD	IMM8	P

ANDL — Logical AND Immediate 8 bit Constant (Low Byte) — ANDL

Operation

RD.L & IMM8 ⇒ RD.L

Performs a bit wise logical AND between the low byte of register RD and an immediate 8 bit constant and stores the result in the destination register RD.L. The high byte of RD is not affected.

CCR Effects

N	Z	V	C
Δ	Δ	0	—

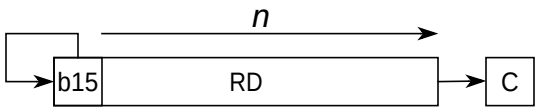
N: Set if bit 7 of the result is set; cleared otherwise.
Z: Set if the 8 bit result is $00; cleared otherwise.
V: 0; cleared.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code							Cycles
ANDL RD, #IMM8	IMM8	1	0	0	0	0	RD	IMM8	P

ASR — Arithmetic Shift Right — ASR

Operation



n = RS or IMM4

Shifts the bits in register RD *n* positions to the right. The higher *n* bits of the register RD become filled with the sign bit (RD[15]). The carry flag will be updated to the bit contained in RD[n-1] before the shift for *n* > 0.

n can range from 0 to 16.

In immediate address mode, *n* is determined by the operand IMM4. *n* is considered to be 16 if IMM4 is equal to 0.

In dyadic address mode, *n* is determined by the content of RS. *n* is considered to be 16 if the content of RS is greater than 15.

CCR Effects

N	Z	V	C
Δ	Δ	Δ	Δ

N: Set if bit 15 of the result is set; cleared otherwise.

Z: Set if the result is $0000; cleared otherwise.

V: Set if a two´s complement overflow resulted from the operation; cleared otherwise.
$RD[15]_{old}$ ^ $RD[15]_{new}$

C: Set if *n* > 0 and RD[n-1] = 1; if n = 0 unaffected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code												Cycles
ASR RD, #IMM4	IMM4	0	0	0	0	1	RD	IMM4		1	0	0	1	P
ASR RD, RS	DYA	0	0	0	0	1	RD	RS	1	0	0	0	1	P

BCC — Branch if Carry Cleared (Same as BHS) — BCC

Operation

If C = 0, then PC + $0002 + (REL9 << 1) ⇒ PC

Tests the Carry flag and branches if C = 0.

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code								Cycles
BCC REL9	REL9	0	0	1	0	0	0	0	REL9	PP/P

BCS

Branch if Carry Set (Same as BLO)

BCS

Operation

If C = 1, then PC + $0002 + (REL9 << 1) ⇒ PC

Tests the Carry flag and branches if C = 1.

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code								Cycles
BCS REL9	REL9	0	0	1	0	0	0	1	REL9	PP/P

BEQ — Branch if Equal — BEQ

Operation

If Z = 1, then PC + $0002 + (REL9 << 1) ⇒ PC

Tests the Zero flag and branches if Z = 1.

CCR Effect

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code								Cycles
BEQ REL9	REL9	0	0	1	0	0	1	1	REL9	PP/P

BFEXT Bit Field Extract BFEXT

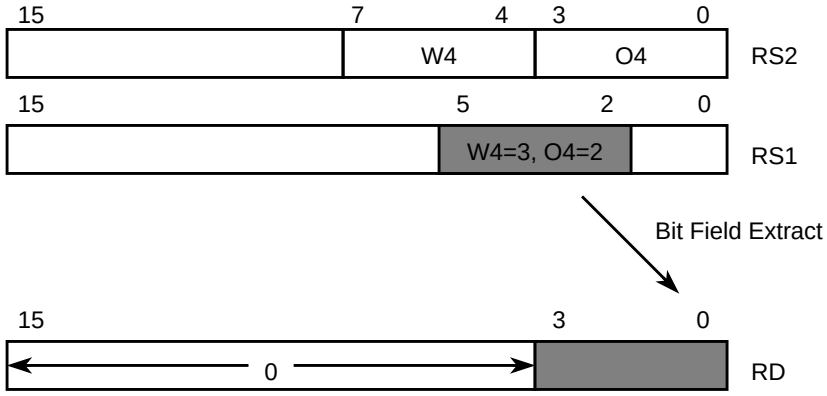
Operation

RS1[(*o*+*w*):*o*] ⇒ RD[*w*:0]; 0 ⇒ RD[15:(*w*+1)]

w = (RS2[7:4])

o = (RS2[3:0])

Extracts *w+1* bits from register RS1 starting at position *o* and writes them right aligned into register RD. The remaining bits in RD will be cleared. If (o+w) > 15 only bits [15:o] get extracted.



CCR Effects

N	Z	V	C
Δ	Δ	0	—

N: Set if bit 15 of the result is set; cleared otherwise.

Z: Set if the result is $0000; cleared otherwise.

V: 0; cleared.

C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code										Cycles
BFEXT RD, RS1, RS2	TRI	0	1	1	0	0	RD	RS1	RS2	1	1	P

BFFO Bit Field Find First One BFFO

Operation

FirstOne(RS) ⇒ RD;

Searches the first “1” in register RS (from MSB to LSB) and writes the bit position into the destination register RD. The upper bits of RD are cleared. In case the content of RS is equal to $0000, RD will be cleared and the carry flag will be set. This is used to distinguish a “1” in position 0 versus no “1” in the whole RS register at all.

CCR Effects

N	Z	V	C
0	Δ	0	Δ

N: 0; cleared.

Z: Set if the result is $0000; cleared otherwise.

V: 0; cleared.

C: Set if RS = $0000(1); cleared otherwise.

1. Before executing the instruction

Code and CPU Cycles

Source Form	Address Mode	Machine Code												Cycles
BFFO RD, RS	DYA	0	0	0	0	1	RD	RS	1	0	0	0	0	P

BFINS Bit Field Insert BFINS

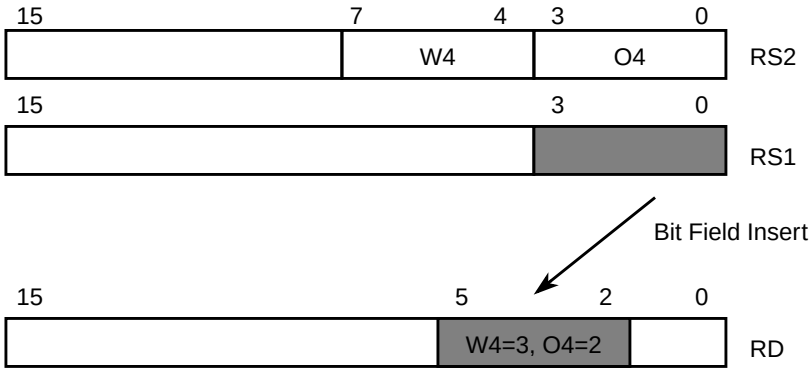
Operation

RS1[*w*:*0*] ⇒ RD[(*w*+*o*):o];

w = (RS2[7:4])

o = (RS2[3:0])

Extracts *w+1* bits from register RS1 starting at position 0 and writes them into register RD starting at position *o*. The remaining bits in RD are not affected. If (o+w) > 15 the upper bits are ignored. Using R0 as a RS1, this command can be used to clear bits.



CCR Effects

N	Z	V	C
Δ	Δ	0	—

N: Set if bit 15 of the result is set; cleared otherwise.

Z: Set if the result is $0000; cleared otherwise.

V: 0; cleared.

C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code										Cycles
BFINS RD, RS1, RS2	TRI	0	1	1	0	1	RD	RS1	RS2	1	1	P

BFINSI Bit Field Insert and Invert BFINSI

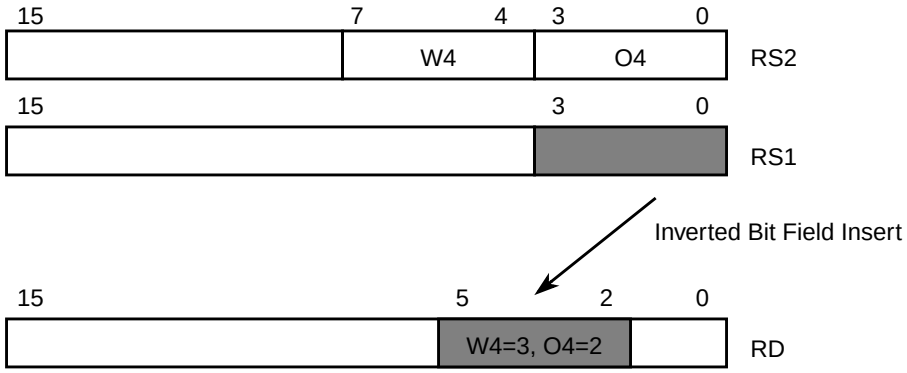
Operation

!RS1[*w*:*0*] ⇒ RD[*w*+o:o];

w = (RS2[7:4])

o = (RS2[3:0])

Extracts *w+1* bits from register RS1 starting at position 0, inverts them and writes into register RD starting at position *o*. The remaining bits in RD are not affected. If (o+w) > 15 the upper bits are ignored. Using R0 as a RS1, this command can be used to set bits.



CCR Effects

N	Z	V	C
Δ	Δ	0	—

N: Set if bit 15 of the result is set; cleared otherwise.
Z: Set if the result is $0000; cleared otherwise.
V: 0; cleared.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code										Cycles
BFINSI RD, RS1, RS2	TRI	0	1	1	1	0	RD	RS1	RS2	1	1	P

BFINSX Bit Field Insert and XNOR BFINSX

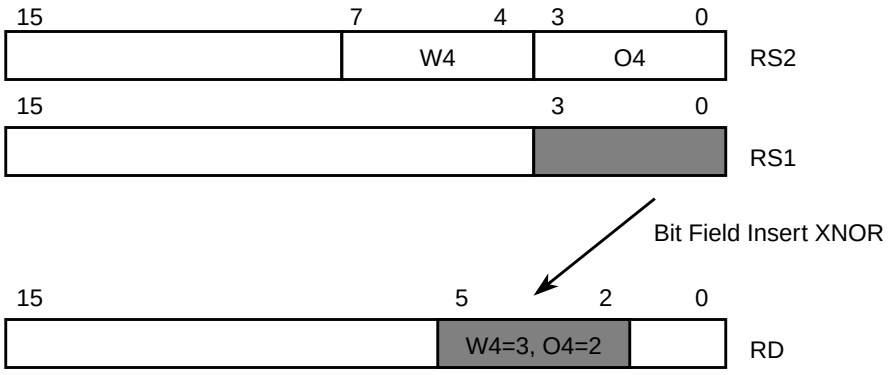
Operation

!(RS1[*w*:*0*] ^ RD[w+o:o]) ⇒ RD[*w*+o:o];

w = (RS2[7:4])

o = (RS2[3:0])

Extracts *w+1* bits from register RS1 starting at position 0, performs an XNOR with RD[w+o:o] and writes the bits back to RD. The remaining bits in RD are not affected. If (o+w) > 15 the upper bits are ignored. Using R0 as a RS1, this command can be used to toggle bits.



CCR Effects

N	Z	V	C
Δ	Δ	0	—

N: Set if bit 15 of the result is set; cleared otherwise.

Z: Set if the result is $0000; cleared otherwise.

V: 0; cleared.

C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code										Cycles
BFINSX RD, RS1, RS2	TRI	0	1	1	1	1	RD	RS1	RS2	1	1	P

BGE Branch if Greater than or Equal to Zero BGE

Operation

If N ^ V = 0, then PC + $0002 + (REL9 << 1) ⇒ PC

Branch instruction to compare signed numbers.

Branch if RS1 ≥ RS2:

```
SUB     R0,RS1,RS2
BGE     REL9
```

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code								Cycles
BGE REL9	REL9	0	0	1	1	0	1	0	REL9	PP/P

BGT Branch if Greater than Zero BGT

Operation

If Z | (N ^ V) = 0, then PC + $0002 + (REL9 << 1) ⇒ PC

Branch instruction to compare signed numbers.

Branch if RS1 > RS2:

```
SUB     R0,RS1,RS2
BGT     REL9
```

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code								Cycles
BGT REL9	REL9	0	0	1	1	1	0	0	REL9	PP/P

BHI Branch if Higher BHI

Operation

If C | Z = 0, then PC + $0002 + (REL9 << 1) ⇒ PC

Branch instruction to compare unsigned numbers.

Branch if RS1 > RS2:

```
SUB     R0,RS1,RS2
BHI     REL9
```

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code								Cycles
BHI REL9	REL9	0	0	1	1	0	0	0	REL9	PP/P

BHS — Branch if Higher or Same (Same as BCC) — BHS

Operation

If C = 0, then PC + $0002 + (REL9 << 1) ⇒ PC

Branch instruction to compare unsigned numbers.

Branch if RS1 ≥ RS2:

```
SUB     R0,RS1,RS2
BHS     REL9
```

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code								Cycles
BHS REL9	REL9	0	0	1	0	0	0	0	REL9	PP/P

BITH — Bit Test Immediate 8 bit Constant (High Byte) — BITH

Operation

RD.H & IMM8 ⇒ NONE

Performs a bit wise logical AND between the high byte of register RD and an immediate 8 bit constant. Only the condition code flags get updated, but no result is written back.

CCR Effects

N	Z	V	C
Δ	Δ	0	—

N: Set if bit 15 of the result is set; cleared otherwise.
Z: Set if the 8 bit result is $00; cleared otherwise.
V: 0; cleared.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code							Cycles
BITH RD, #IMM8	IMM8	1	0	0	1	1	RD	IMM8	P

BITL — Bit Test Immediate 8 bit Constant (Low Byte) — BITL

Operation

RD.L & IMM8 ⇒ NONE

Performs a bit wise logical AND between the low byte of register RD and an immediate 8 bit constant. Only the condition code flags get updated, but no result is written back.

CCR Effects

N	Z	V	C
Δ	Δ	0	—

N: Set if bit 7 of the result is set; cleared otherwise.
Z: Set if the 8 bit result is $00; cleared otherwise.
V: 0; cleared.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code							Cycles
BITL RD, #IMM8	IMM8	1	0	0	1	0	RD	IMM8	P

BLE Branch if Less or Equal to Zero BLE

Operation

If Z | (N ^ V) = 1, then PC + $0002 + (REL9 << 1) ⇒ PC

Branch instruction to compare signed numbers.

Branch if RS1 ≤ RS2:

```
SUB     R0,RS1,RS2
BLE     REL9
```

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code								Cycles
BLE REL9	REL9	0	0	1	1	1	0	1	REL9	PP/P

BLO

Branch if Carry Set (Same as BCS)

BLO

Operation

If C = 1, then PC + $0002 + (REL9 << 1) ⇒ PC

Branch instruction to compare unsigned numbers.

Branch if RS1 < RS2:

```
SUB     R0,RS1,RS2
BLO     REL9
```

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code								Cycles
BLO REL9	REL9	0	0	1	0	0	0	1	REL9	PP/P

BLS Branch if Lower or Same BLS

Operation

If C | Z = 1, then PC + $0002 + (REL9 << 1) ⇒ PC

Branch instruction to compare unsigned numbers.

Branch if RS1 ≤ RS2:

```
SUB     R0,RS1,RS2
BLS     REL9
```

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code								Cycles
BLS REL9	REL9	0	0	1	1	0	0	1	REL9	PP/P

BLT — Branch if Lower than Zero — BLT

Operation

If N ^ V = 1, then PC + $0002 + (REL9 << 1) ⇒ PC

Branch instruction to compare signed numbers.

Branch if RS1 < RS2:

```
SUB     R0,RS1,RS2
BLT     REL9
```

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code								Cycles
BLT REL9	REL9	0	0	1	1	0	1	1	REL9	PP/P

BMI Branch if Minus BMI

Operation

If N = 1, then PC + $0002 + (REL9 << 1) ⇒ PC

Tests the sign flag and branches if N = 1.

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code								Cycles
BMI REL9	REL9	0	0	1	0	1	0	1	REL9	PP/P

BNE Branch if Not Equal BNE

Operation

If Z = 0, then PC + $0002 + (REL9 << 1) ⇒ PC

Tests the Zero flag and branches if Z = 0.

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code								Cycles
BNE REL9	REL9	0	0	1	0	0	1	0	REL9	PP/P

BPL Branch if Plus BPL

Operation

If N = 0, then PC + $0002 + (REL9 << 1) ⇒ PC

Tests the Sign flag and branches if N = 0.

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code								Cycles
BPL REL9	REL9	0	0	1	0	1	0	0	REL9	PP/P

BRA Branch Always BRA

Operation

PC + $0002 + (REL10 << 1) ⇒ PC

Branches always.

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code							Cycles
BRA REL10	REL10	0	0	1	1	1	1	REL10	PP

BRK Break BRK

Operation

Put XGATE into Debug Mode (see Section 14.6.1.0.1, “Entering Debug Mode”) and signals a software breakpoint to the S12X_DBG module (see section 4.9 of the **S12X_DBG Section**).

NOTE

It is not possible to single step over a BRK instruction. This instruction does not advance the program counter.

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code																Cycles
BRK	INH	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	PAff

BVC Branch if Overflow Cleared BVC

Operation

If V = 0, then PC + $0002 + (REL9 << 1) ⇒ PC

Tests the Overflow flag and branches if V = 0.

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code								Cycles
BVC REL9	REL9	0	0	1	0	1	1	0	REL9	PP/P

BVS Branch if Overflow Set BVS

Operation

If V = 1, then PC + $0002 + (REL9 << 1) ⇒ PC

Tests the Overflow flag and branches if V = 1.

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code								Cycles
BVS REL9	REL9	0	0	1	0	1	1	1	REL9	PP/P

CMP — Compare — CMP

Operation

RS1 – RS2 ⇒ NONE (translates to SUB R0, RS1, RS2)
RD – IMM16 ⇒ NONE (translates to CMPL RD, #IMM16[7:0]; CPCH RD, #IMM16[15:8])

Subtracts two 16 bit values and discards the result.

CCR Effects

N	Z	V	C
Δ	Δ	Δ	Δ

N: Set if bit 15 of the result is set; cleared otherwise.

Z: Set if the result is $0000; cleared otherwise.

V: Set if a two´s complement overflow resulted from the operation; cleared otherwise.
RS1[15] & $\overline{RS2[15]}$ & $\overline{result[15]}$ | $\overline{RS1[15]}$ & RS2[15] & result[15]
RD[15] & $\overline{IMM16[15]}$ & $\overline{result[15]}$ | RD[15] & IMM16[15] & result[15]

C: Set if there is a carry from the bit 15 of the result; cleared otherwise.
$\overline{RS1[15]}$ & RS2[15] | $\overline{RS1[15]}$ & result[15] | RS2[15] & result[15]
$\overline{RD[15]}$ & IMM16[15] | $\overline{RD[15]}$ & result[15] | IMM16[15] & result[15]

Code and CPU Cycles

Source Form	Address Mode	Machine Code											Cycles
CMP RS1, RS2	TRI	0	0	0	1	1	0	0	0	RS1	RS2	0	0	P
CMP RS, #IMM16	IMM8	1	1	0	1	0	RS			IMM16[7:0]				P
	IMM8	1	1	0	1	1	RS			IMM16[15:8]				P

CMPL — Compare Immediate 8 bit Constant (Low Byte) — CMPL

Operation

RS.L – IMM8 ⇒ NONE, only condition code flags get updated

Subtracts the 8 bit constant IMM8 contained in the instruction code from the low byte of the source register RS.L using binary subtraction and updates the condition code register accordingly.

Remark: There is no equivalent operation using triadic addressing. Comparing the values of two registers can be performed by using the subtract instruction with R0 as destination register.

CCR Effects

N	Z	V	C
Δ	Δ	Δ	Δ

N: Set if bit 7 of the result is set; cleared otherwise.

Z: Set if the 8 bit result is $00; cleared otherwise.

V: Set if a two´s complement overflow resulted from the 8 bit operation; cleared otherwise.
RS[7] & $\overline{\text{IMM8[7]}}$ & $\overline{\text{result[7]}}$ | $\overline{\text{RS[7]}}$ & IMM8[7] & result[7]

C: Set if there is a carry from the Bit 7 to Bit 8 of the result; cleared otherwise.
$\overline{\text{RS[7]}}$ & IMM8[7] | $\overline{\text{RS[7]}}$ & result[7] | IMM8[7] & result[7]

Code and CPU Cycles

Source Form	Address Mode	Machine Code							Cycles
CMPL RS, #IMM8	IMM8	1	1	0	1	0	RS	IMM8	P

COM One's Complement COM

Operation

~RS ⇒ RD (translates to XNOR RD, R0, RS)
~RD ⇒ RD (translates to XNOR RD, R0, RD)

Performs a one's complement on a general purpose register.

CCR Effects

N	Z	V	C
Δ	Δ	0	—

N: Set if bit 15 of the result is set; cleared otherwise.
Z: Set if the result is $0000; cleared otherwise.
V: 0; cleared.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code											Cycles
COM RD, RS	TRI	0	0	0	1	0	RD	0	0	0	RS	1	1	P
COM RD	TRI	0	0	0	1	0	RD	0	0	0	RD	1	1	P

CPC — Compare with Carry — CPC

Operation

RS1 – RS2 - C ⇒ NONE (translates to SBC R0, RS1, RS2)

Subtracts the carry bit and the content of register RS2 from the content of register RS1 using binary subtraction and discards the result.

CCR Effects

N	Z	V	C
Δ	Δ	Δ	Δ

N: Set if bit 15 of the result is set; cleared otherwise.

Z: Set if the result is $0000; cleared otherwise.

V: Set if a two´s complement overflow resulted from the operation; cleared otherwise.
RS1[15] & $\overline{RS2[15]}$ & $\overline{result[15]}$ | $\overline{RS1[15]}$ & RS2[15] & result[15]

C: Set if there is a carry from the bit 15 of the result; cleared otherwise.
$\overline{RS1[15]}$ & RS2[15] | $\overline{RS1[15]}$ & result[15] | RS2[15] & result[15]

Code and CPU Cycles

Source Form	Address Mode	Machine Code											Cycles
CPC RS1, RS2	TRI	0	0	0	1	1	0	0	0	RS1	RS2	0	1	P

CPCH — Compare Immediate 8 bit Constant with Carry (High Byte) — CPCH

Operation

RS.H - IMM8 - C ⇒ NONE, only condition code flags get updated

Subtracts the carry bit and the 8 bit constant IMM8 contained in the instruction code from the high byte of the source register RD using binary subtraction and updates the condition code register accordingly. The carry bit and Zero bits are taken into account to allow a 16 bit compare in the form of

```
CMPL    R2,#LOWBYTE
CPCH    R2,#HIGHBYTE
BCC                         ; branch condition
```

Remark: There is no equivalent operation using triadic addressing. Comparing the values of two registers can be performed by using the subtract instruction with R0 as destination register.

CCR Effects

N	Z	V	C
Δ	Δ	Δ	Δ

N: Set if bit 15 of the result is set; cleared otherwise.

Z: Set if the result is $00 and Z was set before this operation; cleared otherwise.

V: Set if a two´s complement overflow resulted from the operation; cleared otherwise.
RS[15] & $\overline{\text{IMM8[7]}}$ & $\overline{\text{result[15]}}$ | $\overline{\text{RS[15]}}$ & IMM8[7] & result[15]

C: Set if there is a carry from the bit 15 of the result; cleared otherwise.
$\overline{\text{RS[15]}}$ & IMM8[7] | $\overline{\text{RS[15]}}$ & result[15] | IMM8[7] & result[15]

Code and CPU Cycles

Source Form	Address Mode	Machine Code							Cycles
CPCH RD, #IMM8	IMM8	1	1	0	1	1	RS	IMM8	P

CSEM Clear Semaphore CSEM

Operation

Unlocks a semaphore that was locked by the RISC core.

In monadic address mode, bits RS[2:0] select the semaphore to be cleared.

CCR Effects

N	Z	V	C
—	—	—	—

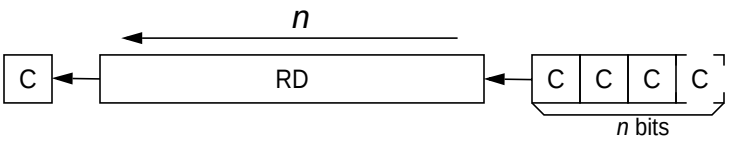
N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code														Cycles
CSEM #IMM3	IMM3	0	0	0	0	0	IMM3	1	1	1	1	0	0	0	0	PA
CSEM RS	MON	0	0	0	0	0	RS	1	1	1	1	0	0	0	1	PA

CSL — Logical Shift Left with Carry — CSL

Operation



n = RS or IMM4

Shifts the bits in register RD *n* positions to the left. The lower *n* bits of the register RD become filled with the carry flag. The carry flag will be updated to the bit contained in RD[16-n] before the shift for *n* > 0. *n* can range from 0 to 16.

In immediate address mode, *n* is determined by the operand IMM4. *n* is considered to be 16 if IMM4 is equal to 0.

In dyadic address mode, *n* is determined by the content of RS. *n* is considered to be 16 if the content of RS is greater than 15.

CCR Effects

N	Z	V	C
Δ	Δ	Δ	Δ

N: Set if bit 15 of the result is set; cleared otherwise.

Z: Set if the result is $0000; cleared otherwise.

V: Set if a two´s complement overflow resulted from the operation; cleared otherwise.
$RD[15]_{old}$ ^ $RD[15]_{new}$

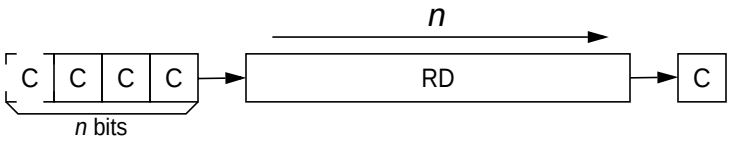
C: Set if *n* > 0 and RD[16-n] = 1; if n = 0 unaffected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code												Cycles
CSL RD, #IMM4	IMM4	0	0	0	0	1	RD	IMM4		1	0	1	0	P
CSL RD, RS	DYA	0	0	0	0	1	RD	RS	1	0	0	1	0	P

CSR Logical Shift Right with Carry CSR

Operation



n = RS or IMM4

Shifts the bits in register RD *n* positions to the right. The higher *n* bits of the register RD become filled with the carry flag. The carry flag will be updated to the bit contained in RD[n-1] before the shift for *n* > 0. *n* can range from 0 to 16.

In immediate address mode, *n* is determined by the operand IMM4. *n* is considered to be 16 if IMM4 is equal to 0.

In dyadic address mode, *n* is determined by the content of RS. *n* is considered to be 16 if the content of RS is greater than 15.

CCR Effects

N	Z	V	C
Δ	Δ	Δ	Δ

N: Set if bit 15 of the result is set; cleared otherwise.

Z: Set if the result is $0000; cleared otherwise.

V: Set if a two´s complement overflow resulted from the operation; cleared otherwise.
$RD[15]_{old}$ ^ $RD[15]_{new}$

C: Set if *n* > 0 and RD[n-1] = 1; if n = 0 unaffected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code												Cycles
CSR RD, #IMM4	IMM4	0	0	0	0	1	RD	IMM4		1	0	1	1	P
CSR RD, RS	DYA	0	0	0	0	1	RD	RS	1	0	0	1	1	P

JAL — Jump and Link — JAL

Operation

PC + $0002 ⇒ RD; RD ⇒ PC

Jumps to the address stored in RD and saves the return address in RD.

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code														Cycles
JAL RD	MON	0	0	0	0	0	RD	1	1	1	1	0	1	1	0	PP

LDB — Load Byte from Memory (Low Byte) — LDB

Operation

M[RB, #OFFS5] ⇒ RD.L; $00 ⇒ RD.H
M[RB, RI] ⇒ RD.L; $00 ⇒ RD.H
M[RB, RI] ⇒ RD.L; $00 ⇒ RD.H; RI+1 ⇒ RI;[1]
RI-1 ⇒ RI; M[RS, RI] ⇒ RD.L; $00 ⇒ RD.H

Loads a byte from memory into the low byte of register RD. The high byte is cleared.

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code										Cycles
LDB RD, (RB, #OFFS5)	IDO5	0	1	0	0	0	RD	RB	OFFS5			Pr
LDB RD, (RS, RI)	IDR	0	1	1	0	0	RD	RB	RI	0	0	Pr
LDB RD, (RS, RI+)	IDR+	0	1	1	0	0	RD	RB	RI	0	1	Pr
LDB RD, (RS, -RI)	-IDR	0	1	1	0	0	RD	RB	RI	1	0	Pr

1. If the same general purpose register is used as index (RI) and destination register (RD), the content of the register will not be incremented after the data move: M[RB, RI] ⇒ RD.L; $00 ⇒ RD.H

LDH — Load Immediate 8 bit Constant (High Byte) — LDH

Operation

IMM8 ⇒ RD.H;

Loads an 8 bit immediate constant into the high byte of register RD. The low byte is not affected.

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code							Cycles
LDH RD, #IMM8	IMM8	1	1	1	1	1	RD	IMM8	P

LDL — Load Immediate 8 bit Constant (Low Byte) — LDL

Operation

IMM8 ⇒ RD.L; $00 ⇒ RD.H

Loads an 8 bit immediate constant into the low byte of register RD. The high byte is cleared.

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code							Cycles
LDL RD, #IMM8	IMM8	1	1	1	1	0	RD	IMM8	P

LDW — Load Word from Memory — LDW

Operation

M[RB, #OFFS5] ⇒ RD
M[RB, RI] ⇒ RD
M[RB, RI] ⇒ RD; RI+2 ⇒ RI[1]
RI-2 ⇒ RI; M[RS, RI] ⇒ RD
IMM16 ⇒ RD (translates to LDL RD, #IMM16[7:0]; LDH RD, #IMM16[15:8])

Loads a 16 bit value into the register RD.

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

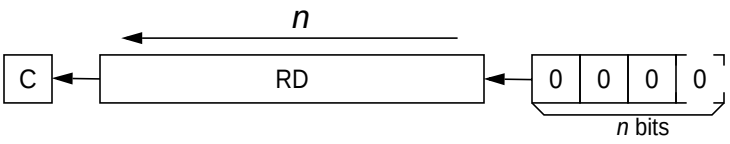
Code and CPU Cycles

Source Form	Address Mode	Machine Code										Cycles
LDW RD, (RB, #OFFS5)	IDO5	0	1	0	0	1	RD	RB	OFFS5			PR
LDW RD, (RB, RI)	IDR	0	1	1	0	1	RD	RB	RI	0	0	PR
LDW RD, (RB, RI+)	IDR+	0	1	1	0	1	RD	RB	RI	0	1	PR
LDW RD, (RB, -RI)	-IDR	0	1	1	0	1	RD	RB	RI	1	0	PR
LDW RD, #IMM16	IMM8	1	1	1	1	0	RD	IMM16[7:0]				P
	IMM8	1	1	1	1	1	RD	IMM16[15:8]				P

1. If the same general purpose register is used as index (RI) and destination register (RD), the content of the register will not be incremented after the data move: M[RB, RI] ⇒ RD

LSL Logical Shift Left LSL

Operation



n = RS or IMM4

Shifts the bits in register RD *n* positions to the left. The lower *n* bits of the register RD become filled with zeros. The carry flag will be updated to the bit contained in RD[16-n] before the shift for *n* > 0.

n can range from 0 to 16.

In immediate address mode, *n* is determined by the operand IMM4. *n* is considered to be 16 in IMM4 is equal to 0.

In dyadic address mode, *n* is determined by the content of RS. *n* is considered to be 16 if the content of RS is greater than 15.

CCR Effects

N	Z	V	C
Δ	Δ	Δ	Δ

N: Set if bit 15 of the result is set; cleared otherwise.

Z: Set if the result is $0000; cleared otherwise.

V: Set if a two´s complement overflow resulted from the operation; cleared otherwise.
$RD[15]_{old}$ ^ $RD[15]_{new}$

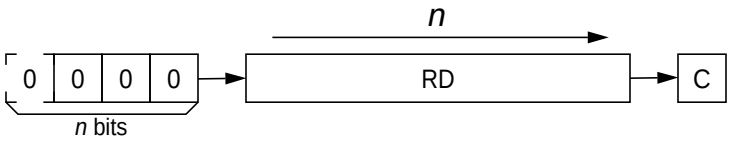
C: Set if *n* > 0 and RD[16-n] = 1; if n = 0 unaffected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code												Cycles
LSL RD, #IMM4	IMM4	0	0	0	0	1	RD	IMM4		1	1	0	0	P
LSL RD, RS	DYA	0	0	0	0	1	RD	RS	1	0	1	0	0	P

LSR — Logical Shift Right — LSR

Operation



n = RS or IMM4

Shifts the bits in register RD *n* positions to the right. The higher *n* bits of the register RD become filled with zeros. The carry flag will be updated to the bit contained in RD[n-1] before the shift for *n* > 0.

n can range from 0 to 16.

In immediate address mode, *n* is determined by the operand IMM4. *n* is considered to be 16 in IMM4 is equal to 0.

In dyadic address mode, *n* is determined by the content of RS. *n* is considered to be 16 if the content of RS is greater than 15.

CCR Effects

N	Z	V	C
Δ	Δ	Δ	Δ

N: Set if bit 15 of the result is set; cleared otherwise.

Z: Set if the result is $0000; cleared otherwise.

V: Set if a two´s complement overflow resulted from the operation; cleared otherwise.
$RD[15]_{old}$ ^ $RD[15]_{new}$

C: Set if *n* > 0 and RD[n-1] = 1; if n = 0 unaffected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code												Cycles
LSR RD, #IMM4	IMM4	0	0	0	0	1	RD	IMM4		1	1	0	1	P
LSR RD, RS	DYA	0	0	0	0	1	RD	RS	1	0	1	0	1	P

MOV — Move Register Content — MOV

Operation

RS ⇒ RD (translates to OR RD, R0, RS)

Copies the content of RS to RD.

CCR Effects

N	Z	V	C
Δ	Δ	0	—

N: Set if bit 15 of the result is set; cleared otherwise.
Z: Set if the result is $0000; cleared otherwise.
V: 0; cleared.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code											Cycles
MOV RD, RS	TRI	0	0	0	1	0	RD	0	0	0	RS	1	0	P

NEG — Two's Complement — NEG

Operation

–RS ⇒ RD (translates to SUB RD, R0, RS)
–RD ⇒ RD (translates to SUB RD, R0, RD)

Performs a two's complement on a general purpose register.

CCR Effects

N	Z	V	C
Δ	Δ	Δ	Δ

N: Set if bit 15 of the result is set; cleared otherwise.
Z: Set if the result is $0000; cleared otherwise.
V: Set if a two´s complement overflow resulted from the operation; cleared otherwise.
$RS[15] \& RD[15]_{new}$
C: Set if there is a carry from the bit 15 of the result; cleared otherwise
$RS[15] | RD[15]_{new}$

Code and CPU Cycles

Source Form	Address Mode	Machine Code										Cycles
NEG RD, RS	TRI	0	0	0	1	1	RD	0	0	0	RS	0	0	P
NEG RD	TRI	0	0	0	1	1	RD	0	0	0	RD	0	0	P

NOP — No Operation — NOP

Operation

No Operation for one cycle.

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code																Cycles
NOP	INH	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	P

OR — Logical OR — OR

Operation

RS1 | RS2 ⇒ RD
RD | IMM16⇒ RD (translates to ORL RD, #IMM16[7:0]; ORH RD, #IMM16[15:8]

Performs a bit wise logical OR between two 16 bit values and stores the result in the destination register RD.

NOTE

When using immediate addressing mode (OR RD, #IMM16), the Z-flag of the first instruction (ORL RD, #IMM16[7:0]) is not considered by the second instruction (ORH RD, #IMM16[15:8]).
⇒ Don’t rely on the Z-Flag.

CCR Effects

N	Z	V	C
Δ	Δ	0	—

N: Set if bit 15 of the result is set; cleared otherwise.
Z: Set if the result is $0000; cleared otherwise.
Refer to ORH instruction for #IMM16 operations.
V: 0; cleared.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code										Cycles
OR RD, RS1, RS2	TRI	0	0	0	1	0	RD	RS1	RS2	1	0	P
OR RD, #IMM16	IMM8	1	0	1	0	0	RD	IMM16[7:0]				P
	IMM8	1	0	1	0	1	RD	IMM16[15:8]				P

ORH — Logical OR Immediate 8 bit Constant (High Byte) — ORH

Operation

RD.H | IMM8 ⇒ RD.H

Performs a bit wise logical OR between the high byte of register RD and an immediate 8 bit constant and stores the result in the destination register RD.H. The low byte of RD is not affected.

CCR Effects

N	Z	V	C
Δ	Δ	0	—

N: Set if bit 15 of the result is set; cleared otherwise.
Z: Set if the 8 bit result is $00; cleared otherwise.
V: 0; cleared.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code							Cycles
ORH RD, #IMM8	IMM8	1	0	1	0	1	RD	IMM8	P

ORL — Logical OR Immediate 8 bit Constant (Low Byte) — ORL

Operation

RD.L | IMM8 ⇒ RD.L

Performs a bit wise logical OR between the low byte of register RD and an immediate 8 bit constant and stores the result in the destination register RD.L. The high byte of RD is not affected.

CCR Effects

N	Z	V	C
Δ	Δ	0	—

N: Set if bit 7 of the result is set; cleared otherwise.
Z: Set if the 8 bit result is $00; cleared otherwise.
V: 0; cleared.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code							Cycles
ORL RD, #IMM8	IMM8	1	0	1	0	0	RD	IMM8	P

PAR Calculate Parity PAR

Operation

Calculates the number of ones in the register RD. The Carry flag will be set if the number is odd, otherwise it will be cleared.

CCR Effects

N	Z	V	C
0	Δ	0	Δ

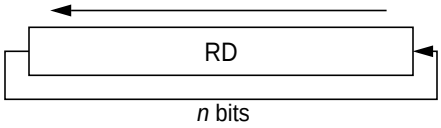
N: 0; cleared.
Z: Set if RD is $0000; cleared otherwise.
V: 0; cleared.
C: Set if the number of ones in the register RD is odd; cleared otherwise.

Code and CPU Cycles

Source Form	Address Mode	Machine Code														Cycles
PAR, RD	MON	0	0	0	0	0	RD	1	1	1	1	0	1	0	1	P

ROL — Rotate Left — ROL

Operation



n = RS or IMM4

Rotates the bits in register RD *n* positions to the left. The lower *n* bits of the register RD are filled with the upper *n* bits. Two source forms are available. In the first form, the parameter *n* is contained in the instruction code as an immediate operand. In the second form, the parameter is contained in the lower bits of the source register RS[3:0]. All other bits in RS are ignored. If *n* is zero, no shift will take place and the register RD will be unaffected; however, the condition code flags will be updated.

CCR Effects

N	Z	V	C
Δ	Δ	0	—

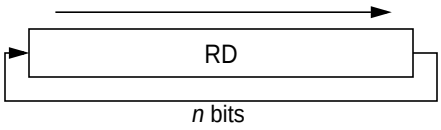
N: Set if bit 15 of the result is set; cleared otherwise.
Z: Set if the result is $0000; cleared otherwise.
V: 0; cleared.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code												Cycles
ROL RD, #IMM4	IMM4	0	0	0	0	1	RD	IMM4		1	1	1	0	P
ROL RD, RS	DYA	0	0	0	0	1	RD	RS	1	0	1	1	0	P

ROR — Rotate Right — ROR

Operation



n = RS or IMM4

Rotates the bits in register RD *n* positions to the right. The upper *n* bits of the register RD are filled with the lower *n* bits. Two source forms are available. In the first form, the parameter *n* is contained in the instruction code as an immediate operand. In the second form, the parameter is contained in the lower bits of the source register RS[3:0]. All other bits in RS are ignored. If *n* is zero no shift will take place and the register RD will be unaffected; however, the condition code flags will be updated.

CCR Effects

N	Z	V	C
Δ	Δ	0	—

N: Set if bit 15 of the result is set; cleared otherwise.
Z: Set if the result is $0000; cleared otherwise.
V: 0; cleared.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code												Cycles
ROR RD, #IMM4	IMM4	0	0	0	0	1	RD	IMM4		1	1	1	1	P
ROR RD, RS	DYA	0	0	0	0	1	RD	RS	1	0	1	1	1	P

RTS — Return to Scheduler — RTS

Operation

Terminates the current thread of program execution.

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code																Cycles
RTS	INH	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	PA

SBC — Subtract with Carry — SBC

Operation

RS1 - RS2 - C ⇒ RD

Subtracts the content of register RS2 and the value of the Carry bit from the content of register RS1 using binary subtraction and stores the result in the destination register RD. Also the zero flag is carried forward from the previous operation allowing 32 and more bit subtractions.

Example:

```
SUB     R6,R4,R2
SBC     R7,R5,R3          ; R7:R6 = R5:R4 - R3:R2
BCC                       ; conditional branch on 32 bit subtraction
```

CCR Effects

N	Z	V	C
Δ	Δ	Δ	Δ

N: Set if bit 15 of the result is set; cleared otherwise.

Z: Set if the result is $0000 and Z was set before this operation; cleared otherwise.

V: Set if a two´s complement overflow resulted from the operation; cleared otherwise.
$RS1[15] \& \overline{RS2[15]} \& \overline{RD[15]}_{new} \mid \overline{RS1[15]} \& RS2[15] \& RD[15]_{new}$

C: Set if there is a carry from bit 15 of the result; cleared otherwise.
$\overline{RS1[15]} \& RS2[15] \mid \overline{RS1[15]} \& RD[15]_{new} \mid RS2[15] \& RD[15]_{new}$

Code and CPU Cycles

Source Form	Address Mode	Machine Code										Cycles
SBC RD, RS1, RS2	TRI	0	0	0	1	1	RD	RS1	RS2	0	1	P

SEX — Sign Extend Byte to Word — SEX

Operation

The result in RD is the 16 bit sign extended representation of the original two's complement number in the low byte of RD.L.

CCR Effects

N	Z	V	C
Δ	Δ	0	—

N: Set if bit 15 of the result is set; cleared otherwise.
Z: Set if the result is $0000; cleared otherwise.
V: 0; cleared.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code														Cycles
SEX RD	MON	0	0	0	0	0	RD	1	1	1	1	0	1	0	0	P

SIF — Set Interrupt Flag — SIF

Operation

Sets the interrupt flag of an XGATE channel (XGIF). This instruction supports two source forms. If inherent address mode is used, then the interrupt flag of the current channel (XGCHID) will be set. If the monadic address form is used, the interrupt flag associated with the channel id number contained in RS[6:0] is set. The content of RS[15:7] is ignored.

NOTE

Interrupt flags of reserved channels (see Device User Guide) can't be set.

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code																Cycles
SIF	INH	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	PA
SIF RS	MON	0	0	0	0	0	RS			1	1	1	1	0	1	1	1	PA

SSEM Set Semaphore SSEM

Operation

Attempts to set a semaphore. The state of the semaphore will be stored in the Carry-Flag:

1 = Semaphore is locked by the RISC core

0 = Semaphore is locked by the S12X_CPU

In monadic address mode, bits RS[2:0] select the semaphore to be set.

CCR Effects

N	Z	V	C
—	—	—	Δ

N: Not affected.
Z: Not affected.
V: Not affected.
C: Set if semaphore is locked by the RISC core; cleared otherwise.

Code and CPU Cycles

Source Form	Address Mode	Machine Code														Cycles
SSEM #IMM3	IMM3	0	0	0	0	0	IMM3	1	1	1	1	0	0	1	0	PA
SSEM RS	MON	0	0	0	0	0	RS	1	1	1	1	0	0	1	1	PA

STB — Store Byte to Memory (Low Byte) — STB

Operation

RS.L ⇒ M[RB, #OFFS5]
RS.L ⇒ M[RB, RI]
RS.L ⇒ M[RB, RI]; RI+1 ⇒ RI;
RI–1 ⇒ RI; RS.L ⇒ M[RB, RI][1]

Stores the low byte of register RS to memory.

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code										Cycles
STB RS, (RB, #OFFS5),	IDO5	0	1	0	1	0	RS	RB	OFFS5			Pw
STB RS, (RB, RI)	IDR	0	1	1	1	0	RS	RB	RI	0	0	Pw
STB RS, (RB, RI+)	IDR+	0	1	1	1	0	RS	RB	RI	0	1	Pw
STB RS, (RB, -RI)	-IDR	0	1	1	1	0	RS	RB	RI	1	0	Pw

1. If the same general purpose register is used as index (RI) and source register (RS), the unmodified content of the source register is written to the memory: RS.L ⇒ M[RB, RS-1]; RS-1 ⇒ RS

STW Store Word to Memory STW

Operation

RS ⇒ M[RB, #OFFS5]
RS ⇒ M[RB, RI]
RS ⇒ M[RB, RI]; RI+2 ⇒ RI;
RI–2 ⇒ RI; RS ⇒ M[RB, RI][1]

Stores the content of register RS to memory.

CCR Effects

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code										Cycles
STW RS, (RB, #OFFS5)	IDO5	0	1	0	1	1	RS	RB	OFFS5			PW
STW RS, (RB, RI)	IDR	0	1	1	1	1	RS	RB	RI	0	0	PW
STW RS, (RB, RI+)	IDR+	0	1	1	1	1	RS	RB	RI	0	1	PW
STW RS, (RB, -RI)	-IDR	0	1	1	1	1	RS	RB	RI	1	0	PW

1. If the same general purpose register is used as index (RI) and source register (RS), the unmodified content of the source register is written to the memory: RS ⇒ M[RB, RS–2]; RS–2 ⇒ RS

SUB — Subtract without Carry — SUB

Operation

RS1 – RS2 ⇒ RD
RD – IMM16 ⇒ RD (translates to SUBL RD, #IMM16[7:0]; SUBH RD, #IMM16{15:8])

Subtracts two 16 bit values and stores the result in the destination register RD.

NOTE

When using immediate addressing mode (SUB RD, #IMM16), the V-flag and the C-Flag of the first instruction (SUBL RD, #IMM16[7:0]) are not considered by the second instruction (SUBH RD, #IMM16[15:8]).
⇒ Don't rely on the V-Flag if RD - IMM16[7:0] < -2^{15}.
⇒ Don't rely on the C-Flag if RD < IMM16[7:0].

CCR Effects

N	Z	V	C
Δ	Δ	Δ	Δ

N: Set if bit 15 of the result is set; cleared otherwise.

Z: Set if the result is $0000; cleared otherwise.

V: Set if a two´s complement overflow resulted from the operation; cleared otherwise.
$RS1[15] \& \overline{RS2[15]} \& \overline{RD[15]}_{new} \mid \overline{RS1[15]} \& RS2[15] \& RD[15]_{new}$
Refer to SUBH instruction for #IMM16 operations.

C: Set if there is a carry from the bit 15 of the result; cleared otherwise.
$\overline{RS1[15]} \& RS2[15] \mid \overline{RS1[15]} \& RD[15]_{new} \mid RS2[15] \& RD[15]_{new}$
Refer to SUBH instruction for #IMM16 operations.

Code and CPU Cycles

Source Form	Address Mode	Machine Code										Cycles
SUB RD, RS1, RS2	TRI	0	0	0	1	1	RD	RS1	RS2	0	0	P
SUB RD, #IMM16	IMM8	1	1	0	0	0	RD	IMM16[7:0]				P
	IMM8	1	1	0	0	1	RD	IMM16[15:8]				P

SUBH — Subtract Immediate 8 bit Constant (High Byte) — SUBH

Operation

RD – IMM8:$00 ⇒ RD

Subtracts a signed immediate 8 bit constant from the content of high byte of register RD and using binary subtraction and stores the result in the high byte of destination register RD. This instruction can be used after an SUBL for a 16 bit immediate subtraction.

Example:

```
SUBL    R2,#LOWBYTE
SUBH    R2,#HIGHBYTE      ; R2 = R2 - 16 bit immediate
```

CCR Effects

N	Z	V	C
Δ	Δ	Δ	Δ

N: Set if bit 15 of the result is set; cleared otherwise.

Z: Set if the result is $0000; cleared otherwise.

V: Set if a two´s complement overflow resulted from the operation; cleared otherwise.
$RD[15]_{old} \& \overline{IMM8[7]} \& \overline{RD[15]}_{new} \mid \overline{RD[15]}_{old} \& IMM8[7] \& RD[15]_{new}$

C: Set if there is a carry from the bit 15 of the result; cleared otherwise.
$\overline{RD[15]}_{old} \& IMM8[7] \mid \overline{RD[15]}_{old} \& RD[15]_{new} \mid IMM8[7] \& RD[15]_{new}$

Code and CPU Cycles

Source Form	Address Mode	Machine Code							Cycles
SUBH RD, #IMM8	IMM8	1	1	0	0	1	RD	IMM8	P

SUBL — Subtract Immediate 8 bit Constant (Low Byte) — SUBL

Operation

RD – $00:IMM8 ⇒ RD

Subtracts an immediate 8 bit constant from the content of register RD using binary subtraction and stores the result in the destination register RD.

CCR Effects

N	Z	V	C
Δ	Δ	Δ	Δ

N: Set if bit 15 of the result is set; cleared otherwise.

Z: Set if the result is $0000; cleared otherwise.

V: Set if a two´s complement overflow resulted from the 8 bit operation; cleared otherwise.
$RD[15]_{old}$ & $\overline{RD[15]}_{new}$

C: Set if there is a carry from the bit 15 of the result; cleared otherwise.
$\overline{RD[15]}_{old}$ & $RD[15]_{new}$

Code and CPU Cycles

Source Form	Address Mode	Machine Code							Cycles
SUBL RD, #IMM8	IMM8	1	1	0	0	0	RD	IMM8	P

TFR Transfer from and to Special Registers TFR

Operation

TFR RD,CCR: CCR ⇒ RD[3:0]; 0 ⇒ RD[15:4]
TFR CCR,RD: RD[3:0] ⇒ CCR
TFR RD,PC: PC+4 ⇒ RD

Transfers the content of one RISC core register to another.
The TFR RD,PC instruction can be used to implement relative subroutine calls.

Example:

```
        TFR     R7,PC    ;Return address (RETADDR) is stored in R7
        BRA     SUBR     ;Relative branch to subroutine (SUBR)
RETADDR ...

SUBR    ...
        JAL     R7       ;Jump to return address (RETADDR)
```

CCR Effects

TFR RD,CCR, TFR RD,PC:

N	Z	V	C
—	—	—	—

N: Not affected.
Z: Not affected.
V: Not affected.
C: Not affected.

TFR CCR,RS:

N	Z	V	C
Δ	Δ	Δ	Δ

N: RS[3].
Z: RS[2].
V: RS[1].
C: RS[0].

Code and CPU Cycles

Source Form	Address Mode	Machine Code														Cycles
TFR RD,CCR CCR ⇒ RD	MON	0	0	0	0	0	RD	1	1	1	1	1	0	0	0	P
TFR CCR,RS RS ⇒ CCR	MON	0	0	0	0	0	RS	1	1	1	1	1	0	0	1	P
TFR RD,PCPC+4 ⇒ RD	MON	0	0	0	0	0	RD	1	1	1	1	1	0	1	0	P

TST Test Register TST

Operation

RS – 0 ⇒ NONE (translates to SUB R0, RS, R0)

Subtracts zero from the content of register RS using binary subtraction and discards the result.

CCR Effects

N	Z	V	C
Δ	Δ	Δ	Δ

N: Set if bit 15 of the result is set; cleared otherwise.

Z: Set if the result is $0000; cleared otherwise.

V: Set if a two´s complement overflow resulted from the operation; cleared otherwise.
RS[15] & $\overline{\text{result[15]}}$

C: Set if there is a carry from the bit 15 of the result; cleared otherwise.
$\overline{\text{RS1[15]}}$ & result[15]

Code and CPU Cycles

Source Form	Address Mode	Machine Code												Cycles
TST RS	TRI	0	0	0	1	1	0	0	0	RS1	0	0	0	P

XNOR Logical Exclusive NOR XNOR

Operation

~(RS1 ^ RS2) ⇒ RD
~(RD ^ IMM16)⇒ RD
(translates to XNOR RD, #IMM16{15:8]; XNOR RD, #IMM16[7:0])

Performs a bit wise logical exclusive NOR between two 16 bit values and stores the result in the destination register RD.

Remark: Using R0 as a source registers will calculate the one's complement of the other source register. Using R0 as both source operands will fill RD with $FFFF.

NOTE

When using immediate addressing mode (XNOR RD, #IMM16), the Z-flag of the first instruction (XNORL RD, #IMM16[7:0]) is not considered by the second instruction (XNORH RD, #IMM16[15:8]).
⇒ Don't rely on the Z-Flag.

CCR Effects

N	Z	V	C
Δ	Δ	0	—

N: Set if bit 15 of the result is set; cleared otherwise.
Z: Set if the result is $0000; cleared otherwise.
Refer to XNORH instruction for #IMM16 operations.
V: 0; cleared.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code										Cycles
XNOR RD, RS1, RS2	TRI	0	0	0	1	0	RD	RS1	RS2	1	1	P
XNOR RD, #IMM16	IMM8	1	0	1	1	0	RD	IMM16[7:0]				P
	IMM8	1	0	1	1	1	RD	IMM16[15:8]				P

XNORH Logical Exclusive NOR Immediate 8 bit Constant (High Byte) XNORH

Operation

~(RD.H ^ IMM8) ⇒ RD.H

Performs a bit wise logical exclusive NOR between the high byte of register RD and an immediate 8 bit constant and stores the result in the destination register RD.H. The low byte of RD is not affected.

CCR Effects

N	Z	V	C
Δ	Δ	0	—

N: Set if bit 15 of the result is set; cleared otherwise.
Z: Set if the 8 bit result is $00; cleared otherwise.
V: 0; cleared.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code							Cycles
XNORH RD, #IMM8	IMM8	1	0	1	1	1	RD	IMM8	P

XNORL — Logical Exclusive NOR Immediate 8 bit Constant (Low Byte) — XNORL

Operation

~(RD.L ^ IMM8) ⇒ RD.L

Performs a bit wise logical exclusive NOR between the low byte of register RD and an immediate 8 bit constant and stores the result in the destination register RD.L. The high byte of RD is not affected.

CCR Effects

N	Z	V	C
Δ	Δ	0	—

N: Set if bit 7 of the result is set; cleared otherwise.
Z: Set if the 8 bit result is $00; cleared otherwise.
V: 0; cleared.
C: Not affected.

Code and CPU Cycles

Source Form	Address Mode	Machine Code							Cycles
XNORL RD, #IMM8	IMM8	1	0	1	1	0	RD	IMM8	P

14.8.6 Instruction Coding

Table 14-24 summarizes all XGATE instructions in the order of their machine coding.

Table 14-24. Instruction Set Summary (Sheet 1 of 3)

Functionality	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Return to Scheduler and Others																
BRK	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
NOP	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
RTS	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
SIF	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0
Semaphore Instructions																
CSEM IMM3	0	0	0	0	0	IMM3			1	1	1	1	0	0	0	0
CSEM RS	0	0	0	0	0	RS			1	1	1	1	0	0	0	1
SSEM IMM3	0	0	0	0	0	IMM3			1	1	1	1	0	0	1	0
SSEM RS	0	0	0	0	0	RS			1	1	1	1	0	0	1	1
Single Register Instructions																
SEX RD	0	0	0	0	0	RD			1	1	1	1	0	1	0	0
PAR RD	0	0	0	0	0	RD			1	1	1	1	0	1	0	1
JAL RD	0	0	0	0	0	RD			1	1	1	1	0	1	1	0
SIF RS	0	0	0	0	0	RS			1	1	1	1	0	1	1	1
Special Move instructions																
TFR RD,CCR	0	0	0	0	0	RD			1	1	1	1	1	0	0	0
TFR CCR,RS	0	0	0	0	0	RS			1	1	1	1	1	0	0	1
TFR RD,PC	0	0	0	0	0	RD			1	1	1	1	1	0	1	0
Shift instructions Dyadic																
BFFO RD, RS	0	0	0	0	1	RD			RS			1	0	0	0	0
ASR RD, RS	0	0	0	0	1	RD			RS			1	0	0	0	1
CSL RD, RS	0	0	0	0	1	RD			RS			1	0	0	1	0
CSR RD, RS	0	0	0	0	1	RD			RS			1	0	0	1	1
LSL RD, RS	0	0	0	0	1	RD			RS			1	0	1	0	0
LSR RD, RS	0	0	0	0	1	RD			RS			1	0	1	0	1
ROL RD, RS	0	0	0	0	1	RD			RS			1	0	1	1	0
ROR RD, RS	0	0	0	0	1	RD			RS			1	0	1	1	1
Shift instructions immediate																
ASR RD, #IMM4	0	0	0	0	1	RD			IMM4				1	0	0	1
CSL RD, #IMM4	0	0	0	0	1	RD			IMM4				1	0	1	0
CSR RD, #IMM4	0	0	0	0	1	RD			IMM4				1	0	1	1
LSL RD, #IMM4	0	0	0	0	1	RD			IMM4				1	1	0	0
LSR RD, #IMM4	0	0	0	0	1	RD			IMM4				1	1	0	1
ROL RD, #IMM4	0	0	0	0	1	RD			IMM4				1	1	1	0
ROR RD, #IMM4	0	0	0	0	1	RD			IMM4				1	1	1	1
Logical Triadic																
AND RD, RS1, RS2	0	0	0	1	0	RD			RS1			RS2			0	0
OR RD, RS1, RS2	0	0	0	1	0	RD			RS1			RS2			1	0
XNOR RD, RS1, RS2	0	0	0	1	0	RD			RS1			RS2			1	1
Arithmetic Triadic	For compare use SUB R0,Rs1,Rs2															
SUB RD, RS1, RS2	0	0	0	1	1	RD			RS1			RS2			0	0
SBC RD, RS1, RS2	0	0	0	1	1	RD			RS1			RS2			0	1
ADD RD, RS1, RS2	0	0	0	1	1	RD			RS1			RS2			1	0
ADC RD, RS1, RS2	0	0	0	1	1	RD			RS1			RS2			1	1

Table 14-24. Instruction Set Summary (Sheet 2 of 3)

Functionality	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Branches																
BCC REL9	0	0	1	0	0	0	0	REL9								
BCS REL9	0	0	1	0	0	0	1	REL9								
BNE REL9	0	0	1	0	0	1	0	REL9								
BEQ REL9	0	0	1	0	0	1	1	REL9								
BPL REL9	0	0	1	0	1	0	0	REL9								
BMI REL9	0	0	1	0	1	0	1	REL9								
BVC REL9	0	0	1	0	1	1	0	REL9								
BVS REL9	0	0	1	0	1	1	1	REL9								
BHI REL9	0	0	1	1	0	0	0	REL9								
BLS REL9	0	0	1	1	0	0	1	REL9								
BGE REL9	0	0	1	1	0	1	0	REL9								
BLT REL9	0	0	1	1	0	1	1	REL9								
BGT REL9	0	0	1	1	1	0	0	REL9								
BLE REL9	0	0	1	1	1	0	1	REL9								
BRA REL10	0	0	1	1	1	1	REL10									
Load and Store Instructions																
LDB RD, (RB, #OFFS5)	0	1	0	0	0	RD			RB			OFFS5				
LDW RD, (RB, #OFFS5)	0	1	0	0	1	RD			RB			OFFS5				
STB RS, (RB, #OFFS5)	0	1	0	1	0	RS			RB			OFFS5				
STW RS, (RB, #OFFS5)	0	1	0	1	1	RS			RB			OFFS5				
LDB RD, (RB, RI)	0	1	1	0	0	RD			RB			RI			0	0
LDW RD, (RB, RI)	0	1	1	0	1	RD			RB			RI			0	0
STB RS, (RB, RI)	0	1	1	1	0	RS			RB			RI			0	0
STW RS, (RB, RI)	0	1	1	1	1	RS			RB			RI			0	0
LDB RD, (RB, RI+)	0	1	1	0	0	RD			RB			RI			0	1
LDW RD, (RB, RI+)	0	1	1	0	1	RD			RB			RI			0	1
STB RS, (RB, RI+)	0	1	1	1	0	RS			RB			RI			0	1
STW RS, (RB, RI+)	0	1	1	1	1	RS			RB			RI			0	1
LDB RD, (RB, –RI)	0	1	1	0	0	RD			RB			RI			1	0
LDW RD, (RB, –RI)	0	1	1	0	1	RD			RB			RI			1	0
STB RS, (RB, –RI)	0	1	1	1	0	RS			RB			RI			1	0
STW RS, (RB, –RI)	0	1	1	1	1	RS			RB			RI			1	0
Bit Field Instructions																
BFEXT RD, RS1, RS2	0	1	1	0	0	RD			RS1			RS2			1	1
BFINS RD, RS1, RS2	0	1	1	0	1	RD			RS1			RS2			1	1
BFINSI RD, RS1, RS2	0	1	1	1	0	RD			RS1			RS2			1	1
BFINSX RD, RS1, RS2	0	1	1	1	1	RD			RS1			RS2			1	1
Logic Immediate Instructions																
ANDL RD, #IMM8	1	0	0	0	0	RD			IMM8							
ANDH RD, #IMM8	1	0	0	0	1	RD			IMM8							
BITL RD, #IMM8	1	0	0	1	0	RD			IMM8							
BITH RD, #IMM8	1	0	0	1	1	RD			IMM8							
ORL RD, #IMM8	1	0	1	0	0	RD			IMM8							
ORH RD, #IMM8	1	0	1	0	1	RD			IMM8							
XNORL RD, #IMM8	1	0	1	1	0	RD			IMM8							
XNORH RD, #IMM8	1	0	1	1	1	RD			IMM8							

Table 14-24. Instruction Set Summary (Sheet 3 of 3)

Functionality	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Arithmetic Immediate Instructions																
SUBL RD, #IMM8	1	1	0	0	0	RD			IMM8							
SUBH RD, #IMM8	1	1	0	0	1	RD			IMM8							
CMPL RS, #IMM8	1	1	0	1	0	RS			IMM8							
CPCH RS, #IMM8	1	1	0	1	1	RS			IMM8							
ADDL RD, #IMM8	1	1	1	0	0	RD			IMM8							
ADDH RD, #IMM8	1	1	1	0	1	RD			IMM8							
LDL RD, #IMM8	1	1	1	1	0	RD			IMM8							
LDH RD, #IMM8	1	1	1	1	1	RD			IMM8							

14.9 Initialization and Application Information

14.9.1 Initialization

The recommended initialization of the XGATE is as follows:

1. Clear the XGE bit to suppress any incoming service requests.
2. Make sure that no thread is running on the XGATE. This can be done in several ways:
   a) Poll the XGCHID register until it reads $00. Also poll XGDBG and XGSWEF to make sure that the XGATE has not been stopped.
   b) Enter Debug Mode by setting the XGDBG bit. Clear the XGCHID register. Clear the XGDBG bit.

   The recommended method is a).
3. Set the XGVBR register to the lowest address of the XGATE vector space.
4. Clear all Channel ID flags.
5. Copy XGATE vectors and code into the RAM.
6. Initialize the S12X_INT module.
7. Enable the XGATE by setting the XGE bit.

The following code example implements the XGATE initialization sequence.

14.9.2 Code Example (Transmit "Hello World!" on SCI)

```
                        CPU   S12X
                        ;###########################################
                        ;#                  SYMBOLS                #
                        ;###########################################
SCI_REGS                EQU   $00C8           ;SCI register space
SCIBDH                  EQU   SCI_REGS+$00;   ;SCI Baud Rate Register
SCIBDL                  EQU   SCI_REGS+$00    ;SCI Baud Rate Register
SCICR2                  EQU   SCI_REGS+$03    ;SCI Control Register 2
SCISR1                  EQU   SCI_REGS+$04    ;SCI Status  Register 1
SCIDRL                  EQU   SCI_REGS+$07    ;SCI Control Register 2
TIE                     EQU   $80             ;TIE bit mask
TE                      EQU   $08             ;TE bit mask
RE                      EQU   $04             ;RE bit mask
```

```
SCI_VEC                 EQU   $D6             ;SCI vector number

INT_REGS                EQU   $0120           ;S12X_INT register space
INT_CFADDR              EQU   INT_REGS+$07    ;Interrupt Configuration Address Register
INT_CFDATA              EQU   INT_REGS+$08    ;Interrupt Configuration Data Registers
RQST                    EQU   $80             ;RQST bit mask

XGATE_REGS              EQU   $0380           ;XGATE register space
XGMCTL                  EQU   XGATE_REGS+$00  ;XGATE Module Control Register
XGMCTL_CLEAR            EQU   $FA02           ;Clear all XGMCTL bits
XGMCTL_ENABLE           EQU   $8282           ;Enable XGATE
XGCHID                  EQU   XGATE_REGS+$02  ;XGATE Channel ID Register
XGISPSEL                EQU   XGATE_REGS+$05  ;XGATE Channel ID Register
XGVBR                   EQU   XGATE_REGS+$06  ;XGATE ISP Select Register
XGIF                    EQU   XGATE_REGS+$08  ;XGATE Interrupt Flag Vector
XGSWT                   EQU   XGATE_REGS+$18  ;XGATE Software Trigger Register
XGSEM                   EQU   XGATE_REGS+$1A  ;XGATE Semaphore Register

RPAGE                   EQU   $0016

RAM_SIZE                EQU   32*$400         ;32k RAM

RAM_START               EQU   $1000
RAM_START_XG            EQU   $10000-RAM_SIZE
RAM_START_GLOB          EQU   $100000-RAM_SIZE

XGATE_VECTORS           EQU   RAM_START
XGATE_VECTORS_XG        EQU   RAM_START_XG

XGATE_DATA              EQU   RAM_START+(4*128)
XGATE_DATA_XG           EQU   RAM_START_XG+(4*128)

XGATE_CODE              EQU   XGATE_DATA+(XGATE_CODE_FLASH-XGATE_DATA_FLASH)
XGATE_CODE_XG           EQU   XGATE_DATA_XG+(XGATE_CODE_FLASH-XGATE_DATA_FLASH)

BUS_FREQ_HZ             EQU   40000000

                        ;###########################################
                        ;#           S12XE VECTOR TABLE            #
                        ;###########################################
                        ORG   $FF10           ;non-maskable interrupts
                        DW    DUMMY_ISR DUMMY_ISR DUMMY_ISR DUMMY_ISR

                        ORG   $FFF4           ;non-maskable interrupts
                        DW    DUMMY_ISR DUMMY_ISR DUMMY_ISR

                        ORG   $FFFA           ;resets
                        DW    START_OF_CODE START_OF_CODE START_OF_CODE

                        ;###########################################
                        ;#                DISABLE COP              #
                        ;###########################################
                        ORG   $FF0E
                        DW    $FFFE

                        ORG   $C000
START_OF_CODE
```

```
                        ;###########################################
                        ;#           INITIALIZE S12XE CORE         #
                        ;###########################################
                        SEI
                        MOVB  #(RAM_START_GLOB>>12), RPAGE    ;set RAM page

                        ;###########################################
                        ;#              INITIALIZE SCI             #
                        ;###########################################
INIT_SCI                MOVW  #(BUS_FREQ_HZ/(16*9600)), SCIBDH;set baud rate
                        MOVB  #(TIE|TE), SCICR2;enable tx buffer empty interrupt

                        ;###########################################
                        ;#           INITIALIZE S12X_INT           #
                        ;###########################################
INIT_INT                MOVB  #(SCI_VEC&$F0), INT_CFADDR      ;switch SCI interrupts to XGATE
                        MOVB  #RQST|$01, INT_CFDATA+((SCI_VEC&$0F)>>1)

                        ;###########################################
                        ;#             INITIALIZE XGATE            #
                        ;###########################################
INIT_XGATE              MOVW  #XGMCTL_CLEAR, XGMCTL           ;clear all XGMCTL bits

INIT_XGATE_BUSY_LOOP    TST   XGCHID          ;wait until current thread is finished
                        BNE   INIT_XGATE_BUSY_LOOP

                        LDX   #XGIF           ;clear all channel interrupt flags
                        LDD   #$FFFF
                        STD   2,X+
                        STD   2,X+
                        STD   2,X+
                        STD   2,X+
                        STD   2,X+
                        STD   2,X+
                        STD   2,X+
                        STD   2,X+

                        CLR   XGISPSEL        ;set vector base register
                        MOVW  #XGATE_VECTORS_XG, XGVBR
                        MOVW  #$FF00, XGSWT   ;clear all software triggers

                        ;###########################################
                        ;#      INITIALIZE XGATE VECTOR TABLE      #
                        ;###########################################
                        LDAA  #128            ;build XGATE vector table
                        LDY   #XGATE_VECTORS
INIT_XGATE_VECTAB_LOOP  MOVW  #XGATE_DUMMY_ISR_XG, 4,Y+
                        DBNE  A, INIT_XGATE_VECTAB_LOOP

                        MOVW  #XGATE_CODE_XG, RAM_START+(2*SCI_VEC)
                        MOVW  #XGATE_DATA_XG, RAM_START+(2*SCI_VEC)+2

                        ;###########################################
                        ;#              COPY XGATE CODE            #
                        ;###########################################
COPY_XGATE_CODE         LDX   #XGATE_DATA_FLASH
COPY_XGATE_CODE_LOOP    MOVW  2,X+, 2,Y+
```

```
                        MOVW  2,X+, 2,Y+
                        MOVW  2,X+, 2,Y+
                        MOVW  2,X+, 2,Y+
                        CPX   #XGATE_CODE_FLASH_END
                        BLS   COPY_XGATE_CODE_LOOP

                        ;###########################################
                        ;#                START XGATE              #
                        ;###########################################
START_XGATE             MOVW  #XGMCTL_ENABLE, XGMCTL          ;enable XGATE
                        BRA   *

                        ;###########################################
                        ;#     DUMMY INTERRUPT SERVICE ROUTINE     #
                        ;###########################################
DUMMY_ISR               RTI

                        CPU   XGATE
                        ;###########################################
                        ;#                 XGATE DATA              #
                        ;###########################################
                        ALIGN 1
XGATE_DATA_FLASH        EQU   *
XGATE_DATA_SCI          EQU   *-XGATE_DATA_FLASH
                        DW    SCI_REGS                        ;pointer to SCI register space
XGATE_DATA_IDX          EQU   *-XGATE_DATA_FLASH
                        DB    XGATE_DATA_MSG                  ;string pointer
XGATE_DATA_MSG          EQU   *-XGATE_DATA_FLASH
                        FCC   "Hello World!                   ;ASCII string
                        DB    $0D                             ;CR

                        ;###########################################
                        ;#                XGATE CODE               #
                        ;###########################################
                        ALIGN 1
XGATE_CODE_FLASH        LDW   R2,(R1,#XGATE_DATA_SCI)         ;SCI -> R2
                        LDB   R3,(R1,#XGATE_DATA_IDX)         ;msg -> R3
                        LDB   R4,(R1,R3+)                     ;curr. char -> R4
                        STB   R3,(R1,#XGATE_DATA_IDX)         ;R3 -> idx
                        LDB   R0,(R2,#(SCISR1-SCI_REGS))      ;initiate SCI transmit
                        STB   R4,(R2,#(SCIDRL-SCI_REGS))      ;initiate SCI transmit
                        CMPL  R4,#$0D
                        BEQ   XGATE_CODE_DONE
                        RTS
XGATE_CODE_DONE         LDL   R4,#$00                         ;disable SCI interrupts
                        STB   R4,(R2,#(SCICR2-SCI_REGS))
                        LDL   R3,#XGATE_DATA_MSG;reset R3
                        STB   R3,(R1,#XGATE_DATA_IDX)
XGATE_CODE_FLASH_END    RTS
XGATE_DUMMY_ISR_XG      EQU   (XGATE_CODE_FLASH_END-XGATE_CODE_FLASH)+XGATE_CODE_XG
```

14.9.3 Stack Support

To simplify the implementation of a program stack the XGATE can be configured to set RISC core register R7 to the beginning of a stack region before executing a thread. Two separate stack regions can be defined: One for threads of priority level 7 to 4 (refer to Section 14.3.1.5, “XGATE Initial Stack Pointer for

Interrupt Priorities 7 to 4 (XGISP74)") and one for threads of priority level 3 to 1 (refer to Section 14.3.1.6, "XGATE Initial Stack Pointer for Interrupt Priorities 3 to 1 (XGISP31)").

Chapter 15
Background Debug Module (S12XBDMV2)

Table 15-1. Revision History

Revision Number	Revision Date	Sections Affected	Description of Changes
V02.00	07 Mar 2006		- First version of S12XBDMV2
V02.01	14 May 2008		- Introduced standardized Revision History Table

15.1 Introduction

This section describes the functionality of the background debug module (BDM) sub-block of the HCS12X core platform.

The background debug module (BDM) sub-block is a single-wire, background debug system implemented in on-chip hardware for minimal CPU intervention. All interfacing with the BDM is done via the BKGD pin.

The BDM has enhanced capability for maintaining synchronization between the target and host while allowing more flexibility in clock rates. This includes a sync signal to determine the communication rate and a handshake signal to indicate when an operation is complete. The system is backwards compatible to the BDM of the S12 family with the following exceptions:

- TAGGO command no longer supported by BDM
- External instruction tagging feature now part of DBG module
- BDM register map and register content extended/modified
- Global page access functionality
- Enabled but not active out of reset in emulation modes (if modes available)
- CLKSW bit set out of reset in emulation modes (if modes available).
- Family ID readable from firmware ROM at global address 0x7FFF0F (value for HCS12X devices is 0xC1)

15.1.1 Features

The BDM includes these distinctive features:

- Single-wire communication with host development system
- Enhanced capability for allowing more flexibility in clock rates
- SYNC command to determine communication rate
- GO_UNTIL command
- Hardware handshake protocol to increase the performance of the serial communication

- Active out of reset in special single chip mode
- Nine hardware commands using free cycles, if available, for minimal CPU intervention
- Hardware commands not requiring active BDM
- 14 firmware commands execute from the standard BDM firmware lookup table
- Software control of BDM operation during wait mode
- Software selectable clocks
- Global page access functionality
- Enabled but not active out of reset in emulation modes (if modes available)
- CLKSW bit set out of reset in emulation modes (if modes available).
- When secured, hardware commands are allowed to access the register space in special single chip mode, if the non-volatile memory erase test fail.
- Family ID readable from firmware ROM at global address 0x7FFF0F (value for HCS12X devices is 0xC1)
- BDM hardware commands are operational until system stop mode is entered (all bus masters are in stop mode)

15.1.2 Modes of Operation

BDM is available in all operating modes but must be enabled before firmware commands are executed. Some systems may have a control bit that allows suspending thefunction during background debug mode.

15.1.2.1 Regular Run Modes

All of these operations refer to the part in run mode and not being secured. The BDM does not provide controls to conserve power during run mode.

- Normal modes

  General operation of the BDM is available and operates the same in all normal modes.
- Special single chip mode

  In special single chip mode, background operation is enabled and active out of reset. This allows programming a system with blank memory.
- Emulation modes (if modes available)

  In emulation mode, background operation is enabled but not active out of reset. This allows debugging and programming a system in this mode more easily.

15.1.2.2 Secure Mode Operation

If the device is in secure mode, the operation of the BDM is reduced to a small subset of its regular run mode operation. Secure operation prevents BDM and CPU accesses to non-volatile memory (Flash and/or EEPROM) other than allowing erasure. For more information please see Section 15.4.1, “Security”.

15.1.2.3 Low-Power Modes

The BDM can be used until all bus masters (e.g., CPU or XGATE or others depending on which masters are available on the SOC) are in stop mode. When CPU is in a low power mode (wait or stop mode) all BDM firmware commands as well as the hardware BACKGROUND command can not be used respectively are ignored. In this case the CPU can not enter BDM active mode, and only hardware read and write commands are available. Also the CPU can not enter a low power mode during BDM active mode.

If all bus masters are in stop mode, the BDM clocks are stopped as well. When BDM clocks are disabled and one of the bus masters exits from stop mode the BDM clocks will restart and BDM will have a soft reset (clearing the instruction register, any command in progress and disable the ACK function). The BDM is now ready to receive a new command.

15.1.3 Block Diagram

A block diagram of the BDM is shown in Figure 15-1.

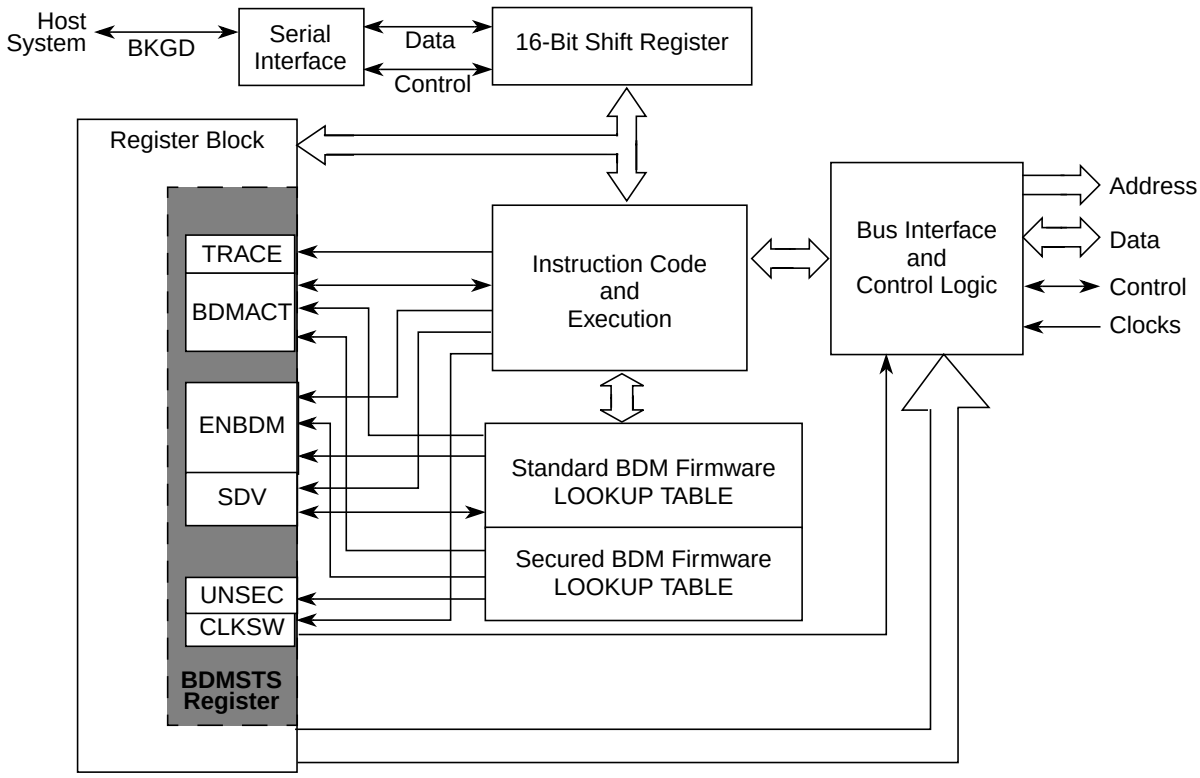


Figure 15-1. BDM Block Diagram

15.2 External Signal Description

A single-wire interface pin called the background debug interface (BKGD) pin is used to communicate with the BDM system. During reset, this pin is a mode select input which selects between normal and special modes of operation. After reset, this pin becomes the dedicated serial interface pin for the background debug mode.

15.3 Memory Map and Register Definition

15.3.1 Module Memory Map

Table 15-2 shows the BDM memory map when BDM is active.

Table 15-2. BDM Memory Map

Global Address	Module	Size (Bytes)
0x7FFF00–0x7FFF0B	BDM registers	12
0x7FFF0C–0x7FFF0E	BDM firmware ROM	3
0x7FFF0F	Family ID (part of BDM firmware ROM)	1
0x7FFF10–0x7FFFFF	BDM firmware ROM	240

15.3.2 Register Descriptions

A summary of the registers associated with the BDM is shown in Figure 15-2. Registers are accessed by host-driven communications to the BDM hardware using READ_BD and WRITE_BD commands.

Global Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x7FFF00	Reserved	R	X	X	X	X	X	X	0	0
		W								
0x7FFF01	BDMSTS	R	ENBDM	BDMACT	0	SDV	TRACE	CLKSW	UNSEC	0
		W								
0x7FFF02	Reserved	R	X	X	X	X	X	X	X	X
		W								
0x7FFF03	Reserved	R	X	X	X	X	X	X	X	X
		W								
0x7FFF04	Reserved	R	X	X	X	X	X	X	X	X
		W								
0x7FFF05	Reserved	R	X	X	X	X	X	X	X	X
		W								
0x7FFF06	BDMCCRL	R	CCR7	CCR6	CCR5	CCR4	CCR3	CCR2	CCR1	CCR0
		W								

= Unimplemented, Reserved

= Implemented (do not alter)

X = Indeterminate

0 = Always read zero

Figure 15-2. BDM Register Summary

Global Address	Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x7FFF07	BDMCCRH	R	0	0	0	0	0	CCR10	CCR9	CCR8
		W								
0x7FFF08	BDMGPR	R	BGAE	BGP6	BGP5	BGP4	BGP3	BGP2	BGP1	BGP0
		W								
0x7FFF09	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x7FFF0A	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x7FFF0B	Reserved	R	0	0	0	0	0	0	0	0
		W								

= Unimplemented, Reserved; = Implemented (do not alter)

X = Indeterminate; 0 = Always read zero

Figure 15-2. BDM Register Summary (continued)

15.3.2.1 BDM Status Register (BDMSTS)

Register Global Address 0x7FFF01

	7	6	5	4	3	2	1	0
R	ENBDM	BDMACT	0	SDV	TRACE	CLKSW	UNSEC	0
W								
Reset								
Special Single-Chip Mode	0(1)	1	0	0	0	0	0(3)	0
Emulation Modes (if modes available)	1	0	0	0	0	1(2)	0	0
All Other Modes	0	0	0	0	0	0	0	0

= Unimplemented, Reserved; = Implemented (do not alter)

0 = Always read zero

1. ENBDM is read as 1 by a debugging environment in special single chip mode when the device is not secured or secured but fully erased (non-volatile memory). This is because the ENBDM bit is set by the standard firmware before a BDM command can be fully transmitted and executed.
2. CLKSW is read as 1 by a debugging environment in emulation modes when the device is not secured and read as 0 when secured if emulation modes available.
3. UNSEC is read as 1 by a debugging environment in special single chip mode when the device is secured and fully erased, else it is 0 and can only be read if not secure (see also bit description).

Figure 15-3. BDM Status Register (BDMSTS)

Read: All modes through BDM operation when not secured

Write: All modes through BDM operation when not secured, but subject to the following:

- ENBDM should only be set via a BDM hardware command if the BDM firmware commands are needed. (This does not apply in special single chip and emulation modes).
- BDMACT can only be set by BDM hardware upon entry into BDM. It can only be cleared by the standard BDM firmware lookup table upon exit from BDM active mode.
- CLKSW can only be written via BDM hardware WRITE_BD commands.
- All other bits, while writable via BDM hardware or standard BDM firmware write commands, should only be altered by the BDM hardware or standard firmware lookup table as part of BDM command execution.

Table 15-3. BDMSTS Field Descriptions

Field	Description
7 ENBDM	**Enable BDM** — This bit controls whether the BDM is enabled or disabled. When enabled, BDM can be made active to allow firmware commands to be executed. When disabled, BDM cannot be made active but BDM hardware commands are still allowed. 0 BDM disabled 1 BDM enabled **Note:** ENBDM is set by the firmware out of reset in special single chip mode. In emulation modes (if modes available) the ENBDM bit is set by BDM hardware out of reset. In special single chip mode with the device secured, this bit will not be set by the firmware until after the non-volatile memory erase verify tests are complete. In emulation modes (if modes available) with the device secured, the BDM operations are blocked.
6 BDMACT	**BDM Active Status** — This bit becomes set upon entering BDM. The standard BDM firmware lookup table is then enabled and put into the memory map. BDMACT is cleared by a carefully timed store instruction in the standard BDM firmware as part of the exit sequence to return to user code and remove the BDM memory from the map. 0 BDM not active 1 BDM active
4 SDV	**Shift Data Valid** — This bit is set and cleared by the BDM hardware. It is set after data has been transmitted as part of a firmware or hardware read command or after data has been received as part of a firmware or hardware write command. It is cleared when the next BDM command has been received or BDM is exited. SDV is used by the standard BDM firmware to control program flow execution. 0 Data phase of command not complete 1 Data phase of command is complete
3 TRACE	**TRACE1 BDM Firmware Command is Being Executed** — This bit gets set when a BDM TRACE1 firmware command is first recognized. It will stay set until BDM firmware is exited by one of the following BDM commands: GO or GO_UNTIL. 0 TRACE1 command is not being executed 1 TRACE1 command is being executed

Table 15-3. BDMSTS Field Descriptions (continued)

Field	Description
2 CLKSW	**Clock Switch** — The CLKSW bit controls which clock the BDM operates with. It is only writable from a hardware BDM command. A minimum delay of 150 cycles at the clock speed that is active during the data portion of the command send to change the clock source should occur before the next command can be send. The delay should be obtained no matter which bit is modified to effectively change the clock source (either PLLSEL bit or CLKSW bit). This guarantees that the start of the next BDM command uses the new clock for timing subsequent BDM communications. Table 15-4 shows the resulting BDM clock source based on the CLKSW and the PLLSEL (PLL select in the CRG module, the bit is part of the CLKSEL register) bits. **Note:** The BDM alternate clock source can only be selected when CLKSW = 0 and PLLSEL = 1. The BDM serial interface is now fully synchronized to the alternate clock source, when enabled. This eliminates frequency restriction on the alternate clock which was required on previous versions. Refer to the device specification to determine which clock connects to the alternate clock source input. **Note:** If the acknowledge function is turned on, changing the CLKSW bit will cause the ACK to be at the new rate for the write command which changes it. **Note:** In emulation modes (if modes available), the CLKSW bit will be set out of RESET.
1 UNSEC	**Unsecure** — If the device is secured this bit is only writable in special single chip mode from the BDM secure firmware. It is in a zero state as secure mode is entered so that the secure BDM firmware lookup table is enabled and put into the memory map overlapping the standard BDM firmware lookup table. The secure BDM firmware lookup table verifies that the non-volatile memories (e.g. on-chip EEPROM and/or Flash EEPROM) are erased. This being the case, the UNSEC bit is set and the BDM program jumps to the start of the standard BDM firmware lookup table and the secure BDM firmware lookup table is turned off. If the erase test fails, the UNSEC bit will not be asserted. 0 System is in a secured mode. 1 System is in a unsecured mode. **Note:** When UNSEC is set, security is off and the user can change the state of the secure bits in the on-chip Flash EEPROM. Note that if the user does not change the state of the bits to “unsecured” mode, the system will be secured again when it is next taken out of reset.After reset this bit has no meaning or effect when the security byte in the Flash EEPROM is configured for unsecure mode.

Table 15-4. BDM Clock Sources

PLLSEL	CLKSW	BDMCLK
0	0	Bus clock dependent on oscillator
0	1	Bus clock dependent on oscillator
1	0	Alternate clock (refer to the device specification to determine the alternate clock source)
1	1	Bus clock dependent on the PLL

15.3.2.2 BDM CCR LOW Holding Register (BDMCCRL)

Register Global Address 0x7FFF06

	7	6	5	4	3	2	1	0
R W	CCR7	CCR6	CCR5	CCR4	CCR3	CCR2	CCR1	CCR0
Reset								
Special Single-Chip Mode	1	1	0	0	1	0	0	0
All Other Modes	0	0	0	0	0	0	0	0

Figure 15-4. BDM CCR LOW Holding Register (BDMCCRL)

Read: All modes through BDM operation when not secured

Write: All modes through BDM operation when not secured

NOTE

When BDM is made active, the CPU stores the content of its CCR_L register in the BDMCCRL register. However, out of special single-chip reset, the BDMCCRL is set to 0xD8 and not 0xD0 which is the reset value of the CCR_L register in this CPU mode. Out of reset in all other modes the BDMCCRL register is read zero.

When entering background debug mode, the BDM CCR LOW holding register is used to save the low byte of the condition code register of the user's program. It is also used for temporary storage in the standard BDM firmware mode. The BDM CCR LOW holding register can be written to modify the CCR value.

15.3.2.3 BDM CCR HIGH Holding Register (BDMCCRH)

Register Global Address 0x7FFF07

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	CCR10	CCR9	CCR8
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 15-5. BDM CCR HIGH Holding Register (BDMCCRH)

Read: All modes through BDM operation when not secured

Write: All modes through BDM operation when not secured

When entering background debug mode, the BDM CCR HIGH holding register is used to save the high byte of the condition code register of the user's program. The BDM CCR HIGH holding register can be written to modify the CCR value.

15.3.2.4 BDM Global Page Index Register (BDMGPR)

Register Global Address 0x7FFF08

	7	6	5	4	3	2	1	0
R W	BGAE	BGP6	BGP5	BGP4	BGP3	BGP2	BGP1	BGP0
Reset	0	0	0	0	0	0	0	0

Figure 15-6. BDM Global Page Register (BDMGPR)

Read: All modes through BDM operation when not secured

Write: All modes through BDM operation when not secured

Table 15-5. BDMGPR Field Descriptions

Field	Description
7 BGAE	**BDM Global Page Access Enable Bit** — BGAE enables global page access for BDM hardware and firmware read/write instructions The BDM hardware commands used to access the BDM registers (READ_BD_ and WRITE_BD_) can not be used for global accesses even if the BGAE bit is set. 0 BDM Global Access disabled 1 BDM Global Access enabled
6–0 BGP[6:0]	**BDM Global Page Index Bits 6–0** — These bits define the extended address bits from 22 to 16. For more detailed information regarding the global page window scheme, please refer to the S12X_MMC Block Guide.

15.3.3 Family ID Assignment

The family ID is a 8-bit value located in the firmware ROM (at global address: 0x7FFF0F). The read-only value is a unique family ID which is 0xC1 for S12X devices.

15.4 Functional Description

The BDM receives and executes commands from a host via a single wire serial interface. There are two types of BDM commands: hardware and firmware commands.

Hardware commands are used to read and write target system memory locations and to enter active background debug mode, see Section 15.4.3, “BDM Hardware Commands”. Target system memory includes all memory that is accessible by the CPU.

Firmware commands are used to read and write CPU resources and to exit from active background debug mode, see Section 15.4.4, “Standard BDM Firmware Commands”. The CPU resources referred to are the accumulator (D), X index register (X), Y index register (Y), stack pointer (SP), and program counter (PC).

Hardware commands can be executed at any time and in any mode excluding a few exceptions as highlighted (see Section 15.4.3, “BDM Hardware Commands”) and in secure mode (see Section 15.4.1, “Security”). Firmware commands can only be executed when the system is not secure and is in active background debug mode (BDM).

15.4.1 Security

If the user resets into special single chip mode with the system secured, a secured mode BDM firmware lookup table is brought into the map overlapping a portion of the standard BDM firmware lookup table. The secure BDM firmware verifies that the on-chip non-volatile memory (e.g. EEPROM and Flash EEPROM) is erased. This being the case, the UNSEC and ENBDM bit will get set. The BDM program jumps to the start of the standard BDM firmware and the secured mode BDM firmware is turned off and all BDM commands are allowed. If the non-volatile memory does not verify as erased, the BDM firmware sets the ENBDM bit, without asserting UNSEC, and the firmware enters a loop. This causes the BDM hardware commands to become enabled, but does not enable the firmware commands. This allows the BDM hardware to be used to erase the non-volatile memory.

BDM operation is not possible in any other mode than special single chip mode when the device is secured. The device can be unsecured via BDM serial interface in special single chip mode only. For more information regarding security, please see the S12X_9SEC Block Guide.

15.4.2 Enabling and Activating BDM

The system must be in active BDM to execute standard BDM firmware commands. BDM can be activated only after being enabled. BDM is enabled by setting the ENBDM bit in the BDM status (BDMSTS) register. The ENBDM bit is set by writing to the BDM status (BDMSTS) register, via the single-wire interface, using a hardware command such as WRITE_BD_BYTE.

After being enabled, BDM is activated by one of the following[1]:

- Hardware BACKGROUND command
- CPU BGND instruction
- External instruction tagging mechanism[2]
- Breakpoint force or tag mechanism[2]

When BDM is activated, the CPU finishes executing the current instruction and then begins executing the firmware in the standard BDM firmware lookup table. When BDM is activated by a breakpoint, the type of breakpoint used determines if BDM becomes active before or after execution of the next instruction.

NOTE

If an attempt is made to activate BDM before being enabled, the CPU resumes normal instruction execution after a brief delay. If BDM is not enabled, any hardware BACKGROUND commands issued are ignored by the BDM and the CPU is not delayed.

In active BDM, the BDM registers and standard BDM firmware lookup table are mapped to addresses 0x7FFF00 to 0x7FFFFF. BDM registers are mapped to addresses 0x7FFF00 to 0x7FFF0B. The BDM uses these registers which are readable anytime by the BDM. However, these registers are not readable by user programs.

1. BDM is enabled and active immediately out of special single-chip reset.
2. This method is provided by the S12X_DBG module.

15.4.3 BDM Hardware Commands

Hardware commands are used to read and write target system memory locations and to enter active background debug mode. Target system memory includes all memory that is accessible by the CPU on the SOC which can be on-chip RAM, non-volatile memory (e.g. EEPROM, Flash EEPROM), I/O and control registers, and all external memory.

Hardware commands are executed with minimal or no CPU intervention and do not require the system to be in active BDM for execution, although, they can still be executed in this mode. When executing a hardware command, the BDM sub-block waits for a free bus cycle so that the background access does not disturb the running application program. If a free cycle is not found within 128 clock cycles, the CPU is momentarily frozen so that the BDM can steal a cycle. When the BDM finds a free cycle, the operation does not intrude on normal CPU operation provided that it can be completed in a single cycle. However, if an operation requires multiple cycles the CPU is frozen until the operation is complete, even though the BDM found a free cycle.

The BDM hardware commands are listed in Table 15-6.

The READ_BD and WRITE_BD commands allow access to the BDM register locations. These locations are not normally in the system memory map but share addresses with the application in memory. To distinguish between physical memory locations that share the same address, BDM memory resources are enabled just for the READ_BD and WRITE_BD access cycle. This allows the BDM to access BDM locations unobtrusively, even if the addresses conflict with the application memory map.

Table 15-6. Hardware Commands

Command	Opcode (hex)	Data	Description
BACKGROUND	90	None	Enter background mode if firmware is enabled. If enabled, an ACK will be issued when the part enters active background mode.
ACK_ENABLE	D5	None	Enable Handshake. Issues an ACK pulse after the command is executed.
ACK_DISABLE	D6	None	Disable Handshake. This command does not issue an ACK pulse.
READ_BD_BYTE	E4	16-bit address 16-bit data out	Read from memory with standard BDM firmware lookup table in map. Odd address data on low byte; even address data on high byte.
READ_BD_WORD	EC	16-bit address 16-bit data out	Read from memory with standard BDM firmware lookup table in map. Must be aligned access.
READ_BYTE	E0	16-bit address 16-bit data out	Read from memory with standard BDM firmware lookup table out of map. Odd address data on low byte; even address data on high byte.
READ_WORD	E8	16-bit address 16-bit data out	Read from memory with standard BDM firmware lookup table out of map. Must be aligned access.
WRITE_BD_BYTE	C4	16-bit address 16-bit data in	Write to memory with standard BDM firmware lookup table in map. Odd address data on low byte; even address data on high byte.
WRITE_BD_WORD	CC	16-bit address 16-bit data in	Write to memory with standard BDM firmware lookup table in map. Must be aligned access.
WRITE_BYTE	C0	16-bit address 16-bit data in	Write to memory with standard BDM firmware lookup table out of map. Odd address data on low byte; even address data on high byte.

Table 15-6. Hardware Commands (continued)

Command	Opcode (hex)	Data	Description
WRITE_WORD	C8	16-bit address 16-bit data in	Write to memory with standard BDM firmware lookup table out of map. Must be aligned access.

NOTE:
If enabled, ACK will occur when data is ready for transmission for all BDM READ commands and will occur after the write is complete for all BDM WRITE commands.

15.4.4 Standard BDM Firmware Commands

Firmware commands are used to access and manipulate CPU resources. The system must be in active BDM to execute standard BDM firmware commands, see Section 15.4.2, “Enabling and Activating BDM”. Normal instruction execution is suspended while the CPU executes the firmware located in the standard BDM firmware lookup table. The hardware command BACKGROUND is the usual way to activate BDM.

As the system enters active BDM, the standard BDM firmware lookup table and BDM registers become visible in the on-chip memory map at 0x7FFF00–0x7FFFFF, and the CPU begins executing the standard BDM firmware. The standard BDM firmware watches for serial commands and executes them as they are received.

The firmware commands are shown in Table 15-7.

Table 15-7. Firmware Commands

Command[1]	Opcode (hex)	Data	Description
READ_NEXT[2]	62	16-bit data out	Increment X index register by 2 (X = X + 2), then read word X points to.
READ_PC	63	16-bit data out	Read program counter.
READ_D	64	16-bit data out	Read D accumulator.
READ_X	65	16-bit data out	Read X index register.
READ_Y	66	16-bit data out	Read Y index register.
READ_SP	67	16-bit data out	Read stack pointer.
WRITE_NEXT	42	16-bit data in	Increment X index register by 2 (X = X + 2), then write word to location pointed to by X.
WRITE_PC	43	16-bit data in	Write program counter.
WRITE_D	44	16-bit data in	Write D accumulator.
WRITE_X	45	16-bit data in	Write X index register.
WRITE_Y	46	16-bit data in	Write Y index register.
WRITE_SP	47	16-bit data in	Write stack pointer.
GO	08	none	Go to user program. If enabled, ACK will occur when leaving active background mode.
GO_UNTIL[3]	0C	none	Go to user program. If enabled, ACK will occur upon returning to active background mode.
TRACE1	10	none	Execute one user instruction then return to active BDM. If enabled, ACK will occur upon returning to active background mode.
TAGGO -> GO	18	none	(Previous enable tagging and go to user program.) This command will be deprecated and should not be used anymore. Opcode will be executed as a GO command.

1. If enabled, ACK will occur when data is ready for transmission for all BDM READ commands and will occur after the write is complete for all BDM WRITE commands.
2. When the firmware command READ_NEXT or WRITE_NEXT is used to access the BDM address space the BDM resources are accessed rather than user code. Writing BDM firmware is not possible.
3. System stop disables the ACK function and ignored commands will not have an ACK-pulse (e.g., CPU in stop or wait mode). The GO_UNTIL command will not get an Acknowledge if CPU executes the wait or stop instruction before the “UNTIL” condition (BDM active again) is reached (see Section 15.4.7, “Serial Interface Hardware Handshake Protocol” last Note).

15.4.5 BDM Command Structure

Hardware and firmware BDM commands start with an 8-bit opcode followed by a 16-bit address and/or a 16-bit data word depending on the command. All the read commands return 16 bits of data despite the byte or word implication in the command name.

8-bit reads return 16-bits of data, of which, only one byte will contain valid data. If reading an even address, the valid data will appear in the MSB. If reading an odd address, the valid data will appear in the LSB.

16-bit misaligned reads and writes are generally not allowed. If attempted by BDM hardware command, the BDM will ignore the least significant bit of the address and will assume an even address from the remaining bits.

For devices with external bus:

The following cycle count information is only valid when the external wait function is not used (see wait bit of EBI sub-block). During an external wait the BDM can not steal a cycle. Hence be careful with the external wait function if the BDM serial interface is much faster than the bus, because of the BDM soft-reset after time-out (see Section 15.4.11, “Serial Communication Time Out”).

For hardware data read commands, the external host must wait at least 150 bus clock cycles after sending the address before attempting to obtain the read data. This is to be certain that valid data is available in the BDM shift register, ready to be shifted out. For hardware write commands, the external host must wait 150 bus clock cycles after sending the data to be written before attempting to send a new command. This is to avoid disturbing the BDM shift register before the write has been completed. The 150 bus clock cycle delay in both cases includes the maximum 128 cycle delay that can be incurred as the BDM waits for a free cycle before stealing a cycle.

For firmware read commands, the external host should wait at least 48 bus clock cycles after sending the command opcode and before attempting to obtain the read data. This includes the potential of extra cycles when the access is external and stretched (+1 to maximum +7 cycles) or to registers of the PRU (port replacement unit) in emulation modes (if modes available). The 48 cycle wait allows enough time for the requested data to be made available in the BDM shift register, ready to be shifted out.

NOTE

This timing has increased from previous BDM modules due to the new capability in which the BDM serial interface can potentially run faster than the bus. On previous BDM modules this extra time could be hidden within the serial time.

For firmware write commands, the external host must wait 36 bus clock cycles after sending the data to be written before attempting to send a new command. This is to avoid disturbing the BDM shift register before the write has been completed.

The external host should wait at least for 76 bus clock cycles after a TRACE1 or GO command before starting any new serial command. This is to allow the CPU to exit gracefully from the standard BDM firmware lookup table and resume execution of the user code. Disturbing the BDM shift register prematurely may adversely affect the exit from the standard BDM firmware lookup table.

NOTE

If the bus rate of the target processor is unknown or could be changing or the external wait function is used, it is recommended that the ACK (acknowledge function) is used to indicate when an operation is complete. When using ACK, the delay times are automated.

Figure 15-7 represents the BDM command structure. The command blocks illustrate a series of eight bit times starting with a falling edge. The bar across the top of the blocks indicates that the BKGD line idles in the high state. The time for an 8-bit command is 8×16 target clock cycles.[1]

1. Target clock cycles are cycles measured using the target MCU’s serial clock rate. See Section 15.4.6, “BDM Serial Interface” and Section 15.3.2.1, “BDM Status Register (BDMSTS)” for information on how serial clock rate is selected.

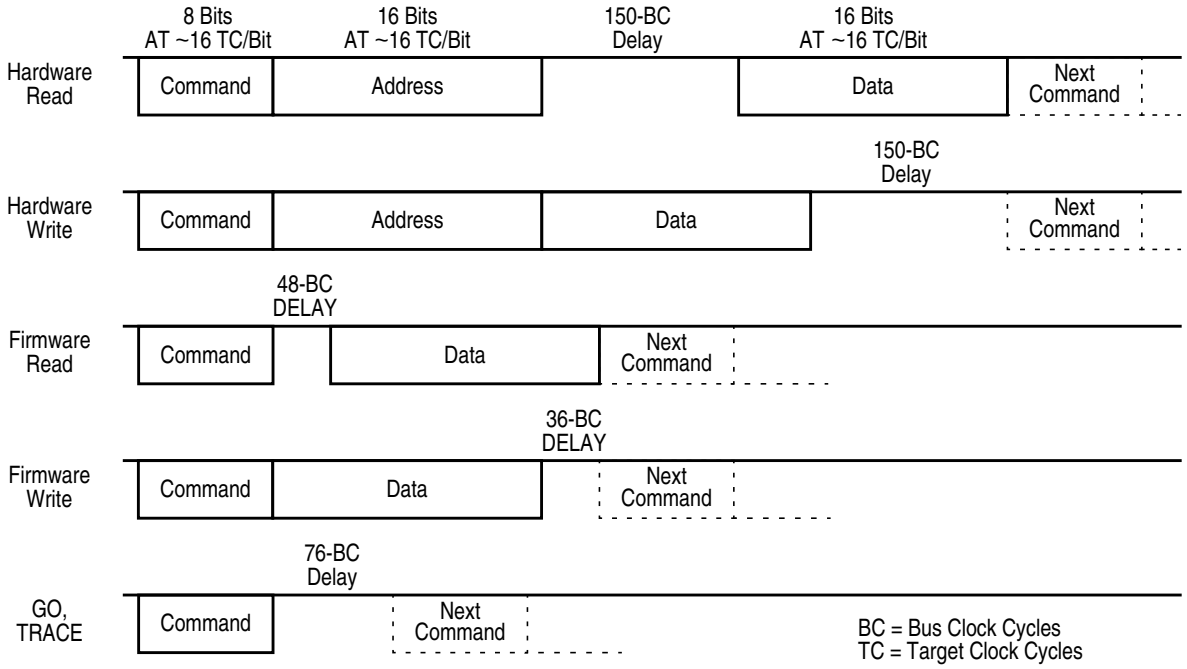


Figure 15-7. BDM Command Structure

15.4.6 BDM Serial Interface

The BDM communicates with external devices serially via the BKGD pin. During reset, this pin is a mode select input which selects between normal and special modes of operation. After reset, this pin becomes the dedicated serial interface pin for the BDM.

The BDM serial interface is timed using the clock selected by the CLKSW bit in the status register see Section 15.3.2.1, “BDM Status Register (BDMSTS)”. This clock will be referred to as the target clock in the following explanation.

The BDM serial interface uses a clocking scheme in which the external host generates a falling edge on the BKGD pin to indicate the start of each bit time. This falling edge is sent for every bit whether data is transmitted or received. Data is transferred most significant bit (MSB) first at 16 target clock cycles per bit. The interface times out if 512 clock cycles occur between falling edges from the host.

The BKGD pin is a pseudo open-drain pin and has an weak on-chip active pull-up that is enabled at all times. It is assumed that there is an external pull-up and that drivers connected to BKGD do not typically drive the high level. Since R-C rise time could be unacceptably long, the target system and host provide brief driven-high (speedup) pulses to drive BKGD to a logic 1. The source of this speedup pulse is the host for transmit cases and the target for receive cases.

The timing for host-to-target is shown in Figure 15-8 and that of target-to-host in Figure 15-9 and Figure 15-10. All four cases begin when the host drives the BKGD pin low to generate a falling edge. Since the host and target are operating from separate clocks, it can take the target system up to one full clock cycle to recognize this edge. The target measures delays from this perceived start of the bit time while the host measures delays from the point it actually drove BKGD low to start the bit up to one target clock cycle

earlier. Synchronization between the host and target is established in this manner at the start of every bit time.

Figure 15-8 shows an external host transmitting a logic 1 and transmitting a logic 0 to the BKGD pin of a target system. The host is asynchronous to the target, so there is up to a one clock-cycle delay from the host-generated falling edge to where the target recognizes this edge as the beginning of the bit time. Ten target clock cycles later, the target senses the bit level on the BKGD pin. Internal glitch detect logic requires the pin be driven high no later that eight target clock cycles after the falling edge for a logic 1 transmission.

Since the host drives the high speedup pulses in these two cases, the rising edges look like digitally driven signals.

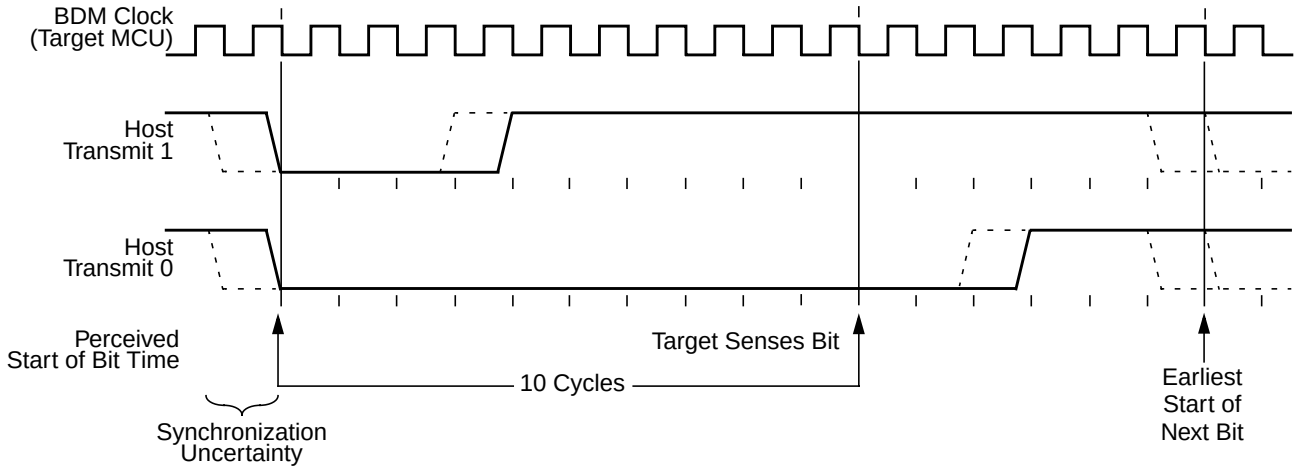


Figure 15-8. BDM Host-to-Target Serial Bit Timing

The receive cases are more complicated. Figure 15-9 shows the host receiving a logic 1 from the target system. Since the host is asynchronous to the target, there is up to one clock-cycle delay from the host-generated falling edge on BKGD to the perceived start of the bit time in the target. The host holds the BKGD pin low long enough for the target to recognize it (at least two target clock cycles). The host must release the low drive before the target drives a brief high speedup pulse seven target clock cycles after the perceived start of the bit time. The host should sample the bit level about 10 target clock cycles after it started the bit time.

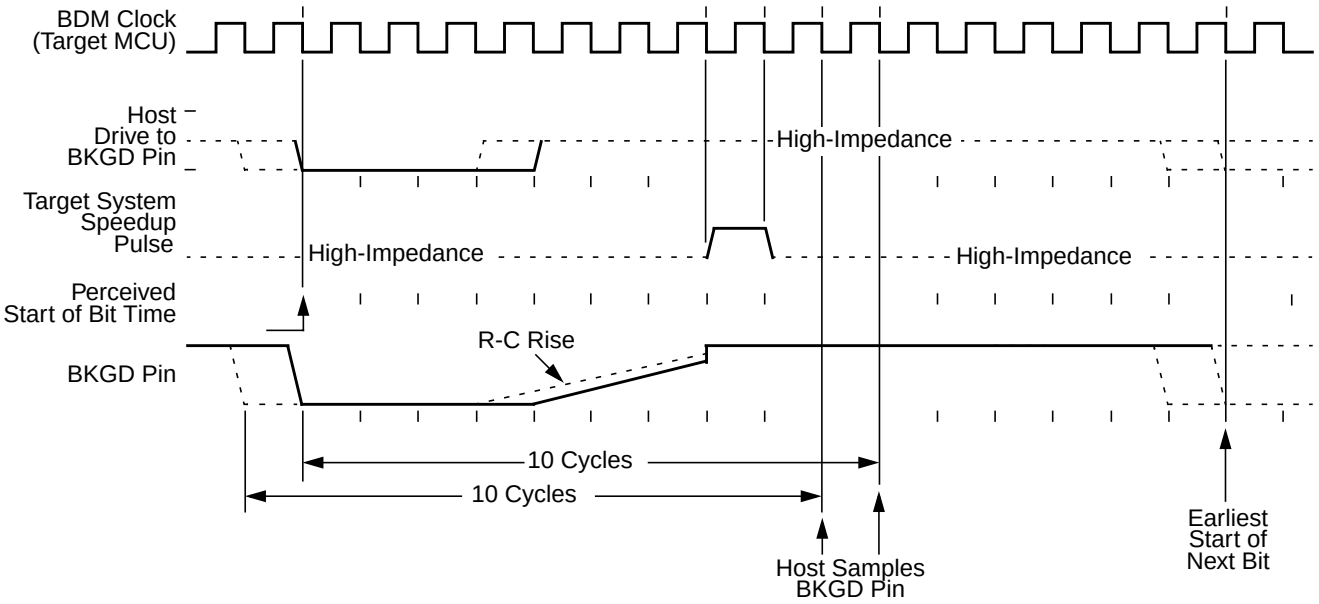


Figure 15-9. BDM Target-to-Host Serial Bit Timing (Logic 1)

Figure 15-10 shows the host receiving a logic 0 from the target. Since the host is asynchronous to the target, there is up to a one clock-cycle delay from the host-generated falling edge on BKGD to the start of the bit time as perceived by the target. The host initiates the bit time but the target finishes it. Since the target wants the host to receive a logic 0, it drives the BKGD pin low for 13 target clock cycles then briefly drives it high to speed up the rising edge. The host samples the bit level about 10 target clock cycles after starting the bit time.

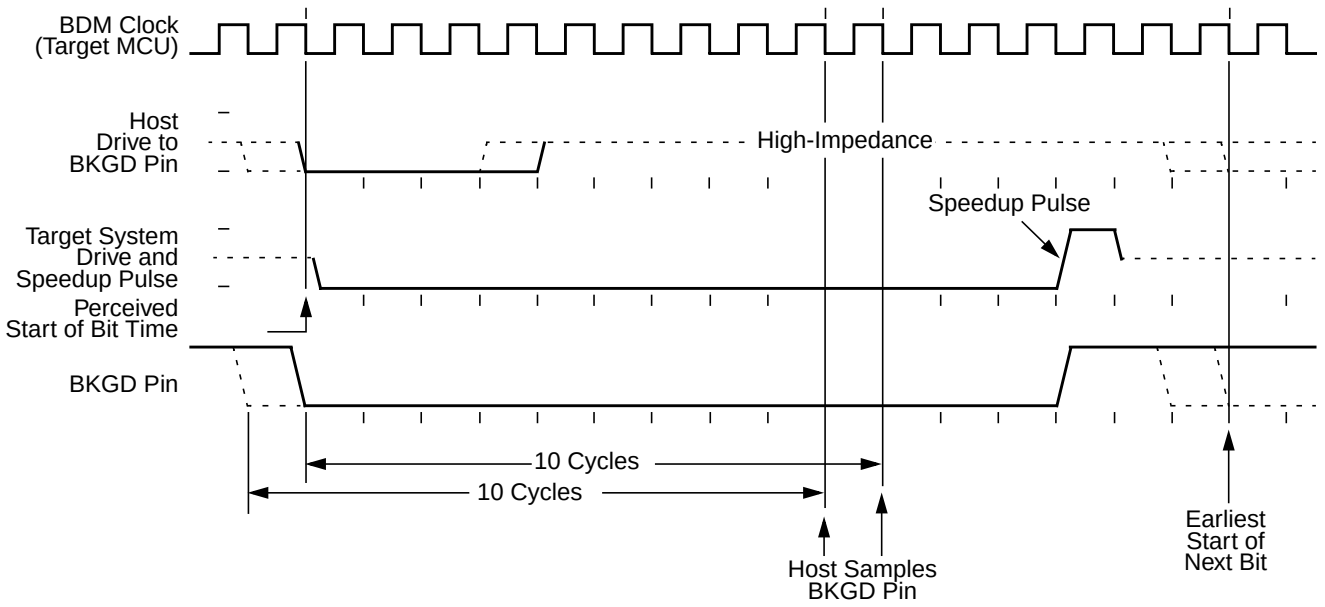


Figure 15-10. BDM Target-to-Host Serial Bit Timing (Logic 0)

15.4.7 Serial Interface Hardware Handshake Protocol

BDM commands that require CPU execution are ultimately treated at the MCU bus rate. Since the BDM clock source can be asynchronously related to the bus frequency, when CLKSW = 0, it is very helpful to provide a handshake protocol in which the host could determine when an issued command is executed by the CPU. The alternative is to always wait the amount of time equal to the appropriate number of cycles at the slowest possible rate the clock could be running. This sub-section will describe the hardware handshake protocol.

The hardware handshake protocol signals to the host controller when an issued command was successfully executed by the target. This protocol is implemented by a 16 serial clock cycle low pulse followed by a brief speedup pulse in the BKGD pin. This pulse is generated by the target MCU when a command, issued by the host, has been successfully executed (see Figure 15-11). This pulse is referred to as the ACK pulse. After the ACK pulse has finished: the host can start the bit retrieval if the last issued command was a read command, or start a new command if the last command was a write command or a control command (BACKGROUND, GO, GO_UNTIL or TRACE1). The ACK pulse is not issued earlier than 32 serial clock cycles after the BDM command was issued. The end of the BDM command is assumed to be the 16th tick of the last bit. This minimum delay assures enough time for the host to perceive the ACK pulse. Note also that, there is no upper limit for the delay between the command and the related ACK pulse, since the command execution depends upon the CPU bus frequency, which in some cases could be very slow

compared to the serial communication rate. This protocol allows a great flexibility for the POD designers, since it does not rely on any accurate time measurement or short response time to any event in the serial communication.

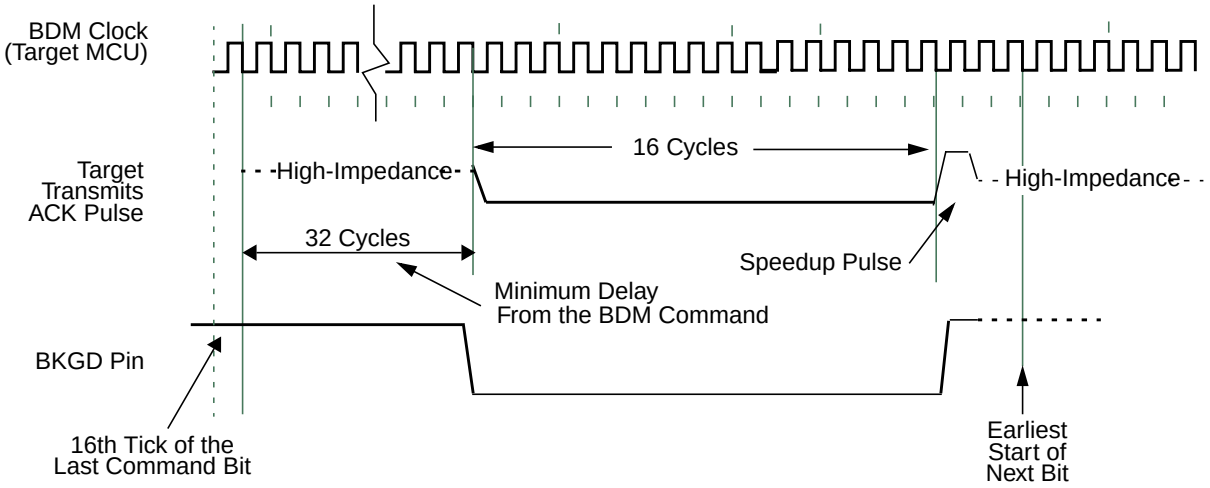


Figure 15-11. Target Acknowledge Pulse (ACK)

NOTE

If the ACK pulse was issued by the target, the host assumes the previous command was executed. If the CPU enters wait or stop prior to executing a hardware command, the ACK pulse will not be issued meaning that the BDM command was not executed. After entering wait or stop mode, the BDM command is no longer pending.

Figure 15-12 shows the ACK handshake protocol in a command level timing diagram. The READ_BYTE instruction is used as an example. First, the 8-bit instruction opcode is sent by the host, followed by the address of the memory location to be read. The target BDM decodes the instruction. A bus cycle is grabbed (free or stolen) by the BDM and it executes the READ_BYTE operation. Having retrieved the data, the BDM issues an ACK pulse to the host controller, indicating that the addressed byte is ready to be retrieved. After detecting the ACK pulse, the host initiates the byte retrieval process. Note that data is sent in the form of a word and the host needs to determine which is the appropriate byte based on whether the address was odd or even.

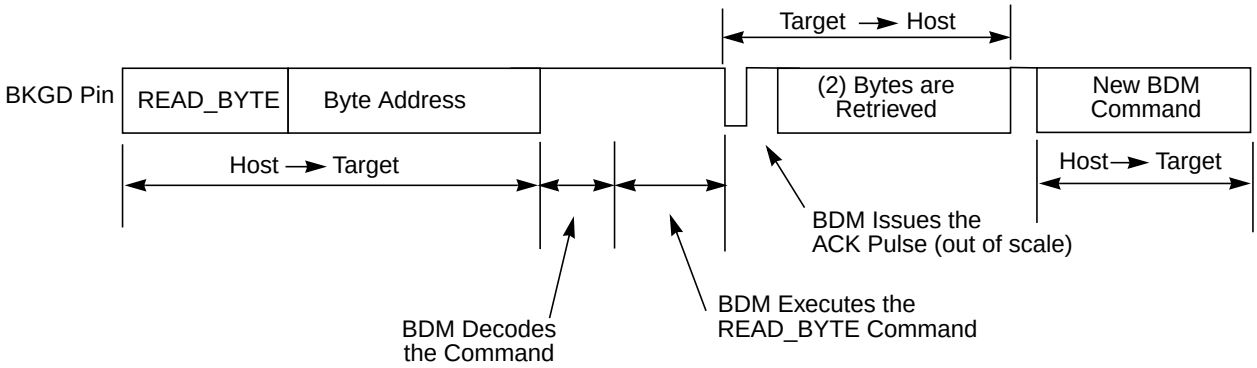


Figure 15-12. Handshake Protocol at Command Level

Differently from the normal bit transfer (where the host initiates the transmission), the serial interface ACK handshake pulse is initiated by the target MCU by issuing a negative edge in the BKGD pin. The hardware handshake protocol in Figure 15-11 specifies the timing when the BKGD pin is being driven, so the host should follow this timing constraint in order to avoid the risk of an electrical conflict in the BKGD pin.

NOTE

The only place the BKGD pin can have an electrical conflict is when one side is driving low and the other side is issuing a speedup pulse (high). Other "highs" are pulled rather than driven. However, at low rates the time of the speedup pulse can become lengthy and so the potential conflict time becomes longer as well.

The ACK handshake protocol does not support nested ACK pulses. If a BDM command is not acknowledge by an ACK pulse, the host needs to abort the pending command first in order to be able to issue a new BDM command. When the CPU enters wait or stop while the host issues a hardware command (e.g., WRITE_BYTE), the target discards the incoming command due to the wait or stop being detected. Therefore, the command is not acknowledged by the target, which means that the ACK pulse will not be issued in this case. After a certain time the host (not aware of stop or wait) should decide to abort any possible pending ACK pulse in order to be sure a new command can be issued. Therefore, the protocol provides a mechanism in which a command, and its corresponding ACK, can be aborted.

NOTE

The ACK pulse does not provide a time out. This means for the GO_UNTIL command that it can not be distinguished if a stop or wait has been executed (command discarded and ACK not issued) or if the "UNTIL" condition (BDM active) is just not reached yet. Hence in any case where the ACK pulse of a command is not issued the possible pending command should be aborted before issuing a new command. See the handshake abort procedure described in Section 15.4.8, "Hardware Handshake Abort Procedure".

15.4.8 Hardware Handshake Abort Procedure

The abort procedure is based on the SYNC command. In order to abort a command, which had not issued the corresponding ACK pulse, the host controller should generate a low pulse in the BKGD pin by driving it low for at least 128 serial clock cycles and then driving it high for one serial clock cycle, providing a speedup pulse. By detecting this long low pulse in the BKGD pin, the target executes the SYNC protocol, see Section 15.4.9, "SYNC — Request Timed Reference Pulse", and assumes that the pending command and therefore the related ACK pulse, are being aborted. Therefore, after the SYNC protocol has been completed the host is free to issue new BDM commands. For Firmware READ or WRITE commands it can not be guaranteed that the pending command is aborted when issuing a SYNC before the corresponding ACK pulse. There is a short latency time from the time the READ or WRITE access begins until it is finished and the corresponding ACK pulse is issued. The latency time depends on the firmware READ or WRITE command that is issued and if the serial interface is running on a different clock rate than the bus. When the SYNC command starts during this latency time the READ or WRITE command will not be aborted, but the corresponding ACK pulse will be aborted. A pending GO, TRACE1 or

GO_UNTIL command can not be aborted. Only the corresponding ACK pulse can be aborted by the SYNC command.

Although it is not recommended, the host could abort a pending BDM command by issuing a low pulse in the BKGD pin shorter than 128 serial clock cycles, which will not be interpreted as the SYNC command. The ACK is actually aborted when a negative edge is perceived by the target in the BKGD pin. The short abort pulse should have at least 4 clock cycles keeping the BKGD pin low, in order to allow the negative edge to be detected by the target. In this case, the target will not execute the SYNC protocol but the pending command will be aborted along with the ACK pulse. The potential problem with this abort procedure is when there is a conflict between the ACK pulse and the short abort pulse. In this case, the target may not perceive the abort pulse. The worst case is when the pending command is a read command (i.e., READ_BYTE). If the abort pulse is not perceived by the target the host will attempt to send a new command after the abort pulse was issued, while the target expects the host to retrieve the accessed memory byte. In this case, host and target will run out of synchronism. However, if the command to be aborted is not a read command the short abort pulse could be used. After a command is aborted the target assumes the next negative edge, after the abort pulse, is the first bit of a new BDM command.

NOTE

The details about the short abort pulse are being provided only as a reference for the reader to better understand the BDM internal behavior. It is not recommended that this procedure be used in a real application.

Since the host knows the target serial clock frequency, the SYNC command (used to abort a command) does not need to consider the lower possible target frequency. In this case, the host could issue a SYNC very close to the 128 serial clock cycles length. Providing a small overhead on the pulse length in order to assure the SYNC pulse will not be misinterpreted by the target. See Section 15.4.9, “SYNC — Request Timed Reference Pulse”.

Figure 15-13 shows a SYNC command being issued after a READ_BYTE, which aborts the READ_BYTE command. Note that, after the command is aborted a new command could be issued by the host computer.

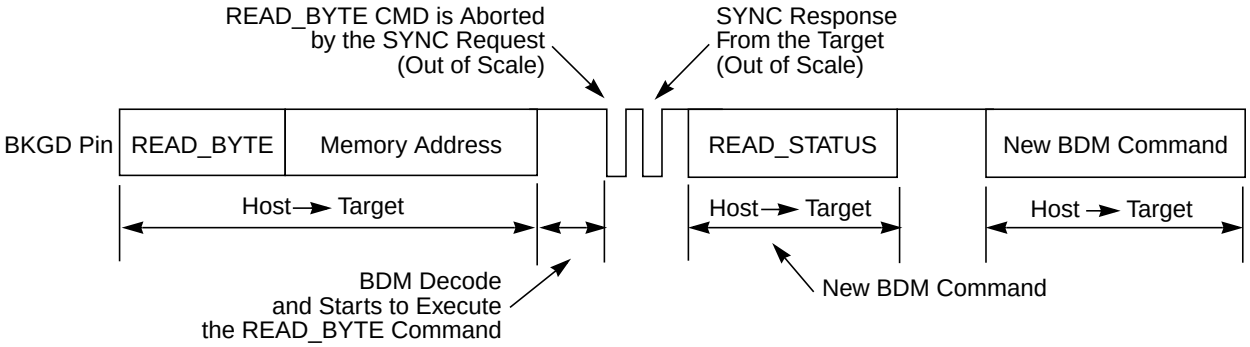


Figure 15-13. ACK Abort Procedure at the Command Level

NOTE

Figure 15-13 does not represent the signals in a true timing scale

Figure 15-14 shows a conflict between the ACK pulse and the SYNC request pulse. This conflict could occur if a POD device is connected to the target BKGD pin and the target is already in debug active mode. Consider that the target CPU is executing a pending BDM command at the exact moment the POD is being connected to the BKGD pin. In this case, an ACK pulse is issued along with the SYNC command. In this case, there is an electrical conflict between the ACK speedup pulse and the SYNC pulse. Since this is not a probable situation, the protocol does not prevent this conflict from happening.

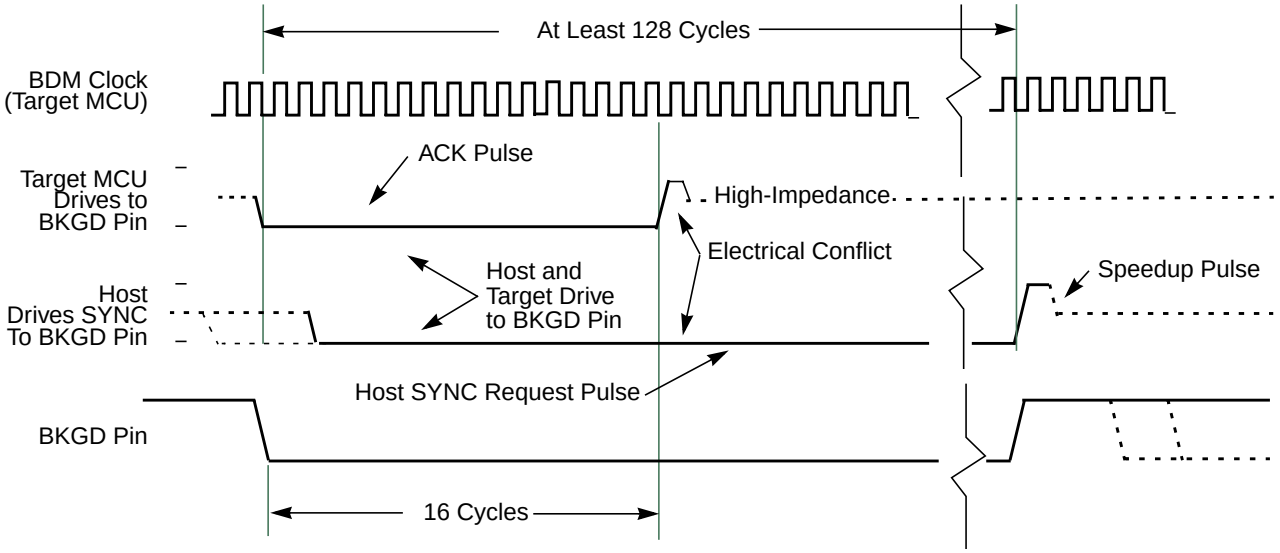


Figure 15-14. ACK Pulse and SYNC Request Conflict

NOTE

This information is being provided so that the MCU integrator will be aware that such a conflict could eventually occur.

The hardware handshake protocol is enabled by the ACK_ENABLE and disabled by the ACK_DISABLE BDM commands. This provides backwards compatibility with the existing POD devices which are not able to execute the hardware handshake protocol. It also allows for new POD devices, that support the hardware handshake protocol, to freely communicate with the target device. If desired, without the need for waiting for the ACK pulse.

The commands are described as follows:

- ACK_ENABLE — enables the hardware handshake protocol. The target will issue the ACK pulse when a CPU command is executed by the CPU. The ACK_ENABLE command itself also has the ACK pulse as a response.
- ACK_DISABLE — disables the ACK pulse protocol. In this case, the host needs to use the worst case delay time at the appropriate places in the protocol.

The default state of the BDM after reset is hardware handshake protocol disabled.

All the read commands will ACK (if enabled) when the data bus cycle has completed and the data is then ready for reading out by the BKGD serial pin. All the write commands will ACK (if enabled) after the data has been received by the BDM through the BKGD serial pin and when the data bus cycle is complete. See Section 15.4.3, “BDM Hardware Commands” and Section 15.4.4, “Standard BDM Firmware Commands” for more information on the BDM commands.

The ACK_ENABLE sends an ACK pulse when the command has been completed. This feature could be used by the host to evaluate if the target supports the hardware handshake protocol. If an ACK pulse is issued in response to this command, the host knows that the target supports the hardware handshake protocol. If the target does not support the hardware handshake protocol the ACK pulse is not issued. In this case, the ACK_ENABLE command is ignored by the target since it is not recognized as a valid command.

The BACKGROUND command will issue an ACK pulse when the CPU changes from normal to background mode. The ACK pulse related to this command could be aborted using the SYNC command.

The GO command will issue an ACK pulse when the CPU exits from background mode. The ACK pulse related to this command could be aborted using the SYNC command.

The GO_UNTIL command is equivalent to a GO command with exception that the ACK pulse, in this case, is issued when the CPU enters into background mode. This command is an alternative to the GO command and should be used when the host wants to trace if a breakpoint match occurs and causes the CPU to enter active background mode. Note that the ACK is issued whenever the CPU enters BDM, which could be caused by a breakpoint match or by a BGND instruction being executed. The ACK pulse related to this command could be aborted using the SYNC command.

The TRACE1 command has the related ACK pulse issued when the CPU enters background active mode after one instruction of the application program is executed. The ACK pulse related to this command could be aborted using the SYNC command.

15.4.9 SYNC — Request Timed Reference Pulse

The SYNC command is unlike other BDM commands because the host does not necessarily know the correct communication speed to use for BDM communications until after it has analyzed the response to the SYNC command. To issue a SYNC command, the host should perform the following steps:

1. Drive the BKGD pin low for at least 128 cycles at the lowest possible BDM serial communication frequency (the lowest serial communication frequency is determined by the crystal oscillator or the clock chosen by CLKSW.)
2. Drive BKGD high for a brief speedup pulse to get a fast rise time (this speedup pulse is typically one cycle of the host clock.)
3. Remove all drive to the BKGD pin so it reverts to high impedance.
4. Listen to the BKGD pin for the sync response pulse.

Upon detecting the SYNC request from the host, the target performs the following steps:

1. Discards any incomplete command received or bit retrieved.
2. Waits for BKGD to return to a logic one.
3. Delays 16 cycles to allow the host to stop driving the high speedup pulse.
4. Drives BKGD low for 128 cycles at the current BDM serial communication frequency.
5. Drives a one-cycle high speedup pulse to force a fast rise time on BKGD.
6. Removes all drive to the BKGD pin so it reverts to high impedance.

The host measures the low time of this 128 cycle SYNC response pulse and determines the correct speed for subsequent BDM communications. Typically, the host can determine the correct communication speed

within a few percent of the actual target speed and the communication protocol can easily tolerate speed errors of several percent.

As soon as the SYNC request is detected by the target, any partially received command or bit retrieved is discarded. This is referred to as a soft-reset, equivalent to a time-out in the serial communication. After the SYNC response, the target will consider the next negative edge (issued by the host) as the start of a new BDM command or the start of new SYNC request.

Another use of the SYNC command pulse is to abort a pending ACK pulse. The behavior is exactly the same as in a regular SYNC command. Note that one of the possible causes for a command to not be acknowledged by the target is a host-target synchronization problem. In this case, the command may not have been understood by the target and so an ACK response pulse will not be issued.

15.4.10 Instruction Tracing

When a TRACE1 command is issued to the BDM in active BDM, the CPU exits the standard BDM firmware and executes a single instruction in the user code. Once this has occurred, the CPU is forced to return to the standard BDM firmware and the BDM is active and ready to receive a new command. If the TRACE1 command is issued again, the next user instruction will be executed. This facilitates stepping or tracing through the user code one instruction at a time.

If an interrupt is pending when a TRACE1 command is issued, the interrupt stacking operation occurs but no user instruction is executed. Once back in standard BDM firmware execution, the program counter points to the first instruction in the interrupt service routine.

Be aware when tracing through the user code that the execution of the user code is done step by step but all peripherals are free running. Hence possible timing relations between CPU code execution and occurrence of events of other peripherals no longer exist.

Do not trace the CPU instruction BGND used for soft breakpoints. Tracing the BGND instruction will result in a return address pointing to BDM firmware address space.

When tracing through user code which contains stop or wait instructions the following will happen when the stop or wait instruction is traced:

The CPU enters stop or wait mode and the TRACE1 command can not be finished before leaving the low power mode. This is the case because BDM active mode can not be entered after CPU executed the stop instruction. However all BDM hardware commands except the BACKGROUND command are operational after tracing a stop or wait instruction and still being in stop or wait mode. If system stop mode is entered (all bus masters are in stop mode) no BDM command is operational.

As soon as stop or wait mode is exited the CPU enters BDM active mode and the saved PC value points to the entry of the corresponding interrupt service routine.

In case the handshake feature is enabled the corresponding ACK pulse of the TRACE1 command will be discarded when tracing a stop or wait instruction. Hence there is no ACK pulse when BDM active mode is entered as part of the TRACE1 command after CPU exited from stop or wait mode. All valid commands sent during CPU being in stop or wait mode or after CPU exited from stop or wait mode will have an ACK pulse. The handshake feature becomes disabled only when system

stop mode has been reached. Hence after a system stop mode the handshake feature must be enabled again by sending the ACK_ENABLE command.

15.4.11 Serial Communication Time Out

The host initiates a host-to-target serial transmission by generating a falling edge on the BKGD pin. If BKGD is kept low for more than 128 target clock cycles, the target understands that a SYNC command was issued. In this case, the target will keep waiting for a rising edge on BKGD in order to answer the SYNC request pulse. If the rising edge is not detected, the target will keep waiting forever without any time-out limit.

Consider now the case where the host returns BKGD to logic one before 128 cycles. This is interpreted as a valid bit transmission, and not as a SYNC request. The target will keep waiting for another falling edge marking the start of a new bit. If, however, a new falling edge is not detected by the target within 512 clock cycles since the last falling edge, a time-out occurs and the current command is discarded without affecting memory or the operating mode of the MCU. This is referred to as a soft-reset.

If a read command is issued but the data is not retrieved within 512 serial clock cycles, a soft-reset will occur causing the command to be disregarded. The data is not available for retrieval after the time-out has occurred. This is the expected behavior if the handshake protocol is not enabled. However, consider the behavior where the BDM is running in a frequency much greater than the CPU frequency. In this case, the command could time out before the data is ready to be retrieved. In order to allow the data to be retrieved even with a large clock frequency mismatch (between BDM and CPU) when the hardware handshake protocol is enabled, the time out between a read command and the data retrieval is disabled. Therefore, the host could wait for more then 512 serial clock cycles and still be able to retrieve the data from an issued read command. However, once the handshake pulse (ACK pulse) is issued, the time-out feature is re-activated, meaning that the target will time out after 512 clock cycles. Therefore, the host needs to retrieve the data within a 512 serial clock cycles time frame after the ACK pulse had been issued. After that period, the read command is discarded and the data is no longer available for retrieval. Any negative edge in the BKGD pin after the time-out period is considered to be a new command or a SYNC request.

Note that whenever a partially issued command, or partially retrieved data, has occurred the time out in the serial communication is active. This means that if a time frame higher than 512 serial clock cycles is observed between two consecutive negative edges and the command being issued or data being retrieved is not complete, a soft-reset will occur causing the partially received command or data retrieved to be disregarded. The next negative edge in the BKGD pin, after a soft-reset has occurred, is considered by the target as the start of a new BDM command, or the start of a SYNC request pulse.

Chapter 16
S12X Debug (S12XDBGV3) Module

Table 16-1. Revision History

Revision Number	Revision Date	Sections Affected	Description of Changes
03.17	20.MAR.2007		Tabulated glossary Renamed S12XCPU to CPU12X
03.18	20.APR.2007		Added “Data Bus Comparison NDB Dependency” section Clarified effect TRIG has on state sequencer
03.17	24.APR.2007		Clarified simultaneous arm and disarm effect

16.1 Introduction

The S12XDBG module provides an on-chip trace buffer with flexible triggering capability to allow non-intrusive debug of application software. The S12XDBG module is optimized for the S12X 16-bit architecture and allows debugging of CPU12Xand XGATE module operations.

Typically the S12XDBG module is used in conjunction with the S12XBDM module, whereby the user configures the S12XDBG module for a debugging session over the BDM interface. Once configured the S12XDBG module is armed and the device leaves BDM Mode returning control to the user program, which is then monitored by the S12XDBG module. Alternatively the S12XDBG module can be configured over a serial interface using SWI routines.

16.1.1 Glossary

Table 16-2. Glossary Of Terms

Term	Definition
COF	Change Of Flow. Change in the program flow due to a conditional branch, indexed jump or interrupt
BDM	Background Debug Mode
DUG	Device User Guide, describing the features of the device into which the DBG is integrated
WORD	16 bit data entity
Data Line	64 bit data entity

Table 16-2. Glossary Of Terms

Term	Definition
CPU	CPU12X module
Tag	Tags can be attached to XGATE or CPU opcodes as they enter the instruction pipe. If the tagged opcode reaches the execution stage a tag hit occurs.

16.1.2 Overview

The comparators monitor the bus activity of the CPU12X and XGATE. When a match occurs the control logic can trigger the state sequencer to a new state. On a transition to the Final State, bus tracing is triggered and/or a breakpoint can be generated.

Independent of comparator matches a transition to Final State with associated tracing and breakpoint can be triggered by the external $\overline{\text{TAGHI}}$ and $\overline{\text{TAGLO}}$ signals, or by an XGATE module S/W breakpoint request or by writing to the TRIG control bit.

The trace buffer is visible through a 2-byte window in the register address map and can be read out using standard 16-bit word reads. Tracing is disabled when the MCU system is secured.

16.1.3 Features

- Four comparators (A, B, C, and D)
  - Comparators A and C compare the full address bus and full 16-bit data bus
  - Comparators A and C feature a data bus mask register
  - Comparators B and D compare the full address bus only
  - Each comparator can be configured to monitor CPU12X or XGATE buses
  - Each comparator features selection of read or write access cycles
  - Comparators B and D allow selection of byte or word access cycles
  - Comparisons can be used as triggers for the state sequencer
- Three comparator modes
  - Simple address/data comparator match mode
  - Inside address range mode, Addmin ≤ Address ≤ Addmax
  - Outside address range match mode, Address < Addmin or Address > Addmax
- Two types of triggers
  - Tagged — This triggers just before a specific instruction begins execution
  - Force — This triggers on the first instruction boundary after a match occurs.
- The following types of breakpoints
  - CPU12X breakpoint entering BDM on breakpoint (BDM)
  - CPU12X breakpoint executing SWI on breakpoint (SWI)
  - XGATE breakpoint
- External CPU12X instruction tagging trigger independent of comparators
- XGATE S/W breakpoint request trigger independent of comparators

- TRIG Immediate software trigger independent of comparators
- Four trace modes
  - Normal: change of flow (COF) PC information is stored (see Section 16.4.5.2.1) for change of flow definition.
  - Loop1: same as Normal but inhibits consecutive duplicate source address entries
  - Detail: address and data for all cycles except free cycles and opcode fetches are stored
  - Pure PC: All program counter addresses are stored.
- 4-stage state sequencer for trace buffer control
  - Tracing session trigger linked to Final State of state sequencer
  - Begin, End, and Mid alignment of tracing to trigger

16.1.4 Modes of Operation

The S12XDBG module can be used in all MCU functional modes.

During BDM hardware accesses and whilst the BDM module is active, CPU12X monitoring is disabled. Thus breakpoints, comparators, and CPU12X bus tracing are disabled but XGATE bus monitoring accessing the S12XDBG registers, including comparator registers, is still possible. While in active BDM or during hardware BDM accesses, XGATE activity can still be compared, traced and can be used to generate a breakpoint to the XGATE module. When the CPU12X enters active BDM Mode through a BACKGROUND command, with the S12XDBG module armed, the S12XDBG remains armed.

The S12XDBG module tracing is disabled if the MCU is secure. However, breakpoints can still be generated if the MCU is secure.

Table 16-3. Mode Dependent Restriction Summary

BDM Enable	BDM Active	MCU Secure	Comparator Matches Enabled	Breakpoints Possible	Tagging Possible	Tracing Possible
x	x	1	Yes	Yes	Yes	No
0	0	0	Yes	Only SWI	Yes	Yes
0	1	0	Active BDM not possible when not enabled			
1	0	0	Yes	Yes	Yes	Yes
1	1	0	XGATE only	XGATE only	XGATE only	XGATE only

16.1.5 Block Diagram

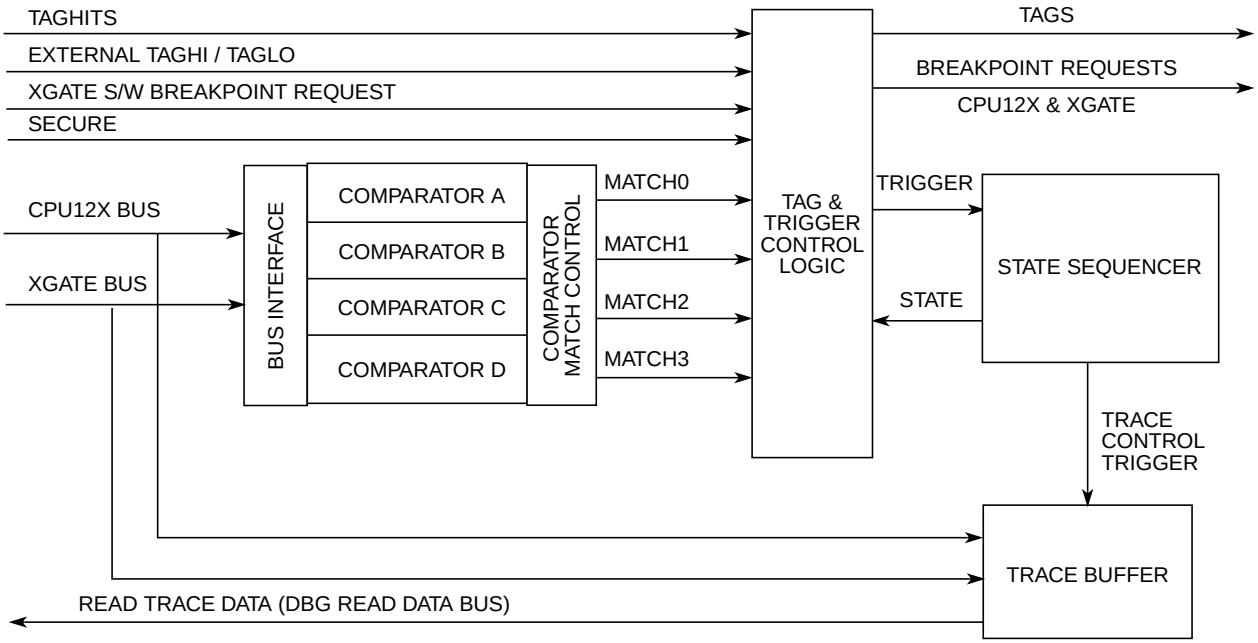


Figure 16-1. Debug Module Block Diagram

16.2 External Signal Description

The S12XDBG sub-module features two external tag input signals. See Device User Guide (DUG) for the mapping of these signals to device pins. These tag pins may be used for the external tagging in emulation modes only.

Table 16-4. External System Pins Associated With S12XDBG

Pin Name	Pin Functions	Description
$\overline{\text{TAGHI}}$ (See DUG)	TAGHI	When instruction tagging is on, tags the high half of the instruction word being read into the instruction queue.
$\overline{\text{TAGLO}}$ (See DUG)	TAGLO	When instruction tagging is on, tags the low half of the instruction word being read into the instruction queue.
$\overline{\text{TAGLO}}$ (See DUG)	Unconditional Tagging Enable	In emulation modes, a low assertion on this pin in the 7th or 8th cycle after the end of reset enables the Unconditional Tagging function.

16.3 Memory Map and Registers

16.3.1 Module Memory Map

A summary of the registers associated with the S12XDBG sub-block is shown in Table 16-2. Detailed descriptions of the registers and bits are given in the subsections that follow.

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0020	DBGC1	R	ARM	0	XGSBPE	BDM	DBGBRK		COMRV	
		W		TRIG						
0x0021	DBGSR	R	TBF	EXTF	0	0	0	SSF2	SSF1	SSF0
		W								
0x0022	DBGTCR	R	TSOURCE		TRANGE		TRCMOD		TALIGN	
		W								
0x0023	DBGC2	R	0	0	0	0	CDCM		ABCM	
		W								
0x0024	DBGTBH	R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		W								
0x0025	DBGTBL	R	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		W								
0x0026	DBGCNT	R	0	CNT						
		W								
0x0027	DBGSCRX	R	0	0	0	0	SC3	SC2	SC1	SC0
		W								
0x0027	DBGMFR	R	0	0	0	0	MC3	MC2	MC1	MC0
		W								
0x0028[1]	DBGXCTL (COMPA/C)	R	0	NDB	TAG	BRK	RW	RWE	SRC	COMPE
		W								
0x0028[2]	DBGXCTL (COMPB/D)	R	SZE	SZ	TAG	BRK	RW	RWE	SRC	COMPE
		W								
0x0029	DBGXAH	R	0	Bit 22	21	20	19	18	17	Bit 16
		W								
0x002A	DBGXAM	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x002B	DBGXAL	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x002C	DBGXDH	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x002D	DBGXDL	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x002E	DBGXDHM	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x002F	DBGXDLM	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								

1 This represents the contents if the Comparator A or C control register is blended into this address.

2 This represents the contents if the Comparator B or D control register is blended into this address

Figure 16-2. Quick Reference to S12XDBG Registers

16.3.2 Register Descriptions

This section consists of the S12XDBG control and trace buffer register descriptions in address order. Each comparator has a bank of registers that are visible through an 8-byte window between 0x0028 and 0x002F in the S12XDBG module register address map. When ARM is set in DBGC1, the only bits in the S12XDBG module registers that can be written are ARM, TRIG, and COMRV[1:0]

16.3.2.1 Debug Control Register 1 (DBGC1)

Address: 0x0020

	7	6	5	4	3	2	1	0
R	ARM	0	XGSBPE	BDM	DBGBRK		COMRV	
W		TRIG						
Reset	0	0	0	0	0	0	0	0

Figure 16-3. Debug Control Register (DBGC1)

Read: Anytime

Write: Bits 7, 1, 0 anytime
Bit 6 can be written anytime but always reads back as 0.
Bits 5:2 anytime S12XDBG is not armed.

NOTE

If a write access to DBGC1 with the ARM bit position set occurs simultaneously to a hardware disarm from an internal trigger event, then the ARM bit is cleared due to the hardware disarm.

NOTE

When disarming the S12XDBG by clearing ARM with software, the contents of bits[5:2] are not affected by the write, since up until the write operation, ARM = 1 preventing these bits from being written. These bits must be cleared using a second write if required.

Table 16-5. DBGC1 Field Descriptions

Field	Description
7 ARM	**Arm Bit** — The ARM bit controls whether the S12XDBG module is armed. This bit can be set and cleared by user software and is automatically cleared on completion of a tracing session, or if a breakpoint is generated with tracing not enabled. On setting this bit the state sequencer enters State1. 0 Debugger disarmed 1 Debugger armed
6 TRIG	**Immediate Trigger Request Bit** — This bit when written to 1 requests an immediate trigger independent of comparator or external tag signal status. When tracing is complete a forced breakpoint may be generated depending upon DBGBRK and BDM bit settings. This bit always reads back a 0. Writing a 0 to this bit has no effect. If TSOURCE are clear no tracing is carried out. If tracing has already commenced using BEGIN- or MID trigger alignment, it continues until the end of the tracing session as defined by the TALIGN bit settings, thus TRIG has no affect. In secure mode tracing is disabled and writing to this bit has no effect. 0 Do not trigger until the state sequencer enters the Final State. 1 Trigger immediately .

Table 16-5. DBGC1 Field Descriptions (continued)

Field	Description
5 XGSBPE	**XGATE S/W Breakpoint Enable** — The XGSBPE bit controls whether an XGATE S/W breakpoint request is passed to the CPU12X. The XGATE S/W breakpoint request is handled by the S12XDBG module, which can request an CPU12X breakpoint depending on the state of this bit. 0 XGATE S/W breakpoint request is disabled 1 XGATE S/W breakpoint request is enabled
4 BDM	**Background Debug Mode Enable** — This bit determines if an S12X breakpoint causes the system to enter Background Debug Mode (BDM) or initiate a Software Interrupt (SWI). If this bit is set but the BDM is not enabled by the ENBDM bit in the BDM module, then breakpoints default to SWI. 0 Breakpoint to Software Interrupt if BDM inactive. Otherwise no breakpoint. 1 Breakpoint to BDM, if BDM enabled. Otherwise breakpoint to SWI
3–2 DBGBRK	**S12XDBG Breakpoint Enable Bits** — The DBGBRK bits control whether the debugger will request a breakpoint to either CPU12X or XGATE or both upon reaching the state sequencer Final State. If tracing is enabled, the breakpoint is generated on completion of the tracing session. If tracing is not enabled, the breakpoint is generated immediately. Please refer to Section 16.4.7 for further details. XGATE software breakpoints are independent of the DBGBRK bits. XGATE software breakpoints force a breakpoint to the CPU12X independent of the DBGBRK bit field configuration. See Table 16-5.
1–0 COMRV	**Comparator Register Visibility Bits** — These bits determine which bank of comparator register is visible in the 8-byte window of the S12XDBG module address map, located between 0x0028 to 0x002F. Furthermore these bits determine which register is visible at the address 0x0027. See Table 16-6.

Table 16-6. DBGBRK Encoding

DBGBRK	Resource Halted by Breakpoint
00	No breakpoint generated
01	XGATE breakpoint generated
10	CPU12X breakpoint generated
11	Breakpoints generated for CPU12X and XGATE

Table 16-7. COMRV Encoding

COMRV	Visible Comparator	Visible Register at 0x0027
00	Comparator A	DBGSCR1
01	Comparator B	DBGSCR2
10	Comparator C	DBGSCR3
11	Comparator D	DBGMFR

16.3.2.2 Debug Status Register (DBGSR)

Address: 0x0021

	7	6	5	4	3	2	1	0
R	TBF	EXTF	0	0	0	SSF2	SSF1	SSF0
W								
Reset	—	0	0	0	0	0	0	0
POR	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 16-4. Debug Status Register (DBGSR)

Read: Anytime

Write: Never

Table 16-8. DBGSR Field Descriptions

Field	Description
7 TBF	**Trace Buffer Full** — The TBF bit indicates that the trace buffer has stored 64 or more lines of data since it was last armed. If this bit is set, then all 64 lines will be valid data, regardless of the value of DBGCNT bits CNT[6:0]. The TBF bit is cleared when ARM in DBGC1 is written to a one. The TBF is cleared by the power on reset initialization. Other system generated resets have no affect on this bit
6 EXTF	**External Tag Hit Flag** — The EXTF bit indicates if a tag hit condition from an external $\overline{\text{TAGHI}}$/$\overline{\text{TAGLO}}$ tag was met since arming. This bit is cleared when ARM in DBGC1 is written to a one. 0 External tag hit has not occurred 1 External tag hit has occurred
2–0 SSF[2:0]	**State Sequencer Flag Bits** — The SSF bits indicate in which state the State Sequencer is currently in. During a debug session on each transition to a new state these bits are updated. If the debug session is ended by software clearing the ARM bit, then these bits retain their value to reflect the last state of the state sequencer before disarming. If a debug session is ended by an internal trigger, then the state sequencer returns to state0 and these bits are cleared to indicate that state0 was entered during the session. On arming the module the state sequencer enters state1 and these bits are forced to SSF[2:0] = 001. See Table 16-8**.**

Table 16-9. SSF[2:0] — State Sequence Flag Bit Encoding

SSF[2:0]	Current State
000	State0 (disarmed)
001	State1
010	State2
011	State3
100	Final State
101,110,111	Reserved

16.3.2.3 Debug Trace Control Register (DBGTCR)

Address: 0x0022

	7	6	5	4	3	2	1	0
R W	TSOURCE		TRANGE		TRCMOD		TALIGN	
Reset	0	0	0	0	0	0	0	0

Figure 16-5. Debug Trace Control Register (DBGTCR)

Read: Anytime

Write: Bits 7:6 only when S12XDBG is neither secure nor armed.
Bits 5:0 anytime the module is disarmed.

Table 16-10. DBGTCR Field Descriptions

Field	Description
7–6 TSOURCE	**Trace Source Control Bits** — The TSOURCE bits select the data source for the tracing session. If the MCU system is secured, these bits cannot be set and tracing is inhibited. See Table 16-10.
5–4 TRANGE	**Trace Range Bits** — The TRANGE bits allow filtering of trace information from a selected address range when tracing from the CPU12X in Detail Mode. The XGATE tracing range cannot be narrowed using these bits. To use a comparator for range filtering, the corresponding COMPE and SRC bits must remain cleared. If the COMPE bit is not clear then the comparator will also be used to generate state sequence triggers. If the corresponding SRC bit is set the comparator is mapped to the XGATE buses, the TRANGE bits have no effect on the valid address range, memory accesses within the whole memory map are traced. See Table 16-11.
3–2 TRCMOD	**Trace Mode Bits** — See Section 16.4.5.2 for detailed Trace Mode descriptions. In Normal Mode, change of flow information is stored. In Loop1 Mode, change of flow information is stored but redundant entries into trace memory are inhibited. In Detail Mode, address and data for all memory and register accesses is stored. See Table 16-12.
1–0 TALIGN	**Trigger Align Bits** — These bits control whether the trigger is aligned to the beginning, end or the middle of a tracing session. See Table 16-13.

Table 16-11. TSOURCE — Trace Source Bit Encoding

TSOURCE	Tracing Source
00	No tracing requested
01	CPU12X
10[(1)]	XGATE
11[1,(2)]	Both CPU12X and XGATE

1. No range limitations are allowed. Thus tracing operates as if TRANGE = 00.
2. No Detail Mode tracing supported. If TRCMOD = 10, no information is stored.

Table 16-12. TRANGE Trace Range Encoding

TRANGE	Tracing Range
00	Trace from all addresses (No filter)
01	Trace only in address range from $00000 to Comparator D
10	Trace only in address range from Comparator C to $7FFFFF
11	Trace only in range from Comparator C to Comparator D

Table 16-13. TRCMOD Trace Mode Bit Encoding

TRCMOD	Description
00	Normal
01	Loop1
10	Detail
11	Pure PC

Table 16-14. TALIGN Trace Alignment Encoding

TALIGN	Description
00	Trigger at end of stored data
01	Trigger before storing data
10	Trace buffer entries before and after trigger
11	Reserved

16.3.2.4 Debug Control Register2 (DBGC2)

Address: 0x0023

	7	6	5	4	3	2	1	0
R	0	0	0	0	CDCM		ABCM	
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 16-6. Debug Control Register2 (DBGC2)

Read: Anytime

Write: Anytime the module is disarmed.

This register configures the comparators for range matching.

Table 16-15. DBGC2 Field Descriptions

Field	Description
3–2 CDCM[1:0]	**C and D Comparator Match Control** — These bits determine the C and D comparator match mapping as described in Table 16-15.
1–0 ABCM[1:0]	**A and B Comparator Match Control** — These bits determine the A and B comparator match mapping as described in Table 16-16.

Table 16-16. CDCM Encoding

CDCM	Description
00	Match2 mapped to comparator C match..... Match3 mapped to comparator D match.
01	Match2 mapped to comparator C/D inside range..... Match3 disabled.
10	Match2 mapped to comparator C/D outside range..... Match3 disabled.
11	Reserved(1)

1. Currently defaults to Match2 mapped to comparator C : Match3 mapped to comparator D

Table 16-17. ABCM Encoding

ABCM	Description
00	Match0 mapped to comparator A match..... Match1 mapped to comparator B match.
01	Match 0 mapped to comparator A/B inside range..... Match1 disabled.
10	Match 0 mapped to comparator A/B outside range..... Match1 disabled.
11	Reserved(1)

1. Currently defaults to Match0 mapped to comparator A : Match1 mapped to comparator B

16.3.2.5 Debug Trace Buffer Register (DBGTBH:DBGTBL)

Address: 0x0024, 0x0025

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
POR	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Other Resets	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—

Figure 16-7. Debug Trace Buffer Register (DBGTB)

Read: Only when unlocked AND not secured AND not armed AND with a TSOURCE bit set.

Write: Aligned word writes when disarmed unlock the trace buffer for reading but do not affect trace buffer contents.

Table 16-18. DBGTB Field Descriptions

Field	Description
15–0 Bit[15:0]	**Trace Buffer Data Bits** — The Trace Buffer Register is a window through which the 64-bit wide data lines of the Trace Buffer may be read 16 bits at a time. Each valid read of DBGTB increments an internal trace buffer pointer which points to the next address to be read. When the ARM bit is written to 1 the trace buffer is locked to prevent reading. The trace buffer can only be unlocked for reading by writing to DBGTB with an aligned word write when the module is disarmed. The DBGTB register can be read only as an aligned word, any byte reads or misaligned access of these registers will return 0 and will not cause the trace buffer pointer to increment to the next trace buffer address. The same is true for word reads while the debugger is armed. The POR state is undefined Other resets do not affect the trace buffer contents. .

16.3.2.6 Debug Count Register (DBGCNT)

Address: 0x0026

	7	6	5	4	3	2	1	0
R	0	CNT						
W								
Reset	0	—	—	—	—	—	—	—
POR	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 16-8. Debug Count Register (DBGCNT)

Read: Anytime

Write: Never

Table 16-19. DBGCNT Field Descriptions

Field	Description
6–0 CNT[6:0]	**Count Value** — The CNT bits [6:0] indicate the number of valid data 64-bit data lines stored in the Trace Buffer. Table 16-19 shows the correlation between the CNT bits and the number of valid data lines in the Trace Buffer. When the CNT rolls over to zero, the TBF bit in DBGSR is set and incrementing of CNT will continue in end-trigger or mid-trigger mode. The DBGCNT register is cleared when ARM in DBGC1 is written to a one. The DBGCNT register is cleared by power-on-reset initialization but is not cleared by other system resets. Thus should a reset occur during a debug session, the DBGCNT register still indicates after the reset, the number of valid trace buffer entries stored before the reset occurred. The DBGCNT register is not decremented when reading from the trace buffer.

Table 16-20. CNT Decoding Table

TBF (DBGSR)	CNT[6:0]	Description
0	0000000	No data valid
0	0000001	32 bits of one line valid(1)
0	0000010 0000100 0000110 .. 1111100	1 line valid 2 lines valid 3 lines valid .. 62 lines valid
0	1111110	63 lines valid
1	0000000	64 lines valid; if using Begin trigger alignment, ARM bit will be cleared and the tracing session ends.
1	0000010 1111110	64 lines valid, oldest data has been overwritten by most recent data

1. This applies to Normal/Loop1/PurePC Modes when tracing from either CPU12X or XGATE only.

16.3.2.7 Debug State Control Registers

There is a dedicated control register for each of the state sequencer states 1 to 3 that determines if transitions from that state are allowed, depending upon comparator matches or tag hits, and defines the

next state for the state sequencer following a match. The three debug state control registers are located at the same address in the register address map (0x0027). Each register can be accessed using the COMRV bits in DBGC1 to blend in the required register. The COMRV = 11 value blends in the match flag register (DBGMFR).

Table 16-21. State Control Register Access Encoding

COMRV	Visible State Control Register
00	DBGSCR1
01	DBGSCR2
10	DBGSCR3
11	DBGMFR

16.3.2.7.1 Debug State Control Register 1 (DBGSCR1)

Address: 0x0027

	7	6	5	4	3	2	1	0
R	0	0	0	0	SC3	SC2	SC1	SC0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 16-9. Debug State Control Register 1 (DBGSCR1)

Read: If COMRV[1:0] = 00

Write: If COMRV[1:0] = 00 and S12XDBG is not armed.

This register is visible at 0x0027 only with COMRV[1:0] = 00. The state control register 1 selects the targeted next state whilst in State1. The matches refer to the match channels of the comparator match control logic as depicted in Figure 16-1 and described in Section 16.3.2.8.1". Comparators must be enabled by setting the comparator enable bit in the associated DBGXCTL control register.

Table 16-22. DBGSCR1 Field Descriptions

Field	Description
3–0 SC[3:0]	These bits select the targeted next state whilst in State1, based upon the match event.

Table 16-23. State1 Sequencer Next State Selection

SC[3:0]	Description
0000	Any match triggers to state2
0001	Any match triggers to state3
0010	Any match triggers to Final State
0011	Match2 triggers to State2..... Other matches have no effect
0100	Match2 triggers to State3..... Other matches have no effect
0101	Match2 triggers to Final State..... Other matches have no effect
0110	Match0 triggers to State2..... Match1 triggers to State3..... Other matches have no effect
0111	Match1 triggers to State3..... Match0 triggers Final State..... Other matches have no effect
1000	Match0 triggers to State2..... Match2 triggers to State3..... Other matches have no effect
1001	Match2 triggers to State3..... Match0 triggers Final State..... Other matches have no effect
1010	Match1 triggers to State2..... Match3 triggers to State3..... Other matches have no effect
1011	Match3 triggers to State3..... Match1 triggers to Final State..... Other matches have no effect
1100	Match3 has no effect..... All other matches (M0,M1,M2) trigger to State2
1101	Reserved
1110	Reserved
1111	Reserved

The trigger priorities described in Table 16-41 dictate that in the case of simultaneous matches, the match on the lower channel number (0,1,2,3) has priority. The SC[3:0] encoding ensures that a match leading to final state has priority over all other matches.

16.3.2.7.2 Debug State Control Register 2 (DBGSCR2)

Address: 0x0027

	7	6	5	4	3	2	1	0
R	0	0	0	0	SC3	SC2	SC1	SC0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 16-10. Debug State Control Register 2 (DBGSCR2)

Read: If COMRV[1:0] = 01

Write: If COMRV[1:0] = 01 and S12XDBG is not armed.

This register is visible at 0x0027 only with COMRV[1:0] = 01. The state control register 2 selects the targeted next state whilst in State2. The matches refer to the match channels of the comparator match control logic as depicted in Figure 16-1 and described in Section 16.3.2.8.1". Comparators must be enabled by setting the comparator enable bit in the associated DBGXCTL control register.

Table 16-24. DBGSCR2 Field Descriptions

Field	Description
3–0 SC[3:0]	These bits select the targeted next state whilst in State2, based upon the match event.

Table 16-25. State2 —Sequencer Next State Selection

SC[3:0]	Description
0000	Any match triggers to state1
0001	Any match triggers to state3
0010	Any match triggers to Final State
0011	Match3 triggers to State1..... Other matches have no effect
0100	Match3 triggers to State3..... Other matches have no effect
0101	Match3 triggers to Final State..... Other matches have no effect
0110	Match0 triggers to State1..... Match1 triggers to State3..... Other matches have no effect
0111	Match1 triggers to State3..... Match0 triggers Final State..... Other matches have no effect
1000	Match0 triggers to State1..... Match2 triggers to State3..... Other matches have no effect
1001	Match2 triggers to State3..... Match0 triggers Final State..... Other matches have no effect
1010	Match1 triggers to State1..... Match3 triggers to State3..... Other matches have no effect
1011	Match3 triggers to State3..... Match1 triggers Final State..... Other matches have no effect
1100	Match2 triggers to State1..... Match3 trigger to Final State
1101	Match2 has no affect, all other matches (M0,M1,M3) trigger to Final State
1110	Reserved
1111	Reserved

The trigger priorities described in Table 16-41 dictate that in the case of simultaneous matches, the match on the lower channel number (0,1,2,3) has priority. The SC[3:0] encoding ensures that a match leading to final state has priority over all other matches.

16.3.2.7.3 Debug State Control Register 3 (DBGSCR3)

Address: 0x0027

	7	6	5	4	3	2	1	0
R	0	0	0	0	SC3	SC2	SC1	SC0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 16-11. Debug State Control Register 3 (DBGSCR3)

Read: If COMRV[1:0] = 10

Write: If COMRV[1:0] = 10 and S12XDBG is not armed.

This register is visible at 0x0027 only with COMRV[1:0] = 10. The state control register three selects the targeted next state whilst in State3. The matches refer to the match channels of the comparator match control logic as depicted in Figure 16-1 and described in Section 16.3.2.8.1". Comparators must be enabled by setting the comparator enable bit in the associated DBGXCTL control register.

Table 16-26. DBGSCR3 Field Descriptions

Field	Description
3–0 SC[3:0]	These bits select the targeted next state whilst in State3, based upon the match event.

Table 16-27. State3 — Sequencer Next State Selection

SC[3:0]	Description
0000	Any match triggers to state1
0001	Any match triggers to state2
0010	Any match triggers to Final State
0011	Match0 triggers to State1..... Other matches have no effect
0100	Match0 triggers to State2..... Other matches have no effect
0101	Match0 triggers to Final State.....Match1 triggers to State1
0110	Match1 triggers to State1..... Other matches have no effect
0111	Match1 triggers to State2..... Other matches have no effect
1000	Match1 triggers to Final State..... Other matches have no effect
1001	Match2 triggers to State2..... Match0 triggers to Final State..... Other matches have no effect
1010	Match1 triggers to State1..... Match3 triggers to State2..... Other matches have no effect
1011	Match3 triggers to State2..... Match1 triggers to Final State..... Other matches have no effect
1100	Match2 triggers to Final State..... Other matches have no effect
1101	Match3 triggers to Final State..... Other matches have no effect
1110	Reserved
1111	Reserved

The trigger priorities described in Table 16-41 dictate that in the case of simultaneous matches, the match on the lower channel number (0,1,2,3) has priority. The SC[3:0] encoding ensures that a match leading to final state has priority over all other matches.

16.3.2.7.4 Debug Match Flag Register (DBGMFR)

Address: 0x0027

	7	6	5	4	3	2	1	0
R	0	0	0	0	MC3	MC2	MC1	MC0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 16-12. Debug Match Flag Register (DBGMFR)

Read: If COMRV[1:0] = 11

Write: Never

DBGMFR is visible at 0x0027 only with COMRV[1:0] = 11. It features four flag bits each mapped directly to a channel. Should a match occur on the channel during the debug session, then the corresponding flag is set and remains set until the next time the module is armed by writing to the ARM bit. Thus the contents are retained after a debug session for evaluation purposes. These flags cannot be cleared by software, they are cleared only when arming the module. A set flag does not inhibit the setting of other flags. Once a flag is set, further triggers on the same channel have no affect.

16.3.2.8 Comparator Register Descriptions

Each comparator has a bank of registers that are visible through an 8-byte window in the S12XDBG module register address map. Comparators A and C consist of 8 register bytes (3 address bus compare registers, two data bus compare registers, two data bus mask registers and a control register).

Comparators B and D consist of four register bytes (three address bus compare registers and a control register).

Each set of comparator registers is accessible in the same 8-byte window of the register address map and can be accessed using the COMRV bits in the DBGC1 register. If the Comparators B or D are accessed through the 8-byte window, then only the address and control bytes are visible, the 4 bytes associated with data bus and data bus masking read as zero and cannot be written. Furthermore the control registers for comparators B and D differ from those of comparators A and C.

Table 16-28. Comparator Register Layout

0x0028	CONTROL	Read/Write	Comparators A,B,C,D
0x0029	ADDRESS HIGH	Read/Write	Comparators A,B,C,D
0x002A	ADDRESS MEDIUM	Read/Write	Comparators A,B,C,D
0x002B	ADDRESS LOW	Read/Write	Comparators A,B,C,D
0x002C	DATA HIGH COMPARATOR	Read/Write	Comparator A and C only
0x002D	DATA LOW COMPARATOR	Read/Write	Comparator A and C only
0x002E	DATA HIGH MASK	Read/Write	Comparator A and C only
0x002F	DATA LOW MASK	Read/Write	Comparator A and C only

16.3.2.8.1 Debug Comparator Control Register (DBGXCTL)

The contents of this register bits 7 and 6 differ depending upon which comparator registers are visible in the 8-byte window of the DBG module register address map.

Address: 0x0028

	7	6	5	4	3	2	1	0
R	0	NDB	TAG	BRK	RW	RWE	SRC	COMPE
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 16-13. Debug Comparator Control Register (Comparators A and C)

Address: 0x0028

	7	6	5	4	3	2	1	0
R	SZE	SZ	TAG	BRK	RW	RWE	SRC	COMPE
W								
Reset	0	0	0	0	0	0	0	0

Figure 16-14. Debug Comparator Control Register (Comparators B and D)

Read: Anytime. See Table 16-28 for visible register encoding.

Write: If DBG not armed. See Table 16-28 for visible register encoding.

The DBGC1_COMRV bits determine which comparator control, address, data and datamask registers are visible in the 8-byte window from 0x0028 to 0x002F as shown in Section Table 16-28.

Table 16-29. Comparator Address Register Visibility

COMRV	Visible Comparator
00	DBGACTL, DBGAAH ,DBGAAM, DBGAAL, DBGADH, DBGADL, DBGADHM, DBGADLM
01	DBGBCTL, DBGBAH, DBGBAM, DBGBAL
10	DBGCCTL, DBGCAH, DBGCAM, DBGCAL, DBGCDH, DBGCDL, DBGCDHM, DBGCDLM
11	DBGDCTL, DBGDAH, DBGDAM, DBGDAL

Table 16-30. DBGXCTL Field Descriptions

Field	Description
7 SZE (Comparators B nd D)	**Size Comparator Enable Bit** — The SZE bit controls whether access size comparison is enabled for the associated comparator. This bit is ignored if the TAG bit in the same register is set. 0 Word/Byte access size is not used in comparison 1 Word/Byte access size is used in comparison
6 NDB (Comparators A and C	**Not Data Bus** — The NDB bit controls whether the match occurs when the data bus matches the comparator register value or when the data bus differs from the register value. Furthermore data bus bits can be individually masked using the comparator data mask registers. This bit is only available for comparators A and C. This bit is ignored if the TAG bit in the same register is set. This bit position has an SZ functionality for comparators B and D. 0 Match on data bus equivalence to comparator register contents 1 Match on data bus difference to comparator register contents

Table 16-30. DBGXCTL Field Descriptions (continued)

Field	Description
6 SZ (Comparators B and D)	**Size Comparator Value Bit** — The SZ bit selects either word or byte access size in comparison for the associated comparator. This bit is ignored if the SZE bit is cleared or if the TAG bit in the same register is set. This bit position has NDB functionality for comparators A and C 0 Word access size will be compared 1 Byte access size will be compared
5 TAG	**Tag Select** — This bit controls whether the comparator match will cause a trigger or tag the opcode at the matched address. Tagged opcodes trigger only if they reach the execution stage of the instruction queue. 0 Trigger immediately on match 1 On match, tag the opcode. If the opcode is about to be executed a trigger is generated
4 BRK	**Break** — This bit controls whether a channel match terminates a debug session immediately, independent of state sequencer state. To generate an immediate breakpoint the module breakpoints must be enabled using DBGBRK. 0 The debug session termination is dependent upon the state sequencer and trigger conditions. 1 A match on this channel terminates the debug session immediately; breakpoints if active are generated, tracing, if active, is terminated and the module disarmed.
3 RW	**Read/Write Comparator Value Bit** — The RW bit controls whether read or write is used in compare for the associated comparator . The RW bit is not used if RWE = 0. 0 Write cycle will be matched 1 Read cycle will be matched
2 RWE	**Read/Write Enable Bit** — The RWE bit controls whether read or write comparison is enabled for the associated comparator. This bit is not used for tagged operations. 0 Read/Write is not used in comparison 1 Read/Write is used in comparison
1 SRC	Determines mapping of comparator to CPU12X or XGATE 0 The comparator is mapped to CPU12X buses 1 The comparator is mapped to XGATE address and data buses
0 COMPE	Determines if comparator is enabled 0 The comparator is not enabled 1 The comparator is enabled for state sequence triggers or tag generation

Table 16-30 shows the effect for RWE and RW on the comparison conditions. These bits are not useful for tagged operations since the trigger occurs based on the tagged opcode reaching the execution stage of the instruction queue. Thus these bits are ignored if tagged triggering is selected.

Table 16-31. Read or Write Comparison Logic Table

RWE Bit	RW Bit	RW Signal	Comment
0	x	0	RW not used in comparison
0	x	1	RW not used in comparison
1	0	0	Write
1	0	1	No match
1	1	0	No match
1	1	1	Read

16.3.2.8.2 Debug Comparator Address High Register (DBGXAH)

Address: 0x0029

	7	6	5	4	3	2	1	0
R	0	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 16-15. Debug Comparator Address High Register (DBGXAH)

Read: Anytime. See Table 16-28 for visible register encoding.

Write: If DBG not armed. See Table 16-28 for visible register encoding.

Table 16-32. DBGXAH Field Descriptions

Field	Description
6–0 Bit[22:16]	**Comparator Address High Compare Bits** — The Comparator address high compare bits control whether the selected comparator will compare the address bus bits [22:16] to a logic one or logic zero. This register byte is ignored for XGATE compares. 0 Compare corresponding address bit to a logic zero 1 Compare corresponding address bit to a logic one

16.3.2.8.3 Debug Comparator Address Mid Register (DBGXAM)

Address: 0x002A

	7	6	5	4	3	2	1	0
R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
W								
Reset	0	0	0	0	0	0	0	0

Figure 16-16. Debug Comparator Address Mid Register (DBGXAM)

Read: Anytime. See Table 16-28 for visible register encoding.

Write: If DBG not armed. See Table 16-28 for visible register encoding.

Table 16-33. DBGXAM Field Descriptions

Field	Description
7–0 Bit[15:8]	**Comparator Address Mid Compare Bits**— The Comparator address mid compare bits control whether the selected comparator will compare the address bus bits [15:8] to a logic one or logic zero. 0 Compare corresponding address bit to a logic zero 1 Compare corresponding address bit to a logic one

16.3.2.8.4 Debug Comparator Address Low Register (DBGXAL)

Address: 0x002B

	7	6	5	4	3	2	1	0
R W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset	0	0	0	0	0	0	0	0

Figure 16-17. Debug Comparator Address Low Register (DBGXAL)

Read: Anytime. See Table 16-28 for visible register encoding.

Write: If DBG not armed. See Table 16-28 for visible register encoding.

Table 16-34. DBGXAL Field Descriptions

Field	Description
7–0 Bits[7:0]	**Comparator Address Low Compare Bits** — The Comparator address low compare bits control whether the selected comparator will compare the address bus bits [7:0] to a logic one or logic zero. 0 Compare corresponding address bit to a logic zero 1 Compare corresponding address bit to a logic one

16.3.2.8.5 Debug Comparator Data High Register (DBGXDH)

Address: 0x002C

	7	6	5	4	3	2	1	0
R W	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Reset	0	0	0	0	0	0	0	0

Figure 16-18. Debug Comparator Data High Register (DBGXDH)

Read: Anytime. See Table 16-28 for visible register encoding.

Write: If DBG not armed. See Table 16-28 for visible register encoding.

Table 16-35. DBGXAH Field Descriptions

Field	Description
7–0 Bits[15:8]	**Comparator Data High Compare Bits** — The Comparator data high compare bits control whether the selected comparator compares the data bus bits [15:8] to a logic one or logic zero. The comparator data compare bits are only used in comparison if the corresponding data mask bit is logic 1. This register is available only for comparators A and C. 0 Compare corresponding data bit to a logic zero 1 Compare corresponding data bit to a logic one

16.3.2.8.6 Debug Comparator Data Low Register (DBGXDL)

Address: 0x002D

	7	6	5	4	3	2	1	0
R W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset	0	0	0	0	0	0	0	0

Figure 16-19. Debug Comparator Data Low Register (DBGXDL)

Read: Anytime. See Table 16-28 for visible register encoding.

Write: If DBG not armed. See Table 16-28 for visible register encoding.

Table 16-36. DBGXDL Field Descriptions

Field	Description
7–0 Bits[7:0]	**Comparator Data Low Compare Bits** — The Comparator data low compare bits control whether the selected comparator compares the data bus bits [7:0] to a logic one or logic zero. The comparator data compare bits are only used in comparison if the corresponding data mask bit is logic 1. This register is available only for comparators A and C. 0 Compare corresponding data bit to a logic zero 1 Compare corresponding data bit to a logic one

16.3.2.8.7 Debug Comparator Data High Mask Register (DBGXDHM)

Address: 0x002E

	7	6	5	4	3	2	1	0
R W	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Reset	0	0	0	0	0	0	0	0

Figure 16-20. Debug Comparator Data High Mask Register (DBGXDHM)

Read: Anytime. See Table 16-28 for visible register encoding.

Write: If DBG not armed. See Table 16-28 for visible register encoding.

Table 16-37. DBGXDHM Field Descriptions

Field	Description
7–0 Bits[15:8]	**Comparator Data High Mask Bits** — The Comparator data high mask bits control whether the selected comparator compares the data bus bits [15:8] to the corresponding comparator data compare bits. This register is available only for comparators A and C. 0 Do not compare corresponding data bit 1 Compare corresponding data bit

16.3.2.8.8 Debug Comparator Data Low Mask Register (DBGXDLM)

Address: 0x002F

	7	6	5	4	3	2	1	0
R W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset	0	0	0	0	0	0	0	0

Figure 16-21. Debug Comparator Data Low Mask Register (DBGXDLM)

Read: Anytime. See Table 16-28 for visible register encoding.

Write: If DBG not armed. See Table 16-28 for visible register encoding.

Table 16-38. DBGXDLM Field Descriptions

Field	Description
7–0 Bits[7:0]	**Comparator Data Low Mask Bits** — The Comparator data low mask bits control whether the selected comparator compares the data bus bits [7:0] to the corresponding comparator data compare bits. This register is available only for comparators A and C. 0 Do not compare corresponding data bit 1 Compare corresponding data bit

16.4 Functional Description

This section provides a complete functional description of the S12XDBG module. If the part is in secure mode, the S12XDBG module can generate breakpoints but tracing is not possible.

16.4.1 S12XDBG Operation

Arming the S12XDBG module by setting ARM in DBGC1 allows triggering, and storing of data in the trace buffer and can be used to cause breakpoints to the CPU12X or the XGATE module. The DBG module is made up of four main blocks, the comparators, control logic, the state sequencer, and the trace buffer.

The comparators monitor the bus activity of the CPU12X and XGATE. Comparators can be configured to monitor address and databus. Comparators can also be configured to mask out individual data bus bits during a compare and to use R/W and word/byte access qualification in the comparison. When a match with a comparator register value occurs the associated control logic can trigger the state sequencer to another state (see Figure 16-22). Either forced or tagged triggers are possible. Using a forced trigger, the trigger is generated immediately on a comparator match. Using a tagged trigger, at a comparator match, the instruction opcode is tagged and only if the instruction reaches the execution stage of the instruction queue is a trigger generated. In the case of a transition to Final State, bus tracing is triggered and/or a breakpoint can be generated. Tracing of both CPU12X and/or XGATE bus activity is possible.

Independent of the state sequencer, a breakpoint can be triggered by the external $\overline{\text{TAGHI}}$ / $\overline{\text{TAGLO}}$ signals or by an XGATE S/W breakpoint request or by writing to the TRIG bit in the DBGC1 control register.

The trace buffer is visible through a 2-byte window in the register address map and can be read out using standard 16-bit word reads.

16.4.2 Comparator Modes

The S12XDBG contains four comparators, A, B, C, and D. Each comparator can be configured to monitor CPU12X or XGATE buses. Each comparator compares the selected address bus with the address stored in DBGXAH, DBGXAM, and DBGXAL. Furthermore, comparators A and C also compare the data buses to the data stored in DBGXDH, DBGXDL and allow masking of individual data bus bits.

S12X comparator matches are disabled in BDM and during BDM accesses.

The comparator match control logic configures comparators to monitor the buses for an exact address or an address range, whereby either an access inside or outside the specified range generates a match condition. The comparator configuration is controlled by the control register contents and the range control by the DBGC2 contents.

On a match a trigger can initiate a transition to another state sequencer state (see Section 16.4.3”). The comparator control register also allows the type of access to be included in the comparison through the use of the RWE, RW, SZE, and SZ bits. The RWE bit controls whether read or write comparison is enabled for the associated comparator and the RW bit selects either a read or write access for a valid match. Similarly the SZE and SZ bits allows the size of access (word or byte) to be considered in the compare. Only comparators B and D feature SZE and SZ.

The TAG bit in each comparator control register is used to determine the triggering condition. By setting TAG, the comparator will qualify a match with the output of opcode tracking logic and a trigger occurs before the tagged instruction executes (tagged-type trigger). Whilst tagging, the RW, RWE, SZE, and SZ bits are ignored and the comparator register must be loaded with the exact opcode address.

If the TAG bit is clear (forced type trigger) a comparator match is generated when the selected address appears on the system address bus. If the selected address is an opcode address, the match is generated when the opcode is fetched from the memory. This precedes the instruction execution by an indefinite number of cycles due to instruction pipe lining. For a comparator match of an opcode at an odd address when TAG = 0, the corresponding even address must be contained in the comparator register. Thus for an opcode at odd address (n), the comparator register must contain address (n–1).

Once a successful comparator match has occurred, the condition that caused the original match is not verified again on subsequent matches. Thus if a particular data value is verified at a given address, this address may not still contain that data value when a subsequent match occurs.

Comparators C and D can also be used to select an address range to trace from. This is determined by the TRANGE bits in the DBGTCR register. The TRANGE encoding is shown in Table 16-11. If the TRANGE bits select a range definition using comparator D, then comparator D is configured for trace range definition and cannot be used for address bus comparisons. Similarly if the TRANGE bits select a range definition using comparator C, then comparator C is configured for trace range definition and cannot be used for address bus comparisons.

Match[0, 1, 2, 3] map directly to Comparators[A, B, C, D] respectively, except in range modes (see Section 16.3.2.4”). Comparator priority rules are described in the trigger priority section (Section 16.4.3.6”).

16.4.2.1 Exact Address Comparator Match (Comparators A and C)

With range comparisons disabled, the match condition is an exact equivalence of address/data bus with the value stored in the comparator address/data registers. Further qualification of the type of access (R/W, word/byte) is possible.

Comparators A and C do not feature SZE or SZ control bits, thus the access size is not compared. The exact address is compared, thus with the comparator address register loaded with address (n) a word access of address (n–1) also accesses (n) but does not cause a match. Table 16-39 lists access considerations without data bus compare. Table 16-38 lists access considerations with data bus comparison. To compare byte accesses DBGXDH must be loaded with the data byte, the low byte must be masked out using the DBGXDLM mask register. On word accesses the data byte of the lower address is mapped to DBGXDH.

Table 16-39. Comparator A and C Data Bus Considerations

Access	Address	DBGxDH	DBGxDL	DBGxDHM	DBGxDLM	Example Valid Match
Word	ADDR[n]	Data[n]	Data[n+1]	$FF	$FF	MOVW #$WORD ADDR[n]
Byte	ADDR[n]	Data[n]	x	$FF	$00	MOVB #$BYTE ADDR[n]
Word	ADDR[n]	Data[n]	x	$FF	$00	MOVW #$WORD ADDR[n]
Word	ADDR[n]	x	Data[n+1]	$00	$FF	MOVW #$WORD ADDR[n]

Comparators A and C feature an NDB control bit to determine if a match occurs when the data bus differs to comparator register contents or when the data bus is equivalent to the comparator register contents.

16.4.2.2 Exact Address Comparator Match (Comparators B and D)

Comparators B and D feature SZ and SZE control bits. If SZE is clear, then the comparator address match qualification functions the same as for comparators A and C.

If the SZE bit is set the access size (word or byte) is compared with the SZ bit value such that only the specified type of access causes a match. Thus if configured for a byte access of a particular address, a word access covering the same address does not lead to match.

Table 16-40. Comparator Access Size Considerations

Comparator	Address	SZE	SZ8	Condition For Valid Match
Comparators A and C	ADDR[n]	—	—	Word and byte accesses of ADDR[n](1) MOVB #$BYTE ADDR[n] MOVW #$WORD ADDR[n]
Comparators B and D	ADDR[n]	0	X	Word and byte accesses of ADDR[n]1 MOVB #$BYTE ADDR[n] MOVW #$WORD ADDR[n]
Comparators B and D	ADDR[n]	1	0	Word accesses of ADDR[n]1 MOVW #$WORD ADDR[n]
Comparators B and D	ADDR[n]	1	1	Byte accesses of ADDR[n] MOVB #$BYTE ADDR[n]

1. A word access of ADDR[n-1] also accesses ADDR[n] but does not generate a match.
The comparator address register must contain the exact address used in the code.

16.4.2.3 Data Bus Comparison NDB Dependency

Comparators A and C each feature an NDB control bit, which allows data bus comparators to be configured to either trigger on equivalence or trigger on difference. This allows monitoring of a difference in the contents of an address location from an expected value.

When matching on an equivalence (NDB=0), each individual data bus bit position can be masked out by clearing the corresponding mask bit (DBGxDHM/DBGxDLM), so that it is ignored in the comparison. A match occurs when all data bus bits with corresponding mask bits set are equivalent. If all mask register bits are clear, then a match is based on the address bus only, the data bus is ignored.

When matching on a difference, mask bits can be cleared to ignore bit positions. A match occurs when any data bus bit with corresponding mask bit set is different. Clearing all mask bits, causes all bits to be ignored and prevents a match because no difference can be detected. In this case address bus equivalence does not cause a match.

Table 16-41. NDB and MASK bit dependency

NDB	DBGxDHM[n] / DBGxDLM[n]	Comment
0	0	Do not compare data bus bit.
0	1	Compare data bus bit. Match on equivalence.
1	0	Do not compare data bus bit.
1	1	Compare data bus bit. Match on difference.

16.4.2.4 Range Comparisons

When using the AB comparator pair for a range comparison, the data bus can also be used for qualification by using the comparator A data and data mask registers. Furthermore the DBGACTL RW and RWE bits can be used to qualify the range comparison on either a read or a write access. The corresponding DBGBCTL bits are ignored. Similarly when using the CD comparator pair for a range comparison, the data bus can also be used for qualification by using the comparator C data and data mask registers. Furthermore the DBGCCTL RW and RWE bits can be used to qualify the range comparison on either a read or a write access if tagging is not selected. The corresponding DBGDCTL bits are ignored. The SZE and SZ control bits are ignored in range mode. The comparator A and C TAG bits are used to tag range comparisons for the AB and CD ranges respectively. The comparator B and D TAG bits are ignored in range modes. In order for a range comparison using comparators A and B, both COMPEA and COMPEB must be set; to disable range comparisons both must be cleared. Similarly for a range CD comparison, both COMPEC and COMPED must be set. If a range mode is selected SRCA and SRCC select the source (S12X or XGATE), SRCB and SRCD are ignored. The comparator A and C BRK bits are used for the AB and CD ranges respectively, the comparator B and D BRK bits are ignored in range mode. When configured for range comparisons and tagging, the ranges are accurate only to word boundaries.

16.4.2.4.1 Inside Range (CompAC_Addr $\leq$ address $\leq$ CompBD_Addr)

In the Inside Range comparator mode, either comparator pair A and B or comparator pair C and D can be configured for range comparisons by the control register (DBGC2). The match condition requires that a valid match for both comparators happens on the same bus cycle. A match condition on only one

comparator is not valid. An aligned word access which straddles the range boundary will cause a trigger only if the aligned address is inside the range.

16.4.2.4.2 Outside Range (address < CompAC_Addr or address > CompBD_Addr)

In the Outside Range comparator mode, either comparator pair A and B or comparator pair C and D can be configured for range comparisons. A single match condition on either of the comparators is recognized as valid. An aligned word access which straddles the range boundary will cause a trigger only if the aligned address is outside the range.

Outside range mode in combination with tagged triggers can be used to detect if the opcode fetches are from an unexpected range. In forced trigger modes the outside range trigger would typically be activated at any interrupt vector fetch or register access. This can be avoided by setting the upper range limit to $7FFFFF or lower range limit to $000000 respectively.

When comparing the XGATE address bus in outside range mode, the initial vector fetch as determined by the vector contained in the XGATE XGVBR register should be taken into consideration. The XGVBR register and hence vector address can be modified.

16.4.3 Trigger Modes

Trigger modes are used as qualifiers for a state sequencer change of state. The control logic determines the trigger mode and provides a trigger to the state sequencer. The individual trigger modes are described in the following sections.

16.4.3.1 Forced Trigger On Comparator Match

If a forced trigger comparator match occurs, the trigger immediately initiates a transition to the next state sequencer state whereby the corresponding flags in DBGSR are set. The state control register for the current state determines the next state for each trigger. Forced triggers are generated as soon as the matching address appears on the address bus, which in the case of opcode fetches occurs several cycles before the opcode execution. For this reason a forced trigger of an opcode address precedes a tagged trigger at the same address by several cycles.

16.4.3.2 Trigger On Comparator Related Taghit

If a CPU12X or XGATE taghit occurs, a transition to another state sequencer state is initiated and the corresponding DBGSR flags are set. For a comparator related taghit to occur, the S12XDBG must first generate tags based on comparator matches. When the tagged instruction reaches the execution stage of the instruction queue a taghit is generated by the CPU12X/XGATE. The state control register for the current state determines the next state for each trigger.

16.4.3.3 External Tagging Trigger

In external tagging trigger mode, the $\overline{\text{TAGLO}}$ and $\overline{\text{TAGHI}}$ pins (mapped to device pins) are used to tag an instruction. This function can be used as another breakpoint source. When the tagged opcode reaches the execution stage of the instruction queue a transition to the disarmed state0 occurs, ending the debug session

and generating a breakpoint, if breakpoints are enabled. External tagging is only possible in device emulation modes.

16.4.3.4 Trigger On XGATE S/W Breakpoint Request

The XGATE S/W breakpoint request issues a forced breakpoint request to the CPU12X immediately independent of S12XDBG settings and triggers the state sequencer into the disarmed state. Active tracing sessions are terminated immediately, thus if tracing has not yet begun, no trace information is stored. XGATE generated breakpoints are independent of the DBGBRK bits. The XGSBPE bit in DBGC1 determines if the XGATE S/W breakpoint function is enabled. The BDM bit in DBGC1 determines if the XGATE requested breakpoint causes the system to enter BDM Mode or initiate a software interrupt (SWI).

16.4.3.5 TRIG Immediate Trigger

Independent of comparator matches or external tag signals it is possible to initiate a tracing session and/or breakpoint by writing the TRIG bit in DBGC1 to a logic “1”. If configured for begin or mid aligned tracing, this triggers the state sequencer into the Final State, if configured for end alignment, setting the TRIG bit disarms the module, ending the session. If breakpoints are enabled, a forced breakpoint request is issued immediately (end alignment) or when tracing has completed (begin or mid alignment).

16.4.3.6 Trigger Priorities

In case of simultaneous triggers, the priority is resolved according to Table 16-41. The lower priority trigger is suppressed. It is thus possible to miss a lower priority trigger if it occurs simultaneously with a trigger of a higher priority. The trigger priorities described in Table 16-41 dictate that in the case of simultaneous matches, the match on the lower channel number (0,1,2,3) has priority. The SC[3:0] encoding ensures that a match leading to final state has priority over all other matches independent of current state sequencer state. When configured for range modes a simultaneous match of comparators A and C generates an active match0 whilst match2 is suppressed.

If a write access to DBGC1 with the ARM bit position set occurs simultaneously to a hardware disarm from an internal trigger event, then the ARM bit is cleared due to the hardware disarm.

Table 16-42. Trigger Priorities

Priority	Source	Action
Highest	**XGATE**	Immediate forced breakpoint.....(Tracing terminated immediately).
	TRIG	Trigger immediately to final state (begin or mid aligned tracing enabled) Trigger immediately to state 0 (end aligned or no tracing enabled)
	External TAGHI/TAGLO	Enter State0
	Match0 (force or tag hit)	Trigger to next state as defined by state control registers
	Match1 (force or tag hit)	Trigger to next state as defined by state control registers
	Match2 (force or tag hit)	Trigger to next state as defined by state control registers
Lowest	**Match3 (force or tag hit)**	Trigger to next state as defined by state control registers

16.4.4 State Sequence Control

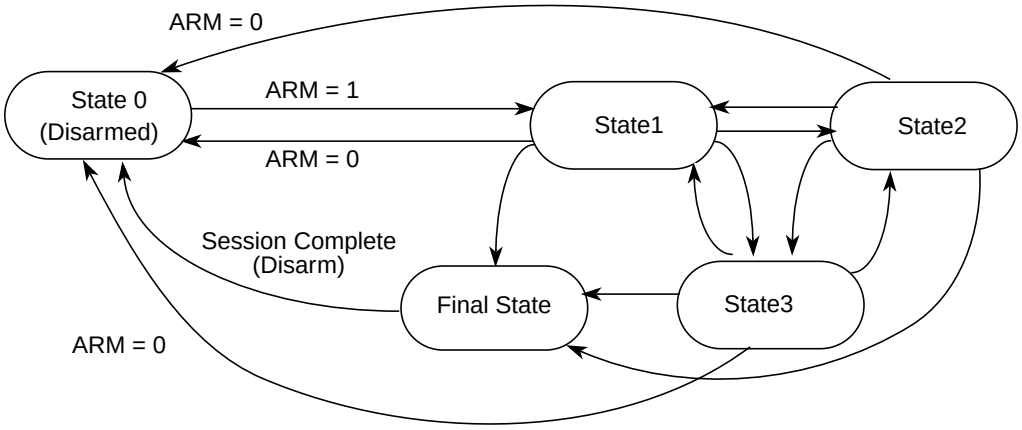


Figure 16-22. State Sequencer Diagram

The state sequencer allows a defined sequence of events to provide a trigger point for tracing of data in the trace buffer. Once the S12XDBG module has been armed by setting the ARM bit in the DBGC1 register, then state1 of the state sequencer is entered. Further transitions between the states are then controlled by the state control registers and depend upon a selected trigger mode condition being met. From Final State the only permitted transition is back to the disarmed state0. Transition between any of the states 1 to 3 is not restricted. Each transition updates the SSF[2:0] flags in DBGSR accordingly to indicate the current state.

Alternatively by setting the TRIG bit in DBGSC1, the state machine can be triggered to state0 or Final State depending on tracing alignment.

A tag hit through $\overline{\text{TAGHI}}/\overline{\text{TAGLO}}$ brings the state sequencer immediately into state0, causes a breakpoint, if breakpoints are enabled, and ends tracing immediately independent of the trigger alignment bits TALIGN[1:0].

Independent of the state sequencer, each comparator channel can be individually configured to generate an immediate breakpoint when a match occurs through the use of the BRK bits in the DBGxCTL registers. Thus it is possible to generate an immediate breakpoint on selected channels, whilst a state sequencer transition can be initiated by a match on other channels. If a debug session is ended by a trigger on a channel with BRK = 1, the state sequencer transitions through Final State for a clock cycle to state0. This is independent of tracing and breakpoint activity, thus with tracing and breakpoints disabled, the state sequencer enters state0 and the debug module is disarmed.

An XGATE S/W breakpoint request, if enabled causes a transition to the State0 and generates a breakpoint request to the CPU12X immediately.

16.4.4.1 Final State

On entering Final State a trigger may be issued to the trace buffer according to the trace position control as defined by the TALIGN field (see Section 16.3.2.3”). If TSOURCE in the trace control register DBGTCR are cleared then the trace buffer is disabled and the transition to Final State can only generate a breakpoint request. In this case or upon completion of a tracing session when tracing is enabled, the ARM

bit in the DBGC1 register is cleared, returning the module to the disarmed state0. If tracing is enabled a breakpoint request can occur at the end of the tracing session. If neither tracing nor breakpoints are enabled then when the final state is reached it returns automatically to state0 and the debug module is disarmed.

16.4.5 Trace Buffer Operation

The trace buffer is a 64 lines deep by 64-bits wide RAM array. The S12XDBG module stores trace information in the RAM array in a circular buffer format. The RAM array can be accessed through a register window (DBGTBH:DBGTBL) using 16-bit wide word accesses. After each complete 64-bit trace buffer line is read, an internal pointer into the RAM is incremented so that the next read will receive fresh information. Data is stored in the format shown in Table 16-42. After each store the counter register bits DBGCNT[6:0] are incremented. Tracing of CPU12X activity is disabled when the BDM is active but tracing of XGATE activity is still possible. Reading the trace buffer whilst the DBG is armed returns invalid data and the trace buffer pointer is not incremented.

16.4.5.1 Trace Trigger Alignment

Using the TALIGN bits (see Section 16.3.2.3”) it is possible to align the trigger with the end, the middle, or the beginning of a tracing session.

If End or Mid tracing is selected, tracing begins when the ARM bit in DBGC1 is set and State1 is entered. The transition to Final State if End is selected signals the end of the tracing session. The transition to Final State if Mid is selected signals that another 32 lines will be traced before ending the tracing session. Tracing with Begin-Trigger starts at the opcode of the trigger.

16.4.5.1.1 Storing with Begin-Trigger

Storing with Begin-Trigger, data is not stored in the Trace Buffer until the Final State is entered. Once the trigger condition is met the S12XDBG module will remain armed until 64 lines are stored in the Trace Buffer. If the trigger is at the address of the change-of-flow instruction the change of flow associated with the trigger will be stored in the Trace Buffer. Using Begin-trigger together with tagging, if the tagged instruction is about to be executed then the trace is started. Upon completion of the tracing session the breakpoint is generated, thus the breakpoint does not occur at the tagged instruction boundary.

16.4.5.1.2 Storing with Mid-Trigger

Storing with Mid-Trigger, data is stored in the Trace Buffer as soon as the S12XDBG module is armed. When the trigger condition is met, another 32 lines will be traced before ending the tracing session, irrespective of the number of lines stored before the trigger occurred, then the S12XDBG module is disarmed and no more data is stored. Using Mid-trigger with tagging, if the tagged instruction is about to be executed then the trace is continued for another 32 lines. Upon tracing completion the breakpoint is generated, thus the breakpoint does not occur at the tagged instruction boundary.

16.4.5.1.3 Storing with End-Trigger

Storing with End-Trigger, data is stored in the Trace Buffer until the Final State is entered, at which point the S12XDBG module will become disarmed and no more data will be stored. If the trigger is at the address of a change of flow instruction the trigger event will not be stored in the Trace Buffer.

16.4.5.2 Trace Modes

The S12XDBG module can operate in four trace modes. The mode is selected using the TRCMOD bits in the DBGTCR register. In each mode tracing of XGATE or CPU12X information is possible. The source for the trace is selected using the TSOURCE bits in the DBGTCR register. The modes are described in the following subsections. The trace buffer organization is shown in Table 16-42.

16.4.5.2.1 Normal Mode

In Normal Mode, change of flow (COF) program counter (PC) addresses will be stored.

COF addresses are defined as follows for the CPU12X:

- Source address of taken conditional branches (long, short, bit-conditional, and loop primitives)
- Destination address of indexed JMP, JSR, and CALL instruction.
- Destination address of RTI, RTS, and RTC instructions
- Vector address of interrupts, except for SWI and BDM vectors

LBRA, BRA, BSR, BGND as well as non-indexed JMP, JSR, and CALL instructions are not classified as change of flow and are not stored in the trace buffer.

COF addresses are defined as follows for the XGATE:

- Source address of taken conditional branches
- Destination address of indexed JAL instructions.
- First XGATE code address in a thread

Change-of-flow addresses stored include the full 23-bit address bus of CPU12X, the 16-bit address bus for the XGATE module and an information byte, which contains a source/destination bit to indicate whether the stored address was a source address or destination address.

NOTE

When an CPU12X COF instruction with destination address is executed, the destination address is stored to the trace buffer on instruction completion, indicating the COF has taken place. If an interrupt occurs simultaneously then the next instruction carried out is actually from the interrupt service routine. The instruction at the destination address of the original program flow gets exectuted after the interrupt service routine.

In the following example an IRQ interrupt occurs during execution of the indexed JMP at address MARK1. The BRN at the destination (SUB_1) is not executed until after the IRQ service routine but the destination address is entered into the trace buffer to indicate that the indexed JMP COF has taken place.

```
         LDX     #SUB_1
MARK1    JMP     0,X                  ; IRQ interrupt occurs during execution of this
MARK2    NOP                          ;

SUB_1    BRN     *                    ; JMP Destination address TRACE BUFFER ENTRY 1
                                      ; RTI Destination address TRACE BUFFER ENTRY 3
         NOP                          ;
ADDR1    DBNE    A,PART5              ; Source address TRACE BUFFER ENTRY 4

IRQ_ISR  LDAB    #$F0                 ; IRQ Vector $FFF2 = TRACE BUFFER ENTRY 2
         STAB    VAR_C1
         RTI                          ;
```

The execution flow taking into account the IRQ is as follows

```
         LDX     #SUB_1
MARK1    JMP     0,X                  ;
IRQ_ISR  LDAB    #$F0                 ;
         STAB    VAR_C1
         RTI                          ;
SUB_1    BRN     *
         NOP                          ;
ADDR1    DBNE    A,PART5              ;
```

16.4.5.2.2 Loop1 Mode

Loop1 Mode, similarly to Normal Mode also stores only COF address information to the trace buffer, it however allows the filtering out of redundant information.

The intent of Loop1 Mode is to prevent the Trace Buffer from being filled entirely with duplicate information from a looping construct such as delays using the DBNE instruction or polling loops using BRSET/BRCLR instructions. Immediately after address information is placed in the Trace Buffer, the S12XDBG module writes this value into a background register. This prevents consecutive duplicate address entries in the Trace Buffer resulting from repeated branches.

Loop1 Mode only inhibits consecutive duplicate source address entries that would typically be stored in most tight looping constructs. It does not inhibit repeated entries of destination addresses or vector addresses, since repeated entries of these would most likely indicate a bug in the user's code that the S12XDBG module is designed to help find.

NOTE

In certain very tight loops, the source address will have already been fetched again before the background comparator is updated. This results in the source address being stored twice before further duplicate entries are suppressed. This condition occurs with branch-on-bit instructions when the branch is fetched by the first P-cycle of the branch or with loop-construct instructions in which the branch is fetched with the first or second P cycle. See examples below:

```
LOOP     INX                            ; 1-byte instruction fetched by 1st P-cycle of BRCLR
         BRCLR       CMPTMP,#$0c, LOOP  ; the BRCLR instruction also will be fetched by 1st
                                        ; P-cycle of BRCLR

LOOP2    BRN     *                      ; 2-byte instruction fetched by 1st P-cycle of DBNE
         NOP                            ; 1-byte instruction fetched by 2nd P-cycle of DBNE
         DBNE        A,LOOP2            ; this instruction also fetched by 2nd P-cycle of DBNE
```

16.4.5.2.3 Detail Mode

In Detail Mode, address and data for all memory and register accesses is stored in the trace buffer. In the case of XGATE tracing this means that initialization of the R1 register during a vector fetch is not traced. This mode is intended to supply additional information on indexed, indirect addressing modes where storing only the destination address would not provide all information required for a user to determine where the code is in error. This mode also features information byte entries to the trace buffer, for each address byte entry. The information byte indicates the size of access (word or byte) and the type of access (read or write).

When tracing CPU12X activity in Detail Mode, all cycles are traced except those when the CPU12X is either in a free or opcode fetch cycle. In this mode the XGATE program counter is also traced to provide a snapshot of the XGATE activity. CXINF information byte bits indicate the type of XGATE activity occurring at the time of the trace buffer entry. When tracing CPU12X activity alone in Detail Mode, the address range can be limited to a range specified by the TRANGE bits in DBGTCR. This function uses comparators C and D to define an address range inside which CPU12X activity should be traced (see Table 16-42). Thus the traced CPU12X activity can be restricted to particular register range accesses.

When tracing XGATE activity in Detail Mode, all load and store cycles are traced. Additionally the CPU12X program counter is stored at the time of the XGATE trace buffer entry to provide a snapshot of CPU12X activity.

16.4.5.2.4 Pure PC Mode

In Pure PC Mode, tracing from the CPU the PC addresses of all executed opcodes, including illegal opcodes, are stored. In Pure PC Mode, tracing from the XGATE the PC addresses of all executed opcodes are stored.

16.4.5.3 Trace Buffer Organization

Referring to Table 16-42. An X prefix denotes information from the XGATE module, a C prefix denotes information from the CPU12X. ADRH, ADRM, ADRL denote address high, middle and low byte respectively. INF bytes contain control information (R/W, S/D etc.). The numerical suffix indicates which tracing step. The information format for Loop1 Mode and PurePC Mode is the same as that of Normal Mode. Whilst tracing from XGATE or CPU12X only, in Normal or Loop1 modes each array line contains 2 data entries, thus in this case the DBGCNT[0] is incremented after each separate entry. In Detail mode DBGCNT[0] remains cleared whilst the other DBGCNT bits are incremented on each trace buffer entry.

XGATE and CPU12X COFs occur independently of each other and the profile of COFs for the two sources is totally different. When both sources are being traced in Normal or Loop1 mode, for each COF from one source, there may be many COFs from the other source, depending on user code. COF events could occur far from each other in the time domain, on consecutive cycles or simultaneously. When a COF occurs in either source (S12X or XGATE) a trace buffer entry is made and the corresponding CDV or XDV bit is set. The current PC of the other source is simultaneously stored to the trace buffer even if no COF has occurred, in which case CDV/XDV remains cleared indicating the address is not associated with a COF, but is simply a snapshot of the PC contents at the time of the COF from the other source.

Single byte data accesses in Detail Mode are always stored to the low byte of the trace buffer (CDATAL or XDATAL) and the high byte is cleared. When tracing word accesses, the byte at the lower address is always stored to trace buffer byte3 and the byte at the higher address is stored to byte2

Table 16-43. Trace Buffer Organization

Mode	8-Byte Wide Word Buffer							
	7	6	5	4	3	2	1	0
XGATE Detail	CXINF1	CADRH1	CADRM1	CADRL1	XDATAH1	XDATAL1	XADRM1	XADRL1
	CXINF2	CADRH2	CADRM2	CADRL2	XDATAH2	XDATAL2	XADRM2	XADRL2
CPU12X Detail	CXINF1	CADRH1	CADRM1	CADRL1	CDATAH1	CDATAL1	XADRM1	XADRL1
	CXINF2	CADRH2	CADRM2	CADRL2	CDATAH2	CDATAL2	XADRM2	XADRL2
Both Other Modes	XINF0		XPCM0	XPCL0	CINF0	CPCH0	CPCM0	CPCL0
	XINF1		XPCM1	XPCL1	CINF1	CPCH1	CPCM1	CPCL1
XGATE Other Modes	XINF1		XPCM1	XPCL1	XINF0		XPCM0	XPCL0
	XINF3		XPCM3	XPCL3	XINF2		XPCM2	XPCL2
CPU12X Other Modes	CINF1	CPCH1	CPCM1	CPCL1	CINF0	CPCH0	CPCM0	CPCL0
	CINF3	CPCH3	CPCM3	CPCL3	CINF2	CPCH2	CPCM2	CPCL2

16.4.5.3.1 Information Byte Organization

The format of the control information byte is dependent upon the active trace mode as described below. In Normal, Loop1, or Pure PC modes tracing of XGATE activity, XINF is used to store control information. In Normal, Loop1, or Pure PC modes tracing of CPU12X activity, CINF is used to store control information. In Detail Mode, CXINF contains the control information

XGATE Information Byte

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
XSD	XSOT	XCOT	XDV	0	0	0	0

Figure 16-23. XGATE Information Byte XINF

Table 16-44. XINF Field Descriptions

Field	Description
7 XSD	**Source Destination Indicator** — This bit indicates if the corresponding stored address is a source or destination address. This is only used in Normal and Loop1 mode tracing. 0 Source address 1 Destination address or Start of Thread or Continuation of Thread
6 XSOT	**Source Of Thread Indicator** — This bit indicates that the corresponding stored address is a start of thread address. This is only used in Normal and Loop1 mode tracing. **NOTE. This bit only has effect on devices where the XGATE module supports multiple interrupt levels**. 0 Stored address not from a start of thread 1 Stored address from a start of thread
5 XCOT	**Continuation Of Thread Indicator** — This bit indicates that the corresponding stored address is the first address following a return from a higher priority thread. This is only used in Normal and Loop1 mode tracing. **NOTE. This bit only has effect on devices where the XGATE module supports multiple interrupt levels.** 0 Stored address not from a continuation of thread 1 Stored address from a continuation of thread
4 XDV	**Data Invalid Indicator** — This bit indicates if the trace buffer entry is invalid. It is only used when tracing from both sources in Normal, Loop1 and Pure PC modes, to indicate that the XGATE trace buffer entry is valid. 0 Trace buffer entry is invalid 1 Trace buffer entry is valid

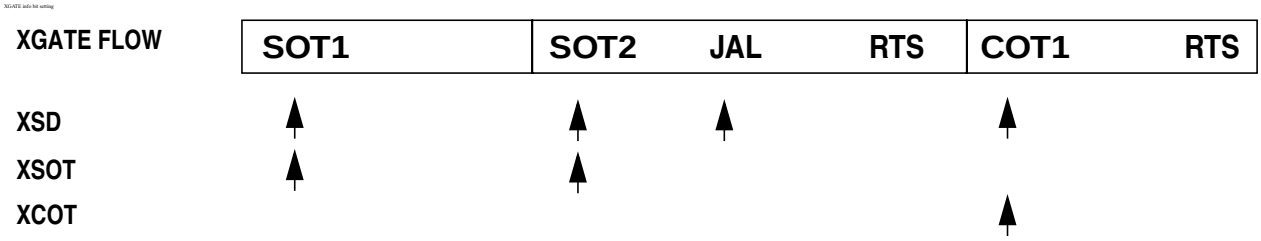


Figure 16-24. XGATE info bit setting

Figure 16-24 indicates the XGATE information bit setting when switching between threads, the initial thread starting at SOT1 and continuing at COT1 after the higher priority thread2 has ended.

CPU12X Information Byte

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CSD	CVA	0	CDV	0	0	0	0

Figure 16-25. CPU12X Information Byte CINF

Table 16-45. CINF Field Descriptions

Field	Description
7 CSD	**Source Destination Indicator** — This bit indicates if the corresponding stored address is a source or destination address. This is only used in Normal and Loop1 mode tracing. 0 Source address 1 Destination address
6 CVA	**Vector Indicator** — This bit indicates if the corresponding stored address is a vector address.. Vector addresses are destination addresses, thus if CVA is set, then the corresponding CSD is also set. This is only used in Normal and Loop1 mode tracing. This bit has no meaning in Pure PC mode. 0 Indexed jump destination address 1 Vector destination address
4 CDV	**Data Invalid Indicator** — This bit indicates if the trace buffer entry is invalid. It is only used when tracing from both sources in Normal, Loop1 and Pure PC modes, to indicate that the CPU12X trace buffer entry is valid. 0 Trace buffer entry is invalid 1 Trace buffer entry is valid

CXINF Information Byte

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CFREE	CSZ	CRW	COCF	XACK	XSZ	XRW	XOCF

Figure 16-26. Information Byte CXINF

This describes the format of the information byte used only when tracing from CPU12X or XGATE in Detail Mode. When tracing from the CPU12X in Detail Mode, information is stored to the trace buffer on all cycles except opcode fetch and free cycles. The XGATE entry stored on the same line is a snapshot of the XGATE program counter. In this case the CSZ and CRW bits indicate the type of access being made by the CPU12X, whilst the XACK and XOCF bits indicate if the simultaneous XGATE cycle is a free cycle (no bus acknowledge) or opcode fetch cycle. Similarly when tracing from the XGATE in Detail Mode, information is stored to the trace buffer on all cycles except opcode fetch and free cycles. The CPU12X entry stored on the same line is a snapshot of the CPU12X program counter. In this case the XSZ and XRW bits indicate the type of access being made by the XGATE, whilst the CFREE and COCF bits indicate if the simultaneous CPU12X cycle is a free cycle or opcode fetch cycle.

Table 16-46. CXINF Field Descriptions

Field	Description
7 CFREE	**CPU12X Free Cycle Indicator** — This bit indicates if the stored CPU12X address corresponds to a free cycle. This bit only contains valid information when tracing the XGATE accesses in Detail Mode. 0 Stored information corresponds to free cycle 1 Stored information does not correspond to free cycle

Table 16-46. CXINF Field Descriptions (continued)

Field	Description
6 CSZ	**Access Type Indicator** — This bit indicates if the access was a byte or word size access.This bit only contains valid information when tracing CPU12X activity in Detail Mode. 0 Word Access 1 Byte Access
5 CRW	**Read Write Indicator** — This bit indicates if the corresponding stored address corresponds to a read or write access. This bit only contains valid information when tracing CPU12X activity in Detail Mode. 0 Write Access 1 Read Access
4 COCF	**CPU12X Opcode Fetch Indicator** — This bit indicates if the stored address corresponds to an opcode fetch cycle. This bit only contains valid information when tracing the XGATE accesses in Detail Mode. 0 Stored information does not correspond to opcode fetch cycle 1 Stored information corresponds to opcode fetch cycle
3 XACK	**XGATE Access Indicator** — This bit indicates if the stored XGATE address corresponds to a free cycle. This bit only contains valid information when tracing the CPU12X accesses in Detail Mode. 0 Stored information corresponds to free cycle 1 Stored information does not correspond to free cycle
2 XSZ	**Access Type Indicator** — This bit indicates if the access was a byte or word size access. This bit only contains valid information when tracing XGATE activity in Detail Mode. 0 Word Access 1 Byte Access
1 XRW	**Read Write Indicator** — This bit indicates if the corresponding stored address corresponds to a read or write access. This bit only contains valid information when tracing XGATE activity in Detail Mode. 0 Write Access 1 Read Access
0 XOCF	**XGATE Opcode Fetch Indicator** — This bit indicates if the stored address corresponds to an opcode fetch cycle.This bit only contains valid information when tracing the CPU12X accesses in Detail Mode. 0 Stored information does not correspond to opcode fetch cycle 1 Stored information corresponds to opcode fetch cycle

16.4.5.4 Reading Data from Trace Buffer

The data stored in the Trace Buffer can be read using either the background debug module (BDM) module, the XGATE or the CPU12X provided the S12XDBG module is not armed, is configured for tracing and the system not secured. When the ARM bit is written to 1 the trace buffer is locked to prevent reading. The trace buffer can only be unlocked for reading by an aligned word write to DBGTB when the module is disarmed.

The Trace Buffer can only be read through the DBGTB register using aligned word reads, any byte or misaligned reads return 0 and do not cause the trace buffer pointer to increment to the next trace buffer address. The Trace Buffer data is read out first-in first-out. By reading CNT in DBGCNT the number of valid 64-bit lines can be determined. DBGCNT will not decrement as data is read.

Whilst reading an internal pointer is used to determine the next line to be read. After a tracing session, the pointer points to the oldest data entry, thus if no overflow has occurred, the pointer points to line0, otherwise it points to the line with the oldest entry. The pointer is initialized by each aligned write to

DBGTBH to point to the oldest data again. This enables an interrupted trace buffer read sequence to be easily restarted from the oldest data entry.

The least significant word of each 64-bit wide array line is read out first. This corresponds to the bytes 1 and 0 of Table 16-42. The bytes containing invalid information (shaded in Table 16-42) are also read out.

Reading the Trace Buffer while the S12XDBG module is armed will return invalid data and no shifting of the RAM pointer will occur.

16.4.5.5 Trace Buffer Reset State

The Trace Buffer contents are not initialized by a system reset. Thus should a system reset occur, the trace session information from immediately before the reset occurred can be read out. The DBGCNT bits are not cleared by a system reset. Thus should a reset occur, the number of valid lines in the trace buffer is indicated by DBGCNT. The internal pointer to the current trace buffer address is initialized by unlocking the trace buffer thus points to the oldest valid data even if a reset occurred during the tracing session. Generally debugging occurrences of system resets is best handled using mid or end trigger alignment since the reset may occur before the trace trigger, which in the begin trigger alignment case means no information would be stored in the trace buffer.

16.4.6 Tagging

A tag follows program information as it advances through the instruction queue. When a tagged instruction reaches the head of the queue a tag hit occurs and triggers the state sequencer.

Each comparator control register features a TAG bit, which controls whether the comparator match will cause a trigger immediately or tag the opcode at the matched address. If a comparator is enabled for tagged comparisons, the address stored in the comparator match address registers must be an opcode address for the trigger to occur.

Both CPU12X and XGATE opcodes can be tagged with the comparator register TAG bits.

Using Begin trigger together with tagging, if the tagged instruction is about to be executed then the transition to the next state sequencer state occurs. If the transition is to the Final State, tracing is started. Only upon completion of the tracing session can a breakpoint be generated. Similarly using Mid trigger with tagging, if the tagged instruction is about to be executed then the trace is continued for another 32 lines. Upon tracing completion the breakpoint is generated. Using End trigger, when the tagged instruction is about to be executed and the next transition is to Final State then a breakpoint is generated immediately, before the tagged instruction is carried out.

R/W monitoring is not useful for tagged operations since the trigger occurs based on the tagged opcode reaching the execution stage of the instruction queue. Similarly access size (SZ) monitoring and data bus monitoring is not useful if tagged triggering is selected, since the tag is attached to the opcode at the matched address and is not dependent on the data bus nor on the size of access. Thus these bits are ignored if tagged triggering is selected.

When configured for range comparisons and tagging, the ranges are accurate only to word boundaries.

S12X tagging is disabled when the BDM becomes active. XGATE tagging is possible when the BDM is active.

16.4.6.1 External Tagging using $\overline{\text{TAGHI}}$ and $\overline{\text{TAGLO}}$

External tagging using the external $\overline{\text{TAGHI}}$ and $\overline{\text{TAGLO}}$ pins can only be used to tag CPU12X opcodes; tagging of XGATE code using these pins is not possible. An external tag triggers the state sequencer into state0 when the tagged opcode reaches the execution stage of the instruction queue.

The pins operate independently, thus the state of one pin does not affect the function of the other. External tagging is possible in emulation modes only. The presence of logic level 0 on either pin at the rising edge of the external clock (ECLK) performs the function indicated in the Table 16-46. It is possible to tag both bytes of an instruction word. If a taghit occurs, a breakpoint can be generated as defined by the DBGBRK and BDM bits in DBGC1. Each time $\overline{\text{TAGHI}}$ or $\overline{\text{TAGLO}}$ are low on the rising edge of ECLK, the old tag is replaced by a new one.

Table 16-47. Tag Pin Function

$\overline{\text{TAGHI}}$	$\overline{\text{TAGLO}}$	Tag
1	1	No tag
1	0	Low byte
0	1	High byte
0	0	Both bytes

16.4.6.2 Unconditional Tagging Function

In emulation modes a low assertion of PE5/$\overline{\text{TAGLO}}$/MODA in the 7th or 8th bus cycle after reset enables the unconditional tagging function, allowing immediate tagging via $\overline{\text{TAGHI}}$/$\overline{\text{TAGLO}}$ with breakpoint to BDM independent of the ARM, BDM and DBGBRK bits. Conversely these bits are not affected by unconditional tagging. The unconditional tagging function remains enabled until the next reset. This function allows an immediate entry to BDM in emulation modes before user code execution. The $\overline{\text{TAGLO}}$ assertion must be in the 7th or 8th bus cycle following the end of reset, whereby the prior $\overline{\text{RESET}}$ pin assertion lasts the full 192 bus cycles.

16.4.7 Breakpoints

Breakpoints can be generated as follows.

- Through XGATE software breakpoint requests.
- From comparator channel triggers to final state.
- Using software to write to the TRIG bit in the DBGC1 register.
- From taghits generated using the external $\overline{\text{TAGHI}}$ and $\overline{\text{TAGLO}}$ pins.

Breakpoints generated by the XGATE module or via the BDM BACKGROUND command have no affect on the CPU12X in STOP or WAIT mode.

16.4.7.1 XGATE Software Breakpoints

The XGATE software breakpoint instruction BRK can request an CPU12X breakpoint, via the S12XDBG module. In this case, if the XGSBPE bit is set, the S12XDBG module immediately generates a forced breakpoint request to the CPU12X, the state sequencer is returned to state0 and tracing, if active, is terminated. If configured for BEGIN trigger and tracing has not yet been triggered from another source, the trace buffer contains no information. Breakpoint requests from the XGATE module do not depend upon the state of the DBGBRK or ARM bits in DBGC1. They depend solely on the state of the XGSBPE and BDM bits. Thus it is not necessary to ARM the DBG module to use XGATE software breakpoints to generate breakpoints in the CPU12X program flow, but it is necessary to set XGSBPE. Furthermore, if a breakpoint to BDM is required, the BDM bit must also be set. When the XGATE requests an CPU12X breakpoint, the XGATE program flow stops by default, independent of the S12XDBG module.

16.4.7.2 Breakpoints From Internal Comparator Channel Final State Triggers

Breakpoints can be generated when internal comparator channels trigger the state sequencer to the Final State. If configured for tagging, then the breakpoint is generated when the tagged opcode reaches the execution stage of the instruction queue.

If a tracing session is selected by TSOURCE, breakpoints are requested when the tracing session has completed, thus if Begin or Mid aligned triggering is selected, the breakpoint is requested only on completion of the subsequent trace (see Table 16-47). If no tracing session is selected, breakpoints are requested immediately.

If the BRK bit is set on the triggering channel, then the breakpoint is generated immediately independent of tracing trigger alignment.

Table 16-48. Breakpoint Setup For Both XGATE and CPU12X Breakpoints

BRK	TALIGN	DBGBRK[n]	Breakpoint Alignment
0	00	0	Fill Trace Buffer until trigger (no breakpoints — keep running)
0	00	1	Fill Trace Buffer until trigger, then breakpoint request occurs
0	01	0	Start Trace Buffer at trigger (no breakpoints — keep running)
0	01	1	Start Trace Buffer at trigger A breakpoint request occurs when Trace Buffer is full
0	10	0	Store a further 32 Trace Buffer line entries after trigger (no breakpoints — keep running)
0	10	1	Store a further 32 Trace Buffer line entries after trigger Request breakpoint after the 32 further Trace Buffer entries
1	00,01,10	1	Terminate tracing and generate breakpoint immediately on trigger
1	00,01,10	0	Terminate tracing immediately on trigger
x	11	x	Reserved

16.4.7.3 Breakpoints Generated Via The TRIG Bit

If a TRIG triggers occur, the Final State is entered. If a tracing session is selected by TSOURCE, breakpoints are requested when the tracing session has completed, thus if Begin or Mid aligned triggering is selected, the breakpoint is requested only on completion of the subsequent trace (see Table 16-47). If no tracing session is selected, breakpoints are requested immediately. TRIG breakpoints are possible even if the S12XDBG module is disarmed.

16.4.7.4 Breakpoints Via TAGHI Or TAGLO Pin Taghits

Tagging using the external $\overline{\text{TAGHI}}$/$\overline{\text{TAGLO}}$ pins always ends the session immediately at the tag hit. It is always end aligned, independent of internal channel trigger alignment configuration.

16.4.7.5 S12XDBG Breakpoint Priorities

XGATE software breakpoints have the highest priority. Active tracing sessions are terminated immediately.

If a TRIG trigger occurs after Begin or Mid aligned tracing has already been triggered by a comparator instigated transition to Final State, then TRIG no longer has an effect. When the associated tracing session is complete, the breakpoint occurs. Similarly if a TRIG is followed by a subsequent trigger from a comparator channel, it has no effect, since tracing has already started.

If a comparator tag hit occurs simultaneously with an external $\overline{\text{TAGHI}}$/$\overline{\text{TAGLO}}$ hit, the state sequencer enters state0. TAGHI/TAGLO triggers are always end aligned, to end tracing immediately, independent of the tracing trigger alignment bits TALIGN[1:0].

16.4.7.5.1 S12XDBG Breakpoint Priorities And BDM Interfacing

Breakpoint operation is dependent on the state of the S12XBDM module. If the S12XBDM module is active, the CPU12X is executing out of BDM firmware and S12X breakpoints are disabled. In addition, while executing a BDM TRACE command, tagging into BDM is disabled. If BDM is not active, the breakpoint will give priority to BDM requests over SWI requests if the breakpoint coincides with a SWI instruction in the user's code. On returning from BDM, the SWI from user code gets executed.

Table 16-49. Breakpoint Mapping Summary

DBGBRK[1] (DBGC1[3])	BDM Bit (DBGC1[4])	BDM Enabled	BDM Active	S12X Breakpoint Mapping
0	X	X	X	No Breakpoint
1	0	X	0	Breakpoint to SWI
1	0	X	1	No Breakpoint
1	1	0	X	Breakpoint to SWI
1	1	1	0	Breakpoint to BDM
1	1	1	1	No Breakpoint

BDM cannot be entered from a breakpoint unless the ENABLE bit is set in the BDM. If entry to BDM via a BGND instruction is attempted and the ENABLE bit in the BDM is cleared, the CPU12X actually

executes the BDM firmware code. It checks the ENABLE and returns if ENABLE is not set. If not serviced by the monitor then the breakpoint is re-asserted when the BDM returns to normal CPU12X flow.

If the comparator register contents coincide with the SWI/BDM vector address then an SWI in user code and DBG breakpoint could occur simultaneously. The CPU12X ensures that BDM requests have a higher priority than SWI requests. Returning from the BDM/SWI service routine care must be taken to avoid re triggering a breakpoint.

NOTE

When program control returns from a tagged breakpoint using an RTI or BDM GO command without program counter modification it will return to the instruction whose tag generated the breakpoint. To avoid re triggering a breakpoint at the same location reconfigure the S12XDBG module in the SWI routine, if configured for an SWI breakpoint, or over the BDM interface by executing a TRACE command before the GO to increment the program flow past the tagged instruction.

An XGATE software breakpoint is forced immediately, the tracing session terminated and the XGATE module execution stops. The user can thus determine if an XGATE breakpoint has occurred by reading out the XGATE program counter over the BDM interface.

Chapter 17
Memory Protection Unit (S12XMPUV2)

Revision History

Table 17-1. Revision History

Version Number	Revision Date	Effective Date	Author	Description of Changes
01.04	14 Sep 2005			- added note to only use the CPU to clear the AE flag. - added disclaimer to avoid changing descriptors while they are in use because of other bus-masters doing accesses
01.05	14 Mar 2005			- clarified that interrupt generation is independent of AEF bit state - corrected preliminary statement about execution of violating accesses
02.00	03 Aug 2006			- new spec version: MPU can now be configured for descriptor granularity and for which address range to cover

17.1 Introduction

The MPU module provides basic functionality required to protect memory mapped resources from undesired accesses. Multiple address range comparators compare memory accesses against eight memory protection descriptors located in the MPU module to determine if each access is valid or not. The comparison is sensitive to which bus master generates the access and the type of the access.

The MPU module can be used to isolate memory ranges accessible by different bus masters. It can be also be used by an operating system or software kernel to isolate the regions of memory “legally” available to specific software tasks, with the kernel re-configuring the task specific memory protection descriptors in supervisor state during task-switching.

17.1.1 Preface

The following terms and abbreviations are used in the document.

Table 17-2. Terminology

Term	Meaning
MCU	Micro-Controller Unit
MPU	Memory Protection Unit
CPU	S12X Central Processing Unit (see S12XCPU Reference Manual)
XGATE	XGATE Co-processor (see XGATE chapter)

Term	Meaning
supervisor state	refers to the supervisor state of the S12XCPU (see S12XCPU Reference Manual)
user state	refers to the user state of the S12XCPU (see S12XCPU Reference Manual)

17.1.2 Overview

The MPU module monitors the bus activity of each bus master. The data describing each access is fed into multiple address range comparators. The output of the comparators is used to determine if a particular access is allowed or represents an access violation. If an access violation caused by the S12X CPU is detected, the MPU module raises an access violation interrupt. If the MPU detects an access violation caused by a bus master other than the S12X CPU, it flags an access error condition to the respective master. In addition to the restrictions defined for memory ranges in the MPU descriptors, accesses to memory inside the MPU address range which are not covered by any MPU descriptor (even read accesses!) are considered access violations.

Figure 17-1 shows a block diagram of the MPU module.

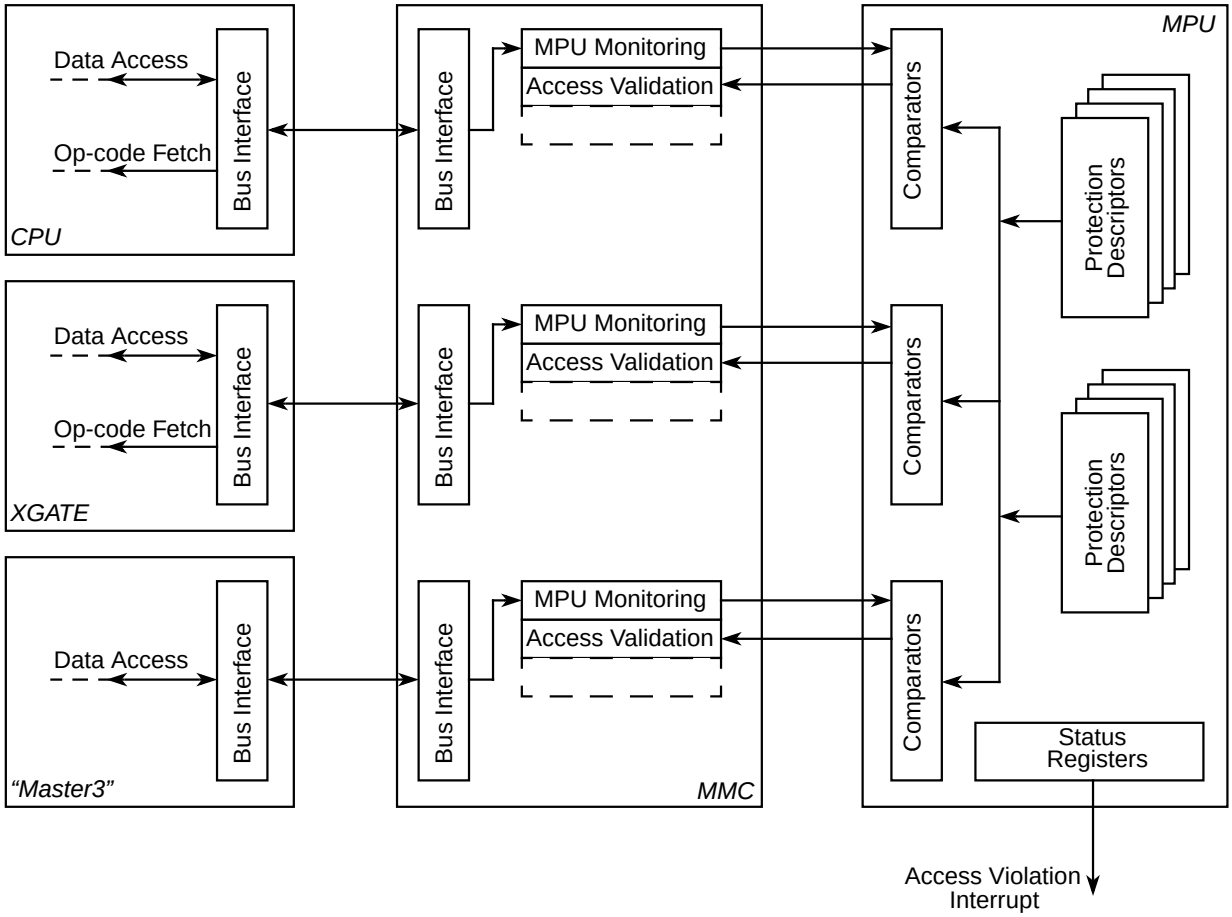


Figure 17-1. Block Diagram

17.1.3 Features

- Protects memory from undesired accesses coming from up to 3 bus masters[1]
- Four memory protection descriptors
  - each descriptor can cover the full MPU address range[2] in the global memory map (up to 8 MBytes)
  - the minimum granularity for each descriptor is 8 Bytes[3]
- Each descriptor can be configured to allow one of four types of access privilege for the defined memory region
  - Bus master has full access (read, write and execute enabled)
  - Bus master can read and execute (write illegal)
  - Bus master can read and write (execution illegal)
  - Bus master can only read (write and execution illegal)
- Accesses to memory in the MPU address range[2] which are not covered by any protection descriptor will cause an access violation

17.1.4 Modes of Operation

The MPU module can be used in all MCU modes.

17.2 External Signal Description

The MPU module has no external signals.

17.3 Memory Map and Register Definition

This section provides a detailed description of address space and registers used by the MPU module.

1. Master 3 can be implemented or left out depending the chip configuration. Please refer to the SoC guide for information about the availability and function of Master 3.
2. The MPU address range is a configuration option defined at SoC level. Please refer to the MCU top-level chapter for details.
3. The MPU descriptor granularity is a configuration option defined at SoC level. Please refer to the MCU top-level chapter for details.

17.3.1 Register Descriptions

This section describes in address order all the MPU registers and their individual bits.

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000 MPUFLG	R	AEF	WPF	NEXF	0	0	0	0	SVSF
	W								
0x0001 MPUASTAT0	R	0	ADDR[22:16]						
	W								
0x0002 MPUASTAT1	R	ADDR[15:8]							
	W								
0x0003 MPUASTAT2	R	ADDR[7:0]							
	W								
0x0004 Reserved	R	0	0	0	0	0	0	0	0
	W								
0x0005 MPUSEL	R	SVSEN	0	0	0	0	SEL[2:0]		
	W								
0x0006 MPUDESC0(1)	R	MSTR0	MSTR1	MSTR2	MSTR3	LOW_ADDR[22:19]			
	W								
0x0007 MPUDESC1[1]	R	LOW_ADDR[18:11]							
	W								
0x0008 MPUDESC2[1]	R	LOW_ADDR[10:3]							
	W								
0x0009 MPUDESC3[1]	R	WP	NEX	0	0	HIGH_ADDR[22:19]			
	W								
0x000A MPUDESC4[1]	R	HIGH_ADDR[18:11]							
	W								
0x000B MPUDESC5[1]	R	HIGH_ADDR[10:3]							
	W								

= Unimplemented or Reserved

1. The module addresses 0x0006–0x000B represent a window in the register map through which different descriptor registers are visible.

Figure 17-2. MPU Register Summary

17.3.1.1 MPU Flag Register (MPUFLG)

Address: Module Base + 0x0000

	7	6	5	4	3	2	1	0
R	AEF	WPF	NEXF	0	0	0	0	SVSF
W								
Reset	0	0	0	0	0	0	0	0

Figure 17-3. MPU Flag Register (MPUFLG)

Read: Anytime

Write: Write of 1 clears flag, write of 0 ignored

Table 17-3. MPUFLG Field Descriptions

Field	Description
7 AEF	**Access Error Flag** — This bit is the CPU access error interrupt flag. It is set if a CPU access violation has occurred. At the same time this bit is set, all the other status flags in this register and the access violation address bits in the MPUASTATn registers are captured. Clear this flag by writing a one. **Note:** If a CPU access error is flagged and both the WPF bit and the NEXF bit are zero, the access violation was caused by an access to memory in the MPU address range which is not covered by the MPU descriptors. **Note:** While this bit is set, the CPU in supervisor state (“Master 0”) can read from and write to the peripheral register space even if there is no memory protection descriptor explicitly allowing this. This is to prevent the case that the CPU cannot clear the AEF bit if the registers are write protected for the CPU in supervisor state. **Note:** This bit should only be cleared by an access from the S12X CPU. Otherwise, when using one of the other masters (such as the XGATE) to clear this bit, the status flags and the address status registers may not get updated correctly if a CPU access causes a violation in the same bus cycle.
6 WPF	**Write-Protect Violation Flag** — This flag is set if the current CPU access violation has occurred because of an attempt to write to memory configured as read-only. The WPF bit is read-only; it will be automatically updated when the next access violation is flagged with the AEF bit.
5 NEXF	**No-Execute Violation Flag** — This bit is set if the current CPU access violation has occurred because of an attempt to fetch code from memory configured as No-Execute. The NEXF bit is read-only; it will be automatically updated when the next access violation is flagged with the AEF bit.
0 SVSF	**Supervisor State Flag** — This bit is set if the current CPU access violation occurred while the CPU was in supervisor state. This bit is cleared if the current CPU access violation occurred while the CPU was in user state. The supervisor state flag is read-only; it will be automatically updated when the next CPU access violation is flagged with the AEF bit.

If the AEF bit is set further violations are not captured into the MPU status registers. The status of the AEF bit has no effect on the access restrictions, i.e. access restrictions for all masters are still enforced if the AEF bit is set. Also, the non-maskable hardware interrupt for violating accesses coming from the S12X CPU is generated regardless of the state of the AEF bit.

17.3.1.2 MPU Address Status Register 0 (MPUASTAT0)

Address: Module Base + 0x0001

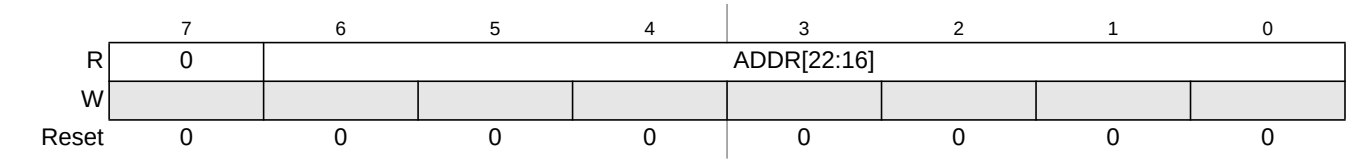


Figure 17-4. MPU Address Status Register 0 (MPUASTAT0)

Read: Anytime

Write: Never

Table 17-4. MPUASTAT0 Field Descriptions

Field	Description
6–0 ADDR[22:16]	**Access violation address bits** — The ADDR[22:16] bits contain bits [22:16] of the global address which caused the current access violation interrupt. These bits are undefined if the access error flag bit (AEF) in the MPUFLG register is not set.

17.3.1.3 MPU Address Status Register 1 (MPUASTAT1)

Address: Module Base + 0x0002

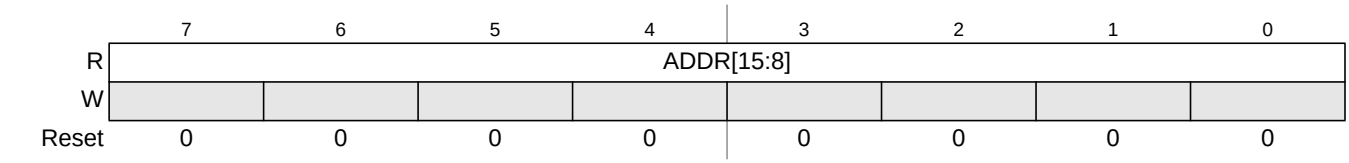


Figure 17-5. MPU Address Status Register 1 (MPUASTAT1)

Read: Anytime

Write: Never

Table 17-5. MPUASTAT1 Field Descriptions

Field	Description
7–0 ADDR[15:8]	**Access violation address bits** — The ADDR[15:8] bits contain bits [15:8] of the global address which caused the current access violation interrupt. These bits are undefined if the access error flag bit (AEF) in the MPUFLG register is not set.

17.3.1.4 MPU Address Status Register 2 (MPUASTAT2)

Address: Module Base + 0x0003

	7	6	5	4	3	2	1	0
R	ADDR[7:0]							
W								
Reset	0	0	0	0	0	0	0	0

Figure 17-6. MPU Address Status Register (MPUASTAT2)

Read: Anytime

Write: Never

Table 17-6. MPUASTAT2 Field Descriptions

Field	Description
7–0 ADDR[7:0]	**Access violation address bits** — The ADDR[7:0] bits contain bits [7:0] of the global address which caused the current access violation interrupt. These bits are undefined if the access error flag bit (AEF) in the MPUFLG register is not set.

17.3.1.5 MPU Descriptor Select Register (MPUSEL)

Address: Module Base + 0x0005

	7	6	5	4	3	2	1	0
R	SVSEN	0	0	0	SEL[3:0]			
W								
Reset	0	0	0	0	0	0	0	0

Figure 17-7. MPU Descriptor Select Register (MPUSEL)

Read: Anytime

Write: Anytime

Table 17-7. MPUSEL Field Descriptions

Field	Description
7 SVSEN	**MPU supervisor state enable bit** — This bit enables the memory protection for the CPU in supervisor state. If this bit is cleared, the MPU does not affect any accesses coming from the CPU in supervisor state. This is to prevent the CPU from locking out itself while configuring the protection descriptors (during initialization after a system reset and during the update of the protection descriptors for a task switch). The memory protection functionality for the other bus-masters is unaffected by this bit. 0 MPU is disabled for the CPU in supervisor state 1 MPU is enabled for the CPU in supervisor state
3–0 SEL[3:0]	**Descriptor select bits** — The SEL[3:0] bits select which descriptor is visible in the MPU Descriptor Register window (MPUDESC0—MPUDESC5). How many of the SEL[3:0] bits are writeable depends on the number of implemented descriptors. The number of implemented descriptors is a configuration option defined at SoC level. Please refer to the MCU toplevel chapter for details.

17.3.1.6 MPU Descriptor Register 0 (MPUDESC0)

Address: Module Base + 0x0006

	7	6	5	4	3	2	1	0
R W	MSTR0	MSTR1	MSTR2	MSTR3	LOW_ADDR[22:19]			
Reset	1[1]	1	1	1[2]	0[3]	0[3]	0[3]	0[3]

1. initialized as set for descriptor 0 only, cleared for all others
2. initialized as set for descriptor 0 only, if respective master is implemented on the device
3. These bits are intialized to the lower boundary of the MPU address range by a system reset. Depending on defined descriptor granularity and MPU address range some of these bits may not be writeable.

Figure 17-8. MPU Descriptor Register 0 (MPUDESC0)

Read: Anytime

Write: Anytime

Table 17-8. MPUDESC0 Field Descriptions

Field	Description
7 MSTR0	**Master 0 select bit** — If this bit is set the descriptor is valid for bus master 0 (CPU in supervisor state).
6 MSTR1	**Master 1 select bit** — If this bit is set the descriptor is valid for bus master 1 (CPU in user state). This bit can only be set if the CPU supports user state.
5 MSTR2	**Master 2 select bit** — If this bit is set the descriptor is valid for bus master 2 (XGATE). This bit can only be set if there is an XGATE implemented on the SoC.
4 MSTR3	**Master 3 select bit** — If this bit is set the descriptor is valid for bus master 3 (FlexRay).[1]
3–0 LOW_ADDR [22:19]	**Memory range lower boundary address bits** — The LOW_ADDR[22:19] bits represent bits [22:19] of the global memory address that is used as the lower boundary for the described memory range. These bits are intialized to the lower boundary of the MPU address range by a system reset.

1. Please refer Refernce Guide for information about the availability and function of Master 3 (see 1.9 MPU Configuration).

A descriptor can be configured as valid for more than one bus-master at the same time by setting multiple Master select bits to one. Setting all Master select bits of a descriptor to zero disables the descriptor. Bits for non-implemented masters cannot be written to and always read zero.

17.3.1.7 MPU Descriptor Register 1 (MPUDESC1)

Address: Module Base + 0x0007

	7	6	5	4	3	2	1	0
R W	LOW_ADDR[18:11]							
Reset	0(1)	0[1]	0[1]	0[1]	0[1]	0[1]	0[1]	0[1]

1. These bits are intialized to the lower boundary of the MPU address range by a system reset. Depending on defined descriptor granularity and MPU address range some of these bits may not be writeable.

Figure 17-9. MPU Descriptor Register 1 (MPUDESC1)

Read: Anytime

Write: Anytime

Table 17-9. MPUDESC1 Field Descriptions

Field	Description
7–0 LOW_ADDR[18:11]	**Memory range lower boundary address bits** — The LOW_ADDR[18:11] bits represent bits [18:11] of the global memory address that is used as the lower boundary for the described memory range.hese bits are intialized to the lower boundary of the MPU address range by a system reset.

17.3.1.8 MPU Descriptor Register 2 (MPUDESC2)

Address: Module Base + 0x0008

	7	6	5	4	3	2	1	0
R W	LOW_ADDR[10:3]							
Reset	0(1)	0[1]	0[1]	0[1]	0[1]	0[1]	0[1]	0[1]

1. These bits are intialized to the lower boundary of the MPU address range by a system reset. Depending on defined descriptor granularity and MPU address range some of these bits may not be writeable.

Figure 17-10. MPU Descriptor Register 2 (MPUDESC2)

Read: Anytime

Write: Anytime

Table 17-10. MPUDESC2 Field Descriptions

Field	Description
7–0 LOW_ADDR[10:3]	**Memory range lower boundary address bits** — The LOW_ADDR[10:3] bits represent bits [10:3] of the global memory address that is used as the lower boundary for the described memory range. These bits are intialized to the lower boundary of the MPU address range by a system reset.

17.3.1.9 MPU Descriptor Register 3 (MPUDESC3)

Address: Module Base + 0x0009

	7	6	5	4	3	2	1	0
R	WP	NEX	0	0	HIGH_ADDR[22:19]			
W								
Reset	0	0	0	0	1(1)	1	1	1

1. These bits are intialized to the upper boundary of the MPU address range by a system reset. Depending on defined descriptor granularity and MPU address range some of these bits may not be writeable.

Figure 17-11. MPU Descriptor Register 3 (MPUDESC3)

Read: Anytime

Write: Anytime

Table 17-11. MPUDESC3 Field Descriptions

Field	Description
7 WP	**Write-Protect bit** — The WP bit causes the described memory range to be treated as write-protected. If this bit is set every attempt to write in the described memory range causes an access violation.
6 NEX	**No-Execute bit** — The NEX bit prevents the described memory range from being used as code memory. If this bit is set every Op-code fetch in this memory range causes an access violation.
3–0 HIGH_ADDR[22:19]	**Memory range upper boundary address bits** — The HIGH_ADDR[22:19] bits represent bits [22:19] of the global memory address that is used as the upper boundary for the described memory range.These bits are intialized to the upper boundary of the MPU address range by a system reset.

17.3.1.10 MPU Descriptor Register 4 (MPUDESC4)

Address: Module Base + 0x000A

	7	6	5	4	3	2	1	0
R	HIGH_ADDR[18:11]							
W								
Reset	1(1)	1	1	1	1	1	1	1

1. These bits are intialized to the upper boundary of the MPU address range by a system reset. Depending on defined descriptor granularity and MPU address range some of these bits may not be writeable.

Figure 17-12. MPU Descriptor Register 4 (MPUDESC4)

Read: Anytime

Write: Anytime

Table 17-12. MPUDESC4 Field Descriptions

Field	Description
7–0 HIGH_ADDR[18:11]	**Memory range upper boundary address bits** — The HIGH_ADDR[18:11] bits represent bits [18:11] of the global memory address that is used as the upper boundary for the described memory range. These bits are intialized to the upper boundary of the MPU address range by a system reset.

17.3.1.11 MPU Descriptor Register 5 (MPUDESC5)

Address: Module Base + 0x000B

	7	6	5	4	3	2	1	0
R W	HIGH_ADDR[10:3]							
Reset	1(1)	1	1	1	1	1	1	1

1. These bits are intialized to the upper boundary of the MPU address range by a system reset. Depending on defined descriptor granularity and MPU address range some of these bits may not be writeable.

Figure 17-13. MPU Descriptor Register 5 (MPUDESC5)

Read: Anytime

Write: Anytime

Table 17-13. MPUDESC5 Field Descriptions

Field	Description
7–0 HIGH_ADDR[10:3]	**Memory range upper boundary address bits** — The HIGH_ADDR[10:3] bits represent bits [10:3] of the global memory address that is used as the upper boundary for the described memory range. These bits are intialized to the upper boundary of the MPU address range by a system reset.

17.4 Functional Description

The MPU module provides memory protection for accesses coming from multiple masters in the system. This is done by monitoring bus traffic of each master and compare this with the configuration information from a set of N[1] programmable descriptors located in the MPU module. If the MPU detects an access violation caused by the S12X CPU, it will assert the CPU access violation interrupt signal. If the MPU detects an access violation caused by a bus master other than the S12X CPU, it raises an access error signal. Please refer to the documentation chapter of the individual master modules (i.e. XGATE, etc.) for more information about the access error condition.

Violating accesses are not executed. The return value of a violating read access is undefined for both 8 bit and 16 bit accesses.

NOTE

Accesses from BDM are not restricted. BDM hardware accesses always bypass the MPU. During execution of BDM firmware code S12X CPU accesses are masked from the MPU as well.

17.4.1 Protection Descriptors

Each of the N protection descriptors can be used to restrict the allowed types of memory accesses for a given memory range. Each of these memory ranges can cover up the entire defined MPU address range. This can be the full 23 bits global memory range (8 MBytes) of the SoC.

1. The number of implemented descriptors is a configuration option defined at SoC level. Please refer to the MCU toplevel chapter for details.

The descriptors are banked in the MPU register map.

Each descriptor can be selected for modifying using the SEL bits in the MPU Descriptor Select (MPUSEL) register.

Table 17-14 gives an overview of the types of accesses that can be configured using the protection descriptors.

Table 17-14. Access Types

WP	NEX	Meaning
0	0	read, write and execute
0	1	read, write
1	0	read and execute
1	1	read only

The minimum granularity of each descriptor is 8 bytes. This means the protection comparators in the MPU module cover only address bits [22:3] of each access. The lower address bits (default [2:0], depending on descriptor granularity) are ignored.

NOTE

A mis-aligned word access to the upper boundary address of a descriptor is always flagged as an access violation.

NOTE

Configuring the lower boundary address of a descriptor to be higher than the upper boundary address of a descriptor causes this descriptor to be ignored by the comparator block. This effectively disables the descriptor.

NOTE

Avoid changing descriptors while they are in active use to validate accesses from bus-masters. This can be done by temporarily disabling the affected master during the update (XGATE, Master 3, switch S12X CPU states). Otherwise accesses from bus-masters affected by a descriptor which is updated concurrently could yield undefined results.

17.4.1.1 Overlapping Descriptors

If the memory ranges of two protection descriptors defined for the same bus-master overlap, the access restrictions for the overlapped memory range are accumulated. For example:

- a memory protection descriptor defines memory range 0x40_0000–0x41_FFFF as WP=1, NEX=0 (read and execute)
- another descriptor defines memory range 0x41_0000–0x43_FFFF as WP=0, NEX=1 (read and write)
- the resulting access rights for the overlapping range 0x41_0000–0x41_FFFF are WP=1, NEX=1 (read only)

17.4.1.2 Implicitly defined memory descriptors

As mentioned in the bit description of the Access Error Flag (AEF) in the MPUFLG register (Table 17-3), there is an additional memory range implicitly defined only while the AEF bit is set: The CPU in supervisor state can read from and write to the peripheral register space even if there is no memory protection descriptor explicitly allowing this. This is to prevent the case that the CPU cannot clear the AEF bit if the registers are write protected for the CPU in supervisor state.

The register address space containing the PAGE registers (EPAGE, RPAGE, GPAGE, PPAGE) at 0x0010–0x0017 gets special treatment. It is defined like this:

- The S12X CPU can always read and write these registers, regardless of the configuration in the descriptors.
- XGATE or Master3 (if available) are never allowed to read or write these registers, even if the descriptor configuration allows accesses for other masters than the S12X CPU.

17.4.1.3 Op-code pre-fetch cycles and the NEX bit

Some bus-masters (CPU, XGATE) do a pre-fetch of program-code past the current instruction. The S12XCPU pre-fetches two words past the current instruction, the XGATE pre-fetches one word, even if the pre-fetched code is not executed. The MPU module has no way of knowing this at the time when the pre-fetch cycles occur. Therefore this will result in an access violation if the op-code pre-fetch accesses a memory range marked as “No-Execute” (NEX=1). This must be taken into account when defining memory ranges with the NEX bit set adjacent to memory used for program code. The best way to do this would be to leave some fill-bytes between the memory ranges in this case, i.e. do not set the upper memory boundary to the address of the last op-code but to a following address which is at least two words (four bytes) away.

17.4.2 Interrupts

This section describes all interrupts originated by the MPU.

17.4.2.1 Description of Interrupt Operation

The MPU generates one interrupt request. It cannot be masked locally in the MPU and is meant to be used as the source of a non-maskable hardware interrupt request for the S12X CPU.

Table 17-15. Interrupt vectors

Interrupt Source	CCR Mask	Local Enable
S12X CPU access error interrupt (AEF)	–	–

17.4.2.2 CPU Access Error Interrupt

An S12X CPU access error interrupt request is generated if the MPU has detected an illegal memory access originating from the S12X CPU. This is a non-maskable hardware interrupt. Due to the non-maskable nature of this interrupt, the de-assertion of this interrupt request is coupled to the S12X CPU interrupt vector fetch instead of the local access error flag (AEF). This means leaving the access error flag (AEF) in

the MPUFLG register set will not cause the same interrupt to be serviced again after leaving the interrupt service routine with “RTI”. Instead, the interrupt request will be asserted again only when the next illegal S12X CPU access is detected.

17.5 Initialization/Application Information

17.5.1 Initialization

After reset the MPU module is in an unconfigured state, with all N protection descriptors covering the whole memory map. The master bits are all set for descriptor “0” and cleared for all other descriptors. The S12XCPU in supervisor state can access everything because the SVSEN bit in the MPUSEL register is cleared by a system reset. After system reset every master has full access to the memory map because of descriptor “0”.

In order to use the MPU to protect memory ranges from undesired accesses, software needs to:

- Initialize the protection descriptors.
- Make sure there are meaningful interrupt service routines defined for the Access Violation interrupts because these are usually non-maskable (See XINT chapter for details).
- Initialize peripherals and other masters for use (i.e. set-up XGATE, Master3 if applicable).
- Enable the MPU protection for the S12X CPU in supervisor state, if desired.
- Switch the S12X CPU to user state, if desired.

Chapter 18
External Bus Interface (S12XEBIV4)

Revision History

Rev. No. (Item No.)	Date (Submitted By)	Sections Affected	Substantial Change(s)
V04.01	12-Sep-05		- Added CSx stretch description.
V04.02	23-May-06		- Internal updates
V04.03	24-Jul-06		- Removed term IVIS

18.1 Introduction

This document describes the functionality of the XEBI block controlling the external bus interface.

The XEBI controls the functionality of a non-multiplexed external bus (a.k.a. ‘expansion bus’) in relationship with the chip operation modes. Dependent on the mode, the external bus can be used for data exchange with external memory, peripherals or PRU, and provide visibility to the internal bus externally in combination with an emulator.

18.1.1 Glossary or Terms

bus clock	System Clock. Refer to CRG Block Guide.
expanded modes	Normal Expanded Mode Emulation Single-Chip Mode Emulation Expanded Mode Special Test Mode
single-chip modes	Normal Single-Chip Mode Special Single-Chip Mode
emulation modes	Emulation Single-Chip Mode Emulation Expanded Mode
normal modes	Normal Single-Chip Mode Normal Expanded Mode
special modes	Special Single-Chip Mode Special Test Mode
NS	Normal Single-Chip Mode
SS	Special Single-Chip Mode
NX	Normal Expanded Mode
ES	Emulation Single-Chip Mode
EX	Emulation Expanded Mode
ST	Special Test Mode
external resource	Addresses outside MCU
PRR	Port Replacement Registers
PRU	Port Replacement Unit
EMULMEM	External emulation memory
access source	CPU or BDM or XGATE

18.1.2 Features

The XEBI includes the following features:

- Output of up to 23-bit address bus and control signals to be used with a non-muxed external bus
- Bidirectional 16-bit external data bus with option to disable upper half
- Visibility of internal bus activity

18.1.3 Modes of Operation

- Single-chip modes

  The external bus interface is not available in these modes.
- Expanded modes

  Address, data, and control signals are activated on the external bus in normal expanded mode and special test mode.
- Emulation modes

  The external bus is activated to interface to an external tool for emulation of normal expanded mode or normal single-chip mode applications.

Refer to the S12X_MMC section for a detailed description of the MCU operating modes.

18.1.4 Block Diagram

Figure 18-1 is a block diagram of the XEBI with all related I/O signals.

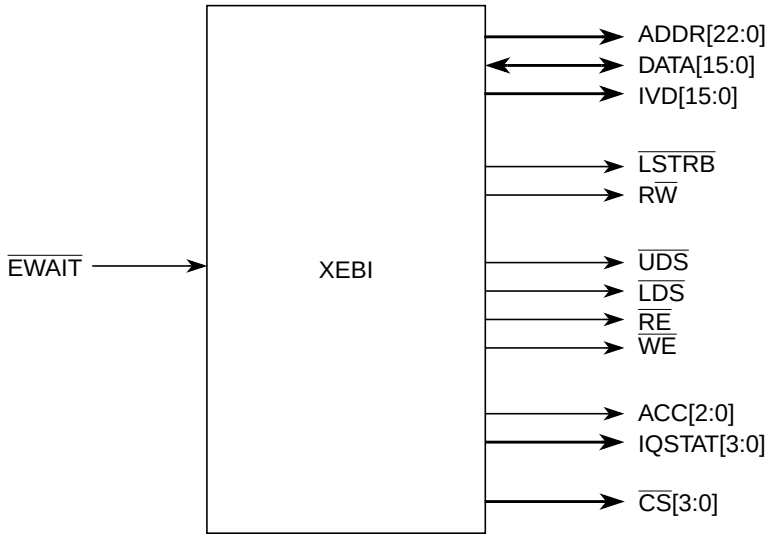


Figure 18-1. XEBI Block Diagram

18.2 External Signal Description

The user is advised to refer to the SoC section for port configuration and location of external bus signals.

NOTE

The following external bus related signals are described in other sections:
ECLK, ECLKX2 (free-running clocks) — PIM section
TAGHI, TAGLO (tag inputs) — PIM section, S12X_DBG section

Table 18-1 outlines the pin names and gives a brief description of their function. Refer to the SoC section and PIM section for reset states of these pins and associated pull-ups or pull-downs.

Table 18-1. External System Signals Associated with XEBI (Sheet 1 of 2)

Signal	I[1]/O	EBI Signal Multiplex (T)ime[2] (F)unction[3]		Description	Available in Modes					
					NS	SS	NX	ES	EX	ST
RE	O	—	—	Read Enable, indicates external read access	No	No	Yes	No	No	No
ADDR[22:20]	O	T	—	External address	No	No	Yes	Yes	Yes	Yes
ACC[2:0]	O		—	Access source	No	No	No	Yes	Yes	Yes
ADDR[19:16]	O	T	—	External address	No	No	Yes	Yes	Yes	Yes
IQSTAT[3:0]	O		—	Instruction Queue Status	No	No	No	Yes	Yes	Yes

Table 18-1. External System Signals Associated with XEBI (Sheet 2 of 2)

Signal	I[1]/O	EBI Signal Multiplex (T)ime[2] (F)unction[3]		Description	Available in Modes					
					NS	SS	NX	ES	EX	ST
ADDR[15:1]	O	T	—	External address	No	No	Yes	Yes	Yes	Yes
IVD[15:1]	O		—	Internal visibility read data	No	No	No	Yes	Yes	Yes
ADDR0	O	T	F	External address	No	No	No	Yes	Yes	Yes
IVD0	O			Internal visibility read data	No	No	No	Yes	Yes	Yes
$\overline{UDS}$	O	—		Upper Data Select, indicates external access to the high byte DATA[15:8]	No	No	Yes	No	No	No
$\overline{LSTRB}$	O	—	F	Low Strobe, indicates valid data on DATA[7:0]	No	No	No	Yes	Yes	Yes
$\overline{LDS}$	O	—		Lower Data Select, indicates external access to the low byte DATA[7:0]	No	No	Yes	No	No	No
R$\overline{W}$	O	—	F	Read/Write, indicates the direction of internal data transfers	No	No	No	Yes	Yes	Yes
$\overline{WE}$	O	—		Write Enable, indicates external write access	No	No	Yes	No	No	No
$\overline{CS}$[3:0]	O	—	—	Chip select	No	No	Yes	No	Yes	No
DATA[15:8]	I/O	—	—	Bidirectional data (even address)	No	No	Yes	Yes	Yes	Yes
DATA[7:0]	I/O	—	—	Bidirectional data (odd address)	No	No	Yes	Yes	Yes	Yes
$\overline{EWAIT}$	I	—	—	External control for external bus access stretches (adding wait states)	No	No	Yes	No	Yes	No

1. All inputs are capable of reducing input threshold level
2. Time-multiplex means that the respective signals share the same pin on chip level and are active alternating in a dedicated time slot (in modes where applicable).
3. Function-multiplex means that one of the respective signals sharing the same pin on chip level continuously uses the pin depending on configuration and reset state.

18.3 Memory Map and Register Definition

This section provides a detailed description of all registers accessible in the XEBI.

18.3.1 Module Memory Map

The registers associated with the XEBI block are shown in Figure 18-2.

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0E EBICTL0	R	ITHRS	0	HDBE	ASIZ4	ASIZ3	ASIZ2	ASIZ1	ASIZ0
	W								
0x0F EBICTL1	R	0	EXSTR12	EXSTR11	EXSTR10	0	EXSTR02	EXSTR01	EXSTR00
	W								

Figure 18-2. XEBI Register Summary

= Unimplemented or Reserved

Figure 18-2. XEBI Register Summary

18.3.2 Register Descriptions

The following sub-sections provide a detailed description of each register and the individual register bits.

All control bits can be written anytime, but this may have no effect on the related function in certain operating modes. This allows specific configurations to be set up before changing into the target operating mode.

NOTE

Depending on the operating mode an available function may be enabled, disabled or depend on the control register bit. Reading the register bits will reflect the status of related function only if the current operating mode allows user control. Please refer the individual bit descriptions.

18.3.2.1 External Bus Interface Control Register 0 (EBICTL0)

Module Base +0x000E (PRR)

	7	6	5	4	3	2	1	0
R	ITHRS	0	HDBE	ASIZ4	ASIZ3	ASIZ2	ASIZ1	ASIZ0
W								
Reset	0	0	1	1	1	1	1	1

= Unimplemented or Reserved

Figure 18-3. External Bus Interface Control Register 0 (EBICTL0)

Read: Anytime. In emulation modes, read operations will return the data from the external bus, in all other modes, the data is read from this register.

Write: Anytime. In emulation modes, write operations will also be directed to the external bus.

This register controls input pin threshold level and determines the external address and data bus sizes in normal expanded mode. If not in use with the external bus interface, the related pins can be used for alternative functions.

External bus is available as programmed in normal expanded mode and always full-sized in emulation modes and special test mode; function not available in single-chip modes.

Table 18-2. EBICTL0 Field Descriptions

Field	Description
7 ITHRS	**Reduced Input Threshold** — This bit selects reduced input threshold on external data bus pins and specific control input signals which are in use with the external bus interface in order to adapt to external devices with a 3.3 V, 5 V tolerant I/O. The reduced input threshold level takes effect depending on ITHRS, the operating mode and the related enable signals of the EBI pin function as summarized in Table 18-3. 0 Input threshold is at standard level on all pins 1 Reduced input threshold level enabled on pins in use with the external bus interface
5 HDBE	**High Data Byte Enable** — This bit enables the higher half of the 16-bit data bus. If disabled, only the lower 8-bit data bus can be used with the external bus interface. In this case the unused data pins and the data select signals ($\overline{UDS}$ and $\overline{LDS}$) are free to be used for alternative functions. 0 DATA[15:8], $\overline{UDS}$, and $\overline{LDS}$ disabled 1 DATA[15:8], $\overline{UDS}$, and $\overline{LDS}$ enabled
4–0 ASIZ[4:0]	**External Address Bus Size** — These bits allow scalability of the external address bus. The programmed value corresponds to the number of available low-aligned address lines (refer to Table 18-4). All address lines ADDR[22:0] start up as outputs after reset in expanded modes. This needs to be taken into consideration when using alternative functions on relevant pins in applications which utilize a reduced external address bus.

Table 18-3. Input Threshold Levels on External Signals

ITHRS	External Signal	NS	SS	NX	ES	EX	ST
0	DATA[15:8] $\overline{TAGHI}$, $\overline{TAGLO}$	Standard	Standard	Standard	Reduced	Reduced	Standard
	DATA[7:0]						
	$\overline{EWAIT}$				Standard	Standard	
1	DATA[15:8] $\overline{TAGHI}$, $\overline{TAGLO}$	Standard	Standard	Reduced if HDBE = 1	Reduced	Reduced	Reduced
	DATA[7:0]			Reduced			
	$\overline{EWAIT}$			Reduced if EWAIT enabled(1)	Standard	Reduced if EWAIT enabled1	Standard

1. EWAIT function is enabled if at least one $\overline{CSx}$ line is configured respectively in MMCCTL0. Refer to S12X_MMC section and Table 18-5.

Table 18-4. External Address Bus Size

ASIZ[4:0]	Available External Address Lines
00000	None
00001	$\overline{UDS}$
00010	ADDR1, $\overline{UDS}$
00011	ADDR[2:1], $\overline{UDS}$
:	:
10110	ADDR[21:1], $\overline{UDS}$

Table 18-4. External Address Bus Size

ASIZ[4:0]	Available External Address Lines
10111 : 11111	ADDR[22:1], $\overline{UDS}$

18.3.2.2 External Bus Interface Control Register 1 (EBICTL1)

Module Base +0x000F (PRR)

	7	6	5	4	3	2	1	0
R	0	EXSTR12	EXSTR11	EXSTR10	0	EXSTR02	EXSTR01	EXSTR00
W								
Reset	0	1	1	1	0	1	1	1

= Unimplemented or Reserved

Figure 18-4. External Bus Interface Control Register 1 (EBICTL1)

Read: Anytime. In emulation modes, read operations will return the data from the external bus, in all other modes the data is read from this register.

Write: Anytime. In emulation modes, write operations will also be directed to the external bus.

This register allows programming of two independent values determining the amount of additional stretch cycles for external accesses (wait states).

With two bits in S12X_MMC register MMCCTL0 for every individual $\overline{CSx}$ line one of the two counter options or the $\overline{EWAIT}$ input is selected as stretch source. The chip select outputs can also be disabled to free up the pins for alternative functions (Table 18-5). Refer also to S12X_MMC section for register bit descriptions.

Table 18-5. Chip select function

CSxE1	CSxE0	Function
0	0	$\overline{CSx}$ disabled
0	1	$\overline{CSx}$ stretched with EXSTR0
1	0	$\overline{CSx}$ stretched with EXSTR1
1	1	$\overline{CSx}$ stretched with $\overline{EWAIT}$

If $\overline{EWAIT}$ input usage is selected in MMCCTL0 the minimum number of stretch cycles is 2 for accesses to the related address range.

If configured respectively, stretch cycles are added as programmed or dependent on $\overline{EWAIT}$ in normal expanded mode and emulation expanded mode; function not available in all other operating modes.

Table 18-6. EBICTL1 Field Descriptions

Field	Description
6–4 EXSTR1[2:0]	**External Access Stretch Option 1 Bits 2, 1, 0** — This three bit field determines the amount of additional clock stretch cycles on every access to the external address space as shown in Table 18-7.
2–0 EXSTR0[2:0]	**External Access Stretch Option 0 Bits 2, 1, 0** — This three bit field determines the amount of additional clock stretch cycles on every access to the external address space as shown in Table 18-7.

Table 18-7. External Access Stretch Bit Definition

EXSTRx[2:0]	Number of Stretch Cycles
000	1
001	2
010	3
011	4
100	5
101	6
110	7
111	8

18.4 Functional Description

This section describes the functions of the external bus interface. The availability of external signals and functions in relation to the operating mode is initially summarized and described in more detail in separate sub-sections.

18.4.1 Operating Modes and External Bus Properties

A summary of the external bus interface functions for each operating mode is shown in Table 18-8.

Table 18-8. Summary of Functions

Properties (if Enabled)	Single-Chip Modes		Expanded Modes			
	Normal Single-Chip	Special Single-Chip	Normal Expanded	Emulation Single-Chip	Emulation Expanded	Special Test
Timing Properties						
PRR access[1]	2 cycles read internal write internal	2 cycles read internal write internal	2 cycles read internal write internal	2 cycles read external write int & ext	2 cycles read external write int & ext	2 cycles read internal write internal
Internal access visible externally	—	—	—	1 cycle	1 cycle	1 cycle
External address access and unimplemented area access[2]	—	—	Max. of 2 to 9 programmed cycles or n cycles of ext. wait[3]	1 cycle	Max. of 2 to 9 programmed cycles or n cycles of ext. wait[3]	1 cycle

Table 18-8. Summary of Functions

Properties (if Enabled)	Single-Chip Modes		Expanded Modes			
	Normal Single-Chip	Special Single-Chip	Normal Expanded	Emulation Single-Chip	Emulation Expanded	Special Test
Flash area address access[4]	—	—	—	1 cycle	1 cycle	1 cycle
Signal Properties						
Bus signals	—	—	ADDR[22:1] DATA[15:0]	ADDR[22:20]/ ACC[2:0] ADDR[19:16]/ IQSTAT[3:0] ADDR[15:0]/ IVD[15:0] DATA[15:0]	ADDR[22:20]/ ACC[2:0] ADDR[19:16]/ IQSTAT[3:0] ADDR[15:0]/ IVD[15:0] DATA[15:0]	ADDR[22:0] DATA[15:0]
Data select signals (if 16-bit data bus)	—	—	$\overline{UDS}$ $\overline{LDS}$	ADDR0 $\overline{LSTRB}$	ADDR0 $\overline{LSTRB}$	ADDR0 $\overline{LSTRB}$
Data direction signals	—	—	$\overline{RE}$ $\overline{WE}$	R$\overline{W}$	R$\overline{W}$	R$\overline{W}$
Chip Selects	—	—	$\overline{CS0}$ $\overline{CS1}$ $\overline{CS2}$ $\overline{CS3}$	—	$\overline{CS0}$ $\overline{CS1}$ $\overline{CS2}$ $\overline{CS3}$	—
External wait feature	—	—	$\overline{EWAIT}$	—	$\overline{EWAIT}$	—
Reduced input threshold enabled on	—	—	Refer to Table 18-3	DATA[15:0] $\overline{EWAIT}$	DATA[15:0] $\overline{EWAIT}$	Refer to Table 18-3

1. Incl. S12X_EBI registers
2. Refer to S12X_MMC section.
3. If EWAIT enabled for at least one $\overline{CSx}$ line (refer to S12X_MMC section), the minimum number of external bus cycles is 3.
4. Available only if configured appropriately by ROMON and EROMON (refer to S12X_MMC section).

18.4.2 Internal Visibility

Internal visibility allows the observation of the internal CPU address and data bus as well as the determination of the access source and the CPU pipe (queue) status through the external bus interface.

Internal visibility is always enabled in emulation single chip mode and emulation expanded mode. Internal CPU accesses are made visible on the external bus interface except CPU execution of BDM firmware instructions.

Internal reads are made visible on ADDRx/IVDx (address and read data multiplexed, see Table 18-11 to Table 18-13), internal writes on ADDRx and DATAx (see Table 18-14 to Table 18-16). R$\overline{W}$ and $\overline{LSTRB}$ show the type of access. External read data are also visible on IVDx.

During ‘no access’ cycles R$\overline{W}$ is held in read position while $\overline{LSTRB}$ is undetermined.

All accesses which make use of the external bus interface are considered external accesses.

18.4.2.1 Access Source Signals (ACC)

The access source can be determined from the external bus control signals ACC[2:0] as shown in Table 18-9.

Table 18-9. Determining Access Source from Control Signals

ACC[2:0]	Access Description
000	Repetition of previous access cycle
001	CPU access
010	BDM external access
011	XGATE PRR access
100	No access(1)
101	CPU access error
110, 111	Reserved

1. Denotes also CPU accesses to BDM firmware and BDM registers (IQSTATx are 'XXXX' and $\overline{RW}$ = 1 in these cases)

18.4.2.2 Instruction Queue Status Signals (IQSTAT)

The CPU instruction queue status (execution-start and data-movement information) is brought out as IQSTAT[3:0] signals. For decoding of the IQSTAT values, refer to the S12X_CPU section.

18.4.2.3 Internal Visibility Data (IVD)

Depending on the access size and alignment, either a word of read data is made visible on the address lines or only the related data byte will be presented in the ECLK low phase. For details refer to Table 18-10.

Invalid IVD are brought out in case of non-CPU read accesses.

Table 18-10. IVD Read Data Output

Access	IVD[15:8]	IVD[7:0]
Word read of data at an even and even+1 address	ivd(even)	ivd(even+1)
Word read of data at an odd and odd+1 internal RAM address (misaligned)	ivd(odd+1)	ivd(odd)
Byte read of data at an even address	ivd(even)	addr[7:0] (rep.)
Byte read of data at an odd address	addr[15:8] (rep.)	ivd(odd)

18.4.2.4 Emulation Modes Timing

A bus access lasts 1 ECLK cycle. In case of a stretched external access (emulation expanded mode), up to an infinite amount of ECLK cycles may be added. ADDRx values will only be shown in ECLK high phases, while ACCx, IQSTATx, and IVDx values will only be presented in ECLK low phases.

Based on this multiplex timing, ACCx are only shown in the current (first) access cycle. IQSTATx and (for read accesses) IVDx follow in the next cycle. If the access takes more than one bus cycle, ACCx display NULL (0x000) in the second and all following cycles of the access. IQSTATx display NULL (0x0000) from the third until one cycle after the access to indicate continuation.

The resulting timing pattern of the external bus signals is outlined in the following tables for read, write and interleaved read/write accesses. Three examples represent different access lengths of 1, 2, and n–1 bus cycles. Non-shaded bold entries denote all values related to Access #0.

The following terminology is used:

‘addr’ — value(ADDRx); small letters denote the logic values at the respective pins
‘x’ — Undefined output pin values
‘z’ — Tristate pins
‘?’ — Dependent on previous access (read or write); IVDx: ‘ivd’ or ‘x’; DATAx: ‘data’ or ‘z’

18.4.2.4.1 Read Access Timing

Table 18-11. Read Access (1 Cycle)

		Access #0				Access #1		
Bus cycle ->	...	**1**		**2**		**3**		...
ECLK phase	...	high	low	high	low	high	low	...
ADDR[22:20] / ACC[2:0]	...		**acc 0**		acc 1		acc 2	...
ADDR[19:16] / IQSTAT[3:0]	...	**addr 0**	iqstat -1	addr 1	**iqstat 0**	addr 2	iqstat 1	...
ADDR[15:0] / IVD[15:0]	...		?		**ivd 0**		ivd 1	...
DATA[15:0] (internal read)	...	?	**z**	**z**	z	z	z	...
DATA[15:0] (external read)	...	?	**z**	**data 0**	z	data 1	z	...
$\overline{RW}$	...	1	1	1	1	1	1	...

Table 18-12. Read Access (2 Cycles)

		Access #0				Access #1		
Bus cycle ->	...	**1**		**2**		**3**		...
ECLK phase	...	high	low	high	low	high	low	...
ADDR[22:20] / ACC[2:0]	...		**acc 0**		**000**		acc 1	...
ADDR[19:16] / IQSTAT[3:0]	...	**addr 0**	iqstat-1	**addr 0**	**iqstat 0**	addr 1	**0000**	...
ADDR[15:0] / IVD[15:0]	...		?		**x**		**ivd 0**	...
DATA[15:0] (internal read)	...	?	**z**	**z**	**z**	**z**	z	...
DATA[15:0] (external read)	...	?	**z**	**z**	**z**	**data 0**	z	...
$\overline{RW}$	...	1	1	1	1	1	1	...

Table 18-13. Read Access (n–1 Cycles)

		Access #0							Access #1		
Bus cycle ->	...	**1**		**2**		**3**		...	**n**		...
ECLK phase	...	high	low	high	low	high	low	...	high	low	...
ADDR[22:20] / ACC[2:0]	...		**acc 0**		**000**		**000**	...		acc 1	...
ADDR[19:16] / IQSTAT[3:0]	...	**addr 0**	iqstat-1	**addr 0**	**iqstat 0**	**addr 0**	**0000**	...	addr 1	**0000**	...
ADDR[15:0] / IVD[15:0]	...		?		**x**		**x**	...		**ivd 0**	...
DATA[15:0] (internal read)	...	?	**z**	**z**	**z**	**z**	**z**	...	**z**	z	...

Table 18-13. Read Access (n–1 Cycles)

DATA[15:0] (external read)	...	?	**z**	**z**	**z**	**z**	**z**	...	**data 0**	z	...
R/$\overline{W}$	...	1	1	1	1	1	1	...	1	1	...

18.4.2.4.2 Write Access Timing

Table 18-14. Write Access (1 Cycle)

		Access #0		**Access #1**		**Access #2**		
Bus cycle ->	...	**1**		**2**		**3**		...
ECLK phase	...	high	low	high	low	high	low	...
ADDR[22:20] / ACC[2:0]	...	**addr 0**	**acc 0**	addr 1	acc 1	addr 2	acc 2	...
ADDR[19:16] / IQSTAT[3:0]	...		iqstat -1		**iqstat 0**		iqstat 1	...
ADDR[15:0] / IVD[15:0]	...		?		**x**		x	...
DATA[15:0] (write)	...	?	**data 0**		data 1		data 2	...
R/$\overline{W}$	...	0	0	1	1	1	1	...

Table 18-15. Write Access (2 Cycles)

		Access #0				**Access #1**		
Bus cycle ->	...	**1**		**2**		**3**		...
ECLK phase	...	high	low	high	low	high	low	...
ADDR[22:20] / ACC[2:0]	...	**addr 0**	**acc 0**	**addr 0**	**000**	addr 1	acc 1	...
ADDR[19:16] / IQSTAT[3:0]	...		iqstat-1		**iqstat 0**		**0000**	...
ADDR[15:0] / IVD[15:0]	...		?		**x**		**x**	...
DATA[15:0] (write)	...	?	**data 0**				x	...
R/$\overline{W}$	...	0	0	0	0	1	1	...

Table 18-16. Write Access (n–1 Cycles)

		Access #0							**Access #1**		
Bus cycle ->	...	**1**		**2**		**3**		...	**n**		...
ECLK phase	...	high	low	high	low	high	low	...	high	low	...
ADDR[22:20] / ACC[2:0]	...	**addr 0**	**acc 0**	**addr 0**	**000**	**addr 0**	**000**	**...**	addr 1	acc 1	...
ADDR[19:16] / IQSTAT[3:0]	...		iqstat-1		**iqstat 0**		**0000**	**...**		**0000**	...
ADDR[15:0] / IVD[15:0]	...		?		**x**		**x**	**...**		**x**	...
DATA[15:0] (write)	...	?	**data 0**							x	...
R/$\overline{W}$	...	0	0	0	0	0	0	...	1	1	...

18.4.2.4.3 Read-Write-Read Access Timing

Table 18-17. Interleaved Read-Write-Read Accesses (1 Cycle)

		Access #0	**Access #1**	**Access #2**	
Bus cycle ->	...	**1**	**2**	**3**	...

Table 18-17. Interleaved Read-Write-Read Accesses (1 Cycle)

ECLK phase	...	high	low	high	low	high	low	...
ADDR[22:20] / ACC[2:0]	...	**addr 0**	**acc 0**	addr 1	acc 1	addr 2	acc 2	...
ADDR[19:16] / IQSTAT[3:0]	...		iqstat -1		**iqstat 0**		iqstat 1	...
ADDR[15:0] / IVD[15:0]	...		?		**ivd 0**		x	...
DATA[15:0] (internal read)	...	?	**z**	**z**	(write) data 1		z	...
DATA[15:0] (external read)	...	?	**z**	**data 0**	(write) data 1		z	...
R/$\overline{W}$	...	1	1	0	0	1	1	...

18.4.3 Accesses to Port Replacement Registers

All read and write accesses to PRR addresses take two bus clock cycles independent of the operating mode. If writing to these addresses in emulation modes, the access is directed to both, the internal register and the external resource while reads will be treated external.

The XEBI control registers also belong to this category.

18.4.4 Stretched External Bus Accesses

In order to allow fast internal bus cycles to coexist in a system with slower external resources, the XEBI supports stretched external bus accesses (wait states) for each external address range related to one of the 4 chip select lines individually.

This feature is available in normal expanded mode and emulation expanded mode for accesses to all external addresses except emulation memory and PRR. In these cases the fixed access times are 1 or 2 cycles, respectively.

Stretched accesses are controlled by:

1. EXSTR1[2:0] and EXSTR0[2:0] bits in the EBICTL1 register configuring a fixed amount of stretch cycles individually for each $\overline{CSx}$ line in MMCCTL0
2. Activation of the external wait feature for each $\overline{CSx}$ line MMCCTL0 register
3. Assertion of the external $\overline{EWAIT}$ signal when at least one $\overline{CSx}$ line is configured for EWAIT

The EXSTRx[2:0] control bits can be programmed for generation of a fixed number of 1 to 8 stretch cycles. If the external wait feature is enabled, the minimum number of additional stretch cycles is 2. An arbitrary amount of stretch cycles can be added using the $\overline{EWAIT}$ input.

$\overline{EWAIT}$ needs to be asserted at least for a minimal specified time window within an external access cycle for the internal logic to detect it and add a cycle (refer to electrical characteristics). Holding it for additional cycles will cause the external bus access to be stretched accordingly.

Write accesses are stretched by holding the initiator in its current state for additional cycles as programmed and controlled by external wait after the data have been driven out on the external bus. This results in an extension of time the bus signals and the related control signals are valid externally.

Read data are not captured by the system in normal expanded mode until the specified setup time before the $\overline{RE}$ rising edge.

Read data are not captured in emulation expanded mode until the specified setup time before the falling edge of ECLK.

In emulation expanded mode, accesses to the internal flash or the emulation memory (determined by EROMON and ROMON bits; see S12X_MMC section for details) always take 1 cycle and stretching is not supported. In case the internal flash is taken out of the map in user applications, accesses are stretched as programmed and controlled by external wait.

18.4.5 Data Select and Data Direction Signals

The S12X_EBI supports byte and word accesses at any valid external address. The big endian system of the MCU is extended to the external bus; however, word accesses are restricted to even aligned addresses. The only exception is the visibility of misaligned word accesses to addresses in the internal RAM as this module exclusively supports these kind of accesses in a single cycle.

With the above restriction, a fixed relationship is implied between the address parity and the dedicated bus halves where the data are accessed: DATA[15:8] is related to even addresses and DATA[7:0] is related to odd addresses.

In expanded modes the data access type is externally determined by a set of control signals, i.e., data select and data direction signals, as described below. The data select signals are not available if using the external bus interface with an 8-bit data bus.

18.4.5.1 Normal Expanded Mode

In normal expanded mode, the external signals $\overline{RE}$, $\overline{WE}$, $\overline{UDS}$, $\overline{LDS}$ indicate the access type (read/write), data size and alignment of an external bus access (Table 18-18).

Table 18-18. Access in Normal Expanded Mode

Access	$\overline{RE}$	$\overline{WE}$	$\overline{UDS}$	$\overline{LDS}$	DATA[15:8]		DATA[7:0]	
					I/O	data(addr)	I/O	data(addr)
Word write of data on DATA[15:0] at an even and even+1 address	1	0	0	0	Out	data(even)	Out	data(odd)
Byte write of data on DATA[7:0] at an odd address	1	0	1	0	In	x	Out	data(odd)
Byte write of data on DATA[15:8] at an even address	1	0	0	1	Out	data(even)	In	x
Word read of data on DATA[15:0] at an even and even+1 address	0	1	0	0	In	data(even)	In	data(odd)
Byte read of data on DATA[7:0] at an odd address	0	1	1	0	In	x	In	data(odd)
Byte read of data on DATA[15:8] at an even address	0	1	0	1	In	data(even)	In	x
Indicates No Access	1	1	1	1	In	x	In	x
Unimplemented	1	1	1	0	In	x	In	x
	1	1	0	1	In	x	In	x

18.4.5.2 Emulation Modes and Special Test Mode

In emulation modes and special test mode, the external signals $\overline{LSTRB}$, R$\overline{W}$, and ADDR0 indicate the access type (read/write), data size and alignment of an external bus access. Misaligned accesses to the

internal RAM and misaligned XGATE PRR accesses in emulation modes are the only type of access that are able to produce $\overline{LSTRB}$ = ADDR0 = 1. This is summarized in Table 18-19.

Table 18-19. Access in Emulation Modes and Special Test Mode

Access	R/$\overline{W}$	$\overline{LSTRB}$	ADDR0	DATA[15:8]		DATA[7:0]	
				I/O	data(addr)	I/O	data(addr)
Word write of data on DATA[15:0] at an even and even+1 address	0	0	0	Out	data(even)	Out	data(odd)
Byte write of data on DATA[7:0] at an odd address	0	0	1	In	x	Out	data(odd)
Byte write of data on DATA[15:8] at an even address	0	1	0	Out	data(odd)	In	x
Word write at an odd and odd+1 internal RAM address (misaligned — only in emulation modes)	0	1	1	Out	data(odd+1)	Out	data(odd)
Word read of data on DATA[15:0] at an even and even+1 address	1	0	0	In	data(even)	In	data(even+1)
Byte read of data on DATA[7:0] at an odd address	1	0	1	In	x	In	data(odd)
Byte read of data on DATA[15:8] at an even address	1	1	0	In	data(even)	In	x
Word read at an odd and odd+1 internal RAM address (misaligned - only in emulation modes)	1	1	1	In	data(odd+1)	In	data(odd)

18.4.6 Low-Power Options

The XEBI does not support any user-controlled options for reducing power consumption.

18.4.6.1 Run Mode

The XEBI does not support any options for reducing power in run mode.

Power consumption is reduced in single-chip modes due to the absence of the external bus interface. Operation in expanded modes results in a higher power consumption, however any unnecessary toggling of external bus signals is reduced to the lowest indispensable activity by holding the previous states between external accesses.

18.4.6.2 Wait Mode

The XEBI does not support any options for reducing power in wait mode.

18.4.6.3 Stop Mode

The XEBI will cease to function in stop mode.

18.5 Initialization/Application Information

This section describes the external bus interface usage and timing. Typical customer operating modes are normal expanded mode and emulation modes, specifically to be used in emulator applications. Taking the availability of the external wait feature into account the use cases are divided into four scenarios:

- Normal expanded mode

  - External wait feature disabled
  - External wait feature enabled
- Emulation modes
  - Emulation single-chip mode (without wait states)
  - Emulation expanded mode (with optional access stretching)

Normal single-chip mode and special single-chip mode do not have an external bus. Special test mode is used for factory test only. Therefore, these modes are omitted here.

All timing diagrams referred to throughout this section are available in the Electrical Characteristics appendix of the SoC section.

18.5.1 Normal Expanded Mode

This mode allows interfacing to external memories or peripherals which are available in the commercial market. In these applications the normal bus operation requires a minimum of 1 cycle stretch for each external access.

18.5.1.1 Example 1a: External Wait Feature Disabled

The first example of bus timing of an external read and write access with the external wait feature disabled is shown in

- Figure ‘Example 1a: Normal Expanded Mode — Read Followed by Write’

The associated supply voltage dependent timing are numbers given in

- Table ‘Example 1a: Normal Expanded Mode Timing V_{DD5} = 5.0 V (EWAIT disabled)’
- Table ‘Example 1a: Normal Expanded Mode Timing V_{DD5} = 3.0 V (EWAIT disabled)’

Systems designed this way rely on the internal programmable access stretching. These systems have predictable external memory access times. The additional stretch time can be programmed up to 8 cycles to provide longer access times.

18.5.1.2 Example 1b: External Wait Feature Enabled

The external wait operation is shown in this example. It can be used to exceed the amount of stretch cycles over the programmed number in EXSTR[2:0]. The feature must be enabled by configuring at least one $\overline{CSx}$ line for EWAIT.

If the $\overline{EWAIT}$ signal is not asserted, the number of stretch cycles is forced to a minimum of 2 cycles. If $\overline{EWAIT}$ is asserted within the predefined time window during the access it will be strobed active and another stretch cycle is added. If strobed inactive, the next cycle will be the last cycle before the access is finished. $\overline{EWAIT}$ can be held asserted as long as desired to stretch the access.

An access with 1 cycle stretch by $\overline{EWAIT}$ assertion is shown in

- Figure ‘Example 1b: Normal Expanded Mode — Stretched Read Access’
- Figure ‘Example 1b: Normal Expanded Mode — Stretched Write Access’

The associated timing numbers for both operations are given in

- Table ‘Example 1b: Normal Expanded Mode Timing V_{DD5} = 5.0 V (EWAIT enabled)’
- Table ‘Example 1b: Normal Expanded Mode Timing V_{DD5} = 3.0 V (EWAIT enabled)’

It is recommended to use the free-running clock (ECLK) at the fastest rate (bus clock rate) to synchronize the $\overline{\text{EWAIT}}$ input signal.

18.5.2 Emulation Modes

In emulation mode applications, the development systems use a custom PRU device to rebuild the single-chip or expanded bus functions which are lost due to the use of the external bus with an emulator.

Accesses to a set of registers controlling the related ports in normal modes (refer to SoC section) are directed to the external bus in emulation modes which are substituted by PRR as part of the PRU. Accesses to these registers take a constant time of 2 cycles.

Depending on the setting of ROMON and EROMON (refer to S12X_MMC section), the program code can be executed from internal memory or an optional external emulation memory (EMULMEM). No wait state operation (stretching) of the external bus access is done in emulation modes when accessing internal memory or emulation memory addresses.

In both modes observation of the internal operation is supported through the external bus (internal visibility).

18.5.2.1 Example 2a: Emulation Single-Chip Mode

This mode is used for emulation systems in which the target application is operating in normal single-chip mode.

Figure 18-5 shows the PRU connection with the available external bus signals in an emulator application.

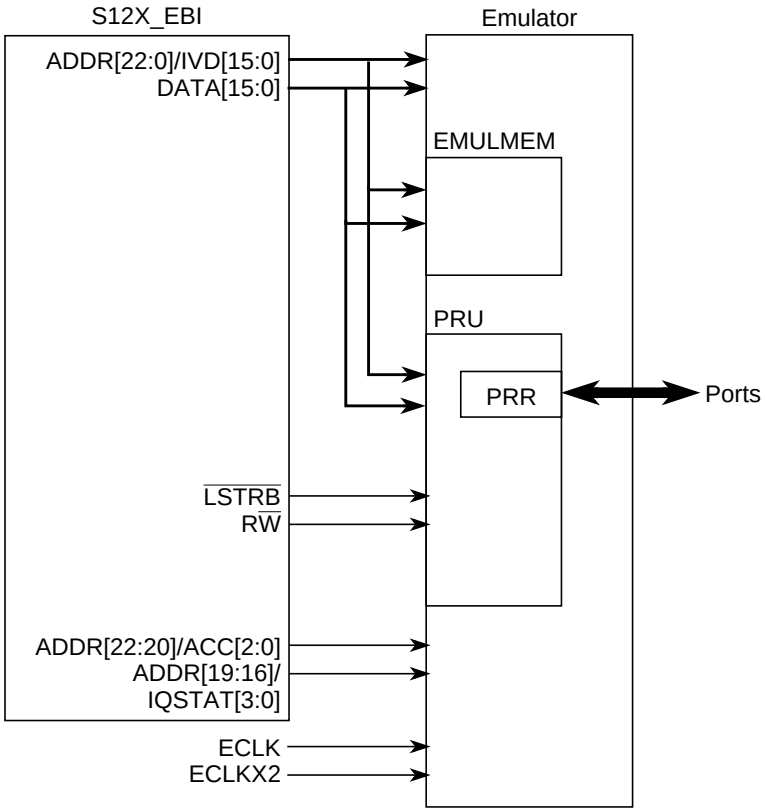


Figure 18-5. Application in Emulation Single-Chip Mode

The timing diagram for this operation is shown in:

- Figure ‘Example 2a: Emulation Single-Chip Mode — Read Followed by Write’

The associated timing numbers are given in:

- Table ‘Example 2a: Emulation Single-Chip Mode Timing (EWAIT disabled)’

Timing considerations:

- Signals muxed with address lines ADDRx, i.e., IVDx, IQSTATx and ACCx, have the same timing.
- $\overline{\text{LSTRB}}$ has the same timing as $\text{R}\overline{\text{W}}$.
- ECLKX2 rising edges have the same timing as ECLK edges.
- The timing for accesses to PRU registers, which take 2 cycles to complete, is the same as the timing for an external non-PRR access with 1 cycle of stretch as shown in example 2b.

18.5.2.2 Example 2b: Emulation Expanded Mode

This mode is used for emulation systems in which the target application is operating in normal expanded mode.

If the external bus is used with a PRU, the external device rebuilds the data select and data direction signals $\overline{\text{UDS}}$, $\overline{\text{LDS}}$, $\overline{\text{RE}}$, and $\overline{\text{WE}}$ from the ADDR0, $\overline{\text{LSTRB}}$, and $\text{R}\overline{\text{W}}$ signals.

Figure 18-6 shows the PRU connection with the available external bus signals in an emulator application.

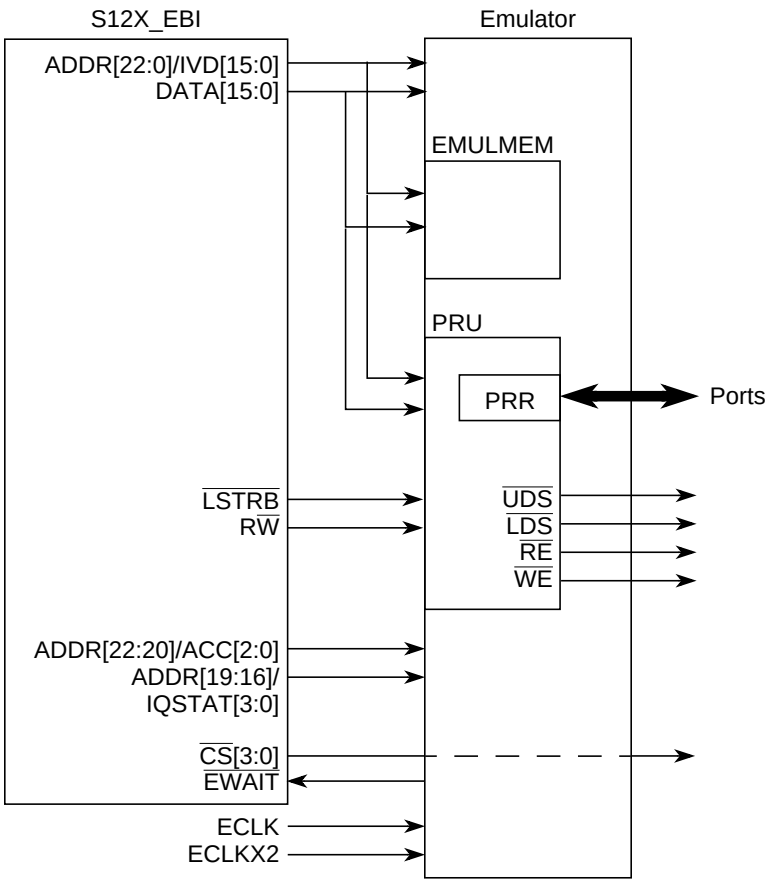


Figure 18-6. Application in Emulation Expanded Mode

The timings of accesses with 1 stretch cycle are shown in

- Figure ‘Example 2b: Emulation Expanded Mode — Read with 1 Stretch Cycle’
- Figure ‘Example 2b: Emulation Expanded Mode — Write with 1 Stretch Cycle’

The associated timing numbers are given in

- Table ‘Example 2b: Emulation Expanded Mode Timing V_{DD5} = 5.0 V (EWAIT disabled)’ (this also includes examples for alternative settings of 2 and 3 additional stretch cycles)

Timing considerations:

- If no stretch cycle is added, the timing is the same as in Emulation Single-Chip Mode.

Chapter 19
Port Integration Module (S12XFPIMV2)

Revision History

Table 19-1. Revision History Table

Rev. No. (Item No.)	Date (Submitted By)	Sections Affected	Substantial Change(s)
V02.01	09 Oct 2006	19.2/19-836	Removed name KWP7 from pin map
V02.02	09 Nov 2006	19.1.1/19-835	Corrected typo
Rev 00.05	02-Nov-2010	19.2/19-836	Removed interrupt function from PJ description

19.1 Introduction

19.1.1 Overview

The S12XF-family Port Integration Module establishes the interface between the peripheral modules including the non-multiplexed External Bus Interface module (S12X_EBI) and the I/O pins for all ports. It controls the electrical pin properties as well as the signal prioritization and multiplexing on shared pins.

This document covers:

- Port A and B used as address output of the S12X_EBI
- Port C and D used as data I/O of the S12X_EBI
- Port E associated with the S12X_EBI control signals and the IRQ, XIRQ interrupt inputs
- Port K associated with address output and control signals of the S12X_EBI
- Port T connected to the Enhanced Capture Timer (ECT) module
- Port S associated with 2 SCI and 1 SPI modules
- Port M associated with 1 SPI, 1 MSCAN, 4 chip select lines and parts of the Pulse Width Modulator with Fault Protection (PMF) module
- Port P connected to parts of the Pulse Width Modulator with Fault Protection (PMF) module
- Port H associated with parts of the FlexRay module
- Port J associated with parts of the FlexRay module and parts of the Pulse Width Modulator with Fault Protection (PMF) module
- Port AD associated with the 16-channel ATD module

Most I/O pins can be configured by register bits to select data direction and drive strength, to enable and select pull-up or pull-down devices.

NOTE

The implementation of the S12XF-family Port Integration Module is device dependent. Therefore some functions are not available on certain derivatives or 112-pin and 64-pin package options.

19.1.2 Features

A full-featured S12XF-family Port Integration Module includes these distinctive registers:

- Data and data direction registers for Ports A, B, C, D, E, K, T, S, M, P, H, J, and AD when used as general-purpose I/O
- Control registers to enable/disable pull-device and select pull-ups/pull-downs on Ports T, S, M, P, H, and J on per-pin basis
- Control registers to enable/disable pull-up devices on Port AD on per-pin basis
- Single control register to enable/disable pull-ups on Ports A, B, C, D, E, and K on per-port basis and on BKGD pin
- Control registers to enable/disable reduced output drive on Ports T, S, M, P, H, J, and AD on per-pin basis
- Single control register to enable/disable reduced output drive on Ports A, B, C, D, E, and K on per-port basis
- Control registers to enable/disable open-drain (wired-or) mode on Ports S and M
- Control register to configure $\overline{\text{IRQ}}$ pin operation
- Free-running clock outputs

A standard port pin has the following minimum features:

- Input/output selection
- 5V output drive with two selectable drive strengths
- 5V digital and analog input
- Input with selectable pull-up or pull-down device

Optional features supported on dedicated pins:

- Open drain for wired-or connections
- Reduced input threshold to support low voltage applications

19.2 External Signal Description

This section lists and describes the signals that do connect off-chip.

Table 19-2 shows all the pins and their functions that are controlled by the S12XFPIMV2. *Refer to the SoC Guide for the availability of the individual pins in the different package options.*

NOTE

If there is more than one function associated with a pin, the priority is indicated by the position in the table from top (highest priority) to bottom (lowest priority).

Table 19-2. Pin Functions and Priorities

Port	Pin Name	Pin Function & Priority[1]	I/O	Description	Pin Function after Reset
-	BKGD	MODC [2]	I	MODC input during $\overline{\text{RESET}}$	BKGD
		BKGD	I/O	S12X_BDM communication pin	
A	PA[7:0]	ADDR[15:8] mux IVD[15:8] [3]	O	High-order external bus address output (multiplexed with IVIS data)	Mode dependent [4]
		GPIO	I/O	General-purpose I/O	
B	PB[7:1]	ADDR[7:1] mux IVD[7:1] [3]	O	Low-order external bus address output (multiplexed with IVIS data)	Mode dependent [4]
		GPIO	I/O	General-purpose I/O	
	PB[0]	ADDR[0] mux IVD0 [3]	O	Low-order external bus address output (multiplexed with IVIS data)	
		$\overline{\text{UDS}}$	O	Upper data strobe	
		GPIO	I/O	General-purpose I/O	
C	PC[7:0]	DATA[15:8]	I/O	High-order bidirectional data input/output Configurable for reduced input threshold	Mode dependent [4]
		GPIO	I/O	General-purpose I/O	
D	PD[7:0]	DATA[7:0]	I/O	Low-order bidirectional data input/output Configurable for reduced input threshold	Mode dependent [4]
		GPIO	I/O	General-purpose I/O	

Port	Pin Name	Pin Function & Priority[1]	I/O	Description	Pin Function after Reset
E	PE[7]	$\overline{\text{XCLKS}}$ [2]	I	External clock selection input during $\overline{\text{RESET}}$	Mode dependent [4]
		ECLKX2	I	Free-running clock output at Core Clock rate (ECLK x 2)	
		GPIO	I/O	General-purpose I/O	
	PE[6]	MODB [2]	I	MODB input during $\overline{\text{RESET}}$	
		$\overline{\text{TAGHI}}$	I	Instruction tagging low pin Configurable for reduced input threshold	
		GPIO	I/O	General-purpose I/O	
	PE[5]	MODA [2]	I	MODA input during $\overline{\text{RESET}}$	
		$\overline{\text{RE}}$	O	Read enable signal	
		$\overline{\text{TAGLO}}$	I	Instruction tagging low pin Configurable for reduced input threshold	
		GPIO	I/O	General-purpose I/O	
	PE[4]	ECLK	O	Free-running clock output at the Bus Clock rate or programmable divided in normal modes	
		GPIO	I/O	General-purpose I/O	
	PE[3]	EROMCTL [2]	I	EROMON bit control input during $\overline{\text{RESET}}$	
		$\overline{\text{LSTRB}}$	O	Low strobe bar output	
		$\overline{\text{LDS}}$	O	Lower data strobe	
		GPIO	I/O	General-purpose I/O	
	PE[2]	R$\overline{\text{W}}$	O	Read/write output for external bus	
		$\overline{\text{WE}}$	O	Write enable signal	
		GPIO	I/O	General-purpose I/O	
	PE[1]	$\overline{\text{IRQ}}$	I	Maskable level- or falling edge-sensitive interrupt input	
		GPI	I	General-purpose input	
	PE[0]	$\overline{\text{XIRQ}}$	I	Non-maskable level-sensitive interrupt input	
		GPI	I	General-purpose input	
K	PK[7]	ROMCTL [2]	I	ROMON bit control input during $\overline{\text{RESET}}$	Mode dependent [4]
		$\overline{\text{EWAIT}}$	I	External Wait signal Configurable for reduced input threshold	
		GPIO	I/O	General-purpose I/O	
	PK[6:4]	ADDR[22:20] mux ACC[2:0] [3]	O	Extended external bus address output (multiplexed with access master output)	
		GPIO	I/O	General-purpose I/O	
	PK[3:0]	ADDR[19:16] mux IQSTAT[3:0] [3]	O	Extended external bus address output (multiplexed with instruction pipe status bits)	
		GPIO	I/O	General-purpose I/O	

Port	Pin Name	Pin Function & Priority[1]	I/O	Description	Pin Function after Reset
T	PT[7:6]	IOC[7:6]	I/O	Enhanced Capture Timer Channels 7 - 6 input/output	GPIO
		GPIO	I/O	General-purpose I/O	
	PT[5]	IOC[5]	I/O	Enhanced Capture Timer Channels 5 input/output	
		VREG_API	O	VREG RC clock test output	
		GPIO	I/O	General-purpose I/O	
	PT[4:0]	IOC[4:0]	I/O	Enhanced Capture Timer Channels 4 - 0 input/output	
		GPIO	I/O	General-purpose I/O	
S	PS7	$\overline{SS0}$	I/O	Serial Peripheral Interface 0 slave select output in master mode, input in slave mode or master mode.	GPIO
		GPIO	I/O	General-purpose I/O	
	PS6	SCK0	I/O	Serial Peripheral Interface 0 serial clock pin	
		GPIO	I/O	General-purpose I/O	
	PS5	MOSI0	I/O	Serial Peripheral Interface 0 master out/slave in pin	
		GPIO	I/O	General-purpose I/O	
	PS4	MISO0	I/O	Serial Peripheral Interface 0 master in/slave out pin	
		GPIO	I/O	General-purpose I/O	
	PS3	TXD1	O	Serial Communication Interface 1 transmit pin	
		GPIO	I/O	General-purpose I/O	
	PS2	RXD1	I	Serial Communication Interface 1 receive pin	
		GPIO	I/O	General-purpose I/O	
	PS1	TXD0	O	Serial Communication Interface 0 transmit pin	
		GPIO	I/O	General-purpose I/O	
	PS0	RXD0	I	Serial Communication Interface 0 receive pin	
		GPIO	I/O	General-purpose I/O	

Port	Pin Name	Pin Function & Priority[1]	I/O	Description	Pin Function after Reset
M	PM7	$\overline{SS1}$	I/O	Serial Peripheral Interface 1 slave select output in master mode, input in slave mode or master mode.	GPIO
		$\overline{CS3}$	O	Chip select 3	
		GPIO	I/O	General-purpose I/O	
	PM6	SCK1	I/O	Serial Peripheral Interface 1 serial clock pin	
		GPIO	I/O	General-purpose I/O	
	PM5	MOSI1	I/O	Serial Peripheral Interface 1 master out/slave in pin	
		GPIO	I/O	General-purpose I/O	
	PM4	MISO1	I/O	Serial Peripheral Interface 1 master in/slave out pin	
		$\overline{CS2}$	O	Chip select 2	
		GPIO	I/O	General-purpose I/O	
	PM3	FAULT3	I	FAULT3 input to PMF	
		$\overline{CS1}$	O	Chip select 1	
		GPIO	I/O	General-purpose I/O	
	PM2	FAULT2	I	FAULT2 input to PMF	
		$\overline{CS0}$	O	Chip select 0	
		GPIO	I/O	General-purpose I/O	
	PM1	TXCAN	O	MSCAN transmit pin	
		GPIO	I/O	General-purpose I/O	
	PM0	RXCAN	I	MSCAN receive pin	
		GPIO	I/O	General-purpose I/O	
P	PP7	FAULT1	I	FAULT1 input to PMF	GPIO
		GPIO	I/O	General-purpose I/O	
	PP6	FAULT0	I	FAULT0 input to PMF	
		GPIO	I/O	General-purpose I/O	
	PP[5:0]	PW0[5:0]	O	Pulse Width Modulator output channels 5 to 0	
		GPIO	I/O	General-purpose I/O	

Port	Pin Name	Pin Function & Priority[1]	I/O	Description	Pin Function after Reset
H	PH7	GPIO	I/O	General-purpose I/O	GPIO
	PH6	$\overline{\text{TXE\_B}}$	O	FlexRay channel B Tx enable, active low	
		GPIO	I/O	General-purpose I/O	
	PH5	TXD_B	O	FlexRay channel B Tx data	
		GPIO	I/O	General-purpose I/O	
	PH4	RXD_B	I	FlexRay channel B Rx data	
		GPIO	I/O	General-purpose I/O	
	PH3	GPIO	I/O	General-purpose I/O	
	PH2	$\overline{\text{TXE\_A}}$	O	FlexRay channel A Tx enable, active low	
		GPIO	I/O	General-purpose I/O	
	PH1	TXD_A	O	FlexRay channel A Tx data	
		GPIO	I/O	General-purpose I/O	
	PH0	RXD_A	I	FlexRay channel A Rx data	
		GPIO	I/O	General-purpose I/O	
J	PJ7	GPIO	I/O	General-purpose I/O	GPIO
	PJ6	STB3	O	FlexRay debug strobe signal 3	
		GPIO	I/O	General-purpose I/O	
	PJ5	STB2	O	FlexRay debug strobe signal 2	
		GPIO	I/O	General-purpose I/O	
	PJ4	STB1	O	FlexRay debug strobe signal 1	
		GPIO	I/O	General-purpose I/O	
	PJ3	STB0	O	FlexRay debug strobe signal 0	
		GPIO	I/O	General-purpose I/O	
	PJ[2:0]	$\overline{\text{IS}}$[2:0]	O	PMF current status outputs 2 to 0	
		GPIO	I/O	General-purpose I/O	
AD	PAD[15:00]	GPIO	I/O	General-purpose I/O	GPIO
		AN[15:0]	I	ATD analog inputs	

1. Signals in brackets denote alternative module routing pins.
2. Function active when $\overline{\text{RESET}}$ asserted.
3. Only available in emulation modes or in Special Test Mode with IVIS on.
4. Refer to S12X_EBI Block Guide.

19.3 Memory Map and Register Definition

This section provides a detailed description of all S12XFPIMV2 registers.

19.3.1 Memory Map

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000 PORTA	R	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
	W								
0x0001 PORTB	R	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
	W								
0x0002 DDRA	R	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0
	W								
0x0003 DDRB	R	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
	W								
0x0004 PORTC	R	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
	W								
0x0005 PORTD	R	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
	W								
0x0006 DDRC	R	DDRC7	DDRC6	DDRC5	DDRC4	DDRC3	DDRC2	DDRC1	DDRC0
	W								
0x0007 DDRD	R	DDRD7	DDRD6	DDRD5	DDRD4	DDRD3	DDRD2	DDRD1	DDRD0
	W								
0x0008 PORTE	R	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
	W								
0x0009 DDRE	R	DDRE7	DDRE6	DDRE5	DDRE4	DDRE3	DDRE2	0	0
	W								
0x000A 0x000B Non-PIM Address Range	R	Non-PIM Address Range							
	W								
0x000C PUCR	R	PUPKE	BKPUE	0	PUPEE	PUPDE	PUPCE	PUPBE	PUPAE
	W								
0x000D RDRIV	R	RDPK	0	0	RDPE	RDPD	RDPC	RDPB	RDPA
	W								

= Unimplemented or Reserved

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x000E–0x001B Non-PIM Address Range	R W	Non-PIM Address Range							
0x001C ECLKCTL	R	NECLK	NCLKX2	DIV16	EDIV4	EDIV3	EDIV2	EDIV1	EDIV0
	W								
0x001D Reserved	R	0	0	0	0	0	0	0	0
	W								
0x001E IRQCR	R	IRQE	IRQEN	0	0	0	0	0	0
	W								
0x001F Reserved	R	0	0	0	0	0	0	0	0
	W								
0x0020–0x0031 Non-PIM Address Range	R W	Non-PIM Address Range							
0x0032 PORTK	R	PK7	PK6	PK5	PK4	PK3	PK2	PK1	PK0
	W								
0x0033 DDRK	R	DDRK7	DDRK6	DDRK5	DDRK4	DDRK3	DDRK2	DDRK1	DDRK0
	W								
0x0034–0x023F Non-PIM Address Range	R W	Non-PIM Address Range							
0x0240 PTT	R	PTT7	PTT6	PTT5	PTT4	PTT3	PTT2	PTT1	PTT0
	W								
0x0241 PTIT	R	PTIT7	PTIT6	PTIT5	PTIT4	PTIT3	PTIT2	PTIT1	PTIT0
	W								
0x0242 DDRT	R	DDRT7	DDRT6	DDRT5	DDRT4	DDRT3	DDRT2	DDRT1	DDRT0
	W								
0x0243 RDRT	R	RDRT7	RDRT6	RDRT5	RDRT4	RDRT3	RDRT2	RDRT1	RDRT0
	W								

= Unimplemented or Reserved

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0244 PERT	R	PERT7	PERT6	PERT5	PERT4	PERT3	PERT2	PERT1	PERT0
	W								
0x0245 PPST	R	PPST7	PPST6	PPST5	PPST4	PPST3	PPST2	PPST1	PPST0
	W								
0x0246 Reserved	R	0	0	0	0	0	0	0	0
	W								
0x0247 Reserved	R	0	0	0	0	0	0	0	0
	W								
0x0248 PTS	R	PTS7	PTS6	PTS5	PTS4	PTS3	PTS2	PTS1	PTS0
	W								
0x0249 PTIS	R	PTIS7	PTIS6	PTIS5	PTIS4	PTIS3	PTIS2	PTIS1	PTIS0
	W								
0x024A DDRS	R	DDRS7	DDRS6	DDRS5	DDRS4	DDRS3	DDRS2	DDRS1	DDRS0
	W								
0x024B RDRS	R	RDRS7	RDRS6	RDRS5	RDRS4	RDRS3	RDRS2	RDRS1	RDRS0
	W								
0x024C PERS	R	PERS7	PERS6	PERS5	PERS4	PERS3	PERS2	PERS1	PERS0
	W								
0x024D PPSS	R	PPSS7	PPSS6	PPSS5	PPSS4	PPSS3	PPSS2	PPSS1	PPSS0
	W								
0x024E WOMS	R	WOMS7	WOMS6	WOMS5	WOMS4	WOMS3	WOMS2	WOMS1	WOMS0
	W								
0x024F Reserved	R	0	0	0	0	0	0	0	0
	W								
0x0250 PTM	R	PTM7	PTM6	PTM5	PTM4	PTM3	PTM2	PTM1	PTM0
	W								
0x0251 PTIM	R	PTIM7	PTIM6	PTIM5	PTIM4	PTIM3	PTIM2	PTIM1	PTIM0
	W								
0x0252 DDRM	R	DDRM7	DDRM6	DDRM5	DDRM4	DDRM3	DDRM2	DDRM1	DDRM0
	W								

= Unimplemented or Reserved

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0253 RDRM	R	RDRM7	RDRM6	RDRM5	RDRM4	RDRM3	RDRM2	RDRM1	RDRM0
	W								
0x0254 PERM	R	PERM7	PERM6	PERM5	PERM4	PERM3	PERM2	PERM1	PERM0
	W								
0x0255 PPSM	R	PPSM7	PPSM6	PPSM5	PPSM4	PPSM3	PPSM2	PPSM1	PPSM0
	W								
0x0256 WOMM	R	WOMM7	WOMM6	WOMM5	WOMM4	WOMM3	WOMM2	WOMM1	WOMM0
	W								
0x0257 Reserved	R	0	0	0	0	0	0	0	0
	W								
0x0258 PTP	R	PTP7	PTP6	PTP5	PTP4	PTP3	PTP2	PTP1	PTP0
	W								
0x0259 PTIP	R	PTIP7	PTIP6	PTIP5	PTIP4	PTIP3	PTIP2	PTIP1	PTIP0
	W								
0x025A DDRP	R	DDRP7	DDRP6	DDRP5	DDRP4	DDRP3	DDRP2	DDRP1	DDRP0
	W								
0x025B RDRP	R	RDRP7	RDRP6	RDRP5	RDRP4	RDRP3	RDRP2	RDRP1	RDRP0
	W								
0x025C PERP	R	PERP7	PERP6	PERP5	PERP4	PERP3	PERP2	PERP1	PERP0
	W								
0x025D PPSP	R	PPSP7	PPSP6	PPSP5	PPSP4	PPSP3	PPSP2	PPSP1	PPSP0
	W								
0x025E Reserved	R	0	0	0	0	0	0	0	0
	W								
0x025F Reserved	R	0	0	0	0	0	0	0	0
	W								
0x0260 PTH	R	PTH7	PTH6	PTH5	PTH4	PTH3	PTH2	PTH1	PTH0
	W								
0x0261 PTIH	R	PTIH7	PTIH6	PTIH5	PTIH4	PTIH3	PTIH2	PTIH1	PTIH0
	W								

= Unimplemented or Reserved

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0262 DDRH	R	DDRH7	DDRH6	DDRH5	DDRH4	DDRH3	DDRH2	DDRH1	DDRH0
	W								
0x0263 RDRH	R	RDRH7	RDRH6	RDRH5	RDRH4	RDRH3	RDRH2	RDRH1	RDRH0
	W								
0x0264 PERH	R	PERH7	PERH6	PERH5	PERH4	PERH3	PERH2	PERH1	PERH0
	W								
0x0265 PPSH	R	PPSH7	PPSH6	PPSH5	PPSH4	PPSH3	PPSH2	PPSH1	PPSH0
	W								
0x0266 Reserved	R	0	0	0	0	0	0	0	0
	W								
0x0267 Reserved	R	0	0	0	0	0	0	0	0
	W								
0x0268 PTJ	R	PTJ7	PTJ6	PTJ5	PTJ4	PTJ3	PTJ2	PTJ1	PTJ0
	W								
0x0269 PTIJ	R	PTIJ7	PTIJ6	PTIJ5	PTIJ4	PTIJ3	PTIJ2	PTIJ1	PTIJ0
	W								
0x026A DDRJ	R	DDRJ7	DDRJ6	DDRJ5	DDRJ4	DDRJ3	DDRJ2	DDRJ1	DDRJ0
	W								
0x026B RDRJ	R	RDRJ7	RDRJ6	RDRJ5	RDRJ4	RDRJ3	RDRJ2	RDRJ1	RDRJ0
	W								
0x026C PERJ	R	PERJ7	PERJ6	PERJ5	PERJ4	PERJ3	PERJ2	PERJ1	PERJ0
	W								
0x026D PPSJ	R	PPSJ7	PPSJ6	PPSJ5	PPSJ4	PPSJ3	PPSJ2	PPSJ1	PPSJ0
	W								
0x026E Reserved	R	0	0	0	0	0	0	0	0
	W								
0x026F Reserved	R	0	0	0	0	0	0	0	0
	W								
0x0270 PT0AD	R	PT0AD15	PT0AD14	PT0AD13	PT0AD12	PT0AD11	PT0AD10	PT0AD9	PT0AD8
	W								

= Unimplemented or Reserved

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0271 PT1AD	R W	PT1AD7	PT1AD6	PT1AD5	PT1AD4	PT1AD3	PT1AD2	PT1AD1	PT1AD0
0x0272 DDR0AD	R W	DDR0AD15	DDR0AD14	DDR0AD13	DDR0AD12	DDR0AD11	DDR0AD10	DDR0AD9	DDR0AD8
0x0273 DDR1AD	R W	DDR1AD7	DDR1AD6	DDR1AD5	DDR1AD4	DDR1AD3	DDR1AD2	DDR1AD1	DDR1AD0
0x0274 RDR0AD	R W	RDR0AD15	RDR0AD14	RDR0AD13	RDR0AD12	RDR0AD11	RDR0AD10	RDR0AD9	RDR0AD8
0x0275 RDR1AD	R W	RDR1AD7	RDR1AD6	RDR1AD5	RDR1AD4	RDR1AD3	RDR1AD2	RDR1AD1	RDR1AD0
0x0276 PER0AD	R W	PER0AD15	PER0AD14	PER0AD13	PER0AD12	PER0AD11	PER0AD10	PER0AD9	PER0AD8
0x0277 PER1AD	R W	PER1AD7	PER1AD6	PER1AD5	PER1AD4	PER1AD3	PER1AD2	PER1AD1	PER1AD0

= Unimplemented or Reserved

19.3.2 Register Descriptions

The following table summarizes the effect of the various configuration bits, i.e. data direction (DDR), output level (IO), reduced drive (RDR), pull enable (PE), pull select (PS) on the pin function and pull device activity.

Table 19-3. Pin Configuration Summary

DDR	IO	RDR	PE	PS (1)	Function	Pull Device
0	X	X	0	X	Input	Disabled
0	X	X	1	0	Input	Pull Up
0	X	X	1	1	Input	Pull Down
0	X	X	0	0	Input	Disabled
0	X	X	0	1	Input	Disabled
0	X	X	1	0	Input	Pull Up
0	X	X	1	1	Input	Pull Down
1	0	0	X	X	Output, full drive to 0	Disabled
1	1	0	X	X	Output, full drive to 1	Disabled
1	0	1	X	X	Output, reduced drive to 0	Disabled
1	1	1	X	X	Output, reduced drive to 1	Disabled
1	0	0	X	0	Output, full drive to 0	Disabled
1	1	0	X	1	Output, full drive to 1	Disabled
1	0	1	X	0	Output, reduced drive to 0	Disabled
1	1	1	X	1	Output, reduced drive to 1	Disabled

1. Always “0” on Port A, B, C, D, E, K, and AD.

NOTE

All register bits in this module are completely synchronous to internal clocks during a register read.

19.3.3 Port A Data Register (PORTA)

Address 0x0000 (PRR) Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
Altern. Function	ADDR15 mux IVD15	ADDR14 mux IVD14	ADDR13 mux IVD13	ADDR12 mux IVD12	ADDR11 mux IVD11	ADDR10 mux IVD10	ADDR9 mux IVD9	ADDR8 mux IVD8
Reset	0	0	0	0	0	0	0	0

Figure 19-1. Port A Data Register (PORTA)

1. Read: Anytime. In emulation modes, read operations will return the data from the external bus, in all other modes the data source is depending on the data direction value.
   Write: Anytime. In emulation modes, write operations will also be directed to the external bus.

Table 19-4. PORTA Register Field Descriptions

Field	Description
7-0 PA	**Port A general purpose input/output data**—Data Register Port A pins 7 through 0 are associated with address outputs ADDR[15:8] respectively in expanded modes. In emulation modes the address is multiplexed with IVD[15:8]. When not used with the alternative function, these pins can be used as general purpose I/O. If the associated data direction bits of these pins are set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.

19.3.4 Port B Data Register (PORTB)

Address 0x0001 (PRR) Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
Altern. Function	ADDR7 mux IVD7	ADDR6 mux IVD6	ADDR5 mux IVD5	ADDR4 mux IVD4	ADDR3 mux IVD3	ADDR2 mux IVD2	ADDR1 mux IVD1	ADDR0 mux IVD0 or $\overline{\text{UDS}}$
Reset	0	0	0	0	0	0	0	0

Figure 19-2. Port B Data Register (PORTB)

1. Read: Anytime. In emulation modes, read operations will return the data from the external bus, in all other modes the data source is depending on the data direction value.
   Write: Anytime. In emulation modes, write operations will also be directed to the external bus.

Table 19-5. PORTB Register Field Descriptions

Field	Description
7-0 PB	**Port B general purpose input/output data**—Data Register Port B pins 7 through 0 are associated with address outputs ADDR[7:0] respectively in expanded modes. In emulation modes the address is multiplexed with IVD[7:0]. In normal expanded mode pin 0 is related to the $\overline{\text{UDS}}$ input. When not used with the alternative function, these pins can be used as general purpose I/O. If the associated data direction bits of these pins are set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.

19.3.5 Port A Data Direction Register (DDRA)

Address 0x0002 (PRR) Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0
Reset	0	0	0	0	0	0	0	0

Figure 19-3. Port A Data Direction Register (DDRA)

1. Read: Anytime. In emulation modes, read operations will return the data from the external bus, in all other modes the data source is depending on the data direction value.
   Write: Anytime. In emulation modes, write operations will also be directed to the external bus.

Table 19-6. DDRA Register Field Descriptions

Field	Description
7-0 DDRA	**Port A Data Direction**— This register controls the data direction of pins 7 through 0. The external bus function forces the I/O state to be outputs for all associated pins. In this case the data direction bits will not change. When operating a pin as a general purpose I/O, the associated data direction bit determines whether it is an input or output. 1 Associated pin is configured as output. 0 Associated pin is configured as high-impedance input.

19.3.6 Port B Data Direction Register (DDRB)

Address 0x0003 (PRR) Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
Reset	0	0	0	0	0	0	0	0

Figure 19-4. Port B Data Direction Register (DDRB)

1. Read: Anytime. In emulation modes, read operations will return the data from the external bus, in all other modes the data source is depending on the data direction value.
   Write: Anytime. In emulation modes, write operations will also be directed to the external bus.

Table 19-7. DDRB Register Field Descriptions

Field	Description
7-0 DDRB	**Port B Data Direction**— This register controls the data direction of pins 7 through 0. The external bus function forces the I/O state to be outputs for all associated pins. In this case the data direction bits will not change. When operating a pin as a general purpose I/O, the associated data direction bit determines whether it is an input or output. 1 Associated pin is configured as output. 0 Associated pin is configured as high-impedance input.

19.3.7 Port C Data Register (PORTC)

Address 0x0004 (PRR) Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
Altern. Function	DATA15	DATA14	DATA13	DATA12	DATA11	DATA10	DATA9	DATA8
Reset	0	0	0	0	0	0	0	0

Figure 19-5. Port C Data Register (PORTC)

1. Read: Anytime. In emulation modes, read operations will return the data from the external bus, in all other modes the data source is depending on the data direction value.
   Write: Anytime. In emulation modes, write operations will also be directed to the external bus.

Table 19-8. PORTC Register Field Descriptions

Field	Description
7-0 PC	**Port C general purpose input/output data**—Data Register Port C pins 7 through 0 are associated with data I/O lines DATA[15:8] respectively in expanded modes. When not used with the alternative function, these pins can be used as general purpose I/O. If the associated data direction bits of these pins are set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.

19.3.8 Port D Data Register (PORTD)

Address 0x0005 (PRR) Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
Altern. Function	DATA7	DATA6	DATA5	DATA4	DATA3	DATA2	DATA1	DATA0
Reset	0	0	0	0	0	0	0	0

Figure 19-6. Port D Data Register (PORTD)

1. Read: Anytime. In emulation modes, read operations will return the data from the external bus, in all other modes the data source is depending on the data direction value.
   Write: Anytime. In emulation modes, write operations will also be directed to the external bus.

Table 19-9. PORTD Register Field Descriptions

Field	Description
7-0 PD	**Port D general purpose input/output data**—Data Register Port D pins 7 through 0 are associated with data I/O lines DATA[7:0] respectively in expanded modes. When not used with the alternative function, these pins can be used as general purpose I/O. If the associated data direction bits of these pins are set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.

19.3.9 Port C Data Direction Register (DDRC)

Address 0x0006 (PRR) Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	DDRC7	DDRC6	DDRC5	DDRC4	DDRC3	DDRC2	DDRC1	DDRC0
Reset	0	0	0	0	0	0	0	0

Figure 19-7. Port C Data Direction Register (DDRC)

1. Read: Anytime. In emulation modes, read operations will return the data from the external bus, in all other modes the data source is depending on the data direction value.
   Write: Anytime. In emulation modes, write operations will also be directed to the external bus.

Table 19-10. DDRC Register Field Descriptions

Field	Description
7-0 DDRC	Port C Data Direction— This register controls the data direction of pins 7 through 0. The external bus function controls the data direction for the associated pins. In this case the data direction bits will not change. When operating a pin as a general purpose I/O, the associated data direction bit determines whether it is an input or output. 1 Associated pin is configured as output. 0 Associated pin is configured as high-impedance input.

19.3.10 Port D Data Direction Register (DDRD)

Address 0x0007 (PRR) Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	DDRD7	DDRD6	DDRD5	DDRD4	DDRD3	DDRD2	DDRD1	DDRD0
Reset	0	0	0	0	0	0	0	0

Figure 19-8. Port D Data Direction Register (DDRD)

1. Read: Anytime. In emulation modes, read operations will return the data from the external bus, in all other modes the data source is depending on the data direction value.
   Write: Anytime. In emulation modes, write operations will also be directed to the external bus.

Table 19-11. DDRD Register Field Descriptions

Field	Description
7-0 DDRD	**Port D Data Direction**— This register controls the data direction of pins 7 through 0. When used with the external bus this function controls the data direction for the associated pins. In this case the data direction bits will not change. When operating a pin as a general purpose I/O, the associated data direction bit determines whether it is an input or output. 1 Associated pin is configured as output. 0 Associated pin is configured as high-impedance input.

19.3.11 Port E Data Register (PORTE)

Address 0x0008 (PRR) Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
W								
Altern. Function	$\overline{XCLKS}$ or ECLKX2	MODB or $\overline{TAGHI}$	MODA or $\overline{RE}$ or $\overline{TAGLO}$	ECLK	EROMCTL or $\overline{LSTRB}$ or $\overline{LDS}$	R$\overline{W}$ or $\overline{WE}$	$\overline{IRQ}$	$\overline{XIRQ}$
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 19-9. Port E Data Register (PORTE)

1. Read: Anytime. In emulation modes, read operations will return the data from the external bus, in all other modes the data source is depending on the data direction value.
   Write: Anytime. In emulation modes, write operations will also be directed to the external bus.

Table 19-12. PORTE Register Field Descriptions

Field	Description
7-0 PE	**Port E general purpose input/output data**—Data Register Port E bits 7 through 0 are associated with external bus control signals and interrupt inputs. These include mode select (MODB, MODA), E clock, double frequency E clock, Instruction Tagging High and Low ($\overline{TAGHI}$, $\overline{TAGLO}$), Read/Write (R$\overline{W}$), Read Enable and Write Enable ($\overline{RE}$, $\overline{WE}$), Lower Data Select ($\overline{LDS}$), $\overline{IRQ}$, and $\overline{XIRQ}$. When not used with the alternative functions, Port E pins 7-2 can be used as general purpose I/O and pins 1-0 can be used as general purpose inputs. If the associated data direction bits of these pins are set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read. Pins 6 and 5 are inputs with enabled pull-down devices while RESET pin is low. Pins 7 and 3 are inputs with enabled pull-up devices while RESET pin is low.

19.3.12 Port E Data Direction Register (DDRE)

Address 0x0009 (PRR) Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	DDRE7	DDRE6	DDRE5	DDRE4	DDRE3	DDRE2	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 19-10. Port E Data Direction Register (DDRE)

1. Read: Anytime. In emulation modes, read operations will return the data from the external bus, in all other modes the data source is depending on the data direction value.
   Write: Anytime. In emulation modes, write operations will also be directed to the external bus.

Table 19-13. DDRE Register Field Descriptions

Field	Description
7-2 DDRE	**Port E Data Direction**— This register controls the data direction of pins 7 through 2. The external bus function controls the data direction for the associated pins. In this case the data direction bits will not change. When operating a pin as a general purpose I/O, the associated data direction bit determines whether it is an input or output. 1 Associated pin is configured as output. 0 Associated pin is configured as high-impedance input.
1-0	**Reserved**— Port E bit 1 (associated with $\overline{IRQ}$) and bit 0 (associated with $\overline{XIRQ}$) cannot be configured as outputs. Port E, bits 1 and 0, can be read regardless of whether the alternate interrupt function is enabled.

19.3.13 S12X_EBI ports, BKGD pin Pull-up Control Register (PUCR)

Address 0x000C (PRR) Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	PUPKE	BKPUE	0	PUPEE	PUPDE	PUPCE	PUPBE	PUPAE
W								
Reset	1	1	0	1	0	0	0	0

= Unimplemented or Reserved

Figure 19-11. S12X_EBI ports, BKGD pin Pull-up Control Register (PUCR)

1. Read:Anytime in single-chip modes.
   Write:Anytime, except BKPUE which is writable in Special Test Mode only.

Table 19-14. PUCR Register Field Descriptions

Field	Description
7 PUPKE	**Pull-up Port K Enable**—Enable pull-up devices on all Port K input pins This bit configures whether pull-up devices are activated, if the pins are used as inputs. This bit has no effect if the pins are used as outputs. Out of reset the pull-up devices are enabled. 1 Pull-up devices enabled. 0 Pull-up devices disabled.
6 BKPUE	**BKGD pin pull-up Enable**—Enable pull-up devices on BKGD pin This bit configures whether a pull-up device is activated, if the pin is used as input. This bit has no effect if the pin is used as outputs. Out of reset the pull-up device is enabled. 1 Pull-up device enabled. 0 Pull-up device disabled.
5	**Reserved**—
4 PUPEE	**Pull-up Port E Enable**—Enable pull-up devices on all Port E input pins except on pins 5 and 6 which have pull-down devices only enabled during reset. This bit has no effect on these pins. This bit configures whether pull-up devices are activated, if the pins are used as inputs. This bit has no effect if the pins are used as outputs. Out of reset the pull-up devices are enabled. 1 Pull-up devices enabled. 0 Pull-up devices disabled.

Table 19-14. PUCR Register Field Descriptions (continued)

Field	Description
3 PUPDE	**Pull-up Port D Enable**—Enable pull-up devices on all Port D input pins This bit configures whether pull-up devices are activated, if the pins are used as inputs. This bit has no effect if the pins are used as outputs. Out of reset the pull-up devices are disabled. 1 Pull-up devices enabled. 0 Pull-up devices disabled.
2 PUPCE	**Pull-up Port C Enable**—Enable pull-up devices on all Port C input pins This bit configures whether pull-up devices are activated, if the pins are used as inputs. This bit has no effect if the pins are used as outputs. Out of reset the pull-up devices are disabled. 1 Pull-up devices enabled. 0 Pull-up devices disabled.
1 PUPBE	**Pull-up Port B Enable**—Enable pull-up devices on all Port B input pins This bit configures whether pull-up devices are activated, if the pins are used as inputs. This bit has no effect if the pins are used as outputs. Out of reset the pull-up devices are disabled. 1 Pull-up devices enabled. 0 Pull-up devices disabled.
0 PUPAE	**Pull-up Port A Enable**—Enable pull-up devices on all Port A input pins This bit configures whether pull-up devices are activated, if the pins are used as inputs. This bit has no effect if the pins are used as outputs. Out of reset the pull-up devices are disabled. 1 Pull-up devices enabled. 0 Pull-up devices disabled.

19.3.14 S12X_EBI ports Reduced Drive Register (RDRIV)

Address 0x000D (PRR) Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	RDPK	0	0	RDPE	RDPD	RDPC	RDPB	RDPA
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved u = Unaffected by reset

Figure 19-12. S12X_EBI ports Reduced Drive Register (RDRIV)

1. Read: Anytime. In emulation modes, read operations will return the data from the external bus, in all other modes the data source is depending on the data direction value.
   Write: Anytime. In emulation modes, write operations will also be directed to the external bus.

This register is used to select reduced drive for the pins associated with the S12X_EBI ports A, B, C, D, E, and K. If enabled, the pins drive at about 1/6 of the full drive strength. The reduced drive function is independent of which function is being used on a particular pin.

The reduced drive functionality does not take effect on the pins in emulation modes.

Table 19-15. RDRIV Register Field Descriptions

Field	Description
7 RDPK	Port K **reduced drive**—Select reduced drive for outputs This bit configures the drive strength of all Port K output pins as either full or reduced. If a pin is used as input this bit has no effect. 1 Reduced drive selected (1/6 of the full drive strength). 0 Full drive strength enabled.
6-5	**Reserved**—
4 RDPE	Port E reduced drive—Select reduced drive for outputs This bit configures the drive strength of all Port E output pins as either full or reduced. If a pin is used as input this bit has no effect. 1 Reduced drive selected (1/6 of the full drive strength). 0 Full drive strength enabled.
3 RDPD	Port D reduced drive—Select reduced drive for outputs This bit configures the drive strength of all output pins as either full or reduced. If a pin is used as input this bit has no effect. 1 Reduced drive selected (1/6 of the full drive strength). 0 Full drive strength enabled.
2 RDPC	Port C reduced drive—Select reduced drive for outputs This bit configures the drive strength of all output pins as either full or reduced. If a pin is used as input this bit has no effect. 1 Reduced drive selected (1/6 of the full drive strength). 0 Full drive strength enabled.
1 RDPB	Port B reduced drive—Select reduced drive for outputs This bit configures the drive strength of all output pins as either full or reduced. If a pin is used as input this bit has no effect. 1 Reduced drive selected (1/6 of the full drive strength). 0 Full drive strength enabled.
0 RDPA	Port A reduced drive—Select reduced drive for outputs This bit configures the drive strength of all output pins as either full or reduced. If a pin is used as input this bit has no effect. 1 Reduced drive selected (1/6 of the full drive strength). 0 Full drive strength enabled.

19.3.15 ECLK Control Register (ECLKCTL)

Address 0x001C (PRR) Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	NECLK	NCLKX2	DIV16	EDIV4	EDIV3	EDIV2	EDIV1	EDIV0
Reset(2):	Mode Depen-dent	1	0	0	0	0	0	0
SS	0	1	0	0	0	0	0	0
ES	1	1	0	0	0	0	0	0
ST	0	1	0	0	0	0	0	0
EX	0	1	0	0	0	0	0	0
NS	1	1	0	0	0	0	0	0
NX	0	1	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 19-13. ECLK Control Register (ECLKCTL)

1. Read: Anytime. In emulation modes, read operations will return the data from the external bus, in all other modes the data source is depending on the data direction value.
   Write: Anytime. In emulation modes, write operations will also be directed to the external bus.
2. Reset values in emulation modes are identical to those of the target mode.

The ECLKCTL register is used to control the availability of the free-running clocks and the free-running clock divider.

Table 19-16. ECLKCTL Register Field Descriptions

Field	Description
7 NECLK	**No ECLK**—Disable ECLK output This bit controls the availability of a free-running clock on the ECLK pin. Clock output is always active in emulation modes and if enabled in all other operating modes. 1 ECLK disabled 0 ECLK enabled
6 NCLKX2	**No ECLKX2**—Disable ECLKX2 output This bit controls the availability of a free-running clock on the ECLKX2 pin. This clock has a fixed rate of twice the internal Bus Clock. Clock output is always active in emulation modes and if enabled in all other operating modes. 1 ECLKX2 disabled 0 ECLKX2 enabled

Table 19-16. ECLKCTL Register Field Descriptions (continued)

Field	Description
5 DIV16	**Free-running ECLK predivider**—Divide by 16 This bit enables a divide-by-16 stage on the selected EDIV rate. 1 Divider enabled: ECLK rate = EDIV rate divided by 16 0 Divider disabled: ECLK rate = EDIV rate
4-0 EDIV	**Free-running ECLK Divider**—Configure ECLK rate These bits determine the rate of the free-running clock on the ECLK pin. Divider is always disabled in emulation modes and active as programmed in all other operating modes. 00000 ECLK rate = Bus Clock rate 00001 ECLK rate = Bus Clock rate divided by 2 00010 ECLK rate = Bus Clock rate divided by 3, ... 11111 ECLK rate = Bus Clock rate divided by 32

19.3.16 PIM Reserved Register

Address 0x001D (PRR) Access: User read(1)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 19-14. PIM Reserved Register

1. Read: Always reads 0x00
   Write: Unimplemented

19.3.17 IRQ Control Register (IRQCR)

Address 0x001E Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	IRQE	IRQEN	0	0	0	0	0	0
W								
Reset	0	1	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 19-15. IRQ Control Register (IRQCR)

1. Read: See individual bit descriptions below.
   Write: See individual bit descriptions below.

Table 19-17. IRQCR Register Field Descriptions

Field	Description
7 IRQE	**IRQ select edge sensitive only**— Special modes: Read or write anytime. Normal & emulation modes: Read anytime, write once. 1 $\overline{IRQ}$ configured to respond only to falling edges. Falling edges on the $\overline{IRQ}$ pin will be detected anytime IRQE = 1 and will be cleared only upon a reset or the servicing of the $\overline{IRQ}$ interrupt. 0 $\overline{IRQ}$ configured for low level recognition.
6 IRQEN	**External IRQ enable**— Read or write anytime. 1 External IRQ pin is connected to interrupt logic. 0 External IRQ pin is disconnected from interrupt logic.
5-0	**Reserved**—

19.3.18 PIM Reserved Register

Address 0x001F — Access: User read(1)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 19-16. PIM Reserved Register

1. Read: Always reads 0x00
   Write: Unimplemented

19.3.19 Port K Data Register (PORTK)

Address 0x0032 (PRR) — Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PK7	PK6	PK5	PK4	PK3	PK2	PK1	PK0
Altern. Function	ROMCTL or $\overline{EWAIT}$	ADDR22 mux NOACC	ADDR21	ADDR20	ADDR19 mux IQSTAT3	ADDR18 mux IQSTAT2	ADDR17 mux IQSTAT1	ADDR16 mux IQSTAT0
Reset	0	0	0	0	0	0	0	0

Figure 19-17. Port K Data Register (PORTK)

1. Read: Anytime. In emulation modes, read operations will return the data from the external bus, in all other modes the data source is depending on the data direction value.
   Write: Anytime. In emulation modes, write operations will also be directed to the external bus.

Table 19-18. PORTK Register Field Descriptions

Field	Description
7-0 PK	**Port K general purpose input/output data**—Data Register Port K pins 7 through 0 are associated with external bus control signals and internal memory expansion emulation pins. These include ADDR[22:16], No-Access (NOACC), External Wait ($\overline{\text{EWAIT}}$) and instruction pipe signals IQSTAT[3:0]. Bits 6-0 carry the external addresses in all expanded modes. In emulation modes the address is multiplexed with the alternate functions NOACC and IQSTAT on the respective pins. When not used with the alternative function, these pins can be used as general purpose I/O. If the associated data direction bits of these pins are set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.

19.3.20 Port K Data Direction Register (DDRK)

Address 0x0033 (PRR) Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	DDRK7	DDRK6	DDRK5	DDRK4	DDRK3	DDRK2	DDRK1	DDRK0
Reset	0	0	0	0	0	0	0	0

Figure 19-18. Port K Data Direction Register (DDRK)

1. Read: Anytime. In emulation modes, read operations will return the data from the external bus, in all other modes the data source is depending on the data direction value.
   Write: Anytime. In emulation modes, write operations will also be directed to the external bus.

Table 19-19. DDRK Register Field Descriptions

Field	Description
7-0 DDRK	**Port K Data Direction**— This register controls the data direction of pins 7 through 0. The external bus function controls the data direction for the associated pins. In this case the data direction bits will not change. When operating a pin as a general purpose I/O, the associated data direction bit determines whether it is an input or output. 1 Associated pin is configured as output. 0 Associated pin is configured as high-impedance input.

19.3.21 Port T Data Register (PTT)

Address 0x0240 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PTT7	PTT6	PTT5	PTT4	PTT3	PTT2	PTT1	PTT0
Altern. Function	IOC7	IOC6	IOC5	IOC4	IOC3	IOC2	IOC1	IOC0
	—	—	VREG_API	—	—	—	—	—
Reset	0	0	0	0	0	0	0	0

Figure 19-19. Port T Data Register (PTT)

1. Read: Anytime.
   Write: Anytime.

Table 19-20. PTT Register Field Descriptions

Field	Description
7-6 PTT	**Port T general purpose input/output data**—Data Register Port T pins 7 through 0 are associated with ECT channels IOC7 and IOC6. When not used with the alternative function, these pins can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
5 PTT	**Port T general purpose input/output data**—Data Register Port T pins 5 is associated with ECT channel IOC5 and the VREG_API output. The ECT function takes precedence over the VREG_API and the general purpose I/O function if the related channel is enabled. When not used with the alternative function, these pins can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
4-0 PTT	**Port T general purpose input/output data**—Data Register Port T pins 4 through 0 are associated with ECT channels IOC4 through IOC0. When not used with the alternative function, these pins can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.

19.3.22 Port T Input Register (PTIT)

Address 0x0241 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	PTIT7	PTIT6	PTIT5	PTIT4	PTIT3	PTIT2	PTIT1	PTIT0
W								
Reset	u	u	u	u	u	u	u	u

= Unimplemented or Reserved u = Unaffected by reset

Figure 19-20. Port T Input Register (PTIT)

1. Read: Anytime.
   Write:Never, writes to this register have no effect.

Table 19-21. PTIT Register Field Descriptions

Field	Description
7-0 PTIT	**Port T input data**— This register always reads back the buffered state of the associated pins. This can also be used to detect overload or short circuit conditions on output pins.

19.3.23 Port T Data Direction Register (DDRT)

Address 0x0242 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	DDRT7	DDRT6	DDRT5	DDRT4	DDRT3	DDRT2	DDRT1	DDRT0
Reset	0	0	0	0	0	0	0	0

Figure 19-21. Port T Data Direction Register (DDRT)

1. Read: Anytime.
   Write: Anytime.

Table 19-22. DDRT Register Field Descriptions

Field	Description
7-0 DDRT	**Port T data direction**— This register controls the data direction of pins 7 through 0. The ECT forces the I/O state to be an output for each timer port associated with an enabled output compare. In this case the data direction bits will not change. The data direction bits revert to controlling the I/O direction of a pin when the associated timer output compare is disabled. The timer Input Capture always monitors the state of the pin. 1 Associated pin is configured as output. 0 Associated pin is configured as high-impedance input.

NOTE

Due to internal synchronization circuits, it can take up to 2 bus clock cycles until the correct value is read on PTT or PTIT registers, when changing the DDRT register.

19.3.24 Port T Reduced Drive Register (RDRT)

Address 0x0243 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	RDRT7	RDRT6	RDRT5	RDRT4	RDRT3	RDRT2	RDRT1	RDRT0
Reset	0	0	0	0	0	0	0	0

Figure 19-22. Port T Reduced Drive Register (RDRT)

1. Read: Anytime.
   Write: Anytime.

Table 19-23. RDRT Register Field Descriptions

Field	Description
7-0 RDRT	**Port T reduced drive**—Select reduced drive for outputs This register configures the drive strength of output pins 7 through 0 as either full or reduced. If a pin is used as input this bit has no effect. 1 Reduced drive selected (1/6 of the full drive strength). 0 Full drive strength enabled.

19.3.25 Port T Pull Device Enable Register (PERT)

Address 0x0244 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PERT7	PERT6	PERT5	PERT4	PERT3	PERT2	PERT1	PERT0
Reset	0	0	0	0	0	0	0	0

Figure 19-23. Port T Pull Device Enable Register (PERT)

1. Read: Anytime.
   Write: Anytime.

Table 19-24. PERT Register Field Descriptions

Field	Description
7-0 PERT	**Port T pull device enable**—Enable pull devices on input pins These bits configure whether a pull device is activated, if the associated pin is used as an input. This bit has no effect if the pin is used as an output. Out of reset no pull device is enabled. 1 Pull device enabled. 0 Pull device disabled.

19.3.26 Port T Polarity Select Register (PPST)

Address 0x0245 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PPST7	PPST6	PPST5	PPST4	PPST3	PPST2	PPST1	PPST0
Reset	0	0	0	0	0	0	0	0

Figure 19-24. Port T Polarity Select Register (PPST)

1. Read: Anytime.
   Write: Anytime.

Table 19-25. PPST Register Field Descriptions

Field	Description
7-0 PPST	**Port T pull device select**—Determine pull device polarity on input pins This register selects whether a pull-down or a pull-up device is connected to the pin. 1 A pull-down device is connected to the associated pin, if enabled and if the pin is used as input. 0 A pull-up device is connected to the associated pin, if enabled and if the pin is used as input.

19.3.27 PIM Reserved Register

Address 0x0246 Access: User read(1)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 19-25. PIM Reserved Register

1. Read: Always reads 0x00
   Write: Unimplemented

19.3.28 PIM Reserved Register

Address 0x0247 Access: User read(1)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 19-26. PIM Reserved Register

1. Read: Always reads 0x00
   Write: Unimplemented

19.3.29 Port S Data Register (PTS)

Address 0x0248 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PTS7	PTST6	PTS5	PTS4	PTS3	PTS2	PTS1	PTS0
Altern. Function	$\overline{SS0}$	SCK0	MOSI0	MISO0	TXD1	RXD1	TXD0	RXD0
Reset	0	0	0	0	0	0	0	0

Figure 19-27. Port S Data Register (PTS)

1. Read: Anytime.
   Write: Anytime.

Table 19-26. PTS Register Field Descriptions

Field	Description
7 PTS	**Port S general purpose input/output data**—Data Register Port S pin 7 is associated with the $\overline{SS}$ signal of the SPI0 module. When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
6 PTS	**Port S general purpose input/output data**—Data Register Port S pin 6 is associated with the SCK signal of the SPI0 module. When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
5 PTS	**Port S general purpose input/output data**—Data Register Port S pin 5 is associated with the MOSI signal of the SPI0 module. When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
4 PTS	**Port S general purpose input/output data**—Data Register Port S pin 4 is associated with the MISO signal of the SPI0 module. When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
3 PTS	**Port S general purpose input/output data**—Data Register Port S pin 3 is associated with the TXD signal of the SCI1 module. When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
2 PTS	**Port S general purpose input/output data**—Data Register Port S bits 2 is associated with the RXD signal of the SCI1 module . When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.

Table 19-26. PTS Register Field Descriptions (continued)

Field	Description
1 PTS	**Port S general purpose input/output data**—Data Register Port S pin 3 is associated with the TXD signal of the SCI0 module. When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
0 PTS	**Port S general purpose input/output data**—Data Register Port S bits 2 is associated with the RXD signal of the SCI0 module. When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.

19.3.30 Port S Input Register (PTIS)

Address 0x0249 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	PTIS7	PTIS6	PTIS5	PTIS4	PTIS3	PTIS2	PTIS1	PTIS0
W								
Reset	u	u	u	u	u	u	u	u

= Unimplemented or Reserved u = Unaffected by reset

Figure 19-28. Port S Input Register (PTIS)

1. Read: Anytime.
   Write:Never, writes to this register have no effect.

Table 19-27. PTIS Register Field Descriptions

Field	Description
7-0 PTIS	**Port S input data**— This register always reads back the buffered state of the associated pins. This can also be used to detect overload or short circuit conditions on output pins.

19.3.31 Port S Data Direction Register (DDRS)

Address 0x024A Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	DDRS7	DDRS6	DDRS5	DDRS4	DDRS3	DDRS2	DDRS1	DDRS0
Reset	0	0	0	0	0	0	0	0

Figure 19-29. Port S Data Direction Register (DDRS)

1. Read: Anytime.
   Write: Anytime.

Table 19-28. DDRS Register Field Descriptions

Field	Description
7-0 DDRS	**Port S data direction**— This register controls the data direction of pins 7 through 0.This register configures each Port S pin as either input or output. If SPI0 is enabled, the SPI0 determines the pin direction. *Refer to SPI Block Guide for details*. If the associated SCI transmit or receive channel is enabled this register has no effect on the pins. The pin is forced to be an output if a SCI transmit channel is enabled, it is forced to be an input if the SCI receive channel is enabled. The data direction bits revert to controlling the I/O direction of a pin when the associated channel is disabled. 1 Associated pin is configured as output. 0 Associated pin is configured as input.

NOTE

Due to internal synchronization circuits, it can take up to 2 bus clock cycles until the correct value is read on PTS or PTIS registers, when changing the DDRS register.

19.3.32 Port S Reduced Drive Register (RDRS)

Address 0x024B

Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	RDRS7	RDRS6	RDRS5	RDRS4	RDRS3	RDRS2	RDRS1	RDRS0
Reset	0	0	0	0	0	0	0	0

Figure 19-30. Port S Reduced Drive Register (RDRS)

1. Read: Anytime.
   Write: Anytime.

Table 19-29. RDRS Register Field Descriptions

Field	Description
7-0 RDRS	**Port S reduced drive**—Select reduced drive for outputs This register configures the drive strength of output pins 7 through 0 as either full or reduced. If a pin is used as input this bit has no effect. 1 Reduced drive selected (1/6 of the full drive strength). 0 Full drive strength enabled.

19.3.33 Port S Pull Device Enable Register (PERS)

Address 0x024C Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PERS7	PERS6	PERS5	PERS4	PERS3	PERS2	PERS1	PERS0
Reset	1	1	1	1	1	1	1	1

Figure 19-31. Port S Pull Device Enable Register (PERS)

1. Read: Anytime.
   Write: Anytime.

Read: Anytime.

Write: Anytime.

Table 19-30. PERS Register Field Descriptions

Field	Description
7-0 PERS	**Port S pull device enable**—Enable pull devices on input pins These bits configure whether a pull device is activated, if the associated pin is used as an input. This bit has no effect if the pin is used as an output. Out of reset all pull devices are enabled. 1 Pull device enabled. 0 Pull device disabled.

19.3.34 Port S Polarity Select Register (PPSS)

Address 0x024D Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PPSS7	PPSS6	PPSS5	PPSS4	PPSS3	PPSS2	PPSS1	PPSS0
Reset	0	0	0	0	0	0	0	0

Figure 19-32. Port S Polarity Select Register (PPSS)

1. Read: Anytime.
   Write: Anytime.

Table 19-31. PPSS Register Field Descriptions

Field	Description
7-0 PPSS	**Port S pull device select**—Determine pull device polarity on input pins This register selects whether a pull-down or a pull-up device is connected to the pin. 1 A pull-down device is connected to the associated pin, if enabled and if the pin is used as input. 0 A pull-up device is connected to the associated pin, if enabled and if the pin is used as input.

19.3.35 Port S Wired-Or Mode Register (WOMS)

Address 0x024E Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	WOMS7	WOMS6	WOMS5	WOMS4	WOMS3	WOMS2	WOMS1	WOMS0
Reset	0	0	0	0	0	0	0	0

Figure 19-33. Port S Wired-Or Mode Register (WOMS)

1. Read: Anytime.
   Write: Anytime.

Table 19-32. WOMS Register Field Descriptions

Field	Description
7-0 WOMS	**Port S wired-or mode**—Enable wired-or functionality This register configures the output pins as wired-or. If enabled the output is driven active low only (open-drain). A logic level of “1” is not driven.This allows a multipoint connection of several serial modules. These bits have no influence on pins used as inputs. 1 Output buffers operate as open-drain outputs. 0 Output buffers operate as push-pull outputs.

19.3.36 PIM Reserved Register

Address 0x024F Access: User read(1)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved u = Unaffected by reset

Figure 19-34. PIM Reserved Register

1. Read: Always reads 0x00
   Write: Unimplemented

19.3.37 Port M Data Register (PTM)

Address 0x0250 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PTM7	PTM6	PTM5	PTM4	PTM3	PTM2	PTM1	PTM0
Altern. Function	$\overline{SS1}$ $\overline{CS3}$	SCK1	MOSI1	MISO1 $\overline{CS2}$	FAULT3 $\overline{CS1}$	FAULT2 $\overline{CS0}$	TXCAN	RXCAN
Reset	0	0	0	0	0	0	0	0

Figure 19-35. Port M Data Register (PTM)

1. Read: Anytime.
   Write: Anytime.

Table 19-33. PTM Register Field Descriptions

Field	Description
7 PTM	**Port M general purpose input/output data**—Data Register Port M pin 7 is associated with the $\overline{SS}$ signal of the SPI1 module and the chip select output $\overline{CS3}$. The SPI function takes precedence over the chip select. When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
6 PTM	**Port M general purpose input/output data**—Data Register Port M pin 6 is associated with the SCK signal of the SPI1 module. When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
5 PTM	**Port M general purpose input/output data**—Data Register Port M pin 5 is associated with the MOSI signal of the SPI1 module. When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
4 PTM	**Port M general purpose input/output data**—Data Register Port M pin 4 is associated with the MISO signal of the SPI1 module and the chip select output $\overline{CS2}$. The SPI function takes precedence over the chip select. When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
3-2 PTM	**Port M general purpose input/output data**—Data Register Port M pins 3 and 2 are associated with the PMF fault inputs FAULT[3:2] and chip select outputs $\overline{CS}$[1:0]. The PMF function takes precedence over the chip selects. When not used with the alternative functions, these pins can be used as general purpose I/O. If the associated data direction bits of these pins are set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.

Table 19-33. PTM Register Field Descriptions (continued)

Field	Description
1 PTM	**Port M general purpose input/output data**—Data Register Port M pin 1 is associated with the TXCAN signal of the CAN module. When not used with the alternative functions, these pins can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
0 PTM	**Port M general purpose input/output data**—Data Register Port M pin 0 is associated with the RXCAN signal of the CAN module. When not used with the alternative functions, these pins can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.

19.3.38 Port M Input Register (PTIM)

Address 0x0251 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	PTIM7	PTIM6	PTIM5	PTIM4	PTIM3	PTIM2	PTIM1	PTIM0
W								
Reset	u	u	u	u	u	u	u	u

= Unimplemented or Reserved u = Unaffected by reset

Figure 19-36. Port M Input Register (PTIM)

1. Read: Anytime.
   Write:Never, writes to this register have no effect.

Table 19-34. PTIM Register Field Descriptions

Field	Description
7-0 PTIM	**Port M input data**— This register always reads back the buffered state of the associated pins. This can also be used to detect overload or short circuit conditions on output pins.

19.3.39 Port M Data Direction Register (DDRM)

Address 0x0252 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	DDRM7	DDRM6	DDRM5	DDRM4	DDRM3	DDRM2	DDRM1	DDRM0
Reset	0	0	0	0	0	0	0	0

Figure 19-37. Port M Data Direction Register (DDRM)

1. Read: Anytime.
   Write: Anytime.

Table 19-35. DDRM Register Field Descriptions

Field	Description
7-6 DDRM	**Port M data direction**— This register controls the data direction of pins 7 through 6. The PMF forces the I/O state to be an to be an input on the associated FAULT[3:2] pins if fault protection functions are enabled. *Refer to PMF Block Guide.* In this case the data direction bit will not change. 1 Associated pin is configured as output. 0 Associated pin is configured as input.
5-2 DDRM	**Port M data direction**— This register controls the data direction of pins 5 through 2. 1 Associated pin is configured as output. 0 Associated pin is configured as input.
1 DDRM	**Port M data direction**— This register controls the data direction of pin 1. The CAN forces the I/O state to be an output (TXCAN) if enabled . In this case the data direction bit will not change. 1 Pin is configured as output. 0 Associated pin is configured as input.
0 DDRM	**Port M data direction**— This register controls the data direction of pin 0. The CAN forces the I/O state to be an input (RXCAN) if enabled . In this case the data direction bit will not change. 1 Pin is configured as output. 0 Pin is configured as input.

NOTE

Due to internal synchronization circuits, it can take up to 2 bus clock cycles until the correct value is read on PTM or PTIM registers, when changing the DDRM register.

19.3.40 Port M Reduced Drive Register (RDRM)

Address 0x0253 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	RDRM7	RDRM6	RDRM5	RDRM4	RDRM3	RDRM2	RDRM1	RDRM0
Reset	0	0	0	0	0	0	0	0

Figure 19-38. Port M Reduced Drive Register (RDRM)

1. Read: Anytime.
   Write: Anytime.

Table 19-36. RDRM Register Field Descriptions

Field	Description
7-0 RDRM	**Port M reduced drive**—Select reduced drive for outputs This register configures the drive strength of Port M output pins 7 through 0 as either full or reduced. If a pin is used as input this bit has no effect. 1 Reduced drive selected (1/6 of the full drive strength). 0 Full drive strength enabled.

19.3.41 Port M Pull Device Enable Register (PERM)

Address 0x0254 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PERM7	PERM6	PERM5	PERM4	PERM3	PERM2	PERM1	PERM0
Reset	0	0	0	0	0	0	0	0

Figure 19-39. Port M Pull Device Enable Register (PERM)

1. Read: Anytime.
   Write: Anytime.

Table 19-37. PERM Register Field Descriptions

Field	Description
7-0 PERM	**Port M pull device enable**—Enable pull-up devices on input pins These bits configure whether a pull device is activated, if the associated pin is used as an input. This bit has no effect if the pin is used as an output. Out of reset no pull device is enabled. 1 Pull device enabled. 0 Pull device disabled.

19.3.42 Port M Polarity Select Register (PPSM)

Address 0x0255 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PPSM7	PPSM6	PPSM5	PPSM4	PPSM3	PPSM2	PPSM1	PPSM0
Reset	0	0	0	0	0	0	0	0

Figure 19-40. Port M Polarity Select Register (PPSM)

1. Read: Anytime.
   Write: Anytime.

Table 19-38. PPSM Register Field Descriptions

Field	Description
7-0 PPSM	**Port M pull device select**—Determine pull device polarity on input pins This register selects whether a pull-down or a pull-up device is connected to the pin. 1 A pull-down device is connected to the associated pin, if enabled and if the pin is used as input. 0 A pull-up device is connected to the associated pin, if enabled and if the pin is used as input.

19.3.43 Port M Wired-Or Mode Register (WOMM)

Address 0x0256 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	WOMM7	WOMM6	WOMM5	WOMM4	WOMM3	WOMM2	WOMM1	WOMM0
Reset	0	0	0	0	0	0	0	0

Figure 19-41. Port M Wired-Or Mode Register (WOMM)

1. Read: Anytime.
   Write: Anytime.

Table 19-39. WOMM Register Field Descriptions

Field	Description
7-0 WOMM	**Port M wired-or mode**—Enable wired-or functionality This register configures the output pins as wired-or. If enabled the output is driven active low only (open-drain). A logic level of “1” is not driven.This allows a multipoint connection of several serial modules. These bits have no influence on pins used as inputs. 1 Output buffers operate as open-drain outputs. 0 Output buffers operate as push-pull outputs.

19.3.44 PIM Reserved Register

Address 0x0257 Access: User read(1)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 19-42. PIM Reserved Register

1. Read: Always reads 0x00
   Write: Unimplemented

19.3.45 Port P Data Register (PTP)

Address 0x0258 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PTP7	PTP6	PTP5	PTP4	PTP3	PTP2	PTP1	PTP0
Altern. Function	FAULT1	FAULT0	PW05	PW04	PW03	PW02	PW01	PW00
Reset	0	0	0	0	0	0	0	0

Figure 19-43. Port P Data Register (PTP)

1. Read: Anytime.
   Write: Anytime.

Table 19-40. PTP Register Field Descriptions

Field	Description
7-6 PTP	**Port P general purpose input/output data**—Data Register Port P pins 7 and 6 are associated with the PMF fault inputs FAULT[1:0]. When not used with the alternative functions, these pins can be used as general purpose I/O. If the associated data direction bits of these pins are set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
5-0 PTP	**Port P general purpose input/output data**—Data Register Port P pins 5 through 0 are associated with the PMF output channels PW0[5:0]. When not used with the alternative functions, these pins can be used as general purpose I/O. If the associated data direction bits of these pins are set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.

19.3.46 Port P Input Register (PTIP)

Address 0x0259 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	PTIP7	PTIP6	PTIP5	PTIP4	PTIP3	PTIP2	PTIP1	PTIP0
W								
Reset	u	u	u	u	u	u	u	u

= Unimplemented or Reserved u = Unaffected by reset

Figure 19-44. Port P Input Register (PTIP)

1. Read: Anytime.
   Write:Never, writes to this register have no effect.

Table 19-41. PTIP Register Field Descriptions

Field	Description
7-0 PTIP	**Port P input data**—This register always reads back the buffered state of the associated pins. This can also be used to detect overload or short circuit conditions on output pins.

19.3.47 Port P Data Direction Register (DDRP)

Address 0x025A — Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	DDRP7	DDRP6	DDRP5	DDRP4	DDRP3	DDRP2	DDRP1	DDRP0
Reset	0	0	0	0	0	0	0	0

Figure 19-45. Port P Data Direction Register (DDRP)

1. Read: Anytime.
   Write: Anytime.

Table 19-42. DDRP Register Field Descriptions

Field	Description
7-6 DDRP	**Port P data direction**— This register controls the data direction of pins 7 through 6. The PMF forces the I/O state to be an to be an input on the associated FAULT[1:0] pins if fault protection functions are enabled. *Refer to PMF Block Guide.* In this case the data direction bit will not change. 1 Associated pin is configured as output. 0 Associated pin is configured as input.
5-0 DDRP	**Port P data direction**— This register controls the data direction of pins 5 through 0. The PMF forces the I/O state to be an output for each pin associated with an enabled PW[05:00] channel. In this case the data direction bit will not change. *Refer to PMF Block Guide for details.* 1 Associated pin is configured as output. 0 Associated pin is configured as input.

NOTE

Due to internal synchronization circuits, it can take up to 2 bus clock cycles until the correct value is read on PTP or PTIP registers, when changing the DDRP register.

19.3.48 Port P Reduced Drive Register (RDRP)

Address 0x025B — Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	RDRP7	RDRP6	RDRP5	RDRP4	RDRP3	RDRP2	RDRP1	RDRP0
Reset	0	0	0	0	0	0	0	0

Figure 19-46. Port P Reduced Drive Register (RDRP)

1. Read: Anytime.
   Write: Anytime.

Table 19-43. RDRP Register Field Descriptions

Field	Description
7-0 RDRP	**Port P reduced drive**—Select reduced drive for outputs This register configures the drive strength of output pins 7 through 0 as either full or reduced. If a pin is used as input this bit has no effect. 1 Reduced drive selected (1/6 of the full drive strength). 0 Full drive strength enabled.

19.3.49 Port P Pull Device Enable Register (PERP)

Address 0x025C — Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PERP7	PERP6	PERP5	PERP4	PERP3	PERP2	PERP1	PERP0
Reset	0	0	0	0	0	0	0	0

Figure 19-47. Port P Pull Device Enable Register (PERP)

1. Read: Anytime.
   Write: Anytime.

Table 19-44. PERP Register Field Descriptions

Field	Description
7-0 PERP	**Port P pull device enable**—Enable pull devices on input pins These bits configure whether a pull device is activated, if the associated pin is used as an input. This bit has no effect if the pin is used as an output. Out of reset no pull device is enabled. 1 Pull device enabled. 0 Pull device disabled.

19.3.50 Port P Polarity Select Register (PPSP)

Address 0x025D — Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PPSP7	PPSP6	PPSP5	PPSP4	PPSP3	PPSP2	PPSP1	PPSP0
Reset	0	0	0	0	0	0	0	0

Figure 19-48. Port P Polarity Select Register (PPSP)

1. Read: Anytime.
   Write: Anytime.

Table 19-45. PPSP Register Field Descriptions

Field	Description
7-0 PPSP	**Port P pull device select**—Determine pull device polarity on input pins This register selects whether a pull-down or a pull-up device is connected to the pin. 1 A pull-down device is connected to the associated pin, if enabled and if the pin is used as input. 0 A pull-up device is connected to the associated pin, if enabled and if the pin is used as input.

19.3.51 PIM Reserved Register

Address 0x025E Access: User read(1)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 19-49. PIM Reserved Register

1. Read: Always reads 0x00
   Write: Unimplemented

19.3.52 PIM Reserved Register

Address 0x025F Access: User read(1)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 19-50. PIM Reserved Register

1. Read: Always reads 0x00
   Write: Unimplemented

19.3.53 Port H Data Register (PTH)

Address 0x0260 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PTH7	PTH6	PTH5	PTH4	PTH3	PTH2	PTH1	PTH0
Altern. Function	—	$\overline{\text{TXE\_B}}$	TXD_B	RXD_B	—	$\overline{\text{TXE\_A}}$	TXD_A	RXD_A
Reset	0	0	0	0	0	0	0	0

Figure 19-51. Port H Data Register (PTH)

1. Read: Anytime.
   Write: Anytime.

Table 19-46. PTH Register Field Descriptions

Field	Description
7 PTH	**Port H general purpose input/output data**—Data Register This pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
6 PTH	**Port H general purpose input/output data**—Data Register This pin is associated with the FlexRay channel B transmitter enable signal ($\overline{\text{TXE\_B}}$). When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
5 PTH	**Port H general purpose input/output data**—Data Register This pin is associated with the FlexRay channel B transmit pin (TXD_B). When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
4 PTH	**Port H general purpose input/output data**—Data Register This pin is associated with the FlexRay channel B receive pin (RXD_B). When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
3 PTH	**Port H general purpose input/output data**—Data Register This pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
2 PTH	**Port H general purpose input/output data**—Data Register This pin is associated with the FlexRay channel A transmitter enable signal ($\overline{\text{TXE\_A}}$). When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.

Table 19-46. PTH Register Field Descriptions (continued)

Field	Description
1 PTH	**Port H general purpose input/output data**—Data Register This pin is associated with the FlexRay channel A transmit pin (TXD_A). When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
0 PTH	**Port H general purpose input/output data**—Data Register This pin is associated with the FlexRay channel A receive pin (RXD_A). When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.

19.3.54 Port H Input Register (PTIH)

Address 0x0261 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	PTIH7	PTIH6	PTIH5	PTIH4	PTIH3	PTIH2	PTIH1	PTIH0
W								
Reset	u	u	u	u	u	u	u	u

= Unimplemented or Reserved u = Unaffected by reset

Figure 19-52. Port H Input Register (PTIH)

1. Read: Anytime.
   Write:Never, writes to this register have no effect.

Table 19-47. PTIH Register Field Descriptions

Field	Description
7-0 PTIH	**Port H input data**— This register always reads back the buffered state of the associated pins. This can also be used to detect overload or short circuit conditions on output pins.

19.3.55 Port H Data Direction Register (DDRH)

Address 0x0262 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	DDRH7	DDRH6	DDRH5	DDRH4	DDRH3	DDRH2	DDRH1	DDRH0
Reset	0	0	0	0	0	0	0	0

Figure 19-53. Port H Data Direction Register (DDRH)

1. Read: Anytime.
   Write: Anytime.

Table 19-48. DDRH Register Field Descriptions

Field	Description
7 DDRH	**Port H data direction**— This register controls the data direction of pin 7. 1 Pin is configured as output. 0 Pin is configured as input.
6 DDRH	**Port H data direction**— This register controls the data direction of pin 6. The FlexRay forces the I/O state to be an output ($\overline{\text{TXE\_B}}$) if channel B is enabled. *Refer to FlexRay Block Guide.* In this case the data direction bit will not change. 1 Pin is configured as output. 0 Pin is configured as input.
5 DDRH	**Port H data direction**— This register controls the data direction of pin 5. The FlexRay forces the I/O state to be an output (TXD_B) if channel B is enabled. *Refer to FlexRay Block Guide.* In this case the data direction bit will not change. 1 Pin is configured as output. 0 Pin is configured as input.
4 DDRH	**Port H data direction**— This register controls the data direction of pin 4. The FlexRay forces the I/O state to be an input (RXD_B) if channel B is enabled. *Refer to FlexRay Block Guide.* In this case the data direction bit will not change. 1 Pin is configured as output. 0 Pin is configured as input.
3 DDRH	**Port H data direction**— This register controls the data direction of pins 3 1 Pin is configured as output. 0 Pin is configured as input.
2 DDRH	**Port H data direction**— This register controls the data direction of pin 2. The FlexRay forces the I/O state to be an output ($\overline{\text{TXE\_A}}$) if channel A is enabled. *Refer to FlexRay Block Guide.* In this case the data direction bit will not change. 1 Pin is configured as output. 0 Pin is configured as input.
1 DDRH	**Port H data direction**— This register controls the data direction of pin 1. The FlexRay forces the I/O state to be an output (TXD_A) if channel A is enabled. *Refer to FlexRay Block Guide.* In this case the data direction bit will not change. 1 Pin is configured as output. 0 Pin is configured as input.
0 DDRH	**Port H data direction**— This register controls the data direction of pin 0. The FlexRay forces the I/O state to be an input (RXD_A) if channel A is enabled. *Refer to FlexRay Block Guide.* In this case the data direction bit will not change. 1 Pin is configured as output. 0 Pin is configured as input.

NOTE

Due to internal synchronization circuits, it can take up to 2 bus clock cycles until the correct value is read on PTH or PTIH registers, when changing the DDRH register.

19.3.56 Port H Reduced Drive Register (RDRH)

Address 0x0263 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	RDRH7	RDRH6	RDRH5	RDRH4	RDRH3	RDRH2	RDRH1	RDRH0
Reset	0	0	0	0	0	0	0	0

Figure 19-54. Port H Reduced Drive Register (RDRH)

1. Read: Anytime.
   Write: Anytime.

Table 19-49. RDRH Register Field Descriptions

Field	Description
7-0 RDRH	**Port H reduced drive**—Select reduced drive for outputs This register configures the drive strength of output pins 7 through 0 as either full or reduced. If a pin is used as input this bit has no effect. 1 Reduced drive selected (1/6 of the full drive strength). 0 Full drive strength enabled.

19.3.57 Port H Pull Device Enable Register (PERH)

Address 0x0264 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PERH7	PERH6	PERH5	PERH4	PERH3	PERH2	PERH1	PERH0
Reset	0	0	0	0	0	0	0	0

Figure 19-55. Port H Pull Device Enable Register (PERH)

1. Read: Anytime.
   Write: Anytime.

Table 19-50. PERH Register Field Descriptions

Field	Description
7-0 PERH	**Port H pull device enable**—Enable pull devices on input pins These bits configure whether a pull device is activated, if the associated pin is used as an input. This bit has no effect if the pin is used as an output. Out of reset no pull device is enabled. 1 Pull device enabled. 0 Pull device disabled.

19.3.58 Port H Polarity Select Register (PPSH)

Address 0x0265 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PPSH7	PPSH6	PPSH5	PPSH4	PPSH3	PPSH2	PPSH1	PPSH0
Reset	0	0	0	0	0	0	0	0

Figure 19-56. Port H Polarity Select Register (PPSH)

1. Read: Anytime.
   Write: Anytime.

Table 19-51. PPSH Register Field Descriptions

Field	Description
7-0 PPSH	**Port H pull device select**—Determine pull device polarity on input pins This register selects whether a pull-down or a pull-up device is connected to the pin. 1 A pull-down device is connected to the associated pin, if enabled and if the pin is used as input. 0 A pull-up device is connected to the associated pin, if enabled and if the pin is used as input.

19.3.59 PIM Reserved Register

Address 0x0266 Access: User read(1)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 19-57. PIM Reserved Register

1. Read: Always reads 0x00
   Write: Unimplemented

19.3.60 PIM Reserved Register

Address 0x0267 Access: User read(1)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 19-58. PIM Reserved Register

1. Read: Always reads 0x00
   Write: Unimplemented

19.3.61 Port J Data Register (PTJ)

Address 0x0268 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PTJ7	PTJ6	PTJ5	PTJ4	PTJ3	PTJ2	PTJ1	PTJ0
Altern. Function	—	STB3	STB2	STB1	STB0	$\overline{IS2}$	$\overline{IS1}$	$\overline{IS0}$
Reset	0	0	0	0	0	0	0	0

Figure 19-59. Port J Data Register (PTJ)

1. Read: Anytime.
   Write: Anytime.

Table 19-52. PTJ Register Field Descriptions

Field	Description
7 PTJ	**Port J general purpose input/output data**—Data Register This pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
6 PTJ	**Port J general purpose input/output data**—Data Register This pin is associated with the FlexRay debug strobe signal 3. When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
5 PTJ	**Port J general purpose input/output data**—Data Register This pin is associated with the FlexRay debug strobe signal 2. When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
4 PTJ	**Port J general purpose input/output data**—Data Register This pin is associated with the FlexRay debug strobe signal 1. When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
3 PTJ	**Port J general purpose input/output data**—Data Register This pin is associated with the FlexRay debug strobe signal 0. When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
2 PTJ	**Port J general purpose input/output data**—Data Register This pin is associated with the PMF current status signal $\overline{IS2}$. When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.

Table 19-52. PTJ Register Field Descriptions (continued)

Field	Description
1 PTJ	**Port J general purpose input/output data**—Data Register This pin is associated with the PMF current status signal $\overline{IS1}$. When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.
0 PTJ	**Port J general purpose input/output data**—Data Register This pin is associated with the PMF current status signal $\overline{IS0}$. When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bit of this pin is set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.

19.3.62 Port J Input Register (PTIJ)

Address 0x0269 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	PTIJ7	PTIJ6	PTIJ5	PTIJ4	PTIJ3	PTIJ2	PTIJ1	PTIJ0
W								
Reset	u	u	u	u	u	u	u	u

= Unimplemented or Reserved u = Unaffected by reset

Figure 19-60. Port J Input Register (PTIJ)

1. Read: Anytime.
   Write:Never, writes to this register have no effect.

Table 19-53. PTIJ Register Field Descriptions

Field	Description
7-0 PTIJ	**Port J input data**— This register always reads back the buffered state of the associated pins. This can also be used to detect overload or short circuit conditions on output pins.

19.3.63 Port J Data Direction Register (DDRJ)

Address 0x026A Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	DDRJ7	DDRJ6	DDRJ5	DDRJ4	DDRJ3	DDRJ2	DDRJ1	DDRJ0
Reset	0	0	0	0	0	0	0	0

Figure 19-61. Port J Data Direction Register (DDRJ)

1. Read: Anytime.
   Write: Anytime.

Table 19-54. DDRJ Register Field Descriptions

Field	Description
7 DDRJ	**Port J data direction**— This register controls the data direction of pin 7. 1 Associated pin is configured as output. 0 Associated pin is configured as input.
6 DDRJ	**Port J data direction**— This register controls the data direction of pin 6. The FlexRay forces the I/O state to be an output (STB3) if the strobe signal is enabled. *Refer to FlexRay Block Guide.* In this case the data direction bit will not change. 1 Associated pin is configured as output. 0 Associated pin is configured as input.
5 DDRJ	**Port J data direction**— This register controls the data direction of pin 5. The FlexRay forces the I/O state to be an output (STB2) if the strobe signal is enabled. *Refer to FlexRay Block Guide.* In this case the data direction bit will not change. 1 Associated pin is configured as output. 0 Associated pin is configured as input.
4 DDRJ	**Port J data direction**— This register controls the data direction of pin 4. The FlexRay forces the I/O state to be an output (STB1) if the strobe signal is enabled. *Refer to FlexRay Block Guide.* In this case the data direction bit will not change. 1 Associated pin is configured as output. 0 Associated pin is configured as input.
3 DDRJ	**Port J data direction**— This register controls the data direction of pin 3. The FlexRay forces the I/O state to be an output (STB0) if the strobe signal is enabled. *Refer to FlexRay Block Guide.* In this case the data direction bit will not change. 1 Associated pin is configured as output. 0 Associated pin is configured as input.
2-0 DDRJ	**Port J data direction**— This register controls the data direction of pins 2 through 0. The PMF forces the I/O state to be an input on the associated $\overline{IS}$[2:0] pins if current sense functions are enabled. *Refer to PMF Block Guide.* In this case the data direction bit will not change. 1 Associated pin is configured as output. 0 Associated pin is configured as input.

NOTE

Due to internal synchronization circuits, it can take up to 2 bus clock cycles until the correct value is read on PTH or PTIH registers, when changing the DDRH register.

19.3.64 Port J Reduced Drive Register (RDRJ)

Address 0x026B Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	RDRJ7	RDRJ6	RDRJ5	RDRJ4	RDRJ3	RDRJ2	RDRJ1	RDRJ0
Reset	0	0	0	0	0	0	0	0

Figure 19-62. Port J Reduced Drive Register (RDRJ)

1. Read: Anytime.
   Write: Anytime.

Table 19-55. RDRJ Register Field Descriptions

Field	Description
7-0 RDRJ	**Port J reduced drive**—Select reduced drive for outputs This register configures the drive strength of output pins 7 through 0 as either full or reduced. If a pin is used as input this bit has no effect. 1 Reduced drive selected (1/6 of the full drive strength). 0 Full drive strength enabled.

19.3.65 Port J Pull Device Enable Register (PERJ)

Address 0x026C Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PERJ7	PERJ6	PERJ5	PERJ4	PERJ3	PERJ2	PERJ1	PERJ0
Reset	1	1	1	1	1	1	1	1

Figure 19-63. Port J Pull Device Enable Register (PERJ)

1. Read: Anytime.
   Write: Anytime.

Table 19-56. PERJ Register Field Descriptions

Field	Description
7-0 PERJ	**Port J pull device enable**—Enable pull devices on input pins These bits configure whether a pull device is activated, if the associated pin is used as an input. This bit has no effect if the pin is used as an output. Out of reset all pull device are enabled. 1 Pull device enabled. 0 Pull device disabled.

19.3.66 Port J Polarity Select Register (PPSJ)

Address 0x026D Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PPSJ7	PPSJ6	PPSJ5	PPSJ4	PPSJ3	PPSJ2	PPSJ1	PPSJ0
Reset	0	0	0	0	0	0	0	0

Figure 19-64. Port J Polarity Select Register (PPSJ)

1. Read: Anytime.
   Write: Anytime.

Table 19-57. PPSJ Register Field Descriptions

Field	Description
7-0 PPSJ	**Port J pull device select**—Determine pull device polarity on input pins This register selects whether a pull-down or a pull-up device is connected to the pin. 1 A pull-down device is connected to the associated pin, if enabled and if the pin is used as input. 0 A pull-up device is connected to the associated pin, if enabled and if the pin is used as input.

19.3.67 PIM Reserved Register

Address 0x026E Access: User read(1)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 19-65. PIM Reserved Register

1. Read: Always reads 0x00
   Write: Unimplemented

19.3.68 PIM Reserved Register

Address 0x026F Access: User read(1)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 19-66. PIM Reserved Register

1. Read: Always reads 0x00
   Write: Unimplemented

19.3.69 Port AD Data Register 0 (PT0AD)

Address 0x0270 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PT0AD15	PT0AD14	PT0AD13	PT0AD12	PT0AD11	PT0AD10	PT0AD9	PT0AD8
Altern. Function	AN15	AN14	AN13	AN12	AN11	AN10	AN9	AN8
Reset	0	0	0	0	0	0	0	0

Figure 19-67. Port AD Data Register 0 (PT0AD)

1. Read: Anytime.
   Write: Anytime.

Table 19-58. PT0AD Register Field Descriptions

Field	Description
7-0 PT0AD	**Port AD general purpose input/output data**—Data Register This register is associated with ATD analog inputs PAD[15:08]. When not used with the alternative function, this pin can be used as general purpose I/O. If the associated data direction bits of these pins are set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.

19.3.70 Port AD Data Register 1 (PT1AD)

Address 0x0271 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PT1AD7	PT1AD6	PT1AD5	PT1AD4	PT1AD3	PT1AD2	PT1AD1	PT1AD0
Altern. Function	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0
Reset	0	0	0	0	0	0	0	0

Figure 19-68. Port AD Data Register 1 (PT1AD)

1. Read: Anytime.
   Write: Anytime.

Table 19-59. PT1AD Register Field Descriptions

Field	Description
7-0 PT1AD	**Port AD general purpose input/output data**—Data Register This register is associated with ATD analog inputs PAD[07:00]. When not used with the alternative function, these pins can be used as general purpose I/O. If the associated data direction bits of these pins are set to 1, a read returns the value of the port register, otherwise the buffered pin input state is read.

19.3.71 Port AD Data Direction Register 0 (DDR0AD)

Address 0x0272 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	DDR0AD15	DDR0AD14	DDR0AD13	DDR0AD12	DDR0AD11	DDR0AD10	DDR0AD9	DDR0AD8
Reset	0	0	0	0	0	0	0	0

Figure 19-69. Port AD Data Direction Register 0 (DDR0AD)

1. Read: Anytime.
   Write: Anytime.

Table 19-60. DDR0AD Register Field Descriptions

Field	Description
7-0 DDR0AD	**Port AD data direction**— This register controls the data direction of pins 15 through 8. 1 Associated pin is configured as output. 0 Associated pin is configured as input.

NOTE

Due to internal synchronization circuits, it can take up to 2 bus clock cycles until the correct value is read on PT0AD registers, when changing the DDR0AD register.

NOTE

To use the digital input function on Port AD the ATD Digital Input Enable Register (ATDDIEN1) has to be set to logic level “1”.

19.3.72 Port AD Data Direction Register 1 (DDR1AD)

Address 0x0273 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	DDR1AD7	DDR1AD6	DDR1AD5	DDR1AD4	DDR1AD3	DDR1AD2	DDR1AD1	DDR1AD0
Reset	0	0	0	0	0	0	0	0

Figure 19-70. Port AD Data Direction Register 1 (DDR1AD)

1. Read: Anytime.
   Write: Anytime.

Table 19-61. DDR1AD Register Field Descriptions

Field	Description
7-0 DDR1AD	**Port AD data direction**— This register controls the data direction of pins 7 through 0. 1 Associated pin is configured as output. 0 Associated pin is configured as input.

NOTE

Due to internal synchronization circuits, it can take up to 2 bus clock cycles until the correct value is read on PT0AD registers, when changing the DDR1AD register.

NOTE

To use the digital input function on Port AD the ATD Digital Input Enable Register (ATDDIEN1) has to be set to logic level “1”.

19.3.73 Port AD Reduced Drive Register 0 (RDR0AD)

Address 0x0274 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	RDR0AD15	RDR0AD14	RDR0AD13	RDR0AD12	RDR0AD11	RDR0AD10	RDR0AD9	RDR0AD8
Reset	0	0	0	0	0	0	0	0

Figure 19-71. Port AD Reduced Drive Register 0 (RDR0AD)

1. Read: Anytime.
   Write: Anytime.

Table 19-62. RDR0AD Register Field Descriptions

Field	Description
7-0 RDR0AD	**Port AD reduced drive**—Select reduced drive for Port AD outputs This register configures the drive strength of Port AD output pins 15 through 8 as either full or reduce. If a pin is used as input this bit has no effect. 1 Reduced drive selected (1/6 of the full drive strength). 0 Full drive strength enabled.

19.3.74 Port AD Reduced Drive Register 1 (RDR1AD)

Address 0x0275 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	RDR1AD15	RDR1AD14	RDR1AD13	RDR1AD12	RDR1AD11	RDR1AD10	RDR1AD9	RDR1AD8
Reset	0	0	0	0	0	0	0	0

Figure 19-72. Port AD Reduced Drive Register 1 (RDR1AD)

1. Read: Anytime.
   Write: Anytime.

Table 19-63. RDR1AD Register Field Descriptions

Field	Description
7-0 RDR1AD	**Port AD reduced drive**—Select reduced drive for Port AD outputs This register configures the drive strength of Port AD output pins 7 through 0 as either full or reduced. If a pin is used as input this bit has no effect. 1 Reduced drive selected (1/6 of the full drive strength). 0 Full drive strength enabled.

19.3.75 Port AD Pull Up Enable Register 0 (PER0AD)

Address 0x0276 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PER0AD15	PER0AD14	PER0AD13	PER0AD12	PER0AD11	PER0AD10	PER0AD9	PER0AD8
Reset	0	0	0	0	0	0	0	0

Figure 19-73. Port AD Pull Device Up Register 0 (PER0AD)

1. Read: Anytime.
   Write: Anytime.

Table 19-64. PER0AD Register Field Descriptions

Field	Description
7-0 PER0AD	**Port AD pull device enable**—Enable pull devices on input pins These bits configure whether a pull device is activated, if the associated pin is used as an input. This bit has no effect if the pin is used as an output. Out of reset no pull device is enabled. 1 Pull device enabled. 0 Pull device disabled.

19.3.76 Port AD Pull Up Enable Register 1 (PER1AD)

Address 0x0277 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PER1AD7	PER1AD6	PER1AD5	PER1AD4	PER1AD3	PER1AD2	PER1AD1	PER1AD0
Reset	0	0	0	0	0	0	0	0

Figure 19-74. Port AD Pull Up Enable Register 1 (PER1AD)

1. Read: Anytime.
   Write: Anytime.

Table 19-65. PER1AD Register Field Descriptions

Field	Description
7-0 PER1AD	**Port AD pull device enable**—Enable pull devices on input pins These bits configure whether a pull device is activated, if the associated pin is used as an input. This bit has no effect if the pin is used as an output. Out of reset no pull device is enabled. 1 Pull device enabled. 0 Pull device disabled.

19.3.77 PIM Reserved Register

Address 0x0278 - 0x027F Access: User read(1)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 19-75. PIM Reserved Register

1. Read: Always reads 0x00
   Write: Unimplemented

19.4 Functional Description

19.4.1 General

Each pin except PE0, PE1, and BKGD can act as general purpose I/O. In addition each pin can act as an output from the external bus interface module or a peripheral module or an input to the external bus interface module or a peripheral module.

19.4.2 Registers

A set of configuration registers is common to all ports with exceptions in the expanded bus interface and ATD ports (Table 19-66). All registers can be written at any time, however a specific configuration might not become active.

Example 19-1. Selecting a pull-up device

```
This device does not become active while the port is used as a push-pull output.
```

Table 19-66. Register availability per port(1)

Port	Data	Input	Data Direction	Reduced Drive	Pull Enable	Polarity Select	Wired-Or Mode
A	✔	-	✔	✔	✔	-	-
B	✔	-	✔			-	-
C	✔	-	✔			-	-
D	✔	-	✔			-	-
E	✔	-	✔			-	-
K	✔	-	✔			-	-
T	✔	✔	✔	✔	✔	✔	-
S	✔	✔	✔	✔	✔	✔	✔
M	✔	✔	✔	✔	✔	✔	✔
P	✔	✔	✔	✔	✔	✔	-
H	✔	✔	✔	✔	✔	✔	-
J	✔	✔	✔	✔	✔	✔	-
AD	✔	-	✔	✔	✔	-	-

1. Each cell represents one register with individual configuration bits

19.4.2.1 Data register (PORTx, PTx)

This register holds the value driven out to the pin if the pin is used as a general purpose I/O.

Writing to this register has only an effect on the pin if the pin is used as general purpose output. When reading this address, the buffered state of the pin is returned if the associated data direction register bit is set to “0”.

If the data direction register bits are set to logic level “1”, the contents of the data register is returned. This is independent of any other configuration (Figure 19-76).

19.4.2.2 Input register (PTIx)

This is a read-only register and always returns the buffered state of the pin (Figure 19-76).

19.4.2.3 Data direction register (DDRx)

This register defines whether the pin is used as an input or an output.

If a peripheral module controls the pin the contents of the data direction register is ignored (Figure 19-76).

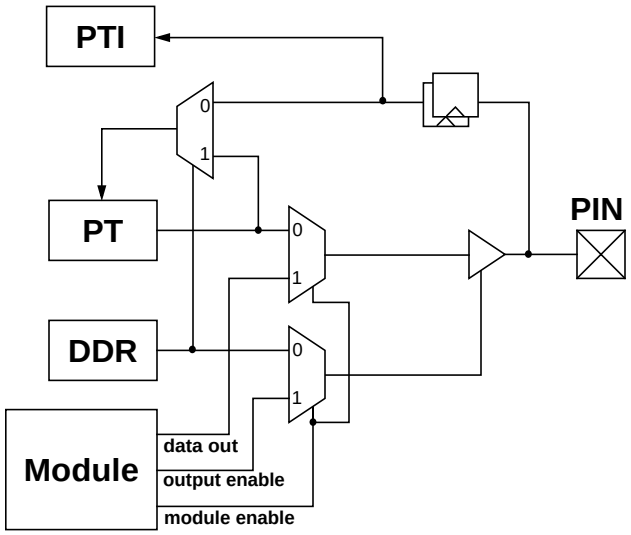


Figure 19-76. Illustration of I/O pin functionality

19.4.2.4 Reduced drive register (RDRx)

If the pin is used as an output this register allows the configuration of the drive strength.

19.4.2.5 Pull device enable register (PERx)

This register turns on a pull-up or pull-down device.

It becomes active only if the pin is used as an input or as a wired-or output.

19.4.2.6 Polarity select register (PPSx)

This register selects either a pull-up or pull-down device if enabled.

It becomes only active if the pin is used as an input. A pull-up device can be activated if the pin is used as a wired-or output.

19.4.2.7 Wired-or mode register (WOMx)

If the pin is used as an output this register turns off the active high drive. This allows wired-or type connections of outputs.

19.4.3 Pins and Ports

NOTE

Please refer to the SoC Guide to determine the pin availability in the different package options.

19.4.3.1 BKGD pin

The BKGD pin is associated with the S12X_BDM and S12X_EBI modules.

During reset, the BKGD pin is used as MODC input.

19.4.3.2 Port A, B

Port A pins PA[7:0] and Port B pins PB[7:0] can be used for either general-purpose I/O with the external bus interface. In this case Port A and Port B are associated with the external address bus outputs ADDR15-ADDR8 and ADDR7-ADDR0, respectively. PB0 is the ADDR0 or $\overline{\text{UDS}}$ output.

19.4.3.3 Port C, D

Port C pins PC[7:0] and Port D pins PD[7:0] can be used for either general-purpose I/O with the external bus interface. In this case Port C and Port D are associated with the external data bus inputs/outputs DATA15-DATA8 and DATA7-DATA0, respectively.

These pins are configured for reduced input threshold in certain operating modes (refer to S12X_EBI Block Guide).

19.4.3.4 Port E

Port E is associated with the external bus control outputs $\overline{\text{RW}}$, $\overline{\text{LSTRB}}$, $\overline{\text{LDS}}$ and $\overline{\text{RE}}$, the free-running clock outputs ECLK and ECLK2X, as well as with the $\overline{\text{TAGHI}}$, $\overline{\text{TAGLO}}$, MODA and MODB and interrupt inputs $\overline{\text{IRQ}}$ and $\overline{\text{XIRQ}}$.

Port E pins PE[7:2] can be used for either general-purpose I/O or with the alternative functions.

Port E pin PE[7] an be used for either general-purpose I/O or as the free-running clock ECLKX2 output running at the Core Clock rate. The clock output is always enabled in emulation modes.

Port E pin PE[6] an be used for either general-purpose I/O, as $\overline{\text{TAGHI}}$ input or as MODB input during reset.

Port E pin PE[5] an be used for either general-purpose I/O, as $\overline{\text{TAGLO}}$ input, $\overline{\text{RE}}$ output or as MODB input during reset.

Port E pin PE[4] an be used for either general-purpose I/O or as the free-running clock ECLK output running at the Bus Clock rate or at the programmed divided clock rate. The clock output is always enabled in emulation modes.

Port E pin PE[3] an be used for either general-purpose I/O, as $\overline{\text{LSTRB}}$ or $\overline{\text{LDS}}$ output, or as EROMCTL input during reset.

Port E pin PE[2] an be used for either general-purpose I/O, or as $\overline{\text{RW}}$ or $\overline{\text{RE}}$ output.

Port E pin PE[1] can be used for either general-purpose input or as the level- or falling edge-sensitive $\overline{\text{IRQ}}$ interrupt input. $\overline{\text{IRQ}}$ will be enabled by setting the IRQEN configuration bit (19.3.17/19-859) and clearing

the I-bit in the CPU condition code register. It is inhibited at reset so this pin is initially configured as a simple input with a pull-up.

Port E pin PE[0] can be used for either general-purpose input or as the level-sensitive $\overline{\text{XIRQ}}$ interrupt input. $\overline{\text{XIRQ}}$ can be enabled by clearing the X-bit in the CUP's condition code register. It is inhibited at reset so this pin is initially configured as a high-impedance input with a pull-up.

Port E pins PE[5] and PE[6] are configured for reduced input threshold in certain modes (refer to S12X_EBI Block Guide).

19.4.3.5 Port K

Port K pins PK[7:0] can be used for either general-purpose I/O, or with the external bus interface. In this case Port K pins PK[6:0] are associated with the external address bus outputs ADDR22-ADDR16 and PK7 is associated to the $\overline{\text{EWAIT}}$ input.

Port K pin PE[7] is configured for reduced input threshold in certain modes (refer to S12X_EBI Block Guide).

19.4.3.6 Port T

This port is associated with the ECT module, or with the FlexRay subsystem.

Port T pins PT[7:0] can be used for either general-purpose I/O, or with the channels of the Enhanced Capture Timer.

19.4.3.7 Port S

This port is associated with SCI0, SCI1 and SPI0.

Port S pins PS[7:4] can be used either for general-purpose I/O, or with the SPI0 subsystem.

Port S pins PS[3:2] can be used either for general-purpose I/O, or with the SCI1 subsystem.

Port S pins PS[1:0] can be used either for general-purpose I/O, or with the SCI0 subsystem.

19.4.3.8 Port M

This port is associated with the CAN0, SPI1, PMF and chip selects $\overline{\text{CS0}}$, $\overline{\text{CS1}}$, $\overline{\text{CS2}}$ and $\overline{\text{CS3}}$.

Port M pins PM[7:4] can be used for either general purpose I/O, or with the SPI1 subsystem.

Port M pins PM[3:2] can be used for either general purpose I/O, or as fault inputs of the PMF subsystem.

Port M pins PM[1:0] can be used for either general purpose I/O, or with CAN0 subsystem.

Port M pins PM[7] and PM[4:2] can also be used as chip select outputs.

19.4.3.9 Port P

This port is associated with the PMF.

Port P pins PP[7:0] can be used for either general purpose I/O, or with the PMF subsystem.

19.4.3.10 Port H

This port is associated with the FlexRay.

Port H pins PH[6:4] and PH[2:0] can be used for either general purpose I/O, or with FlexRay subsystem.

Port H pins PH[7] and PH[3] can be used as general purpose I/O.

19.4.3.11 Port J

This port is associated with the FlexRay and PMF.

Port J pins PJ[7:4] can be used for either general purpose I/O, or with the FlexRay subsystem.

Port J pins PJ[2:0] can be used for either general purpose I/O, or with the PMF subsystem.

19.4.3.12 Port AD

This port is associated with the ATD.

Port AD pins PAD[15:00] can be used for either general purpose I/O, or with the ATD subsystem.

19.5 Initialization Information

19.5.1 Port Data and Data Direction Register writes

It is not recommended to write PORTx/PTx and DDRx in a word access. When changing the register pins from inputs to outputs, the data may have extra transitions during the write access. Initialize the port data register before enabling the outputs.

Chapter 20
Pulse Width Modulator with Fault Protection (PMF15B6C) Module

Table 20-1. Revision History

Revision Number	Revision Date	Sections Affected	Description of Changes
V01.05	19 AUG 2002		Updates after review with verification team.
V02.05	15 MAY 2003		Updated version number to match clearcase label
V02.06	10 NOV 2010	20.3.2.24/20-924 20.3.2.29/20-927	Fixed PMFDTMA,PMFDTMB maximum cycle count (4096)

20.1 Introduction

The Pulse width Modulator with Fault protection (PMF) module can be configured for one, two, or three complementary pairs. For example:

- One complementary pair and four independent PWM outputs
- Two complementary pair and two independent PWM outputs
- Three complementary pair and zero independent PWM outputs
- Zero complementary pair and six independent PWM outputs

All PWM outputs can be generated from the same counter, or each pair can have its own counter for three independent PWM frequencies. Complementary operation permits programmable dead-time insertion, distortion correction through current sensing by software, and separate top and bottom output polarity control. Each counter value is programmable to support a continuously variable PWM frequency. Both edge- and center-aligned synchronous pulse width-control and full range modulation from 0 percent to 100 percent, are supported. The PMF is capable of controlling most motor types: AC induction motors (ACIM), both brushless (BLDC) and brush DC motors (BDC), switched (SRM) and variable reluctance motors (VRM), and stepper motors.

20.1.1 Features

- Three complementary PWM signal pairs, or six independent PWM signals
- Three 15-bit counters
- Features of complementary channel operation
- Deadtime insertion
- Separate top and bottom pulse width correction via current status inputs or software
- Separate top and bottom polarity control

- Edge-aligned or center-aligned PWM signals
- Half-cycle reload capability
- Integral reload rates from 1 to 16
- Individual software-controlled PWM output
- Programmable fault protection
- Polarity control

20.1.2 Modes of Operation

Care must be exercised when using this module in the modes listed in Table 20-2. PWM outputs are placed in their inactive states in STOP mode, and optionally under WAIT and FREEZE modes. PWM outputs will be reactivated (assuming they were active to begin with) when these modes are exited

Table 20-2. Modes When PWM Operation is Restricted

Mode	Description
STOP	PWM outputs are disabled
WAIT	PWM outputs are disabled as a function of the PMFWAI bit.
FREEZE	PWM outputs are disabled as a function of the PMFFRZ bit.

20.1.3 Block Diagrams

Figure 20-1 provides an overview of the PMF module.

The Mux/Swap/Current Sense block is tightly integrated with the dead time insertion block. This detail is shown in Figure 20-2.

NOTE

It is possible to have both channels of a complementary pair to be high. For example, if the TOPNEGA (negative polarity for PWM0), BOTNEGA (negative polarity for PWM1), MASK0 and MASK1 bits are set, both the PWM complementary outputs of generator A will be high. See Section 20.3.2.2, “PMF Configure 1 Register (PMFCFG1)” for the description of TOPNEG and BOTNEG bits, and Section 20.3.2.3, “PMF Configure 2 Register (PMFCFG2)” for the description of the MSK0 and MSK1 bits.

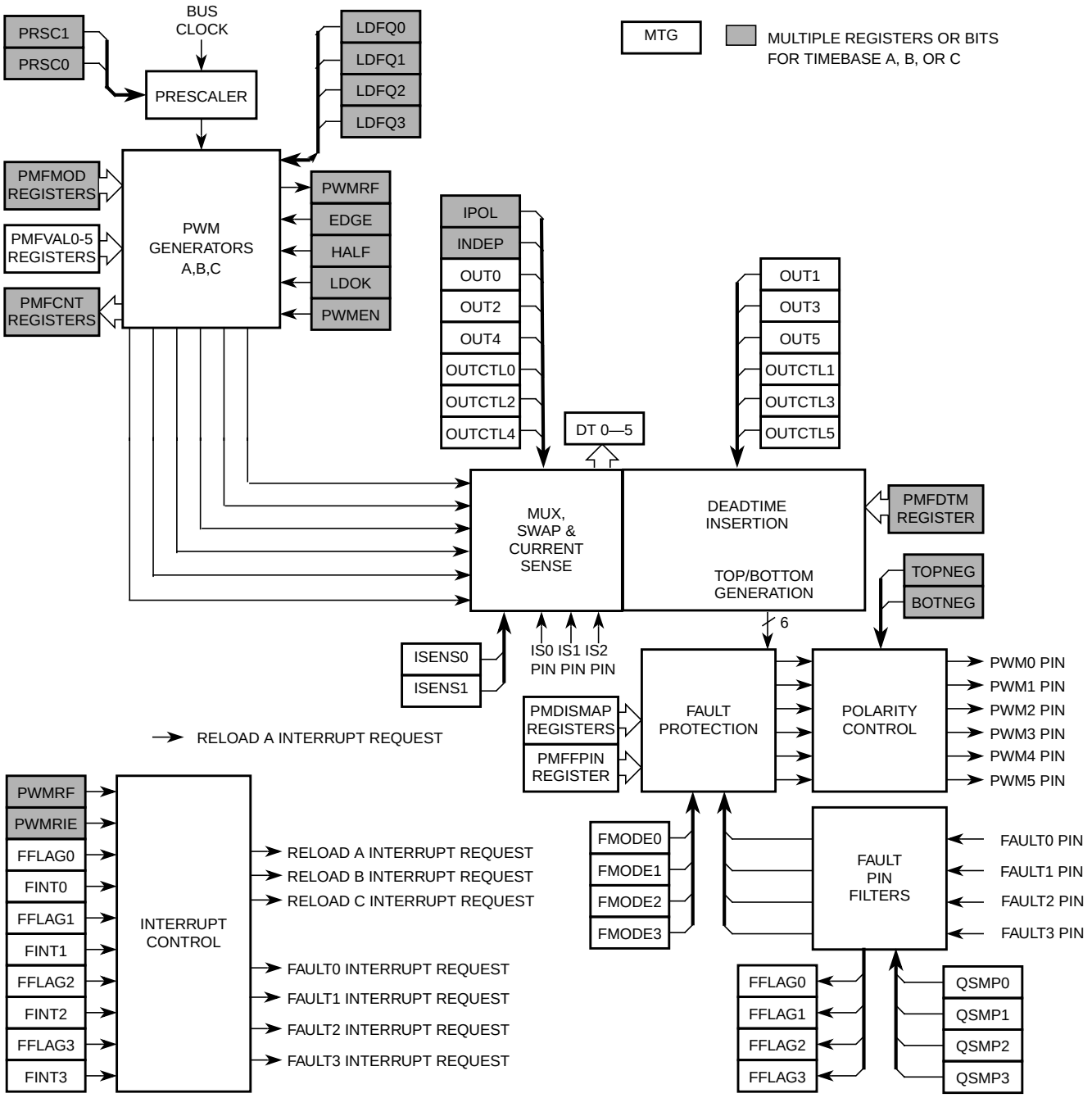


Figure 20-1. PMF Block Diagram

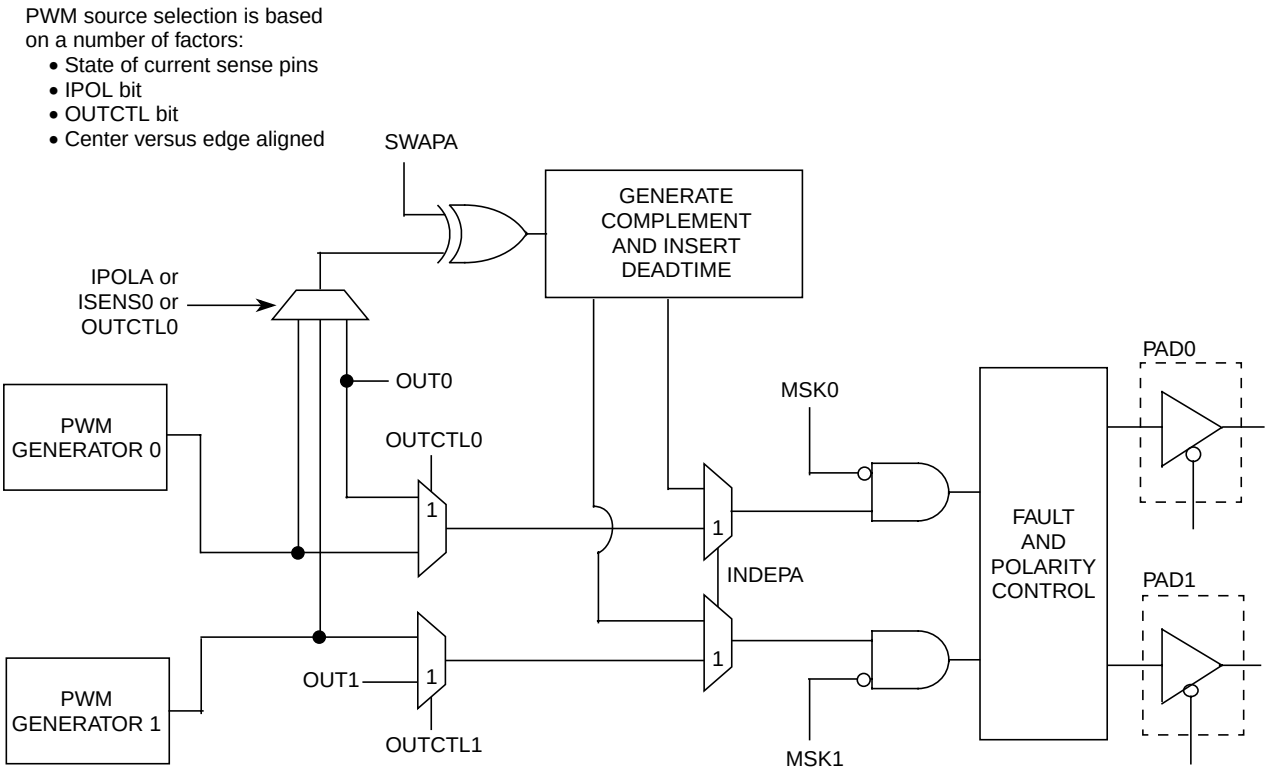


Figure 20-2. Detail of Mux, Swap, and Deadtime Functions

20.2 Signal Descriptions

The pulse width modulator has external pins named PWM0–5, FAULT0–3, and $\overline{IS0}$–$\overline{IS2}$.

20.2.1 PWM0–PWM5 Pins

PWM0–PWM5 are the output pins of the six PWM channels.

20.2.2 FAULT0–FAULT3 Pins

FAULT0–FAULT3 are input pins for disabling selected PWM outputs.

20.2.3 $\overline{IS0}$–$\overline{IS2}$ Pins

$\overline{IS0}$–$\overline{IS2}$ are current status pins for top/bottom pulse width correction in complementary channel operation while deadtime is asserted.

20.3 Memory Map and Registers

20.3.1 Module Memory Map

A summary of the registers associated with the PMF module is shown in Figure 20-3. Detailed descriptions of the registers and bits are given in the subsections that follow.

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
$0000	PMFCFG0	R W	WP	MTG	EDGEC	EDGEB	EDGEA	INDEPC	INDEPB	INDEPA
$0001	PMFCFG1	R W	ENHA	0	BOTNEGC	TOPNEGC	BOTNEGB	TOPNEGB	BOTNEGA	TOPNEGA
$0002	PMFCFG2	R W	0	0	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0
$0003	PMFCFG3	R W	PMFWAI	PMFFRZ	0	VLMODE		SWAPC	SWAPB	SWAPA
$0004	PMFFCTL	R W	FMODE3	FIE3	FMODE2	FIE2	FMODE1	FIE1	FMODE0	FIE0
$0005	PMFFPIN	R W	0	FPINE3	0	FPINE2	0	FPINE1	0	FPINE0
$0006	PMFFSTA	R W	0	FFLAG3	0	FFLAG2	0	FFLAG1	0	FFLAG0
$0007	PMFQSMP	R W	QSMP3		QSMP2		QSMP1		QSMP0	
$0008	PMFDMPA	R W	DMP13	DMP12	DMP11	DMP10	DMP03	DMP02	DMP01	DMP00
$0009	PMFDMPB	R W	DMP33	DMP32	DMP31	DMP30	DMP23	DMP22	DMP21	DMP20
$000A	PMFDMPC	R W	DMP53	DMP52	DMP51	DMP50	DMP43	DMP42	DMP41	DMP53
$000B	Reserved	R W								
$000C	PMFOUTC	R W	0	0	OUTCTL5	OUTCTL4	OUTCTL3	OUTCTL2	OUTCTL1	OUTCTL0
$000D	PMFOUTB	R W	0	0	OUT5	OUT4	OUT3	OUT2	OUT1	OUT0

= Unimplemented or Reserved

Figure 20-3. Quick Reference to PMF Registers (Sheet 1 of 4)

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
$000E	PMFDTMS	R	0	0	DT5	DT4	DT3	DT2	DT1	DT0
		W								
$000F	PMFCCTL	R	0	0	ISENS		0	IPOLC	IPOLB	IPOLA
		W								
$0010	PMFVAL0	R	PMFVAL0							
		W								
$0011	PMFVAL0	R	PMFVAL0							
		W								
$0012	PMFVAL1	R	PMFVAL1							
		W								
$0013	PMFVAL1	R	PMFVAL1							
		W								
$0014	PMFVAL2	R	PMFVAL2							
		W								
$0015	PMFVAL2	R	PMFVAL2							
		W								
$0016	PMFVAL3	R	PMFVAL3							
		W								
$0017	PMFVAL3	R	PMFVAL3							
		W								
$0018	PMFVAL4	R	PMFVAL4							
		W								
$0019	PMFVAL4	R	PMFVAL4							
		W								
$001A	PMFVAL5	R	PMFVAL5							
		W								
$001B	PMFVAL5	R	PMFVAL5							
		W								
$001C–$001F	Reserved	R								
		W								
$0020	PMFENCA	R	PWMENA	0	0	0	0	0	LDOKA	PWMRIEA
		W								
$0021	PMFFQCA	R	LDFQA				HALFA	PRSCA		PWMRFA
		W								

= Unimplemented or Reserved

Figure 20-3. Quick Reference to PMF Registers (Sheet 2 of 4)

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
$0022	PMFCNTA	R	0	PMFCNTA						
		W								
$0023	PMFCNTA	R	PMFCNTA							
		W								
$0024	PMFMODA	R	0	PMFMODA						
		W								
$0025	PMFMODA	R	PMFMODA							
		W								
$0026	PMFDTMA	R	0	0	0	0	PMFDTMA			
		W								
$0027	PMFDTMA	R	PMFDTMA							
		W								
$0028	PMFENCB	R	PWMENB	0	0	0	0	0	LDOKB	PWMRIEB
		W								
$0029	PMFFQCB	R	LDFQB				HALFB	PRSCB		PWMRFB
		W								
$002A	PMFCNTB	R	0	PMFCNTB						
		W								
$002B	PMFCNTB	R	PMFCNTB							
		W								
$002C	PMFMODB	R	0	PMFMODB						
		W								
$002D	PMFMODB	R	PMFMODB							
		W								
$002E	PMFDTMB	R	0	0	0	0	PMFDTMB			
		W								
$002F	PMFDTMB	R	PMFDTMB							
		W								
$0030	PMFENCC	R	PWMENC	0	0	0	0	0	LDOKC	PWMRIEC
		W								
$0031	PMFFQCC	R	LDFQC				HALFC	PRSCC		PWMRFC
		W								

= Unimplemented or Reserved

Figure 20-3. Quick Reference to PMF Registers (Sheet 3 of 4)

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
$0032	PMFCNTC	R	0	PMFCNTC						
		W								
$0033	PMFCNTC	R	PMFCNTC							
		W								
$0034	PMFMODC	R	0	PMFMODC						
		W								
$0035	PMFMODC	R	PMFMODC							
		W								
$0036	PMFDTMC	R	0	0	0	0	PMFDTMC			
		W								
$0037	PMFDTMC	R	PMFDTMC							
		W								
$0038–$003F	Reserved	R								
		W								

= Unimplemented or Reserved

Figure 20-3. Quick Reference to PMF Registers (Sheet 4 of 4)

20.3.2 Register Descriptions

The address of a register is the sum of a base address and an address offset. The base address is defined at the chip level and the address offset is defined at the module level.

20.3.2.1 PMF Configure 0 Register (PMFCFG0)

Address: $0000

	7	6	5	4	3	2	1	0
R W	WP	MTG	EDGEC	EDGEB	EDGEA	INDEPC	INDEPB	INDEPA
Reset	0	0	0	0	0	0	0	0

Figure 20-4. PMF Configure 0 Register (PMFCFG0)

Read anytime. See bit description for write conditions.

Table 20-3. PMFCFG0 Field Descriptions

Field	Description
7 WP	**Write Protect** — This bit enables write protection to be used for all write-protectable registers. While clear, WP allows write-protected registers to be written. When set, WP prevents any further writes to write-protected registers. Once set, WP can be cleared only by reset. 0 Write-protectable registers may be written. 1 Write-protectable registers are write-protected.

Table 20-3. PMFCFG0 Field Descriptions (continued)

Field	Description
6 MTG	**Multiple Timebase Generators** — This bit determines the number of timebase counters used. Once set, MTG can be cleared only by reset. If MTG is set, PWM generators B and C and registers $0028 – $0037 are available. The three generators have their own variable frequencies and are not synchronized. If MTG is cleared, PMF registers from $0028 – $0037 can not be written and read zeroes, and bits EDGEC and EDGEB are ignored. Pair A, Pair B and Pair C PWMs are synchronized to PWM generator A and use registers from $0020 – $0027. 0 Single timebase generator. 1 Multiple timebase generators.
5 EDGEC	**Edge-Aligned or Center-Aligned PWM for Pair C** — This bit determines whether PWM4 and PWM5 channels will use edge-aligned or center-aligned waveforms. This bit has no effect if MTG bit is cleared. This bit cannot be modified after the WP bit is set. 0 PWM4 and PWM5 are center-aligned PWMs 1 PWM4 and PWM5 are edge-aligned PWMs
4 EDGEB	**Edge-Aligned or Center-Aligned PWM for Pair B** — This bit determines whether PWM2 and PWM3 channels will use edge-aligned or center-aligned waveforms. This bit has no effect if MTG bit is cleared. This bit cannot be modified after the WP bit is set. 0 PWM2 and PWM3 are center-aligned PWMs 1 PWM2 and PWM3 are edge-aligned PWMs
3 EDGEA	**Edge-Aligned or Center-Aligned PWM for Pair A**— This bit determines whether PWM0 and PWM1 channels will use edge-aligned or center-aligned waveforms. It determines waveforms for Pair B and Pair C if the MTG bit is cleared. This bit cannot be modified after the WP bit is set. 0 PWM0 and PWM1 are center-aligned PWMs 1 PWM0 and PWM1 are edge-aligned PWMs
2 INDEPC	**Independent or Complimentary Operation for Pair C**— This bit determines if the PWM channels 4 and 5 will be independent PWMs or complementary PWMs. This bit cannot be modified after the WP bit is set. 0 PWM4 and PWM5 are complementary PWM pair 1 PWM4 and PWM5 are independent PWMs
1 INDEPB	**Independent or Complimentary Operation for Pair B**— This bit determines if the PWM channels 2 and 3 will be independent PWMs or complementary PWMs. This bit cannot be modified after the WP bit is set. 0 PWM2 and PWM3 are complementary PWM pair 1 PWM2 and PWM3 are independent PWMs
0 INDEPA	**Independent or Complimentary Operation for Pair A**— This bit determines if the PWM channels 0 and 1 will be independent PWMs or complementary PWMs. This bit cannot be modified after the WP bit is set. 0 PWM0 and PWM1 are complementary PWM pair 1 PWM0 and PWM1 are independent PWMs

20.3.2.2 PMF Configure 1 Register (PMFCFG1)

Address: $0001

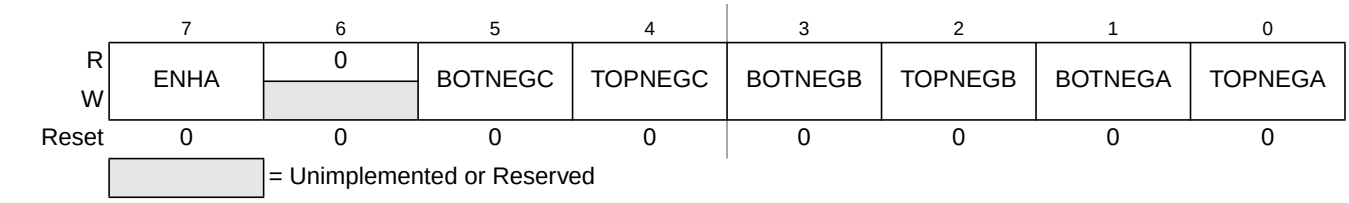


Figure 20-5. PMF Configure 1 Register (PMFCFG1)

Read anytime. This register cannot be modified after the WP bit is set.

A normal PWM output or positive polarity means that the PWM channel outputs high when the counter value is smaller than or equal to the pulse width value and outputs low otherwise. An inverted output or negative polarity means that the PWM channel outputs low when the counter value is smaller than or equal to the pulse width value and outputs high otherwise.

Table 20-4. PMFCFG1 Field Descriptions

Field	Description
7 ENHA	**Enable Hardware Acceleration** — This bit enables writing to the VLMODE[1:0], SWAPC, SWAPB, and SWAPA bits in the PMFCFG3 register. This bit cannot be modified after the WP bit is set. 0 Disable writing to VLMODE[1:0], SWAPC, SWAPB, and SWAPA bits 1 Enable writing to VLMODE[1:0], SWAPC, SWAPB, and SWAPA bits
5 BOTNEGC	**Pair C Bottom-Side PWM Polarity** — This bit determines the polarity for Pair C bottom-side PWM (PWM5). This bit cannot be modified after the WP bit is set. 0 Positive PWM5 polarity 1 Negative PWM5 polarity
4 TOPNEGC	**Pair C Top-Side PWM Polarity** — This bit determines the polarity for Pair C top-side PWM (PWM4). This bit cannot be modified after the WP bit is set. 0 Positive PWM4 polarity 1 Negative PWM4 polarity
3 BOTNEGB	**Pair B Bottom-Side PWM Polarity** — This bit determines the polarity for Pair B bottom-side PWM (PWM3). This bit cannot be modified after the WP bit is set. 0 Positive PWM3 polarity 1 Negative PWM3 polarity
2 TOPNEGB	**Pair B Top-Side PWM Polarity** — This bit determines the polarity for Pair B top-side PWM (PWM2). This bit cannot be modified after the WP bit is set. 0 Positive PWM2 polarity 1 Negative PWM2 polarity
1 BOTNEGA	**Pair A Bottom-Side PWM Polarity** — This bit determines the polarity for Pair A bottom-side PWM (PWM1). This bit cannot be modified after the WP bit is set. 0 Positive PWM1 polarity 1 Negative PWM1 polarity
0 TOPNEGA	**Pair A Top-Side PWM Polarity** — This bit determines the polarity for Pair A top-side PWM (PWM0). This bit cannot be modified after the WP bit is set. 0 Positive PWM0 polarity 1 Negative PWM0 polarity

20.3.2.3 PMF Configure 2 Register (PMFCFG2)

Address: $0002

	7	6	5	4	3	2	1	0
R	0	0	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 20-6. PMF Configure 2 Register (PMFCFG2)

Read and write anytime.

Table 20-5. PMFCFG2 Field Descriptions

Field	Description
5–0 MSK[5:0]	**Mask PWMx** 0 PWMx is unmasked. 1 PWMx is masked and the channel is set to a value of 0 percent duty cycle. where x is 0, 1, 2, 3, 4, and 5

CAUTION

When using the TOPNEG/BOTNEG bits and the MSKx bits at the same time, when in complementary mode, it is possible to have both PMF channel outputs of a channel pair set to one.

20.3.2.4 PMF Configure 3 Register (PMFCFG3)

Address: $0003

	7	6	5	4	3	2	1	0
R	PMFWAI	PMFFRZ	0	VLMODE		SWAPC	SWAPB	SWAPA
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 20-7. PMF Configure 3 Register (PMFCFG3)

Read and write anytime.

Table 20-6. PMFCFG3 Field Descriptions

Field	Description
7 PMFWAI	**PMF Stops While in WAIT Mode** — When set to zero, the PWM generators will continue to run while the chip is in WAIT mode. In this mode, the peripheral clock continues to run but the CPU clock does not. If the device enters WAIT mode and this bit is one, then the PWM outputs will be switched to their inactive state until WAIT mode is exited. At that point the PWM pins will resume operation as programmed in the PWM registers. 0 PMF continues to run in WAIT mode. 1 PMF is disabled in WAIT mode.
6 PMFFRZ	**PMF Stops While in FREEZE Mode** — When set to zero, the PWM generators will continue to run while the chip is in FREEZE mode. If the device enters FREEZE mode and this bit is one, then the PWM outputs will be switched to their inactive state until FREEZE mode is exited. At that point the PWM pins will resume operation as programmed in the PWM registers. 0 PMF continues to run in FREEZE mode. 1 PMF is disabled in FREEZE mode.
4–3 VLMODE	**Value Register Load Mode** — This field determines the way the value registers are being loaded. This field can only be written if ENHA is set. 00 Each value register is accessed independently 01 Writing to value register zero also writes to value registers one to five 10 Writing to value register zero also writes to value registers one to three 11 Reserved (defaults to independent access)

Table 20-6. PMFCFG3 Field Descriptions (continued)

Field	Description
2 SWAPC	**Swap Pair C** — This bit can only be written if ENHA is set. 0 No swap. 1 PWM4 and PWM5 are swapped only in complementary mode.
1 SWAPB	**Swap Pair B** — This bit can only be written if ENHA is set. 0 No swap. 1 PWM2 and PWM3 are swapped only in complementary mode.
0 SWAPA	**Swap Pair A** — This bit can only be written if ENHA is set. 0 No swap. 1 PWM0 and PWM1 are swapped only in complementary mode.

20.3.2.5 PMF Fault Control Register (PMFFCTL)

Address: $0004

	7	6	5	4	3	2	1	0
R W	FMODE3	FIE3	FMODE2	FIE2	FMODE1	FIE1	FMODE0	FIE0
Reset	0	0	0	0	0	0	0	0

Figure 20-8. PMF Fault Control Register (PMFFCTL)

Read and write anytime.

Table 20-7. PMFFCTL Field Descriptions

Field	Description
7, 5, 3, 1 FMODE[3:0]	**Fault x Pin Clearing Mode** — This bit selects automatic or manual clearing of FAULTx pin faults. See Section 20.4.8.2, “Automatic Fault Clearing” and Section 20.4.8.3, “Manual Fault Clearing” for more details. 0 Manual fault clearing of FAULTx pin faults. 1 Automatic fault clearing of FAULTx pin faults. where x is 0, 1, 2 and 3.
6, 4, 2, 0 FIE]3:0]	**Fault x Pin Interrupt Enable** — This bit enables CPU interrupt requests to be generated by the FAULTx pin. The fault protection circuit is independent of the FIEx bit and is active when FPINEx is set. If a fault is detected, the PWM pins are disabled according to the PMF Disable Mapping registers. 0 Fault x CPU interrupt requests disabled. 1 Fault x CPU interrupt requests enabled. where x is 0, 1, 2 and 3.

20.3.2.6 PMF Fault Pin Enable Register (PMFFPIN)

Address: $0005

	7	6	5	4	3	2	1	0
R	0	FPINE3	0	FPINE2	0	FPINE1	0	FPINE0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 20-9. PMF Fault Pin Enable Register (PMFFPIN)

Read anytime. This register cannot be modified after the WP bit is set.

Table 20-8. PMFFPIN Field Descriptions

Field	Description
6, 4, 2, 0 FPINE[3:0]	**Fault x Pin Enable** 0 FAULTx pin is disabled for fault protection. 1 FAULTx pin is enabled for fault protection. where x is 0, 1, 2 and 3

20.3.2.7 PMF Fault Status Register (PMFFSTA)

Address: $0006

	7	6	5	4	3	2	1	0
R	0	FFLAG3	0	FFLAG2	0	FFLAG1	0	FFLAG0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 20-10. PMF Fault Flag Register (PMFFSTA)

Read and write anytime.

Table 20-9. PMFFSTA Field Descriptions

Field	Description
6, 4, 2, 0 FFLAG[3:0]	**Fault x pin Flag** — This flag is set after the required number of samples have been detected after a rising edge on the FAULTx pin. Writing a logic one to FFLAGx clears it. Writing a logic zero has no effect. The fault protection is enabled when FPINEx is set even when the PWMs are not enabled; therefore, a fault will be latched in, requiring to be cleared in order to prevent an interrupt. 0 No fault on the FAULTx pin. 1 Fault on the FAULTx pin. **Note:** Clearing FFLAGx satisfies pending FFLAGx CPU interrupt requests. where x is 0, 1, 2 and 3

20.3.2.8 PMF Fault Qualifying Samples Register (PMFQSMP)

Address: $0007

	7	6	5	4	3	2	1	0
R	QSMP3		QSMP2		QSMP1		QSMP0	
W								
Reset	0	0	0	0	0	0	0	0

Figure 20-11. PMF Fault Qualifying Samples Register (PMFQSMP)

Read anytime. This register cannot be modified after the WP bit is set.

Table 20-10. PMFQSMP Field Descriptions

Field	Description
7–0 QSMP[3:0]	**Fault x Qualifying Samples** — This field indicates the number of consecutive samples taken at the FAULTx pin to determine if a fault is detected. The first sample is qualified after two bus cycles from the time the fault is present and each sample after that is taken every four bus cycles. See Table 20-11. where x is 0, 1, 2 and 3

Table 20-11. Qualifying Samples

QSMPx	Number of Samples
00	1 sample(1)
01	5 samples
10	10 samples
11	15 samples

1. There is an asynchronous path from fault pin to disable PWMs immediately but the fault is qualified in two bus cycles.

20.3.2.9 PMF Disable Mapping Registers

Address: $0008

	7	6	5	4	3	2	1	0
R W	DMP13	DMP12	DMP11	DMP10	DMP03	DMP02	DMP01	DMP00
Reset	0	0	0	0	0	0	0	0

Figure 20-12. PMF Disable Mapping A Register (PMFDMPA)

Address: $0009

	7	6	5	4	3	2	1	0
R W	DMP33	DMP32	DMP31	DMP30	DMP23	DMP22	DMP21	DMP20
Reset	0	0	0	0	0	0	0	0

Figure 20-13. PMF Disable Mapping B Register (PMFDMPB)

Address: $000A

	7	6	5	4	3	2	1	0
R W	DMP53	DMP52	DMP51	DMP50	DMP43	DMP42	DMP41	DMP53
Reset	0	0	0	0	0	0	0	0

Figure 20-14. PMF Disable Mapping C Register (PMFDMPC)

Read anytime. These registers cannot be modified after the WP bit is set.

The fault decoder disables PWM pins selected by the fault logic and the disable mapping registers. See Figure 20-15. Each bank of four bits in the disable mapping registers control the mapping of a single PWM pin. Refer to Table 20-12.

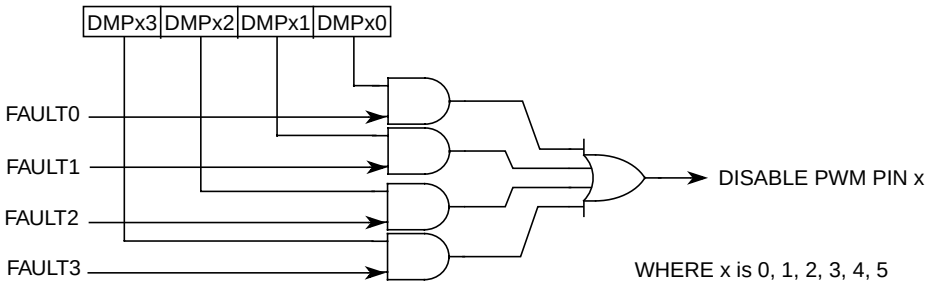


Figure 20-15. Fault Decoder

Table 20-12. Fault Mapping

PWM Pin	Controlling Register Bits
PWM0	DMP03 – DMP00
PWM1	DMP13 – DMP10
PWM2	DMP23 – DMP20
PWM3	DMP33 – DMP30
PWM4	DMP43 – DMP40
PWM5	DMP53 – DMP50

20.3.2.10 PMF Output Control Register (PMFOUTC)

Address: $000C

	7	6	5	4	3	2	1	0
R	0	0	OUTCTL5	OUTCTL4	OUTCTL3	OUTCTL2	OUTCTL1	OUTCTL0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 20-16. PMF Output Control Register (PMFOUTC)

Read and write anytime.

Table 20-13. PMFOUTC Field Descriptions

Field	Description
5–0 OUTCTL[5:0]	**OUTCTLx Bits** — These bits enable software control of their corresponding PWM pin. When OUTCTLx is set, the OUTx bit activates and deactivates the PWMx output. When operating the PWM in complementary mode, these bits must be switched in pairs for proper operation. That is OUTCTL0 and OUTCTL1 must have the same value; OUTCTL2 and OUTCTL3 must have the same value; and OUTCTL4 and OUTCTL5 must have the same value. 0 Software control disabled 1 Software control enabled where X is 0, 1, 2, 3, 4 and 5

20.3.2.11 PMF Output Control Bit Register (PMFOUTB)

Address: $000D

	7	6	5	4	3	2	1	0
R	0	0	OUT5	OUT4	OUT3	OUT2	OUT1	OUT0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 20-17. PMF Output Control Bit Register (PMFOUTB)

Read and write anytime.

Table 20-14. PMFOUTB Field Descriptions

Field	Description
5–0 OUT[5:0]	**OUT Bits** — When the corresponding OUTCTL bit is set, these bits control the PWM pins, illustrated in Table 20-15.

Table 20-15. Software Output Control

OUTx Bit	Complementary Channel Operation	Independent Channel Operation
OUT0	1 — PWM0 is active 0 — PWM0 is inactive	1 — PWM0 is active 0 — PWM0 is inactive
OUT1	1 — PWM1 is complement of PWM0 0 — PWM1 is inactive	1 — PWM1 is active 0 — PWM1 is inactive
OUT2	1 — PWM2 is active 0 — PWM2 is inactive	1 — PWM2 is active 0 — PWM2 is inactive
OUT3	1 — PWM3 is complement of PWM2 0 — PWM3 is inactive	1 — PWM3 is active 0 — PWM3 is inactive
OUT4	1 — PWM4 is active 0 — PWM4 is inactive	1 — PWM4 is active 0 — PWM4 is inactive
OUT5	1 — PWM5 is complement of PWM4 0 — PWM5 is inactive	1 — PWM5 is active 0 — PWM5 is inactive

20.3.2.12 PMF Deadtime Sample Register (PMFDTMS)

Address: $000E

	7	6	5	4	3	2	1	0
R	0	0	DT5	DT4	DT3	DT2	DT1	DT0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 20-18. PMF Deadtime Sample Register (PMFDTMS)

Read anytime and writes have no effect.

Table 20-16. PMFDTMS Field Descriptions

Field	Description
5–0 DT[5:0}	**DTx Bits** — The DTx bits are grouped in pairs, DT0 and DT1, DT2 and DT3, DT4 and DT5. Each pair reflects the corresponding $\overline{ISx}$ pin value as sampled at the end of deadtime.

20.3.2.13 PMF Correction Control Register (PMFCCTL)

Address: $000F

	7	6	5	4	3	2	1	0
R	0	0	ISENS		0	IPOLC	IPOLB	IPOLA
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 20-19. PMF Correction Control Register (PMFCCTL)

Read and write anytime.

Table 20-17. PMFCCTL Field Descriptions

Field	Description
5–4 ISENS	**Current Status Sensing Method** — This field selects the top/bottom correction scheme, illustrated in Table 20-18. **Note:** Assume the user will provide current sensing circuitry causing the voltage at the corresponding input pin to be low for positive current and high for negative current. In addition, it assumes the top PWMs are PWM 0, 2, and 4 while the bottom PWMs are PWM 1, 3, and 5. **Note:** The ISENS bits are not buffered. Changing the current status sensing method can affect the present PWM cycle.
2 IPOLC	**Current Polarity** — This buffered bit selects the PMF Value register for the PWM4 and PWM5 pins in top/bottom software correction in complementary mode. 0 PMF Value 4 register in next PWM cycle. 1 PMF Value 5 register in next PWM cycle.
1 IPOLB	**Current Polarity** — This buffered bit selects the PMF Value register for the PWM2 and PWM3 pins in top/bottom software correction in complementary mode. 0 PMF Value 2 register in next PWM cycle. 1 PMF Value 3 register in next PWM cycle.
0 IPOLA	**Current Polarity** — This buffered bit selects the PMF Value register for the PWM0 and PWM1 pins in top/bottom software correction in complementary mode. 0 PMF Value 0 register in next PWM cycle. 1 PMF Value 1 register in next PWM cycle.

Table 20-18. Correction Method Selection

ISENS	Correction Method
00	No correction(1)
01	Manual correction
10	Current status sample correction on pins $\overline{IS0}$, $\overline{IS1}$, and $\overline{IS2}$ during deadtime(2)

Table 20-18. Correction Method Selection

ISENS	Correction Method
11	Current status sample on pins $\overline{IS0}$, $\overline{IS1}$, and $\overline{IS2}$(3) At the half cycle in center-aligned operation At the end of the cycle in edge-aligned operation

1. The current status pins can be used as general purpose input/output ports.
2. The polarity of the $\overline{ISx}$ pin is latched when both the top and bottom PWMs are off. At the 0% and 100% duty cycle boundaries, there is no deadtime, so no new current value is sensed.
3. Current is sensed even with 0% or 100% duty cycle.

NOTE

The IPOLx bits take effect at the beginning of the next load cycle, regardless of the state of the load okay bit, LDOK. Select top/bottom software correction by writing 01 to the current select bits, ISENS[1:0], in the PWM control register. Reading the IPOLx bits read the buffered value and not necessarily the value currently in effect.

20.3.2.14 PMF Value 0 Register (PMFVAL0)

Address: $0010

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PMFVAL0															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 20-20. PMF Value 0 Register (PMFVAL0)

Read and write anytime.

Table 20-19. PMFVAL0 Field Descriptions

Field	Description
15–0 PMFVAL0	**PMF Value 0 Bits** — The 16-bit signed value in this buffered register is the pulse width in PWM0 clock period. A value less than or equal to zero deactivates the PWM output for the entire PWM period. A value greater than, or equal to the modulus, activates the PWM output for the entire PWM period. See Table 20-37**.** The terms activate and deactivate refer to the high and low logic states of the PWM output. **Note:** PMFVAL0 is buffered. The value written does not take effect until the LDOK bit is set and the next PWM load cycle begins Reading PMFVAL0 reads the value in the buffer and not necessarily the value the PWM generator is currently using.

20.3.2.15 PMF Value 1 Register (PMFVAL1)

Address: $0012

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PMFVAL1															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 20-21. PMF Value 1 Register (PMFVAL1)

Read and write anytime.

Table 20-20. PMFVAL1 Field Descriptions

Field	Description
15–0 PMFVAL1	**PMF Value 1 Bits** — The 16-bit signed value in this buffered register is the pulse width in PWM1 clock period. A value less than or equal to zero deactivates the PWM output for the entire PWM period. A value greater than, or equal to the modulus, activates the PWM output for the entire PWM period. SeeTable 20-37**.** The terms activate and deactivate refer to the high and low logic states of the PWM output. **Note:** PMFVAL1 is buffered. The value written does not take effect until the LDOK bit is set and the next PWM load cycle begins. Reading PMFVAL1 reads the value in the buffer and not necessarily the value the PWM generator is currently using.

20.3.2.16 PMF Value 2 Register (PMFVAL2)

Address: $0014

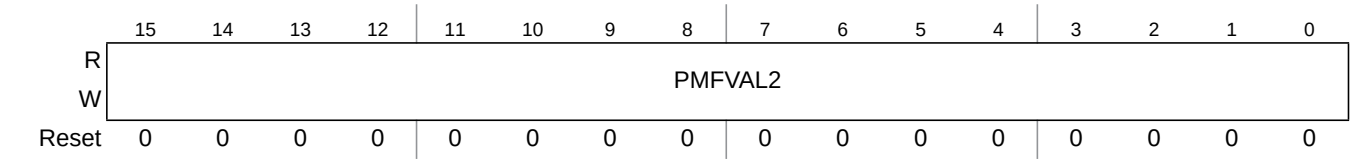


Figure 20-22. PMF Value 2 Register (PMFVAL2)

Read and write anytime.

Table 20-21. PMFVAL2 Field Descriptions

Field	Description
15–0 PMFVAL2	**PMF Value 2 Bits** — The 16-bit signed value in this buffered register is the pulse width in PWM2 clock period. A value less than or equal to zero deactivates the PWM output for the entire PWM period. A value greater than, or equal to the modulus, activates the PWM output for the entire PWM period. See Table 20-37**.** The terms activate and deactivate refer to the high and low logic states of the PWM output. **Note:** PMFVAL2 is buffered. The value written does not take effect until the LDOK bit is set and the next PWM load cycle begins. Reading PMFVAL2 reads the value in the buffer and not necessarily the value the PWM generator is currently using.

20.3.2.17 PMF Value 3 Register (PMFVAL3)

Address: $0016

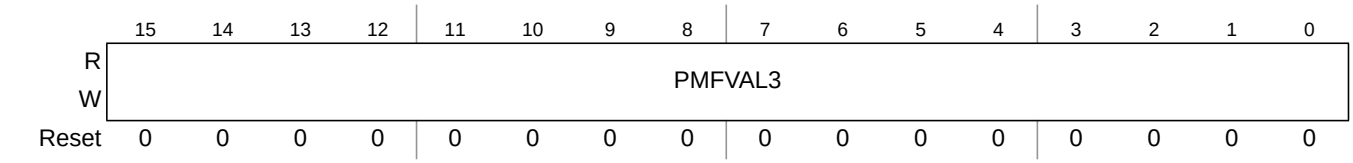


Figure 20-23. PMF Value 3 Register (PMFVAL3)

Read and write anytime.

Table 20-22. PMFVAL3 Field Descriptions

Field	Description
15–0 PMFVAL3	**PMF Value 3 Bits** — The 16-bit signed value in this buffered register is the pulse width in PWM3 clock period. A value less than or equal to zero deactivates the PWM output for the entire PWM period. A value greater than, or equal to the modulus, activates the PWM output for the entire PWM period. See Table 20-37**.** The terms activate and deactivate refer to the high and low logic states of the PWM output. **Note:** PMFVAL3 is buffered. The value written does not take effect until the LDOK bit is set and the next PWM load cycle begins. Reading PMFVAL3 reads the value in the buffer and not necessarily the value the PWM generator is currently using.

20.3.2.18 PMF Value 4 Register (PMFVAL4)

Address: $0018

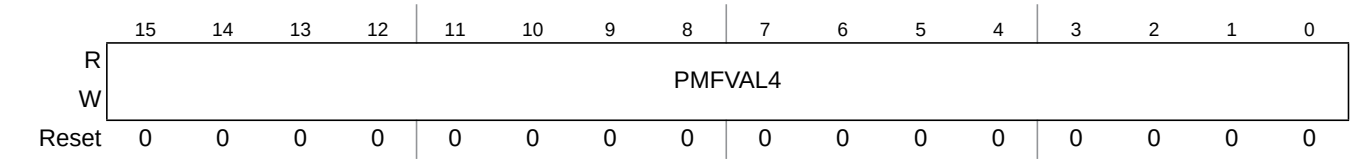


Figure 20-24. PMF Value 4 Register (PMFVAL4)

Read and write anytime.

Table 20-23. PMFVAL4 Field Descriptions

Field	Description
15–0 PMFVAL4	**PMF Value 4 Bits** — The 16-bit signed value in this buffered register is the pulse width in PWM4 clock period. A value less than or equal to zero deactivates the PWM output for the entire PWM period. A value greater than, or equal to the modulus, activates the PWM output for the entire PWM period. See Table 20-37**.** The terms activate and deactivate refer to the high and low logic states of the PWM output. **Note:** PMFVAL4 is buffered. The value written does not take effect until the LDOK bit is set and the next PWM load cycle begins. Reading PMFVAL4 reads the value in the buffer and not necessarily the value the PWM generator is currently using.

20.3.2.19 PMF Value 5 Register (PMFVAL5)

Address: $001A

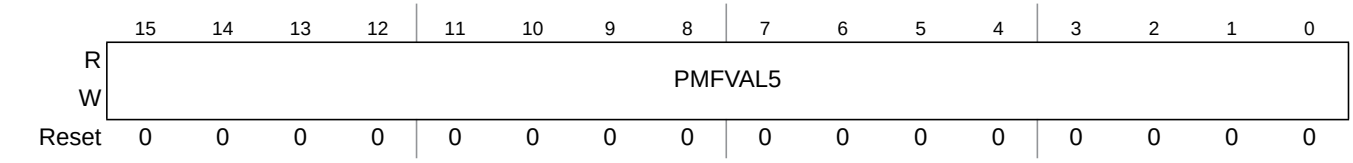


Figure 20-25. PMF Value 5 Register (PMFVAL5)

Read and write anytime.

Table 20-24. PMFVAL5 Field Descriptions

Field	Description
15–0 PMFVAL5	**PMF Value 5 Bits** — The 16-bit signed value in this buffered register is the pulse width in PWM5 clock period. A value less than or equal to zero deactivates the PWM output for the entire PWM period. A value greater than, or equal to the modulus, activates the PWM output for the entire PWM period. See Table 20-37**.** The terms activate and deactivate refer to the high and low logic states of the PWM output. **Note:** PMFVAL5 is buffered. The value written does not take effect until the LDOK bit is set and the next PWM load cycle begins. Reading PMFVAL5 reads the value in the buffer and not necessarily the value the PWM generator is currently using.

20.3.2.20 PMF Enable Control A Register (PMFENCA)

Address: $0020

	7	6	5	4	3	2	1	0
R	PWMENA	0	0	0	0	0	LDOKA	PWMRIEA
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 20-26. PMF Enable Control A Register (PMFENCA)

Read and write anytime.

Table 20-25. PMFENCA Field Descriptions

Field	Description
7 PWMENA	**PWM Generator A Enable** — When MTG is clear, this bit when set enables the PWM generators A, B and C and the PWM0–5 pins. When PWMENA is clear, PWM generators A, B and C are disabled, and the PWM0–5 pins are in their inactive states unless the corresponding OUTCTLx bits are set. When MTG is set, this bit when set enables the PWM generator A and the PWM0 and PWM1 pins. When PWMENA is clear, the PWM generator A is disabled and PWM0 and PWM1 pins are in their inactive states unless the OUTCTL0 and OUTCTL1 bits are set. 0 PWM generator A and PWM0-1 (2–5 if MTG = 0) pins disabled unless the respective OUTCTL bit is set. 1 PWM generator A and PWM0-1 (2–5 if MTG = 0) pins enabled.
1 LDOKA	**Load Okay A** — When MTG is clear, this bit allows loads of the PRSCA bits, the PMFMODA register and the PWMVAL0-5 registers into a set of buffers. The buffered prescaler A divisor, PWM counter modulus A value, and all PWM pulse widths take effect at the next PWM reload. When MTG is set, this bit allows loads of the PRSCA bits, the PMFMODA register and the PWMVAL0–1 registers into a set of buffers. The buffered prescaler divisor A, PWM counter modulus A value, PWM0–1 pulse widths take effect at the next PWM reload. Set LDOKA by reading it when it is logic zero and then writing a logic one to it. LDOKA is automatically cleared after the new values are loaded, or can be manually cleared before a reload by writing a logic zero to it. Reset clears LDOKA. 0 Do not load new modulus A, prescaler A, and PWM0–1 (2–5 if MTG = 0) values 1 Load prescaler A, modulus A, and PWM0–1 (2–5 if MTG = 0) values **Note:** Do not set PWMENA bit before setting the LDOKA bit and do not clear the LDOKA bit at the same time as setting the PWMENA bit.
0 PWMRIEA	**PWM Reload Interrupt Enable A** — This bit enables the PWMRFA flag to generate CPU interrupt requests. 0 PWMRFA CPU interrupt requests disabled 1 PWMRFA CPU interrupt requests enabled

20.3.2.21 PMF Frequency Control A Register (PMFFQCA)

Address: $0021

	7	6	5	4	3	2	1	0
R W	LDFQA				HALFA	PRSCA		PWMRFA
Reset	0	0	0	0	0	0	0	0

Figure 20-27. PMF Frequency Control A Register (PMFFQCA)

Read and write anytime.

Table 20-26. PMFFQCA Field Descriptions

Field	Description
7–4 LDFQA	**Load Frequency A** — This field selects the PWM load frequency according to Table 20-27. See Section 20.4.7.2, “Load Frequency” for more details. **Note:** The LDFQA field takes effect when the current load cycle is complete, regardless of the state of the load okay bit, LDOKA. Reading the LDFQA field reads the buffered value and not necessarily the value currently in effect.
3 HALFA	**Half Cycle Reload A** — This bit enables half-cycle reloads in center-aligned PWM mode. This bit has no effect on edge-aligned PWMs. 0 Half-cycle reloads disabled 1 Half-cycle reloads enabled
2–1 PRSCA	**Prescaler A** — This buffered field selects the PWM clock frequency illustrated in Table 20-28. **Note:** Reading the PRSCA field reads the buffered value and not necessarily the value currently in effect. The PRSCA field takes effect at the beginning of the next PWM cycle and only when the load okay bit, LDOKA, is set.
0 PWMRFA	**PWM Reload Flag A** — This flag is set at the beginning of every reload cycle regardless of the state of the LDOKA bit. Clear PWMRFA by reading PMFFQCA with PWMRFA set and then writing a logic one to the PWMRFA bit. If another reload occurs before the clearing sequence is complete, writing logic one to PWMRFA has no effect. 0 No new reload cycle since last PWMRFA clearing 1 New reload cycle since last PWMRFA clearing **Note:** Clearing PWMRFA satisfies pending PWMRFA CPU interrupt requests.

Table 20-27. PWM Reload Frequency A

LDFQA	PWM Reload Frequency	LDFQ[3:0]	PWM Reload Frequency
0000	Every PWM opportunity	1000	Every 9 PWM opportunities
0001	Every 2 PWM opportunities	1001	Every 10 PWM opportunities
0010	Every 3 PWM opportunities	1010	Every 11 PWM opportunities
0011	Every 4 PWM opportunities	1011	Every 12 PWM opportunities
0100	Every 5 PWM opportunities	1100	Every 13 PWM opportunities
0101	Every 6 PWM opportunities	1101	Every 14 PWM opportunities
0110	Every 7 PWM opportunities	1110	Every 15 PWM opportunities
0111	Every 8 PWM opportunities	1111	Every 16 PWM opportunities

Table 20-28. PWM Prescaler A

PRSCA	PWM Clock Frequency
00	f_{bus}
01	$f_{bus}/2$
10	$f_{bus}/4$
11	$f_{bus}/8$

20.3.2.22 PMF Counter A Register (PMFCNTA)

Address: $0022

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	PMFCNTA														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 20-28. PMF Counter A Register (PMFCNTA)

Read anytime and writes have no effect.

This register displays the state of the 15-bit PWM A counter.

20.3.2.23 PMF Counter Modulo A Register (PMFMODA)

Address: $0024

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	PMFMODA														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 20-29. PMF Counter Modulo A Register (PMFMODA)

Read and write anytime.

The 15-bit unsigned value written to this register is the PWM period in PWM clock periods. Do not write a modulus value of zero.

NOTE

The PWM counter modulo register is buffered. The value written does not take effect until the LDOKA bit is set and the next PWM load cycle begins. Reading PMFMODA reads the value in the buffer. It is not necessarily the value the PWM generator A is currently using.

20.3.2.24 PMF Deadtime A Register (PMFDTMA)

Address: $0026

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	PMFDTMA											
W																
Reset	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1

= Unimplemented or Reserved

Figure 20-30. PMF Deadtime A Register (PMFDTMA)

Read anytime. This register cannot be modified after the WP bit is set.

The 12-bit value written to this register is the number of PWM clock cycles in complementary channel operation. A reset sets the PWM deadtime register to a default value of 0x0FFF, selecting a deadtime of 4096-PWM clock cycles minus one bus clock cycle.

NOTE

Deadtime is affected by changes to the prescaler value. The deadtime duration is determined as follows: DT = P × PMFDTMA – 1, where DT is deadtime, P is the prescaler value, PMFDTMA is the programmed value of dead time. For example: if the prescaler is programmed for a divide-by-two and the PMFDTMA is set to five, then P = 2 and the deadtime value is equal to DT = 2 × 5 – 1 = 9 IPbus clock cycles. A special case exists when the P = 1, then DT = PMFDTMA.

20.3.2.25 PMF Enable Control B Register (PMFENCB)

Address: $0028

	7	6	5	4	3	2	1	0
R	PWMENB	0	0	0	0	0	LDOKB	PWMRIEB
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 20-31. PMF Enable Control B Register (PMFENCB)

Read anytime and write only if MTG is set.

Table 20-29. PMFENCB Field Descriptions

Field	Description
7 PWMENB	**PWM Generator B Enable** — If MTG is clear, this bit reads zero and cannot be written. If MTG is set, this bit when set enables the PWM generator B and the PWM2 and PWM3 pins. When PWMENB is clear, PWM generator B is disabled, and the PWM2 and PWM3 pins are in their inactive states unless the OUTCTL2 and OUTCTL3 bits are set. 0 PWM generator B and PWM2–3 pins disabled unless the respective OUTCTL bit is set. 1 PWM generator B and PWM2–3 pins enabled.

Table 20-29. PMFENCB Field Descriptions (continued)

Field	Description
1 LDOKB	**Load Okay B** — If MTG is clear, this bit reads zero and cannot be written. If MTG is set, this bit loads the PRSCB bits, the PMFMODB register and the PWMVAL2-3 registers into a set of buffers. The buffered prescaler divisor B, PWM counter modulus B value, PWM2–3 pulse widths take effect at the next PWM reload. Set LDOKB by reading it when it is logic zero and then writing a logic one to it. LDOKB is automatically cleared after the new values are loaded, or can be manually cleared before a reload by writing a logic zero to it. Reset clears LDOKB. 0 Do not load new modulus B, prescaler B, and PWM2–3 values. 1 Load prescaler B, modulus B, and PWM2–3 values. **Note:** Do not set PWMENB bit before setting the LDOKB bit and do not clear the LDOKB bit at the same time as setting the PWMENB bit.
0 PWMRIEB	**PWM Reload Interrupt Enable B** — If MTG is clear, this bit reads zero and cannot be written. If MTG is set, this bit enables the PWMRFB flag to generate CPU interrupt requests. 0 PWMRFB CPU interrupt requests disabled 1 PWMRFB CPU interrupt requests enabled

20.3.2.26 PMF Frequency Control B Register (PMFFQCB)

Address: $0029

	7	6	5	4	3	2	1	0
R W	LDFQB				HALFB	PRSCB		PWMRFB
Reset	0	0	0	0	0	0	0	0

Figure 20-32. PMF Frequency Control B Register (PMFFQCB)

Read anytime and write only if MTG is set.

Table 20-30. PMFFQCB Field Descriptions

Field	Description
7–4 LDFQB	**Load Frequency B** — This field selects the PWM load frequency according to Table 20-31. See Section 20.4.7.2, “Load Frequency” for more details. **Note:** The LDFQB field takes effect when the current load cycle is complete, regardless of the state of the load okay bit, LDOKB. Reading the LDFQB field reads the buffered value and not necessarily the value currently in effect.
3 HALFB	**Half Cycle Reload B** — This bit enables half-cycle reloads in center-aligned PWM mode. This bit has no effect on edge-aligned PWMs. 0 Half-cycle reloads disabled 1 Half-cycle reloads enabled
2–1 PRSCB	**Prescaler B** — This buffered field selects the PWM clock frequency illustrated in Table 20-32. **Note:** Reading the PRSCB field reads the buffered value and not necessarily the value currently in effect. The PRSCB field takes effect at the beginning of the next PWM cycle and only when the load okay bit, LDOKB, is set.

Table 20-30. PMFFQCB Field Descriptions (continued)

Field	Description
0 PWMRFB	**PWM Reload Flag B** — This flag is set at the beginning of every reload cycle regardless of the state of the LDOKB bit. Clear PWMRFB by reading PMFFQCB with PWMRFB set and then writing a logic one to the PWMRFB bit. If another reload occurs before the clearing sequence is complete, writing logic one to PWMRFB has no effect. 0 No new reload cycle since last PWMRFB clearing 1 New reload cycle since last PWMRFB clearing **Note:** Clearing PWMRFB satisfies pending PWMRFB CPU interrupt requests.

Table 20-31. PWM Reload Frequency B

LDFQB	PWM Reload Frequency	LDFQ[3:0]	PWM Reload Frequency
0000	Every PWM opportunity	1000	Every 9 PWM opportunities
0001	Every 2 PWM opportunities	1001	Every 10 PWM opportunities
0010	Every 3 PWM opportunities	1010	Every 11 PWM opportunities
0011	Every 4 PWM opportunities	1011	Every 12 PWM opportunities
0100	Every 5 PWM opportunities	1100	Every 13 PWM opportunities
0101	Every 6 PWM opportunities	1101	Every 14 PWM opportunities
0110	Every 7 PWM opportunities	1110	Every 15 PWM opportunities
0111	Every 8 PWM opportunities	1111	Every 16 PWM opportunities

Table 20-32. PWM Prescaler B

PRSCB	PWM Clock Frequency
00	f_{bus}
01	$f_{bus}/2$
10	$f_{bus}/4$
11	$f_{bus}/8$

20.3.2.27 PMF Counter B Register (PMFCNTB)

Address: $002A

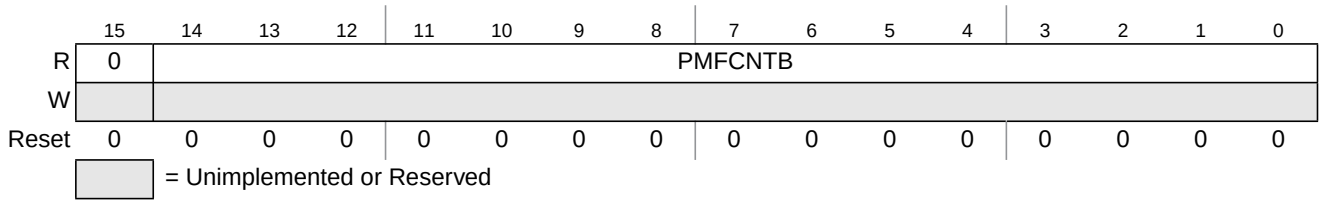


Figure 20-33. PMF Counter B Register (PMFCNTB)

Read anytime and writes have no effect.

This register displays the state of the 15-bit PWM B counter.

20.3.2.28 PMF Counter Modulo B Register (PMFMODB)

Address: $002C

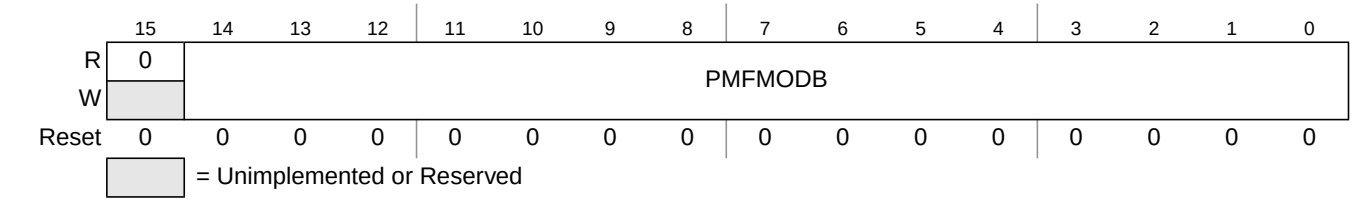


Figure 20-34. PMF Counter Modulo B Register (PMFMODB)

Read anytime and write only if MTG is set.

The 15-bit unsigned value written to this register is the PWM period in PWM clock periods. Do not write a modulus value of zero.

NOTE

The PWM counter modulo register is buffered. The value written does not take effect until the LDOKB bit is set and the next PWM load cycle begins. Reading PMFMODB reads the value in the buffer. It is not necessarily the value the PWM generator B is currently using.

20.3.2.29 PMF Deadtime B Register (PMFDTMB)

Address: $002E

Figure 20-35. PMF Deadtime B Register (PMFDTMB)

Read anytime and write only if MTG is set. This register cannot be modified after the WP bit is set.

The 12-bit value written to this register is the number of PWM clock cycles in complementary channel operation. A reset sets the PWM deadtime register to a default value of 0x0FFF, selecting a deadtime of 4096-PWM clock cycles minus one bus clock cycle.

NOTE

Deadtime is affected by changes to the prescaler value. The deadtime duration is determined as follows: DT = P × PMFDTMB – 1, where DT is deadtime, P is the prescaler value, PMFDTMB is the programmed value of dead time. For example: if the prescaler is programmed for a divide-by-two and the PMFDTMB is set to five, then P = 2 and the deadtime value is equal to DT = 2 × 5 – 1 = 9 IPbus clock cycles. A special case exists when the P = 1, then DT = PMFDTMB.

20.3.2.30 PMF Enable Control C Register (PMFENCC)

Address: $0030

	7	6	5	4	3	2	1	0
R	PWMENC	0	0	0	0	0	LDOKC	PWMRIEC
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 20-36. PMF Enable Control C Register (PMFENCC)

Read anytime and write only if MTG is set.

Table 20-33. PMFENCC Field Descriptions

Field	Description
7 PWMENC	**PWM Generator C Enable** — If MTG is clear, this bit reads zero and cannot be written. If MTG is set, this bit when set enables the PWM generator C and the PWM4 and PWM5 pins. When PWMENC is clear, PWM generator C is disabled, and the PWM4 and PWM5 pins are in their inactive states unless the OUTCTL4 and OUTCTL5 bits are set. 0 PWM generator C and PWM4–5 pins disabled unless the respective OUTCTL bit is set. 1 PWM generator C and PWM4–5 pins enabled.
1 LDOCKC	**Load Okay C** — If MTG is clear, this bit reads zero and can not be written. If MTG is set, this bit loads the PRSCC bits, the PMFMODC register and the PWMVAL4–5 registers into a set of buffers. The buffered prescaler divisor C, PWM counter modulus C value, PWM4–5 pulse widths take effect at the next PWM reload. Set LDOKC by reading it when it is logic zero and then writing a logic one to it. LDOKC is automatically cleared after the new values are loaded, or can be manually cleared before a reload by writing a logic zero to it. Reset clears LDOKC. 0 Do not load new modulus C, prescaler C, and PWM4–5 values. 1 Load prescaler C, modulus C, and PWM4–5 values. **Note:** Do not set PWMENC bit before setting the LDOKC bit and do not clear the LDOKC bit at the same time as setting the PWMENC bit.
0 PWMRIEC	**PWM Reload Interrupt Enable C** — If MTG is clear, this bit reads zero and cannot be written. If MTG is set, this bit enables the PWMRFC flag to generate CPU interrupt requests. 0 PWMRFC CPU interrupt requests disabled 1 PWMRFC CPU interrupt requests enabled

20.3.2.31 PMF Frequency Control C Register (PMFFQCC)

Address: $0031

	7	6	5	4	3	2	1	0
R	LDFQC				HALFC	PRSCC		PWMRFC
W								
Reset	0	0	0	0	0	0	0	0

Figure 20-37. PMF Frequency Control C Register (PMFFQCC)

Read anytime and write only if MTG is set.

Table 20-34. PMFFQCC Field Descriptions

Field	Description
7–4 LDFQC	**Load Frequency C** — This field selects the PWM load frequency according to Table 20-35. See Section 20.4.7.2, “Load Frequency” for more details. **Note:** The LDFQC field takes effect when the current load cycle is complete, regardless of the state of the load okay bit, LDOKC. Reading the LDFQC field reads the buffered value and not necessarily the value currently in effect.
3 HALFC	**Half Cycle Reload C** — This bit enables half-cycle reloads in center-aligned PWM mode. This bit has no effect on edge-aligned PWMs. 0 Half-cycle reloads disabled 1 Half-cycle reloads enabled
2–1 PRSCC	**Prescaler C** — This buffered field selects the PWM clock frequency illustrated in Table 20-36. **Note:** Reading the PRSCC field reads the buffered value and not necessarily the value currently in effect. The PRSCC field takes effect at the beginning of the next PWM cycle and only when the load okay bit, LDOKC, is set.
0 PWMRFC	**PWM Reload Flag C** — This flag is set at the beginning of every reload cycle regardless of the state of the LDOKC bit. Clear PWMRFC by reading PMFFQCC with PWMRFC set and then writing a logic one to the PWMRFC bit. If another reload occurs before the clearing sequence is complete, writing logic one to PWMRFC has no effect. 0 No new reload cycle since last PWMRFC clearing 1 New reload cycle since last PWMRFC clearing **Note:** Clearing PWMRFC satisfies pending PWMRFC CPU interrupt requests.

Table 20-35. PWM Reload Frequency C

LDFQC	PWM Reload Frequency	LDFQ[3:0]	PWM Reload Frequency
0000	Every PWM opportunity	1000	Every 9 PWM opportunities
0001	Every 2 PWM opportunities	1001	Every 10 PWM opportunities
0010	Every 3 PWM opportunities	1010	Every 11 PWM opportunities
0011	Every 4 PWM opportunities	1011	Every 12 PWM opportunities
0100	Every 5 PWM opportunities	1100	Every 13 PWM opportunities
0101	Every 6 PWM opportunities	1101	Every 14 PWM opportunities
0110	Every 7 PWM opportunities	1110	Every 15 PWM opportunities
0111	Every 8 PWM opportunities	1111	Every 16 PWM opportunities

Table 20-36. PWM Prescaler C

PRSCC	PWM Clock Frequency
00	f_{bus}
01	$f_{bus}/2$
10	$f_{bus}/4$
11	$f_{bus}/8$

20.3.2.32 PMF Counter C Register (PMFCNTC)

Address: $0032

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	PMFCNTC														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 20-38. PMF Counter C Register (PMFCNTC)

Read anytime and writes have no effect.

This register displays the state of the 15-bit PWM C counter.

20.3.2.33 PMF Counter Modulo C Register (PMFMODC)

Address: $0034

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	PMFMODC														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 20-39. PMF Counter Modulo C Register (PMFMODC)

Read anytime and write only if MTG is set.

The 15-bit unsigned value written to this register is the PWM period in PWM clock periods. Do not write a modulus value of zero.

NOTE

The PWM counter modulo register is buffered. The value written does not take effect until the LDOKC bit is set and the next PWM load cycle begins. Reading PMFMODC reads the value in the buffer. It is not necessarily the value the PWM generator A is currently using.

20.3.2.34 PMF Deadtime C Register (PMFDTMC)

Address: $0036

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	PMFDTMC											
W																
Reset	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1

= Unimplemented or Reserved

Figure 20-40. PMF Deadtime C Register (PMFDTMC)

Read anytime and write only if MTG is set. This register cannot be modified after the WP bit is set.

The 12-bit value written to this register is the number of PWM clock cycles in complementary channel operation. A reset sets the PWM deadtime register to a default value of 0x0FFF, selecting a deadtime of 4096-PWM clock cycles minus one bus clock cycle.

NOTE

Deadtime is affected by changes to the prescaler value. The deadtime duration is determined as follows: DT = P × PMFDTMC – 1, where DT is deadtime, P is the prescaler value, PMFDTMC is the programmed value of dead time. For example: if the prescaler is programmed for a divide-by-two and the PMFDTMC is set to five, then P = 2 and the deadtime value is equal to DT = 2 × 5 – 1 = 9 IPbus clock cycles. A special case exists when the P = 1, then DT = PMFDTMC.

20.4 Functional Description

20.4.1 Block Diagram

A block diagram of the PMF is shown in Figure 20-1**.** The MTG bit allows the use of multiple PWM generators (A, B, and C) or just a single generator (A). PWM0 and PWM1 constitute Pair A, PWM2 and PWM3 constitute Pair B, and PWM4 and PWM5 constitute Pair C.

20.4.2 Prescaler

To permit lower PWM frequencies, the prescaler produces the PWM clock frequency by dividing the bus clock frequency by one, two, four, and eight. Each PWM generator has its own prescaler divisor. Each prescaler is buffered and will not be used by its PWM generator until the corresponding Load OK bit is set and a new PWM reload cycle begins.

20.4.3 PWM Generator

Each PWM generator contains a 15-bit up/down PWM counter producing output signals with software-selectables:

- Alignment — The logic state of each pair EDGE bit determines whether the PWM pair outputs are edge-aligned or center-aligned
- Period — The value written to each pair PWM counter modulo register is used to determine the PWM pair period. The period can also be varied by using the prescaler
- With edge-aligned output, the modulus is the period of the PWM output in clock cycles
- With center-aligned output, the modulus is one-half of the PWM output period in clock cycles
- Pulse width — The number written to the PWM value register determines the pulse width duty cycle of the PWM output in clock cycles
  - — With center-aligned output, the pulse width is twice the value written to the PWM value register
  - — With edge-aligned output, the pulse width is the value written to the PWM value register

20.4.3.1 Alignment

Each edge-align bit, EDGEx, selects either center-aligned or edge-aligned PWM generator outputs.

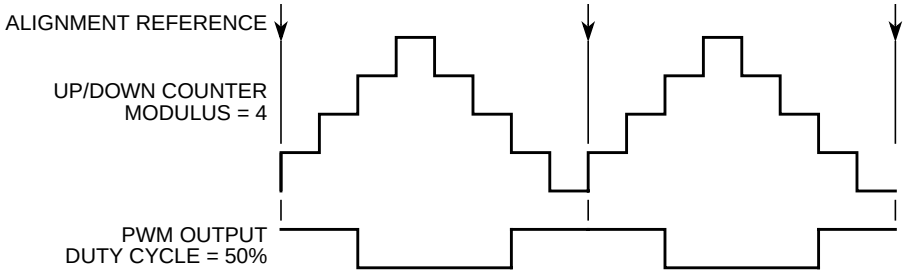


Figure 20-41. Center-Aligned PWM Output

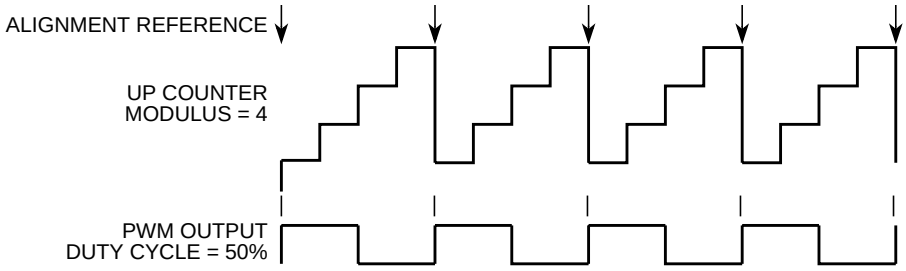


Figure 20-42. Edge-Aligned PWM Output

NOTE

Because of the equals-comparator architecture of this PMF, the modulus equals zero case is considered illegal. Therefore, the modulus register does not return to zero, and a modulus value of zero will result in waveforms inconsistent with the other modulus waveforms. If a modulus of zero is loaded, the counter will continually count down from $7FFF. This operation will not be tested or guaranteed. Consider it illegal. However, the dead-time constraints and fault conditions will still be guaranteed.

20.4.3.2 Period

A PWM period is determined by the value written to the PWM counter modulo register.

The PWM counter is an up/down counter in a center-aligned operation. In this mode the PWM highest output resolution is two bus clock cycles.

$$\text{PWM period} = (\text{PWM modulus}) \times (\text{PWM clock period}) \times 2$$

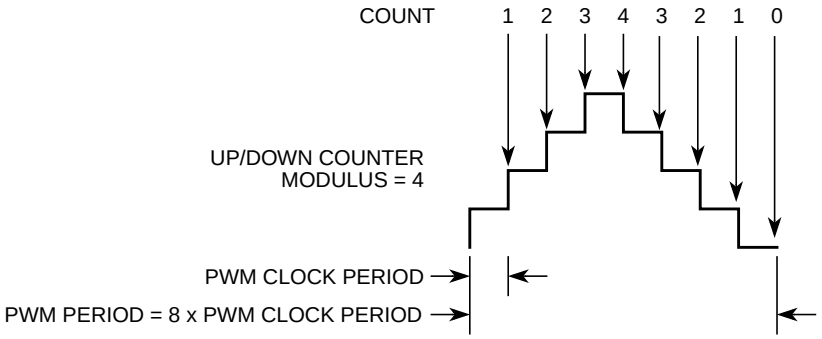


Figure 20-43. Center-Aligned PWM Period

In an edge-aligned operation, the PWM counter is an up counter. The PWM output resolution is one bus clock cycle.

PWM period = PWM modulus × PWM clock period

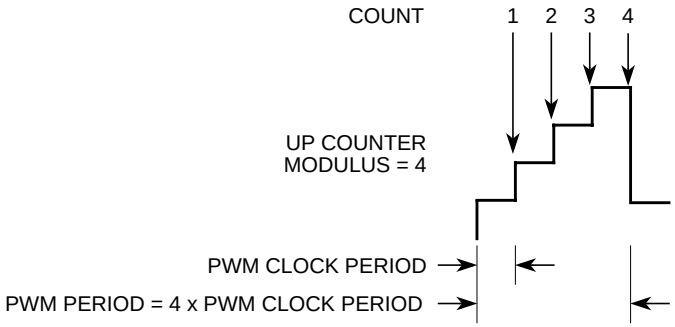


Figure 20-44. Edge-Aligned PWM Period

20.4.3.3 Duty Cycle

The signed 16-bit number written to the PMF value registers is the pulse width in PWM clock periods of the PWM generator output.

$$\text{Duty cycle} = \frac{\text{PMFVAL}}{\text{MODULUS}} \times 100$$

NOTE

A PWM value less than or equal to zero deactivates the PWM output for the entire PWM period. A PWM value greater than or equal to the modulus activates the PWM output for the entire PWM period.

Table 20-37. PWM Value and Underflow Conditions

PMFVALx	Condition	PWM Value Used
$0000–$7FFF	Normal	Value in registers
$8000–$FFFF	Underflow	$0000

Center-aligned operation is illustrated in Figure 20-45.

PWM pulse width = (PWM value) × (PWM clock period) × 2

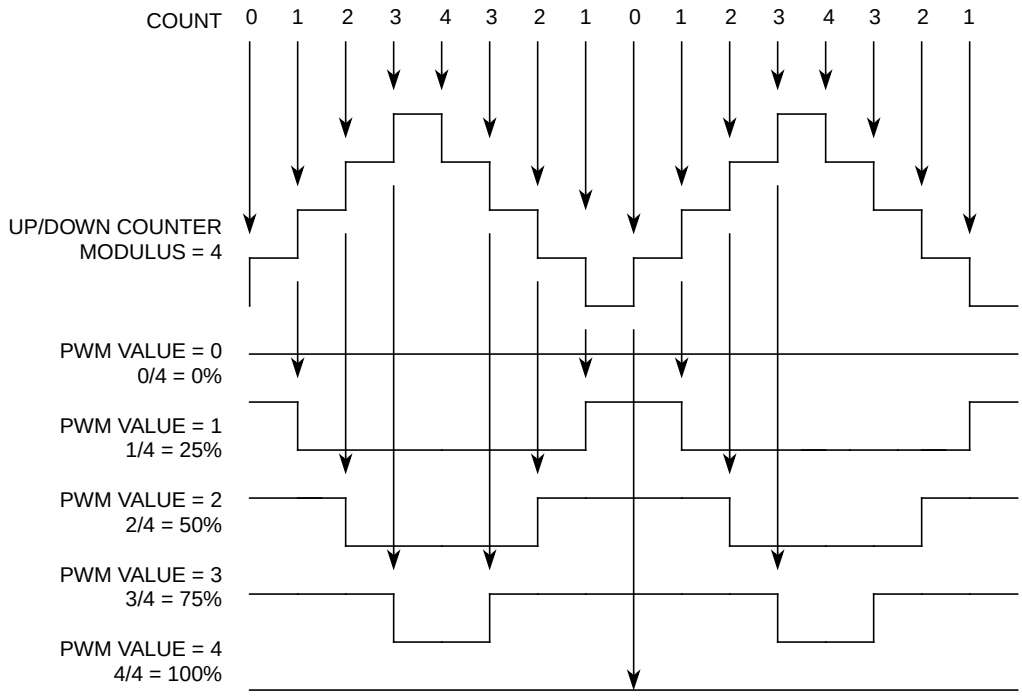


Figure 20-45. Center-Aligned PWM Pulse Width

Edge-aligned operation is illustrated in Figure 20-46.

PWM pulse width = (PWM value) × (PWM clock period)

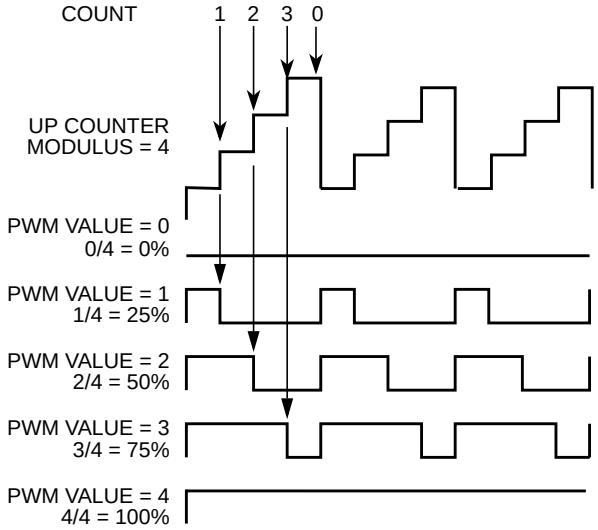


Figure 20-46. Edge-Aligned PWM Pulse Width

20.4.4 Independent or Complementary Channel Operation

Writing a logic one to a INDEPx bit configures a pair of the PWM outputs as two independent PWM channels. Each PWM output has its own PWM value register operating independently of the other channels in independent channel operation.

Writing a logic zero to a INDEPx bit configures the PWM output as a pair of complementary channels. The PWM pins are paired as shown in Figure 20-47 in complementary channel operation.

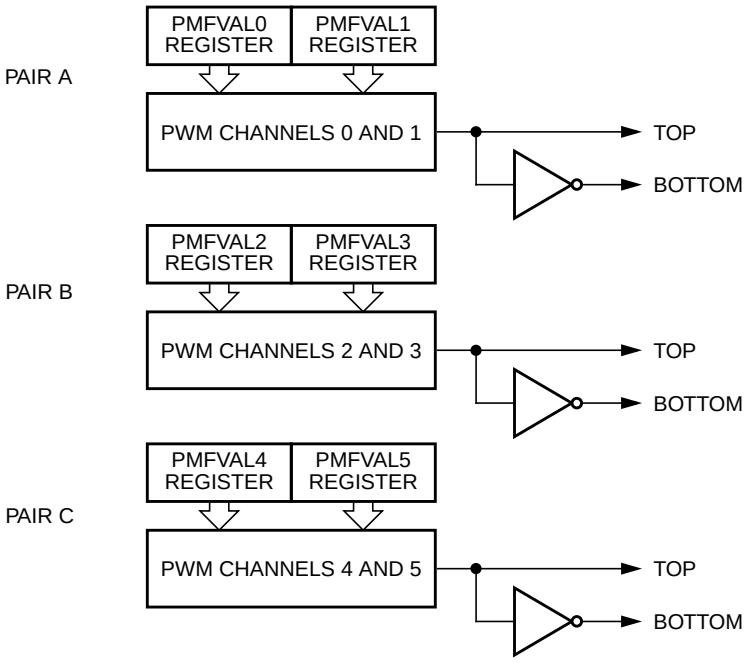


Figure 20-47. Complementary Channel Pairs

The complementary channel operation is for driving top and bottom transistors in a motor drive circuit, such as the one in Figure 20-48.

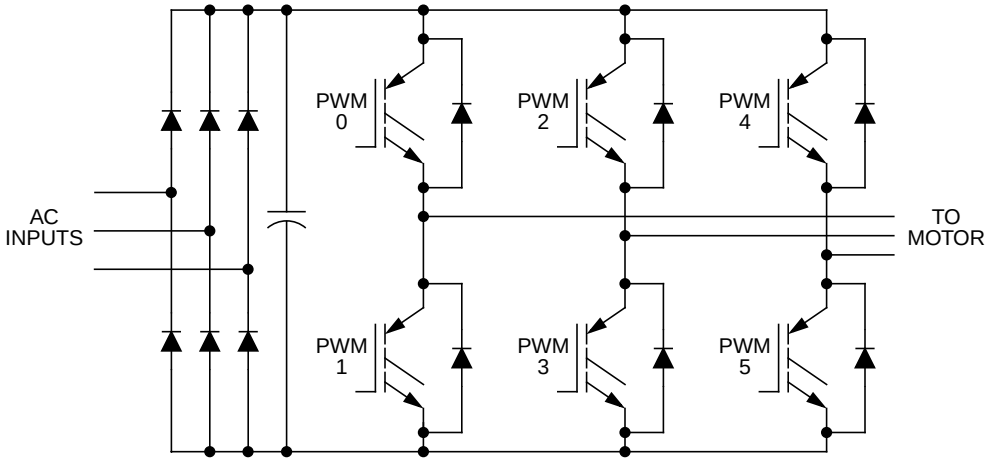


Figure 20-48. Typical 3 Phase AC Motor Drive

In complementary channel operation, there are three additional features:

- Deadtime insertion
- Separate top and bottom pulse width correction for distortions are caused by deadtime inserted and the motor drive characteristics
- Separate top and bottom output polarity control

- Swap functionality

20.4.5 Deadtime Generators

While in the complementary mode, each PWM pair can be used to drive top/bottom transistors, as shown in Figure 20-49. Ideally, the PWM pairs are an inversion of each other. When the top PWM channel is active, the bottom PWM channel is inactive, and vice versa.

NOTE

To avoid a short-circuit on the DC bus and endangering the transistor, there must be no overlap of conducting intervals between top and bottom transistor. But the transistor's characteristics make its switching-off time longer than switching-on time. To avoid the conducting overlap of top and bottom transistors, deadtime needs to be inserted in the switching period.

Deadtime generators automatically insert software-selectable activation delays into each pair of PWM outputs. The deadtime register (PMFDTMx) specifies the number of PWM clock cycles to use for deadtime delay. Every time the deadtime generator input changes state, deadtime is inserted. Deadtime forces both PWM outputs in the pair to the inactive state.

A method of correcting this, adding to or subtracting from the PWM value used, is discussed next.

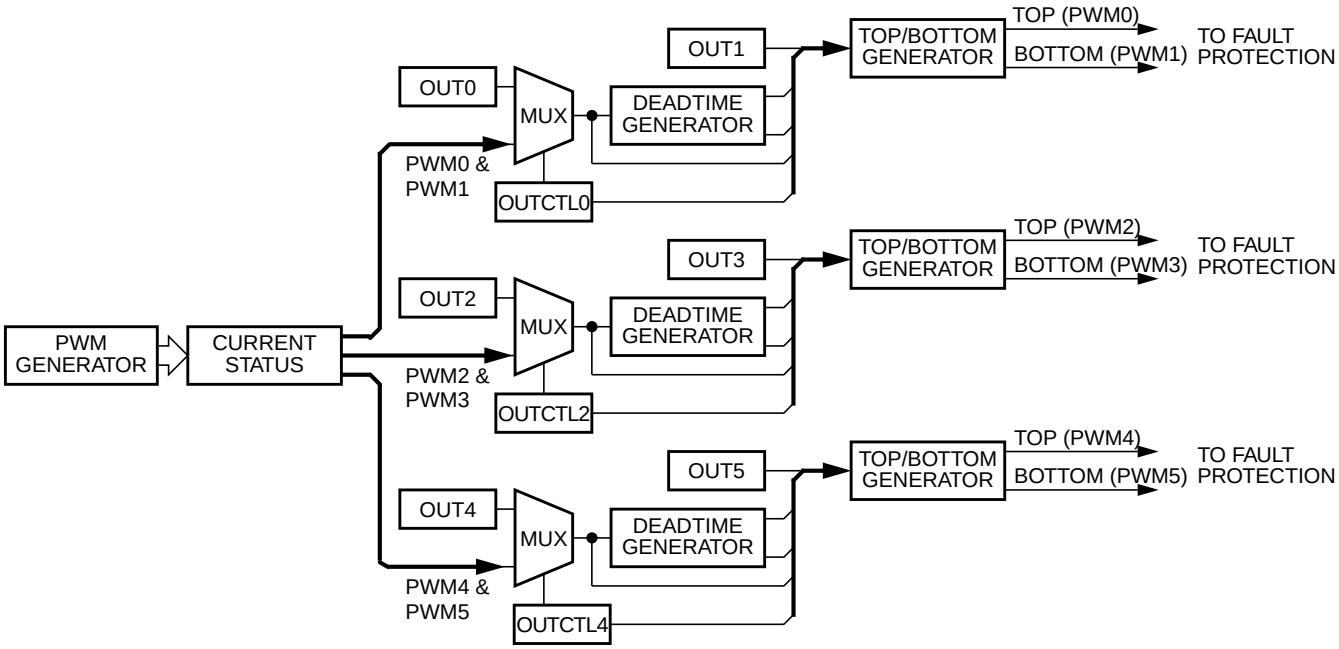


Figure 20-49. Deadtime Generators

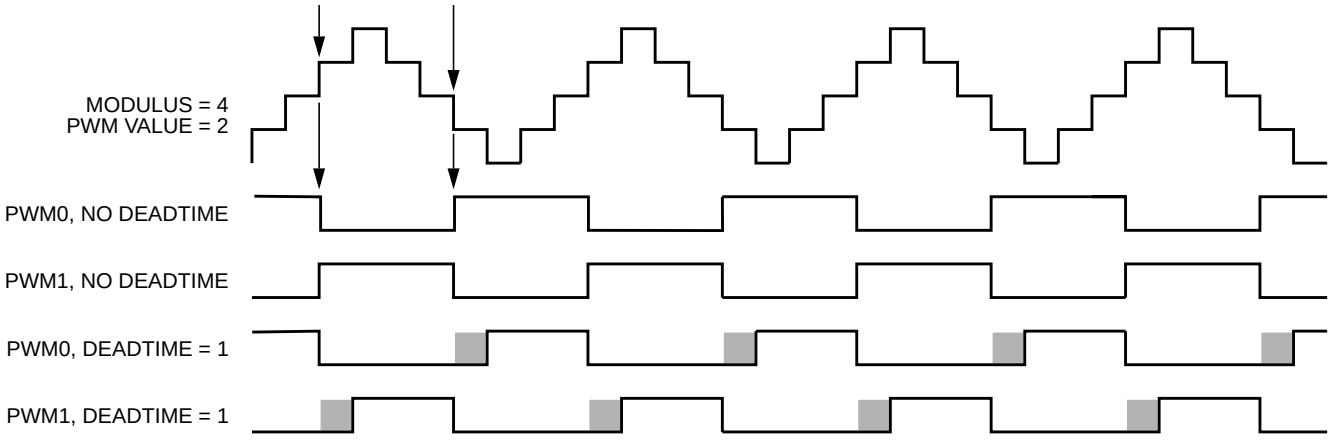


Figure 20-50. Deadtime Insertion, Center Alignment

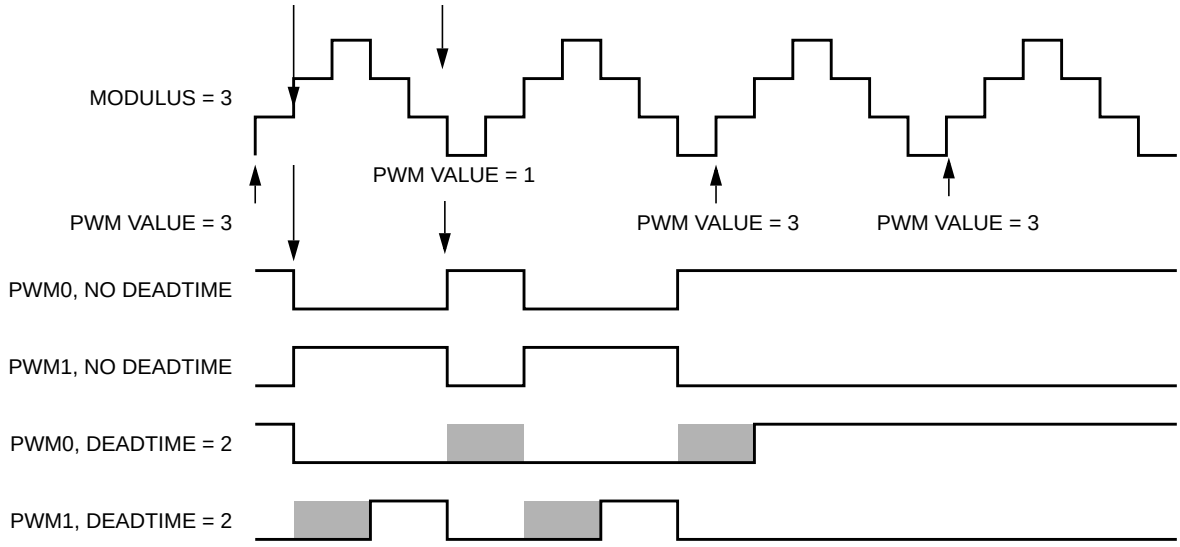


Figure 20-51. Deadtime at Duty Cycle Boundaries

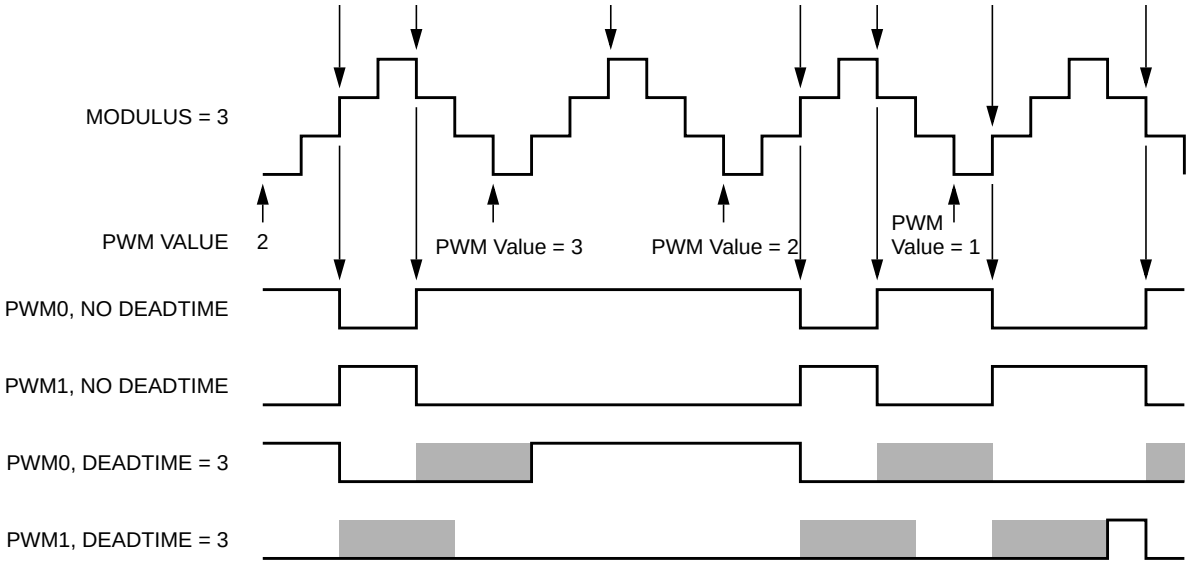


Figure 20-52. Deadtime and Small Pulse Widths

NOTE

The waveform at the pad is delayed by two bus clock cycles for deadtime insertion.

20.4.5.1 Top/Bottom Correction

In complementary mode, either the top or the bottom transistor controls the output voltage. However, deadtime has to be inserted to avoid overlap of conducting interval between the top and bottom transistor. Both transistors in complementary mode are off during deadtime, allowing the output voltage to be determined by the current status of load and introduce distortion in the output voltage. See Figure 20-53. On AC induction motors running open-loop, the distortion typically manifests itself as poor low-speed performance, such as torque ripple and rough operation.

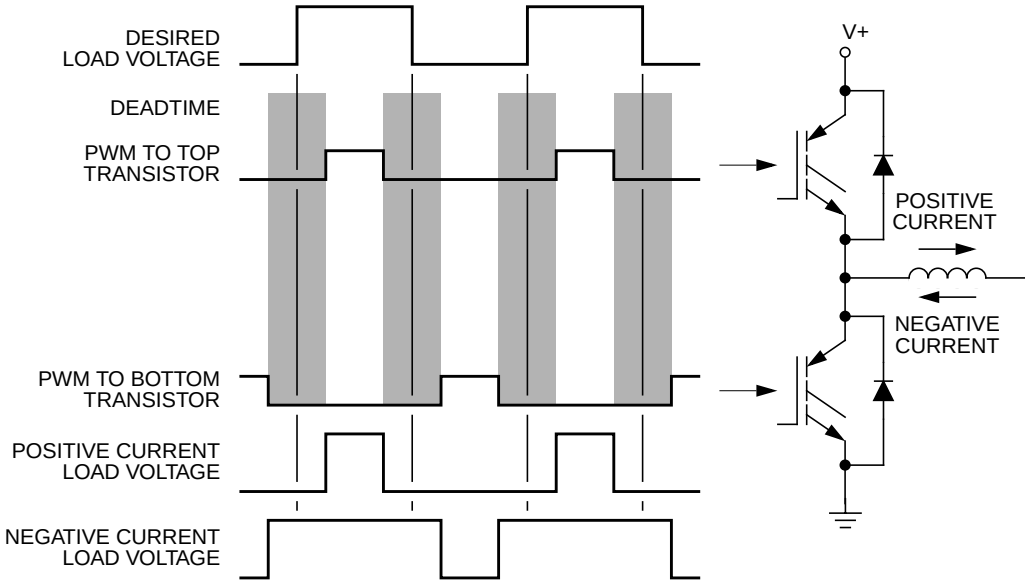


Figure 20-53. Deadtime Distortion

During deadtime, load inductance distorts output voltage by keeping current flowing through the diodes. This deadtime current flow creates a load voltage that varies with current direction. With a positive current flow, the load voltage during deadtime is equal to the bottom supply, putting the top transistor in control. With a negative current flow, the load voltage during deadtime is equal to the top supply putting the bottom transistor in control.

Remembering that the original PWM pulse widths were shortened by deadtime insertion, the averaged sinusoidal output will be less than desired value. However, when deadtime is inserted, it creates a distortion in motor current waveform. This distortion is aggravated by dissimilar turn-on and turn-off delays of each of the transistors. By giving the PWM module information on which transistor is controlling at a given time this distortion can be corrected.

For a typical circuit in complementary channel operation, only one of the transistors will be effective in controlling the output voltage at any given time. This depends on the direction of the motor current for that pair. See Figure 20-53. To correct distortion one of two different factors must be added to the desired PWM value, depending on whether the top or bottom transistor is controlling the output voltage. Therefore, the

software is responsible for calculating both compensated PWM values prior to placing them in an odd-numbered/even numbered PWM register pair. Either the odd or the even PMFVAL register controls the pulse width at any given time. For a given PWM pair, whether the odd or even PMFVAL register is active depends on either:

- The state of the current status pin, $\overline{ISx}$, for that driver
- The state of the odd/even correction bit, IPOLx, for that driver
- To correct deadtime distortion, software can decrease or increase the value in the appropriate PMFVAL register.
- In edge-aligned operation, decreasing or increasing the PWM value by a correction value equal to the deadtime typically compensates for deadtime distortion.
- In center-aligned operation, decreasing or increasing the PWM value by a correction value equal to one-half the deadtime typically compensates for deadtime distortion.
- In the complementary channel operation, ISENS selects one of three correction methods:
- Manual correction
- Automatic current status correction during deadtime
- Automatic current status correction when the PWM counter value equals the value in the PWM counter modulus registers

Table 20-38. Correction Method Selection

ISENS	Correction Method
00	No correction(1)
01	Manual correction
10	Current status sample correction on pins $\overline{IS0}$, $\overline{IS1}$, and $\overline{IS2}$ during deadtime(2)
11	Current status sample on pins $\overline{IS0}$, $\overline{IS1}$, and $\overline{IS2}$(3) At the half cycle in center-aligned operation At the end of the cycle in edge-aligned operation

1. The current status pins can be used as general purpose input/output ports.
2. The polarity of the $\overline{ISx}$ pin is latched when both the top and bottom PWMs are off. At the 0% and 100% duty cycle boundaries, there is no deadtime, so no new current value is sensed.
3. Current is sensed even with 0% or 100% duty cycle.

NOTE

Assume the user will provide current status sensing circuitry causing the voltage at the corresponding input pin to be low for positive current and high for negative current. In addition, it assumes the top PWMs are PWM 0, 2, and 4 while the bottom PWMS are PWM 1, 3, and 5.

20.4.5.2 Manual Correction

The IPOLx bits select either the odd or the even PWM value registers to use in the next PWM cycle.

Table 20-39. Top/Bottom Manual Correction

Bit	Logic state	Output Control
IPOLA	0	PMFVAL0 controls PWM0/PWM1 pair
	1	PMFVAL1 controls PWM0/PWM1 pair
IPOLB	0	PMFVAL2 controls PWM2/PWM3 pair
	1	PMFVAL3 controls PWM2/PWM3 pair
IPOLC	0	PMFVAL4 controls PWM4/PWM5 pair
	1	PMFVAL5 controls PWM4/PWM5 pair

NOTE

IPOLx bits are buffered so only one PWM register is used per PWM cycle. If an IPOLx bit changes during a PWM period, the new value does not take effect until the next PWM period.

IPOLx bits take effect at the end of each PWM cycle regardless of the state of the load okay bit, LDOK.

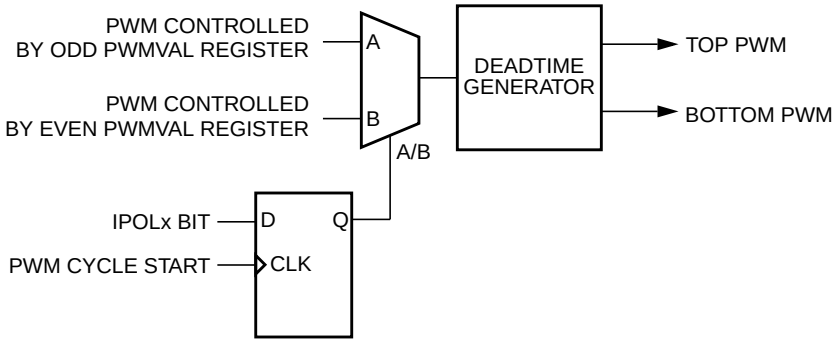


Figure 20-54. Internal Correction Logic when ISENS = 01

To detect the current status, the voltage on each $\overline{ISx}$ pin is sampled twice in a PWM period, at the end of each deadtime. The value is stored in the DTx bits in the PMF Deadtime Sample register (PMFDTMS). The DTx bits are a timing marker especially indicating when to toggle between PWM value registers. Software can then set the IPOLx bit to toggle PMFVAL registers according to DTx values.

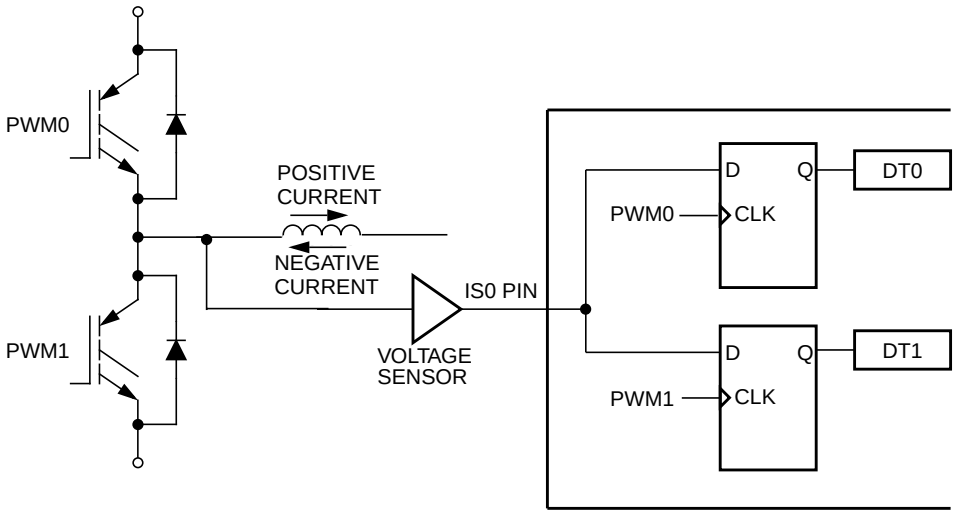


Figure 20-55. Current Status Sense Scheme for Deadtime Correction

If both D flip-flops latch low, DT0 = 0, DT1 = 0, during deadtime periods if current is large and flowing out of the complementary circuit. See Figure 20-55. If both D flip-flops latch the high, DT0 = 1, DT1 = 1, during deadtime periods if current is also large and flowing into the complementary circuit. However, under low-current, the output voltage of the complementary circuit during deadtime is somewhere between the high and low levels. The current cannot free-wheel throughout the opposition anti-body diode, regardless of polarity, giving additional distortion when the current crosses zero. Sampled results will be DT0 = 0 and DT1 = 1. Thus, the best time to change one PWM value register to another is just before the current zero crossing.

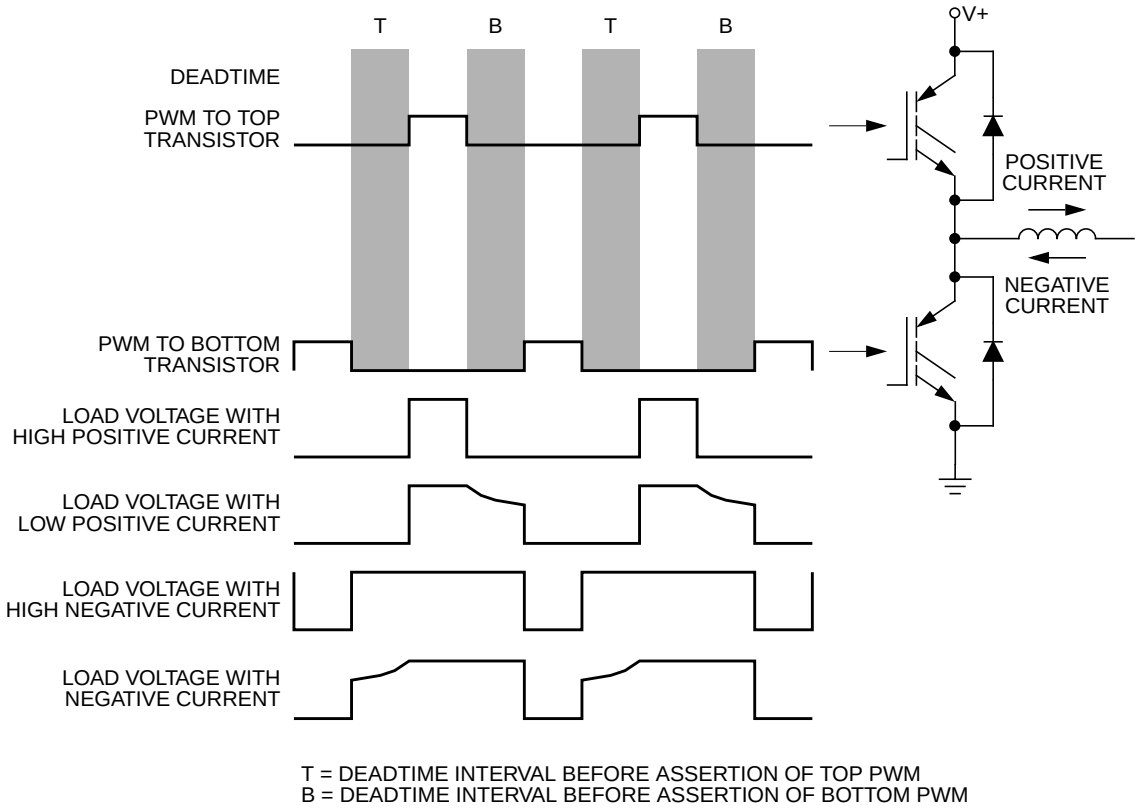


Figure 20-56. Output Voltage Waveforms

20.4.5.3 Current-Sensing Correction

A current sense pin, $\overline{ISx}$, for a PWM pair selects either the odd or the even PWM value registers to use in the next PWM cycle. The selection is based on user-provided current sense circuitry driving the $\overline{ISx}$ pin high for negative current and low for positive current.

Table 20-40. Top/Bottom Current Sense Correction

Pin	Logic State	Output Control
IS0	0	PMFVAL0 controls PWM0/PWM1 pair
	1	PMFVAL1 controls PWM0/PWM1 pair
IS1	0	PMFVAL2 controls PWM2/PWM3 pair
	1	PMFVAL3 controls PWM2/PWM3 pair
IS2	0	PMFVAL4 controls PWM4/PWM5 pair
	1	PMFVAL5 controls PWM4/PWM5 pair

Previously shown, the current direction can be determined by the output voltage during deadtime. Thus, a simple external voltage sensor can be used when current status is completed during deadtime, ISENS = 10. Deadtime does not exists at the 100 percent and zero percent duty cycle boundaries. Therefore, the second automatic mode must be used for correction, ISENS = 11, where current status is sampled at the half cycle in center-aligned operation and at the end of cycle in edge-aligned operation. Using this mode requires external circuitry. It actually senses current direction.

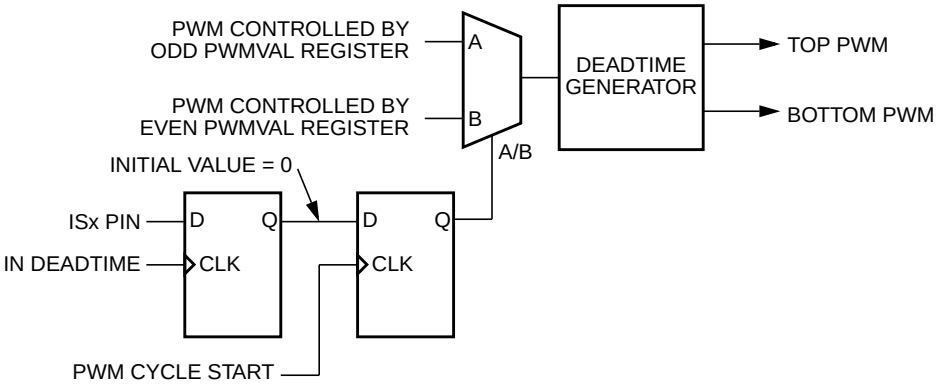


Figure 20-57. Internal Correction Logic when ISENS = 10

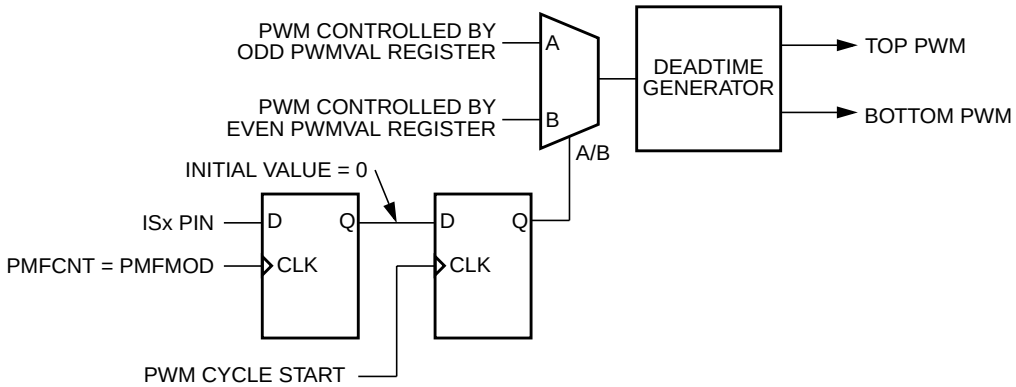


Figure 20-58. Internal Correction Logic when ISENS = 11

NOTE

Values latched on the $\overline{\text{ISx}}$ pins are buffered so only one PWM register is used per PWM cycle. If a current status changes during a PWM period, the new value does not take effect until the next PWM period.

When initially enabled by setting the PWMEN bit, no current status has previously been sampled. PWM value registers one, three, and five initially control the three PWM pairs when configured for current status correction.

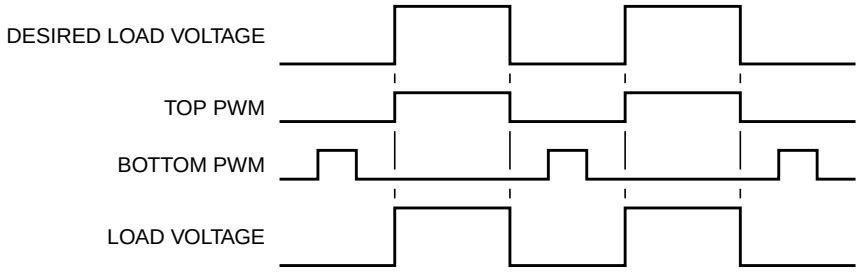


Figure 20-59. Correction with Positive Current

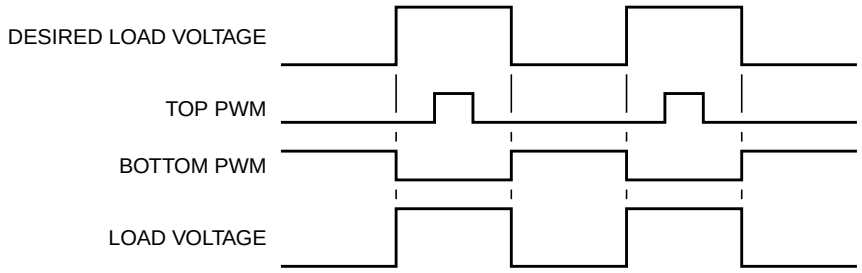


Figure 20-60. Correction with Negative Current

20.4.5.4 Output Polarity

Output polarity of the PWMs is determined by two options: TOPNEG and BOTNEG. The top polarity option, TOPNEG, controls the polarity of PWM0, PWM2 and PWM4. The bottom polarity option, BOTNEG, controls the polarity of PWM1, PWM3 and PWM5. *Positive* polarity means when the PWM is active its output is high. Conversely, *negative* polarity means when the PWM is active its output is low.

The TOPNEG and BOTNEG are in the configure register. TOPNEG is the output of PWM0, PWM2 and PWM4. They are active low. If TOPNEG is set, PWM0, PWM2, and PWM4 outputs become *active-low*. When BOTNEG is set, PWM1, PWM3, and PWM5 outputs are *active-low*. When these bits are clear, their respective PWM pins are *active-high*. See Figure 20-59 and Figure 20-60.

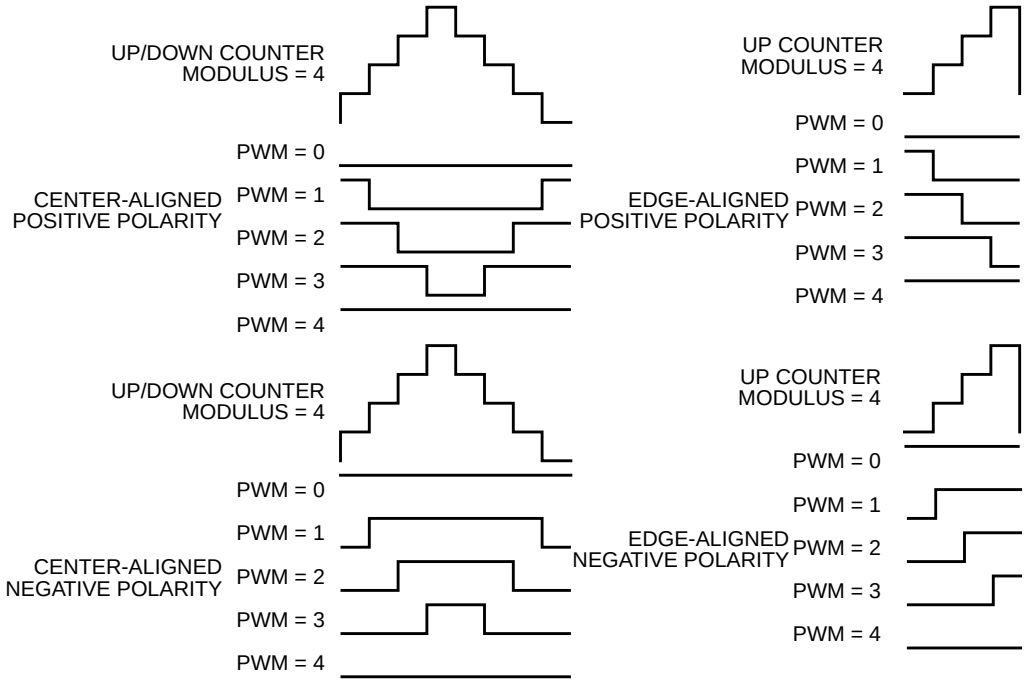


Figure 20-61. PWM Polarity

20.4.6 Software Output Control

Setting output control enable bit, OUTCTLx, enables software to drive the PWM outputs rather than the PWM generator. In an independent mode, with OUTCTLx = 1, the output bit OUTx, controls the PWMx

channel. In a complementary channel operation the even OUTCTL bit is used to enable software output control for the pair. But the OUTCTL bits must be switched in pairs for proper operation. The OUTCTLx and OUTx bits are in the PWM output control register.

NOTE

During software output control, TOPNEG and BOTNEG still control output polarity. It will take up to 3 clock cycles to see the effect of output control on the PWM output pins.

In independent PWM operation, setting or clearing the OUTx bit activates or deactivates the PWMx output.

In complementary channel operation, the even-numbered OUTx bits replace the PWM generator outputs as inputs to the deadtime generators. Complementary channel pairs still cannot be active simultaneously, and the deadtime generators continue to insert deadtime in both channels of that pair, whenever an even OUTx bit toggles. Even OUTx bits control the top PWM signals while the odd OUTx bits control the bottom PWM signals with respect to the even OUTx bits. Setting the odd OUTx bit makes its corresponding PWMx the complement of its even pair, while clearing the odd OUTx bit deactivates the odd PWMx.

Setting the OUTCTLx bits do not disable the PWM generators and current status sensing circuitry. They continue to run, but no longer control the output pins. When the OUTCTLx bits are cleared, the outputs of the PWM generator become the inputs to the deadtime generators at the beginning of the next PWM cycle. Software can drive the PWM outputs even when PWM enable bit (PWMEN) is set to zero.

NOTE

Avoid an unexpected deadtime insertion by clearing the OUTx bits before setting and after clearing the OUTCTLx bits.

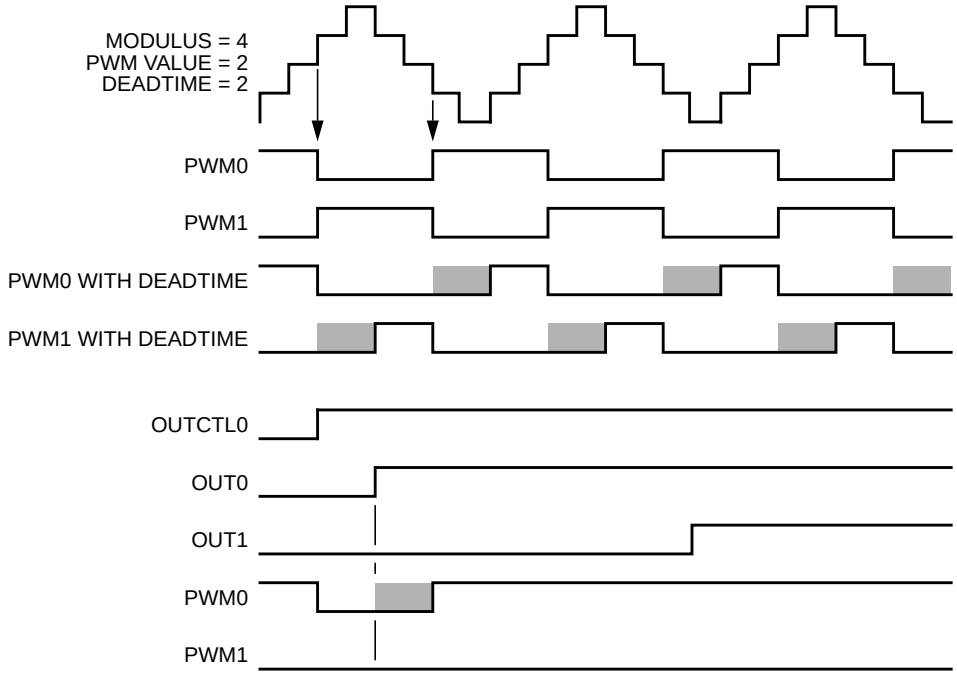


Figure 20-62. Setting OUT0 with OUTCTL Set in Complementary Mode

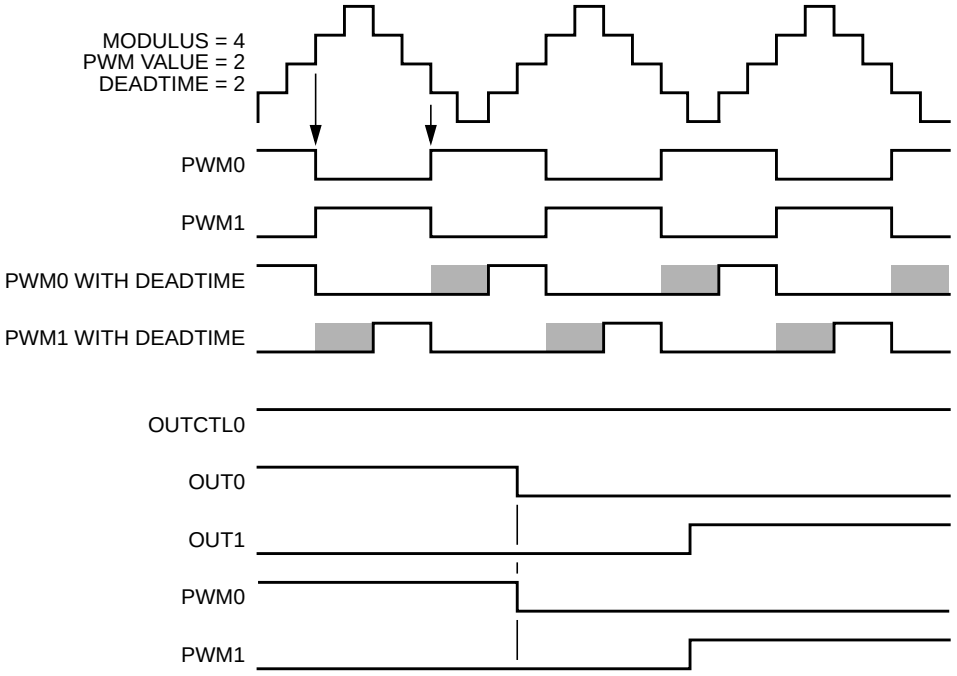


Figure 20-63. Clearing OUT0 with OUTCTL Set In Complementary Mode

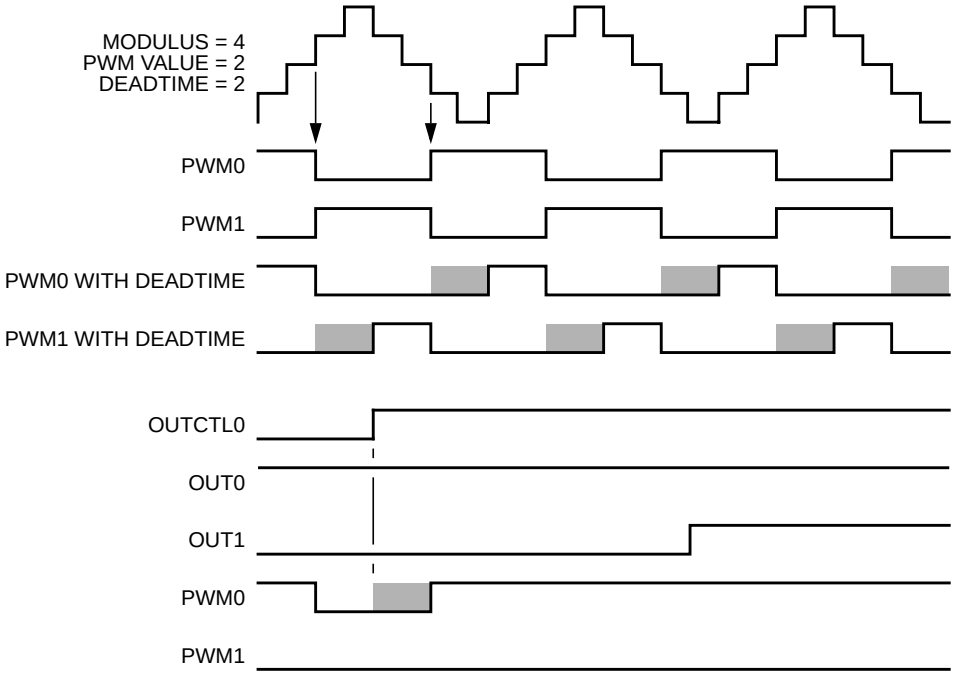


Figure 20-64. Setting OUTCTL with OUT0 Set in Complementary Mode

20.4.7 PWM Generator Loading

20.4.7.1 Load Enable

The load okay bit, LDOK, enables loading the PWM generator with:

- A prescaler divisor—from the PRSC1 and PRSC0 bits in PWM control register
- A PWM period—from the PWM counter modulus registers
- A PWM pulse width—from the PWM value registers

LDOK prevents reloading of these PWM parameters before software is finished calculating them Setting LDOK allows the prescaler bits, PMFMOD and PMFVALx registers to be loaded into a set of buffers. The loaded buffers use the PWM generator at the beginning of the next PWM reload cycle. Set LDOK by reading it when it is a logic zero and then writing a logic one to it. After loading, LDOK is automatically cleared.

20.4.7.2 Load Frequency

The LDFQ3, LDFQ2, LDFQ1, and LDFQ0 bits in the PWM control register (PWMCTL) select an integral loading frequency of one to 16-PWM reload opportunities. The LDFQ bits take effect at every PWM reload opportunity, regardless the state of the load okay bit, LDOK. The *half* bit in the PWMCTL register controls half-cycle reloads for center-aligned PWMs. If the *half* bit is set, a reload opportunity occurs at the beginning of every PWM cycle and half cycle when the count equals the modulus. If the half bit is not set, a reload opportunity occurs only at the beginning of every cycle. Reload opportunities can only occur at the beginning of a PWM cycle in edge-aligned mode.

NOTE

Loading a new modulus on a half cycle will force the count to the new modulus value minus one on the next clock cycle. Half cycle reloads are possible only in center-aligned mode. Enabling or disabling half-cycle reloads in edge-aligned mode will have no effect on the reload rate.

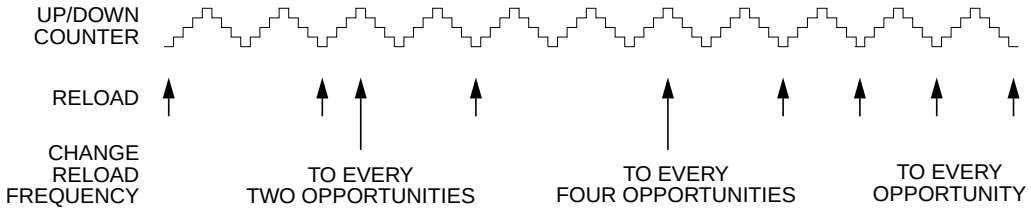


Figure 20-65. Full Cycle Reload Frequency Change

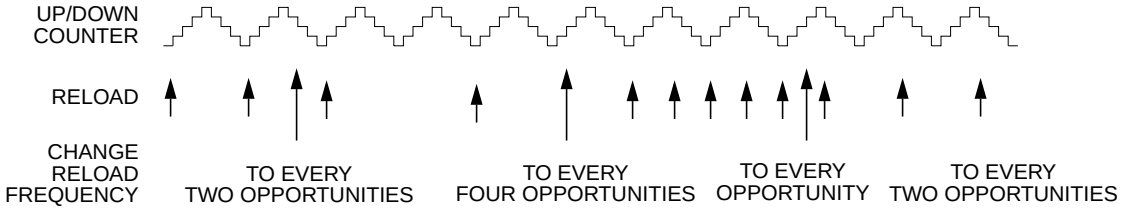


Figure 20-66. Half Cycle Reload Frequency Change

20.4.7.3 Reload Flag

With a reload opportunity, regardless an actual reload occurs as determined by LDOK bit, the PWMF reload flag is set. If the PWM reload interrupt enable bit, PWMRIE is set, the PWMF flag generates CPU interrupt requests allowing software to calculate new PWM parameters in real time. When PWMRIE is not set, reloads still occur at the selected reload rate without generating CPU interrupt requests.

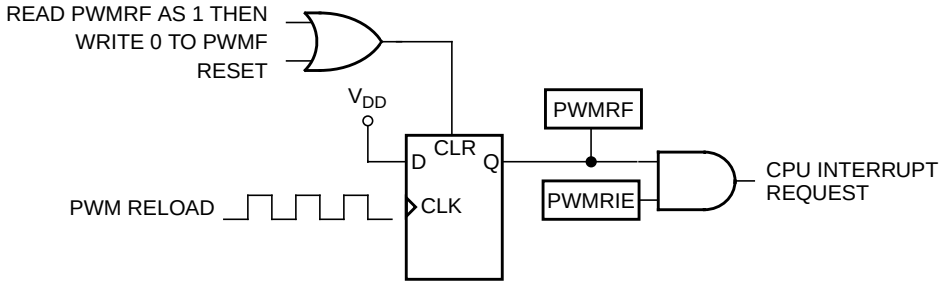


Figure 20-67. PWMRF Reload Interrupt Request

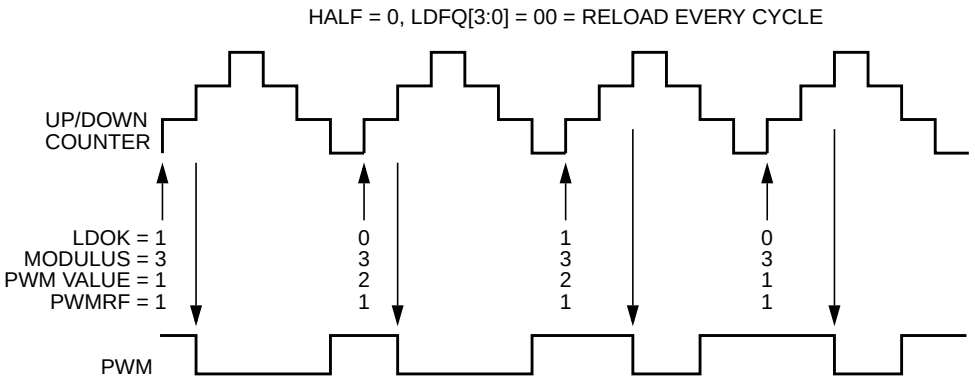


Figure 20-68. Full-Cycle Center-Aligned PWM Value Loading

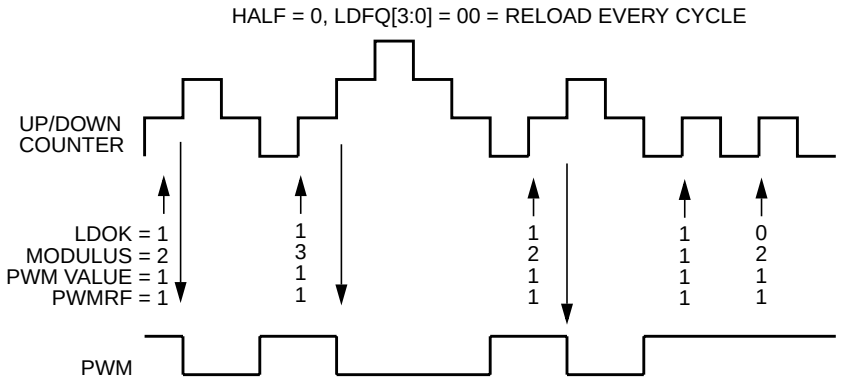


Figure 20-69. Full-Cycle Center-Aligned Modulus Loading

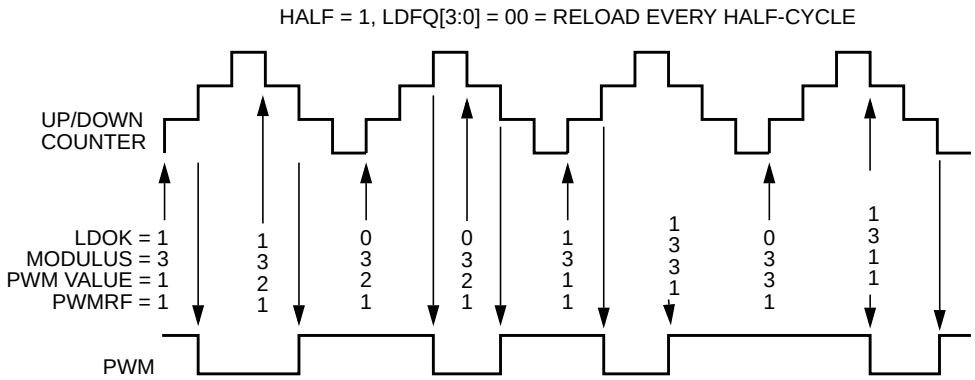


Figure 20-70. Half-Cycle Center-Aligned PWM Value Loading

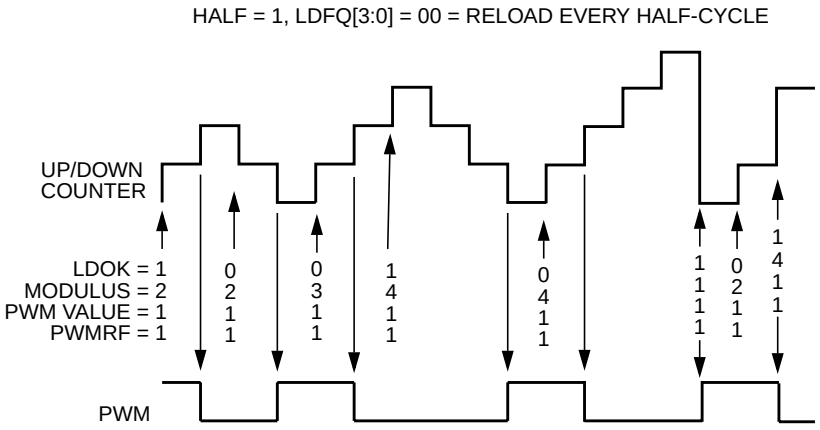


Figure 20-71. Half-Cycle Center-Aligned Modulus Loading

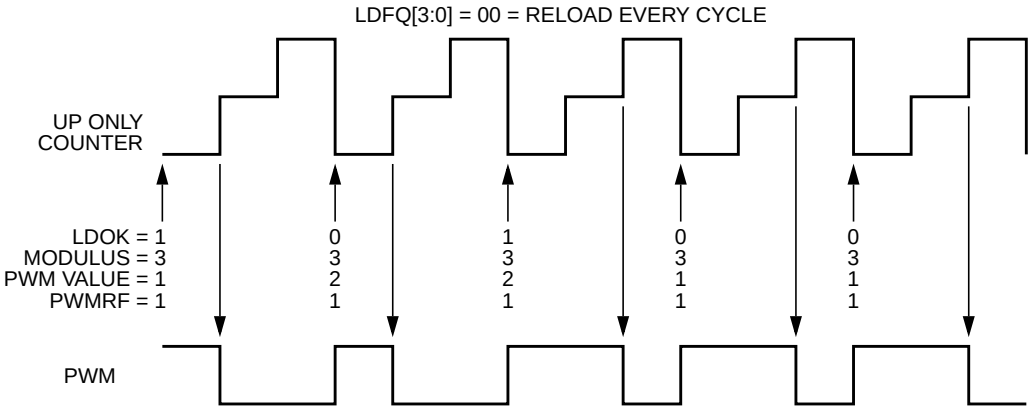


Figure 20-72. Edge-Aligned PWM Value Loading

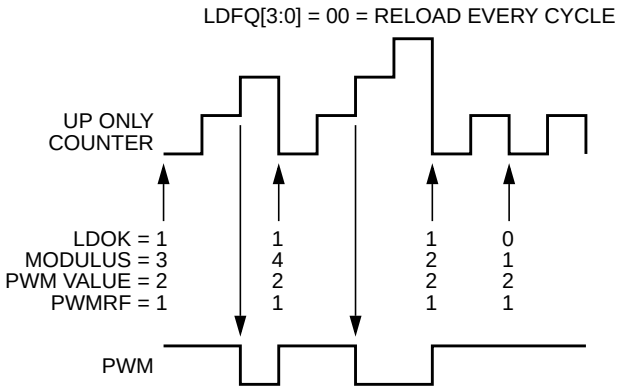


Figure 20-73. Untitled Figure

20.4.7.4 Initialization

Initialize all registers and set the LDOK bit before setting the PWMEN bit. With LDOK set, setting PWMEN for the first time after reset, immediately loads the PWM generator thereby setting the PWMRF flag. PWMRF generates a CPU interrupt request if the PWMRIE bit is set. In complementary channel

operation with current-status correction selected, PWM value registers one, three, and five control the outputs for the first PWM cycle.

NOTE

Even if LDOK is not set, setting PWMEN also sets the PWMRF flag. To prevent a CPU interrupt request, clear the PWMRIE bit before setting PWMEN.

Setting PWMEN for the first time after reset without first setting LDOK loads a prescaler divisor of one, a PWM value of $0000, and an unknown modulus.

The PWM generator uses the last values loaded if PWMEN is cleared and then set while LDOK equals zero.

Initializing the deadtime register, after setting PWMEN or OUTCTLx, can cause an improper deadtime insertion. However, the deadtime can never be shorter than the specified value.

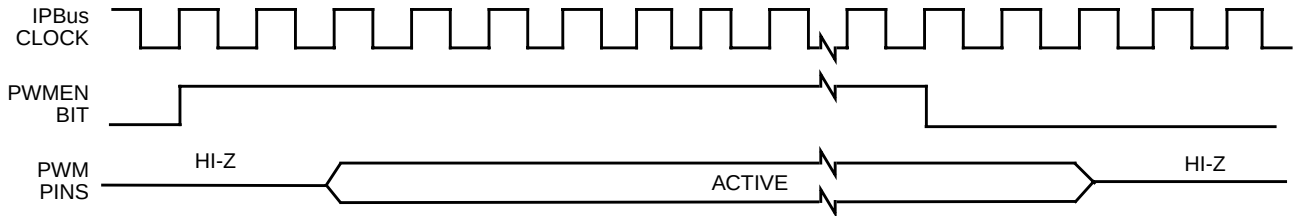


Figure 20-74. PWMEN and PWM Pins in Independent Operation

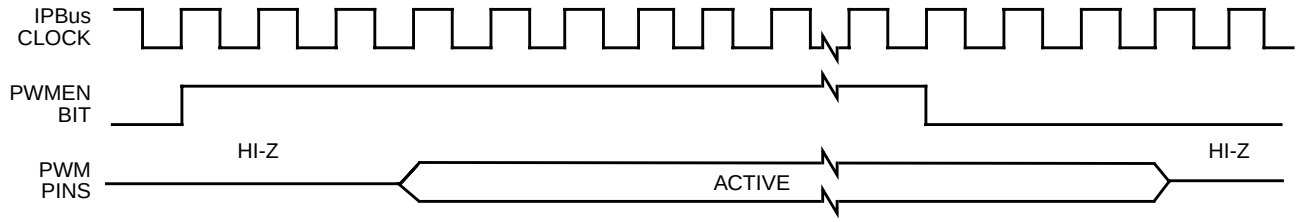


Figure 20-75. PWMEN and PWM Pins in Complementary Operation

When the PWMEN bit is cleared:

- The PWMx outputs will be tri-stated unless OUTCTLx = 1
- The PWM counter is cleared and does not count
- The PWM generator forces its outputs to zero
- The PWMRF flag and pending CPU interrupt requests are not cleared
- All fault circuitry remains active unless FPINEx = 0
- Software output control remains active
- Deadtime insertion continues during software output control

20.4.8 Fault Protection

Fault protection can disable any combination of PWM pins. Faults are generated by a logic one on any of the FAULT pins. Each FAULT pin can be mapped arbitrarily to any of the PWM pins.

When fault protection hardware disables PWM pins, the PWM generator continues to run, only the output pins are deactivated.

The fault decoder disables PWM pins selected by the fault logic and the disable mapping register. See Figure 20-15. Each bank of four bits in the disable mapping register control the mapping for a single PWM pin. Refer to Table 20-12.

The fault protection is enabled even when the PWM is not enabled; therefore, a fault will be latched in and will be cleared in order to prevent an interrupt when the PWM is enabled.

20.4.8.1 Fault Pin Sample Filter

Each fault pin has a sample filter to test for fault conditions. After every bus cycle setting the FAULTx pin at logic zero, the filter synchronously samples the pin once every four bus cycles. QSMP determines the number of consecutive samples that must be logic one for a fault to be detected. When a fault is detected, the corresponding FAULTx pin flag, FFLAGx, is set. Clear FFLAGx by writing a logic one to it.

If the FIEx, FAULTx pin interrupt enable bit is set, the FFLAGx flag generates a CPU interrupt request. The interrupt request latch remains set until:

- Software clears the FFLAGx flag by writing a logic one to it
- Software clears the FIEx bit by writing a logic zero to it
- A reset occurs

20.4.8.2 Automatic Fault Clearing

Setting a fault mode bit, FMODEx, configures faults from the FAULTx pin for automatic clearing.

When FMODEx is set, disabled PWM pins are enabled when the FAULTx pin returns to logic zero and a new PWM half cycle begins. See Figure 20-76. Clearing the FFLAGx flag does not affect disabled PWM pins when FMODEx is set.

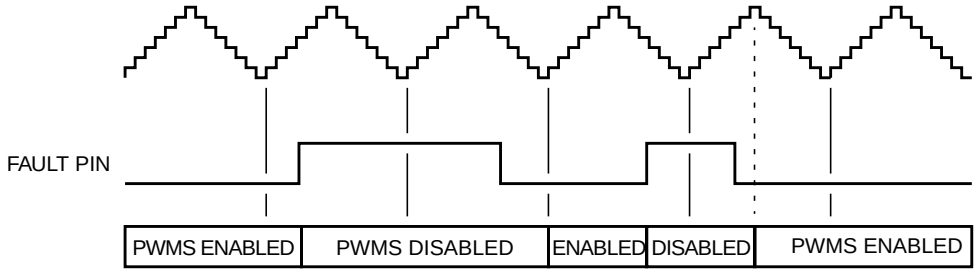


Figure 20-76. Automatic Fault Clearing

20.4.8.3 Manual Fault Clearing

Clearing a fault mode bit, FMODEx, configures faults from the FAULTx pin for manual clearing:

- PWM pins disabled by the FAULT0 pin or the FAULT2 pin are enabled by clearing the corresponding FFLAGx flag. The time at which the PWM pins are enabled depends on the corresponding QSMPx bit setting. If QSMPx = 00, the PWM pins are enabled on the next IP bus cycle when the logic level detected by the filter at the fault pin is logic zero. If QSMPx = 01,10 or

11, the PWMs are enabled when the next PWM half cycle begins regardless of the state of the logic level detected by the filter at the fault. See Figure 20-77 and Figure 20-78.

- PWM pins disabled by the FAULT1 pin or the FAULT3 pin are enabled when
  - — Software clears the corresponding FFLAGx flag
  - — The filter detects a logic zero on the fault pin at the start of the next PWM half cycle boundary. See Figure 20-79.

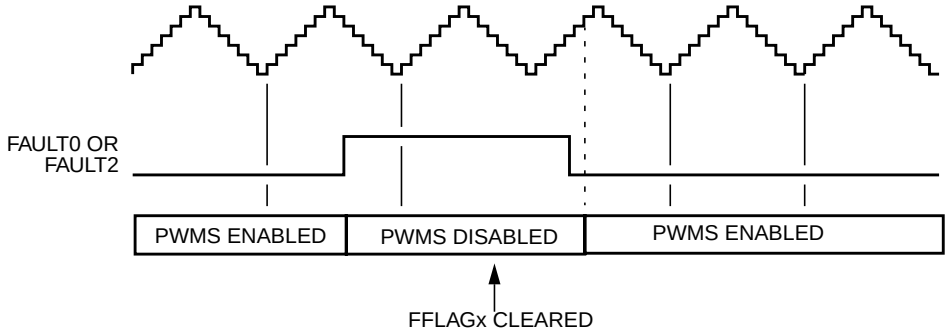


Figure 20-77. Manual Fault Clearing (Faults 0 and 2) — QSMP = 00

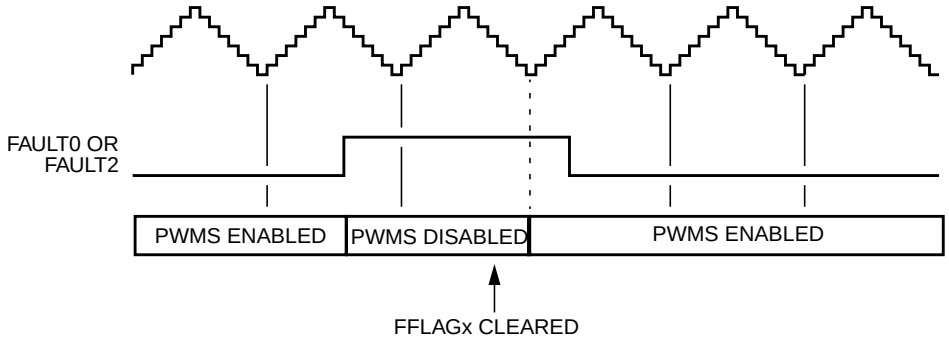


Figure 20-78. Manual Fault Clearing (Faults 0 and 2) — QSMP = 01, 10, or 11

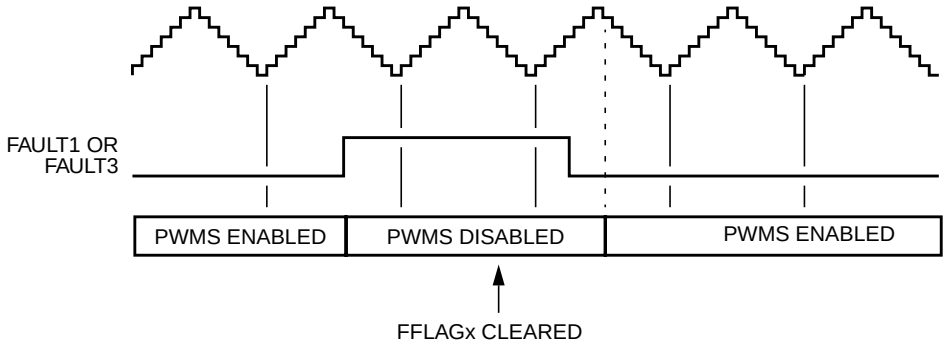


Figure 20-79. Manual Fault Clearing (Faults 1 and 3)

NOTE

PWM half-cycle boundaries occur at both the PWM cycle start and when the counter equals the modulus, so in edge-aligned operation full-cycles and half-cycles are equal.

NOTE

Fault protection also applies during software output control when the OUTCTLx bits are set. Fault clearing still occurs at half PWM cycle boundaries while the PWM generator is engaged, PWMEN equals one. But the OUTx bits can control the PWM pins while the PWM generator is off, PWMEN equals zero. Thus, fault clearing occurs at IPbus cycles while the PWM generator is off and at the start of PWM cycles when the generator is engaged.

20.5 Resets

All PWM registers are reset to their default values upon any system reset.

20.6 Clocks

The system bus clock is the only clock required by this module.

20.7 Interrupts

Seven PWM sources can generate CPU interrupt requests:

- Reload flag x (PWMRFx)—PWMRFx is set at the beginning of every PWM Generator x reload cycle. The reload interrupt enable bit, PWMRIEx, enables PWMRFx to generate CPU interrupt requests.

  where x is A, B and C.
- Fault flag x (FFLAGx)—The FFLAGx bit is set when a logic one occurs on the FAULTx pin. The fault pin interrupt enable x bit, FIEx, enables the FFLAGx flag to generate CPU interrupt requests.

  where x is 0, 1, 2 and 3.

20.8 Electrical Specifications

In general, electrical specifications may vary a bit from chip to chip. This section illustrates typical parameters. Refer to the chip specification electrical and timing specifications for details of a specific implementation.

Table 20-41. DC Electrical Characteristics

Characteristic	Symbol	Min	Typ	Max	Unit
Input high voltage	V_{IH}	2.0	—	5.5	V
Input low voltage	V_{IL}	–0.3	—	0.8	V
Input hysteresis on Schmitt trigger inputs (Fault pins)	V_{HYS}	—	0.3	—	V
Input pullup current	I_{PU}	–50	–100	–170	µA
Input current low (pullups disabled)	I_{IL}	–10	—	10	µA
Input current high (pullups disabled)	I_{IH}	–10	—	10	µA
Output tri-state current low	I_{OZL}	–10	—	10	µA

Table 20-41. DC Electrical Characteristics (continued)

Characteristic	Symbol	Min	Typ	Max	Unit
Output tri-state current high	I_{OZH}	–10	—	10	µA
Output voltage high (at I_{OHP})	V_{OH}	V_{DD} – 0.7	—	—	V
Output voltage low (at I_{OLP})	V_{OL}	—	—	0.4	V
Input capacitance	C_{IN}	—	8	—	pF
Output capacitance	C_{OUT}	—	12	—	pF
PWM pin output high Current(1) (at V_{OH-min})	I_{OHP}	–10	—	—	mA
PWM pin output low current(2) (at V_{OL-min})	I_{OLP}	16	—	—	mA

1. PWM pin output high current measured with 50% duty cycle.
2. PWM pin output low current measured with 50% duty cycle.

Chapter 21
Freescale's Scalable Controller Area Network (S12MSCANV2)

Table 21-1. Revision History

Revision Number	Revision Date	Sections Affected	Description of Changes
V02.14	18 Sep 2002		Added Initialization/Application information; replaced 'MCU' with 'CPU' in several places; cleaned up Mode descriptions; general cleanup.
V02.15	15 Jul 2004		Corrected buffer read/write access definitions; corrected bit time equation.
V02.16	3 Jan 2005		Convert to SRSLite3.2 with single-source document for V02 and V03 (controlled by conditonal text)

21.1 Introduction

Freescale's scalable controller area network (S12MSCANV2) definition is based on the MSCAN12 definition, which is the specific implementation of the MSCAN concept targeted for the M68HC12 microcontroller family.

The module is a communication controller implementing the CAN 2.0A/B protocol as defined in the Bosch specification dated September 1991. For users to fully understand the MSCAN specification, it is recommended that the Bosch specification be read first to familiarize the reader with the terms and concepts contained within this document.

Though not exclusively intended for automotive applications, CAN protocol is designed to meet the specific requirements of a vehicle serial data bus: real-time processing, reliable operation in the EMI environment of a vehicle, cost-effectiveness, and required bandwidth.

MSCAN uses an advanced buffer arrangement resulting in predictable real-time behavior and simplified application software.

21.1.1 Glossary

Table 21-2. Terminology

ACK	Acknowledge of CAN message
CAN	Controller Area Network
CRC	Cyclic Redundancy Code
EOF	End of Frame
FIFO	First-In-First-Out Memory
IFS	Inter-Frame Sequence
SOF	Start of Frame
CPU bus	CPU related read/write data bus
CAN bus	CAN protocol related serial bus
oscillator clock	Direct clock from external oscillator
bus clock	CPU bus related clock
CAN clock	CAN protocol related clock

21.1.2 Block Diagram

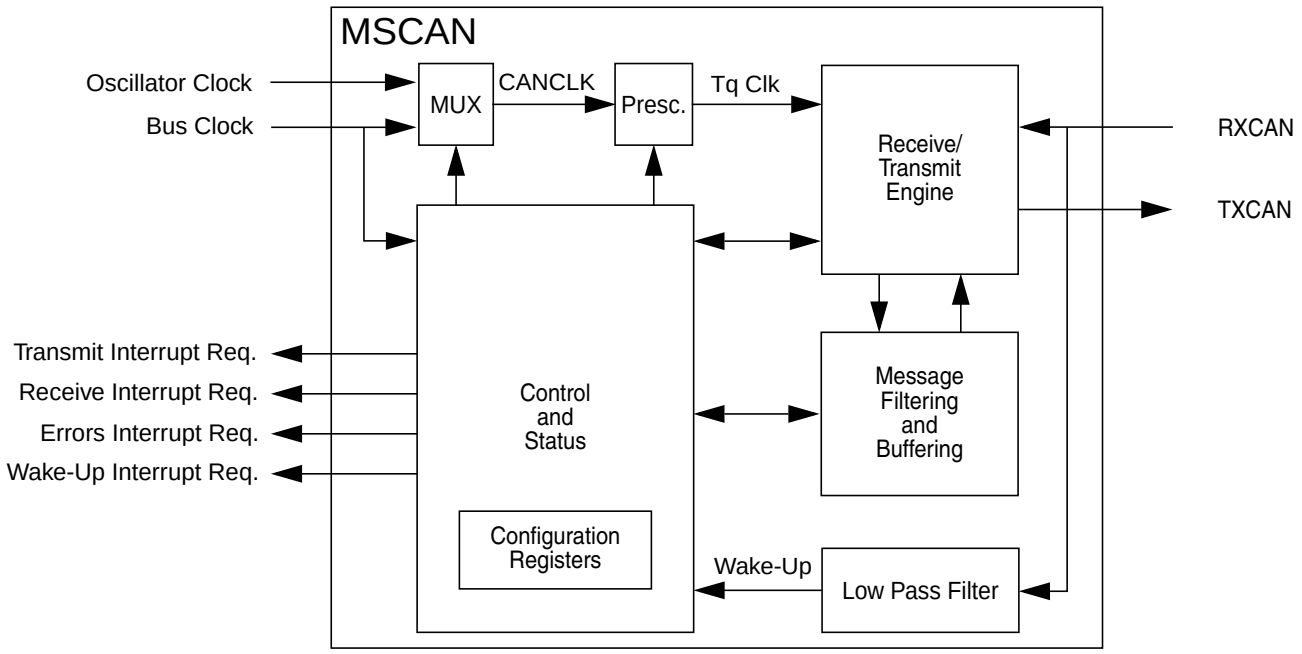


Figure 21-1. MSCAN Block Diagram

21.1.3 Features

The basic features of the MSCAN are as follows:

- Implementation of the CAN protocol — Version 2.0A/B
  - — Standard and extended data frames
  - — Zero to eight bytes data length
  - — Programmable bit rate up to 1 Mbps[1]
  - — Support for remote frames
- Five receive buffers with FIFO storage scheme
- Three transmit buffers with internal prioritization using a "local priority" concept
- Flexible maskable identifier filter supports two full-size (32-bit) extended identifier filters, or four 16-bit filters, or eight 8-bit filters
- Programmable wake-up functionality with integrated low-pass filter
- Programmable loopback mode supports self-test operation
- Programmable listen-only mode for monitoring of CAN bus
- Separate signalling and interrupt capabilities for all CAN receiver and transmitter error states (warning, error passive, bus-off)
- Programmable MSCAN clock source either bus clock or oscillator clock
- Internal timer for time-stamping of received and transmitted messages
- Three low-power modes: sleep, power down, and MSCAN enable
- Global initialization of configuration registers

21.1.4 Modes of Operation

For a description of the specific MSCAN modes and the module operation related to the system operating modes refer to Section 21.4.4, "Modes of Operation".

1. Depending on the actual bit timing and the clock jitter of the PLL.

21.2 External Signal Description

The MSCAN uses two external pins.

NOTE

On MCUs with an integrated CAN physical interface (transceiver) the MSCAN interface is connected internally to the transceiver interface. In these cases the external availability of signals TXCAN and RXCAN is optional.

21.2.1 RXCAN — CAN Receiver Input Pin

RXCAN is the MSCAN receiver input pin.

21.2.2 TXCAN — CAN Transmitter Output Pin

TXCAN is the MSCAN transmitter output pin. The TXCAN output pin represents the logic level on the CAN bus:

0 = Dominant state
1 = Recessive state

21.2.3 CAN System

A typical CAN system with MSCAN is shown in Figure 21-2. Each CAN station is connected physically to the CAN bus lines through a transceiver device. The transceiver is capable of driving the large current needed for the CAN bus and has current protection against defective CAN or defective stations.

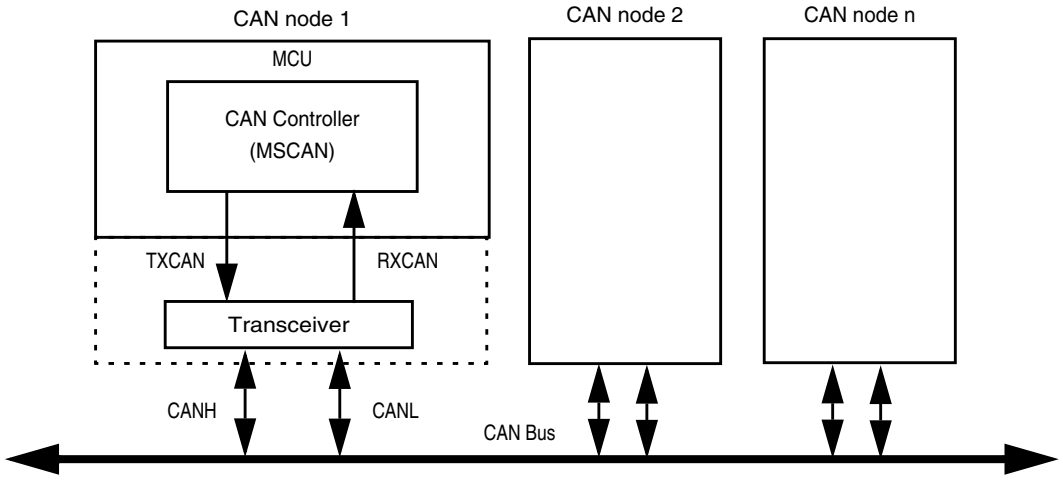


Figure 21-2. CAN System

21.3 Memory Map and Register Definition

This section provides a detailed description of all registers accessible in the MSCAN.

21.3.1 Module Memory Map

Figure 21-3 gives an overview on all registers and their individual bits in the MSCAN memory map. The *register address* results from the addition of *base address* and *address offset*. The *base address* is determined at the MCU level and can be found in the MCU memory map description. The *address offset* is defined at the module level.

The MSCAN occupies 64 bytes in the memory space. The base address of the MSCAN module is determined at the MCU level when the MCU is defined. The register decode map is fixed and begins at the first address of the module address offset.

The detailed register descriptions follow in the order they appear in the register map.

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000 CANCTL0	R	RXFRM	RXACT	CSWAI	SYNCH	TIME	WUPE	SLPRQ	INITRQ
	W								
0x0001 CANCTL1	R	CANE	CLKSRC	LOOPB	LISTEN		WUPM	SLPAK	INITAK
	W								
0x0002 CANBTR0	R	SJW1	SJW0	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0
	W								
0x0003 CANBTR1	R	SAMP	TSEG22	TSEG21	TSEG20	TSEG13	TSEG12	TSEG11	TSEG10
	W								
0x0004 CANRFLG	R	WUPIF	CSCIF	RSTAT1	RSTAT0	TSTAT1	TSTAT0	OVRIF	RXF
	W								
0x0005 CANRIER	R	WUPIE	CSCIE	RSTATE1	RSTATE0	TSTATE1	TSTATE0	OVRIE	RXFIE
	W								
0x0006 CANTFLG	R	0	0	0	0	0	TXE2	TXE1	TXE0
	W								
0x0007 CANTIER	R	0	0	0	0	0	TXEIE2	TXEIE1	TXEIE0
	W								
0x0008 CANTARQ	R	0	0	0	0	0	ABTRQ2	ABTRQ1	ABTRQ0
	W								
0x0009 CANTAAK	R	0	0	0	0	0	ABTAK2	ABTAK1	ABTAK0
	W								
0x000A CANTBSEL	R	0	0	0	0	0	TX2	TX1	TX0
	W								
0x000B CANIDAC	R	0	0	IDAM1	IDAM0	0	IDHIT2	IDHIT1	IDHIT0
	W								
0x000C–0x000D Reserved	R	0	0	0	0	0	0	0	0
	W								
0x000E CANRXERR	R	RXERR7	RXERR6	RXERR5	RXERR4	RXERR3	RXERR2	RXERR1	RXERR0
	W								
0x000F CANTXERR	R	TXERR7	TXERR6	TXERR5	TXERR4	TXERR3	TXERR2	TXERR1	TXERR0
	W								

= Unimplemented or Reserved

Figure 21-3. MSCAN Register Summary

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0010–0x0013 CANIDAR0–3	R	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
	W								
0x0014–0x0017 CANIDMRx	R	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
	W								
0x0018–0x001B CANIDAR4–7	R	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
	W								
0x001C–0x001F CANIDMR4–7	R	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
	W								
0x0020–0x002F CANRXFG	R	See Section 21.3.3, "Programmer's Model of Message Storage"							
	W								
0x0030–0x003F CANTXFG	R	See Section 21.3.3, "Programmer's Model of Message Storage"							
	W								

= Unimplemented or Reserved

Figure 21-3. MSCAN Register Summary (continued)

21.3.2 Register Descriptions

This section describes in detail all the registers and register bits in the MSCAN module. Each description includes a standard register diagram with an associated figure number. Details of register bit and field function follow the register diagrams, in bit order. All bits of all registers in this module are completely synchronous to internal clocks during a register read.

21.3.2.1 MSCAN Control Register 0 (CANCTL0)

The CANCTL0 register provides various control bits of the MSCAN module as described below.

Module Base + 0x0000 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	RXFRM	RXACT	CSWAI	SYNCH	TIME	WUPE	SLPRQ	INITRQ
W								
Reset:	0	0	0	0	0	0	0	1

= Unimplemented

Figure 21-4. MSCAN Control Register 0 (CANCTL0)

1. Read: Anytime
   Write: Anytime when out of initialization mode; exceptions are read-only RXACT and SYNCH, RXFRM (which is set by the module only), and INITRQ (which is also writable in initialization mode)

NOTE

The CANCTL0 register, except WUPE, INITRQ, and SLPRQ, is held in the reset state when the initialization mode is active (INITRQ = 1 and INITAK = 1). This register is writable again as soon as the initialization mode is exited (INITRQ = 0 and INITAK = 0).

Table 21-3. CANCTL0 Register Field Descriptions

Field	Description
7 RXFRM(1)	**Received Frame Flag** — This bit is read and clear only. It is set when a receiver has received a valid message correctly, independently of the filter configuration. After it is set, it remains set until cleared by software or reset. Clearing is done by writing a 1. Writing a 0 is ignored. This bit is not valid in loopback mode. 0 No valid message was received since last clearing this flag 1 A valid message was received since last clearing of this flag
6 RXACT	**Receiver Active Status** — This read-only flag indicates the MSCAN is receiving a message. The flag is controlled by the receiver front end. This bit is not valid in loopback mode. 0 MSCAN is transmitting or idle[2] 1 MSCAN is receiving a message (including when arbitration is lost)(2)
5 CSWAI(3)	**CAN Stops in Wait Mode** — Enabling this bit allows for lower power consumption in wait mode by disabling all the clocks at the CPU bus interface to the MSCAN module. 0 The module is not affected during wait mode 1 The module ceases to be clocked during wait mode
4 SYNCH	**Synchronized Status** — This read-only flag indicates whether the MSCAN is synchronized to the CAN bus and able to participate in the communication process. It is set and cleared by the MSCAN. 0 MSCAN is not synchronized to the CAN bus 1 MSCAN is synchronized to the CAN bus
3 TIME	**Timer Enable** — This bit activates an internal 16-bit wide free running timer which is clocked by the bit clock rate. If the timer is enabled, a 16-bit time stamp will be assigned to each transmitted/received message within the active TX/RX buffer. Right after the EOF of a valid message on the CAN bus, the time stamp is written to the highest bytes (0x000E, 0x000F) in the appropriate buffer (see Section 21.3.3, "Programmer's Model of Message Storage"). The internal timer is reset (all bits set to 0) when disabled. This bit is held low in initialization mode. 0 Disable internal MSCAN timer 1 Enable internal MSCAN timer
2 WUPE(4)	**Wake-Up Enable** — This configuration bit allows the MSCAN to restart from sleep mode or from power down mode (entered from sleep) when traffic on CAN is detected (see Section 21.4.5.5, "MSCAN Sleep Mode"). This bit must be configured before sleep mode entry for the selected function to take effect. 0 Wake-up disabled — The MSCAN ignores traffic on CAN 1 Wake-up enabled — The MSCAN is able to restart

Table 21-3. CANCTL0 Register Field Descriptions (continued)

Field	Description
1 SLPRQ[5]	**Sleep Mode Request** — This bit requests the MSCAN to enter sleep mode, which is an internal power saving mode (see Section 21.4.5.5, "MSCAN Sleep Mode"). The sleep mode request is serviced when the CAN bus is idle, i.e., the module is not receiving a message and all transmit buffers are empty. The module indicates entry to sleep mode by setting SLPAK = 1 (see Section 21.3.2.2, "MSCAN Control Register 1 (CANCTL1)"). SLPRQ cannot be set while the WUPIF flag is set (see Section 21.3.2.5, "MSCAN Receiver Flag Register (CANRFLG)"). Sleep mode will be active until SLPRQ is cleared by the CPU or, depending on the setting of WUPE, the MSCAN detects activity on the CAN bus and clears SLPRQ itself. 0 Running — The MSCAN functions normally 1 Sleep mode request — The MSCAN enters sleep mode when CAN bus idle
0 INITRQ[6],[7]	**Initialization Mode Request** — When this bit is set by the CPU, the MSCAN skips to initialization mode (see Section 21.4.4.5, "MSCAN Initialization Mode"). Any ongoing transmission or reception is aborted and synchronization to the CAN bus is lost. The module indicates entry to initialization mode by setting INITAK = 1 (Section 21.3.2.2, "MSCAN Control Register 1 (CANCTL1)"). The following registers enter their hard reset state and restore their default values: CANCTL0[8], CANRFLG[9], CANRIER[10], CANTFLG, CANTIER, CANTARQ, CANTAAK, and CANTBSEL. The registers CANCTL1, CANBTR0, CANBTR1, CANIDAC, CANIDAR0-7, and CANIDMR0-7 can only be written by the CPU when the MSCAN is in initialization mode (INITRQ = 1 and INITAK = 1). The values of the error counters are not affected by initialization mode. When this bit is cleared by the CPU, the MSCAN restarts and then tries to synchronize to the CAN bus. If the MSCAN is not in bus-off state, it synchronizes after 11 consecutive recessive bits on the CAN bus; if the MSCAN is in bus-off state, it continues to wait for 128 occurrences of 11 consecutive recessive bits. Writing to other bits in CANCTL0, CANRFLG, CANRIER, CANTFLG, or CANTIER must be done only after initialization mode is exited, which is INITRQ = 0 and INITAK = 0. 0 Normal operation 1 MSCAN in initialization mode

1. The MSCAN must be in normal mode for this bit to become set.
2. See the Bosch CAN 2.0A/B specification for a detailed definition of transmitter and receiver states.
3. In order to protect from accidentally violating the CAN protocol, TXCAN is immediately forced to a recessive state when the CPU enters wait (CSWAI = 1) or stop mode (see Section 21.4.5.2, "Operation in Wait Mode" and Section 21.4.5.3, "Operation in Stop Mode")**.**
4. The CPU has to make sure that the WUPE register and the WUPIE wake-up interrupt enable register (see Section 21.3.2.6, "MSCAN Receiver Interrupt Enable Register (CANRIER)) is enabled, if the recovery mechanism from stop or wait is required.
5. The CPU cannot clear SLPRQ before the MSCAN has entered sleep mode (SLPRQ = 1 and SLPAK = 1).
6. The CPU cannot clear INITRQ before the MSCAN has entered initialization mode (INITRQ = 1 and INITAK = 1).
7. In order to protect from accidentally violating the CAN protocol, TXCAN is immediately forced to a recessive state when the initialization mode is requested by the CPU. Thus, the recommended procedure is to bring the MSCAN into sleep mode (SLPRQ = 1 and SLPAK = 1) before requesting initialization mode.
8. Not including WUPE, INITRQ, and SLPRQ.
9. TSTAT1 and TSTAT0 are not affected by initialization mode.
10. RSTAT1 and RSTAT0 are not affected by initialization mode.

21.3.2.2 MSCAN Control Register 1 (CANCTL1)

The CANCTL1 register provides various control bits and handshake status information of the MSCAN module as described below.

Module Base + 0x0001 — Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	CANE	CLKSRC	LOOPB	LISTEN		WUPM	SLPAK	INITAK
W								
Reset:	0	0	0	1	0	0	0	1

= Unimplemented

Figure 21-5. MSCAN Control Register 1 (CANCTL1)

1. Read: AnytimeWrite: Anytime when INITRQ = 1 and INITAK = 1, except CANE which is write once in normal and anytime in special system operation modes when the MSCAN is in initialization mode (INITRQ = 1 and INITAK = 1)

Table 21-4. CANCTL1 Register Field Descriptions

Field	Description
7 CANE	**MSCAN Enable** 0 MSCAN module is disabled 1 MSCAN module is enabled
6 CLKSRC	**MSCAN Clock Source** — This bit defines the clock source for the MSCAN module (only for systems with a clock generation module; Section 21.4.3.2, "Clock System," and Section Figure 21-42., "MSCAN Clocking Scheme,"). 0 MSCAN clock source is the oscillator clock 1 MSCAN clock source is the bus clock
5 LOOPB	**Loopback Self Test Mode** — When this bit is set, the MSCAN performs an internal loopback which can be used for self test operation. The bit stream output of the transmitter is fed back to the receiver internally. The RXCAN input is ignored and the TXCAN output goes to the recessive state (logic 1). The MSCAN behaves as it does normally when transmitting and treats its own transmitted message as a message received from a remote node. In this state, the MSCAN ignores the bit sent during the ACK slot in the CAN frame acknowledge field to ensure proper reception of its own message. Both transmit and receive interrupts are generated. 0 Loopback self test disabled 1 Loopback self test enabled
4 LISTEN	**Listen Only Mode** — This bit configures the MSCAN as a CAN bus monitor. When LISTEN is set, all valid CAN messages with matching ID are received, but no acknowledgement or error frames are sent out (see Section 21.4.4.4, "Listen-Only Mode"). In addition, the error counters are frozen. Listen only mode supports applications which require "hot plugging" or throughput analysis. The MSCAN is unable to transmit any messages when listen only mode is active. 0 Normal operation 1 Listen only mode activated
2 WUPM	**Wake-Up Mode** — If WUPE in CANCTL0 is enabled, this bit defines whether the integrated low-pass filter is applied to protect the MSCAN from spurious wake-up (see Section 21.4.5.5, "MSCAN Sleep Mode"). 0 MSCAN wakes up on any dominant level on the CAN bus 1 MSCAN wakes up only in case of a dominant pulse on the CAN bus that has a length of T_{wup}

Table 21-4. CANCTL1 Register Field Descriptions (continued)

Field	Description
1 SLPAK	**Sleep Mode Acknowledge** — This flag indicates whether the MSCAN module has entered sleep mode (see Section 21.4.5.5, "MSCAN Sleep Mode"). It is used as a handshake flag for the SLPRQ sleep mode request. Sleep mode is active when SLPRQ = 1 and SLPAK = 1. Depending on the setting of WUPE, the MSCAN will clear the flag if it detects activity on the CAN bus while in sleep mode. 0 Running — The MSCAN operates normally 1 Sleep mode active — The MSCAN has entered sleep mode
0 INITAK	**Initialization Mode Acknowledge** — This flag indicates whether the MSCAN module is in initialization mode (see Section 21.4.4.5, "MSCAN Initialization Mode"). It is used as a handshake flag for the INITRQ initialization mode request. Initialization mode is active when INITRQ = 1 and INITAK = 1. The registers CANCTL1, CANBTR0, CANBTR1, CANIDAC, CANIDAR0–CANIDAR7, and CANIDMR0–CANIDMR7 can be written only by the CPU when the MSCAN is in initialization mode. 0 Running — The MSCAN operates normally 1 Initialization mode active — The MSCAN has entered initialization mode

21.3.2.3 MSCAN Bus Timing Register 0 (CANBTR0)

The CANBTR0 register configures various CAN bus timing parameters of the MSCAN module.

Module Base + 0x0002 — Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	SJW1	SJW0	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0
Reset:	0	0	0	0	0	0	0	0

Figure 21-6. MSCAN Bus Timing Register 0 (CANBTR0)

1. Read: Anytime
   Write: Anytime in initialization mode (INITRQ = 1 and INITAK = 1)

Table 21-5. CANBTR0 Register Field Descriptions

Field	Description
7-6 SJW[1:0]	**Synchronization Jump Width** — The synchronization jump width defines the maximum number of time quanta (Tq) clock cycles a bit can be shortened or lengthened to achieve resynchronization to data transitions on the CAN bus (see Table 21-6).
5-0 BRP[5:0]	**Baud Rate Prescaler** — These bits determine the time quanta (Tq) clock which is used to build up the bit timing (see Table 21-7).

Table 21-6. Synchronization Jump Width

SJW1	SJW0	Synchronization Jump Width
0	0	1 Tq clock cycle
0	1	2 Tq clock cycles
1	0	3 Tq clock cycles
1	1	4 Tq clock cycles

Table 21-7. Baud Rate Prescaler

BRP5	BRP4	BRP3	BRP2	BRP1	BRP0	Prescaler value (P)
0	0	0	0	0	0	1
0	0	0	0	0	1	2
0	0	0	0	1	0	3
0	0	0	0	1	1	4
:	:	:	:	:	:	:
1	1	1	1	1	1	64

21.3.2.4 MSCAN Bus Timing Register 1 (CANBTR1)

The CANBTR1 register configures various CAN bus timing parameters of the MSCAN module.

Module Base + 0x0003 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	SAMP	TSEG22	TSEG21	TSEG20	TSEG13	TSEG12	TSEG11	TSEG10
Reset:	0	0	0	0	0	0	0	0

Figure 21-7. MSCAN Bus Timing Register 1 (CANBTR1)

1. Read: Anytime
   Write: Anytime in initialization mode (INITRQ = 1 and INITAK = 1)

Table 21-8. CANBTR1 Register Field Descriptions

Field	Description
7 SAMP	**Sampling** — This bit determines the number of CAN bus samples taken per bit time. 0 One sample per bit. 1 Three samples per bit(1). If SAMP = 0, the resulting bit value is equal to the value of the single bit positioned at the sample point. If SAMP = 1, the resulting bit value is determined by using majority rule on the three total samples. For higher bit rates, it is recommended that only one sample is taken per bit time (SAMP = 0).
6-4 TSEG2[2:0]	**Time Segment 2** — Time segments within the bit time fix the number of clock cycles per bit time and the location of the sample point (see Figure 21-43). Time segment 2 (TSEG2) values are programmable as shown in Table 21-9.
3-0 TSEG1[3:0]	**Time Segment 1** — Time segments within the bit time fix the number of clock cycles per bit time and the location of the sample point (see Figure 21-43). Time segment 1 (TSEG1) values are programmable as shown in Table 21-10.

1. In this case, PHASE_SEG1 must be at least 2 time quanta (Tq).

Table 21-9. Time Segment 2 Values

TSEG22	TSEG21	TSEG20	Time Segment 2
0	0	0	1 Tq clock cycle(1)
0	0	1	2 Tq clock cycles
:	:	:	:
1	1	0	7 Tq clock cycles
1	1	1	8 Tq clock cycles

1. This setting is not valid. Please refer to Table 21-36 for valid settings.

Table 21-10. Time Segment 1 Values

TSEG13	TSEG12	TSEG11	TSEG10	Time segment 1
0	0	0	0	1 Tq clock cycle(1)
0	0	0	1	2 Tq clock cycles1
0	0	1	0	3 Tq clock cycles1
0	0	1	1	4 Tq clock cycles
:	:	:	:	:
1	1	1	0	15 Tq clock cycles
1	1	1	1	16 Tq clock cycles

1. This setting is not valid. Please refer to Table 21-36 for valid settings.

The bit time is determined by the oscillator frequency, the baud rate prescaler, and the number of time quanta (Tq) clock cycles per bit (as shown in Table 21-9 and Table 21-10).

Eqn. 21-1

$$\text{Bit Time} = \frac{(\text{Prescaler value})}{f_{\text{CANCLK}}} \bullet (1 + \text{TimeSegment1} + \text{TimeSegment2})$$

21.3.2.5 MSCAN Receiver Flag Register (CANRFLG)

A flag can be cleared only by software (writing a 1 to the corresponding bit position) when the condition which caused the setting is no longer valid. Every flag has an associated interrupt enable bit in the CANRIER register.

Module Base + 0x0004 — Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	WUPIF	CSCIF	RSTAT1	RSTAT0	TSTAT1	TSTAT0	OVRIF	RXF
W								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 21-8. MSCAN Receiver Flag Register (CANRFLG)

1. Read: Anytime
   Write: Anytime when not in initialization mode, except RSTAT[1:0] and TSTAT[1:0] flags which are read-only; write of 1 clears flag; write of 0 is ignored

NOTE

The CANRFLG register is held in the reset state[1] when the initialization mode is active (INITRQ = 1 and INITAK = 1). This register is writable again as soon as the initialization mode is exited (INITRQ = 0 and INITAK = 0).

Table 21-11. CANRFLG Register Field Descriptions

Field	Description
7 WUPIF	**Wake-Up Interrupt Flag** — If the MSCAN detects CAN bus activity while in sleep mode (see Section 21.4.5.5, "MSCAN Sleep Mode,") and WUPE = 1 in CANTCTL0 (see Section 21.3.2.1, "MSCAN Control Register 0 (CANCTL0)"), the module will set WUPIF. If not masked, a wake-up interrupt is pending while this flag is set. 0 No wake-up activity observed while in sleep mode 1 MSCAN detected activity on the CAN bus and requested wake-up
6 CSCIF	**CAN Status Change Interrupt Flag** — This flag is set when the MSCAN changes its current CAN bus status due to the actual value of the transmit error counter (TEC) and the receive error counter (REC). An additional 4-bit (RSTAT[1:0], TSTAT[1:0]) status register, which is split into separate sections for TEC/REC, informs the system on the actual CAN bus status (see Section 21.3.2.6, "MSCAN Receiver Interrupt Enable Register (CANRIER)"). If not masked, an error interrupt is pending while this flag is set. CSCIF provides a blocking interrupt. That guarantees that the receiver/transmitter status bits (RSTAT/TSTAT) are only updated when no CAN status change interrupt is pending. If the TECs/RECs change their current value after the CSCIF is asserted, which would cause an additional state change in the RSTAT/TSTAT bits, these bits keep their status until the current CSCIF interrupt is cleared again. 0 No change in CAN bus status occurred since last interrupt 1 MSCAN changed current CAN bus status
5-4 RSTAT[1:0]	**Receiver Status Bits** — The values of the error counters control the actual CAN bus status of the MSCAN. As soon as the status change interrupt flag (CSCIF) is set, these bits indicate the appropriate receiver related CAN bus status of the MSCAN. The coding for the bits RSTAT1, RSTAT0 is: 00 RxOK: 0 ≤ receive error counter ≤ 96 01 RxWRN: 96 < receive error counter ≤ 127 10 RxERR: 127 < receive error counter 11 Bus-off(1): transmit error counter > 255
3-2 TSTAT[1:0]	**Transmitter Status Bits** — The values of the error counters control the actual CAN bus status of the MSCAN. As soon as the status change interrupt flag (CSCIF) is set, these bits indicate the appropriate transmitter related CAN bus status of the MSCAN. The coding for the bits TSTAT1, TSTAT0 is: 00 TxOK: 0 ≤ transmit error counter ≤ 96 01 TxWRN: 96 < transmit error counter ≤ 127 10 TxERR: 127 < transmit error counter ≤ 255 11 Bus-Off: transmit error counter > 255

1. The RSTAT[1:0], TSTAT[1:0] bits are not affected by initialization mode.

Table 21-11. CANRFLG Register Field Descriptions (continued)

Field	Description
1 OVRIF	**Overrun Interrupt Flag** — This flag is set when a data overrun condition occurs. If not masked, an error interrupt is pending while this flag is set. 0 No data overrun condition 1 A data overrun detected
0 RXF(2)	**Receive Buffer Full Flag** — RXF is set by the MSCAN when a new message is shifted in the receiver FIFO. This flag indicates whether the shifted buffer is loaded with a correctly received message (matching identifier, matching cyclic redundancy code (CRC) and no other errors detected). After the CPU has read that message from the RxFG buffer in the receiver FIFO, the RXF flag must be cleared to release the buffer. A set RXF flag prohibits the shifting of the next FIFO entry into the foreground buffer (RxFG). If not masked, a receive interrupt is pending while this flag is set. 0 No new message available within the RxFG 1 The receiver FIFO is not empty. A new message is available in the RxFG

1. Redundant information for the most critical CAN bus status which is "bus-off". This only occurs if the Tx error counter exceeds a number of 255 errors. Bus-off affects the receiver state. As soon as the transmitter leaves its bus-off state the receiver state skips to RxOK too. Refer also to TSTAT[1:0] coding in this register.
2. To ensure data integrity, do not read the receive buffer registers while the RXF flag is cleared. For MCUs with dual CPUs, reading the receive buffer registers while the RXF flag is cleared may result in a CPU fault condition.

21.3.2.6 MSCAN Receiver Interrupt Enable Register (CANRIER)

This register contains the interrupt enable bits for the interrupt flags described in the CANRFLG register.

Module Base + 0x0005 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	WUPIE	CSCIE	RSTATE1	RSTATE0	TSTATE1	TSTATE0	OVRIE	RXFIE
Reset:	0	0	0	0	0	0	0	0

Figure 21-9. MSCAN Receiver Interrupt Enable Register (CANRIER)

1. Read: Anytime
   Write: Anytime when not in initialization mode

NOTE

The CANRIER register is held in the reset state when the initialization mode is active (INITRQ=1 and INITAK=1). This register is writable when not in initialization mode (INITRQ=0 and INITAK=0).

The RSTATE[1:0], TSTATE[1:0] bits are not affected by initialization mode.

Table 21-12. CANRIER Register Field Descriptions

Field	Description
7 WUPIE[1]	**Wake-Up Interrupt Enable** 0 No interrupt request is generated from this event. 1 A wake-up event causes a Wake-Up interrupt request.
6 CSCIE	**CAN Status Change Interrupt Enable** 0 No interrupt request is generated from this event. 1 A CAN Status Change event causes an error interrupt request.
5-4 RSTATE[1:0]	**Receiver Status Change Enable** — These RSTAT enable bits control the sensitivity level in which receiver state changes are causing CSCIF interrupts. Independent of the chosen sensitivity level the RSTAT flags continue to indicate the actual receiver state and are only updated if no CSCIF interrupt is pending. 00 Do not generate any CSCIF interrupt caused by receiver state changes. 01 Generate CSCIF interrupt only if the receiver enters or leaves "bus-off" state. Discard other receiver state changes for generating CSCIF interrupt. 10 Generate CSCIF interrupt only if the receiver enters or leaves "RxErr" or "bus-off"[2] state. Discard other receiver state changes for generating CSCIF interrupt. 11 Generate CSCIF interrupt on all state changes.
3-2 TSTATE[1:0]	**Transmitter Status Change Enable** — These TSTAT enable bits control the sensitivity level in which transmitter state changes are causing CSCIF interrupts. Independent of the chosen sensitivity level, the TSTAT flags continue to indicate the actual transmitter state and are only updated if no CSCIF interrupt is pending. 00 Do not generate any CSCIF interrupt caused by transmitter state changes. 01 Generate CSCIF interrupt only if the transmitter enters or leaves "bus-off" state. Discard other transmitter state changes for generating CSCIF interrupt. 10 Generate CSCIF interrupt only if the transmitter enters or leaves "TxErr" or "bus-off" state. Discard other transmitter state changes for generating CSCIF interrupt. 11 Generate CSCIF interrupt on all state changes.
1 OVRIE	**Overrun Interrupt Enable** 0 No interrupt request is generated from this event. 1 An overrun event causes an error interrupt request.
0 RXFIE	**Receiver Full Interrupt Enable** 0 No interrupt request is generated from this event. 1 A receive buffer full (successful message reception) event causes a receiver interrupt request.

1. WUPIE and WUPE (see Section 21.3.2.1, "MSCAN Control Register 0 (CANCTL0)") must both be enabled if the recovery mechanism from stop or wait is required.
2. Bus-off state is only defined for transmitters by the CAN standard (see Bosch CAN 2.0A/B protocol specification). Because the only possible state change for the transmitter from bus-off to TxOK also forces the receiver to skip its current state to RxOK, the coding of the RXSTAT[1:0] flags define an additional bus-off state for the receiver (see Section 21.3.2.5, "MSCAN Receiver Flag Register (CANRFLG)").

21.3.2.7 MSCAN Transmitter Flag Register (CANTFLG)

The transmit buffer empty flags each have an associated interrupt enable bit in the CANTIER register.

Module Base + 0x0006 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	TXE2	TXE1	TXE0
W								
Reset:	0	0	0	0	0	1	1	1

= Unimplemented

Figure 21-10. MSCAN Transmitter Flag Register (CANTFLG)

1. Read: Anytime
   Write: Anytime when not in initialization mode; write of 1 clears flag, write of 0 is ignored

NOTE

The CANTFLG register is held in the reset state when the initialization mode is active (INITRQ = 1 and INITAK = 1). This register is writable when not in initialization mode (INITRQ = 0 and INITAK = 0).

Table 21-13. CANTFLG Register Field Descriptions

Field	Description
2-0 TXE[2:0]	**Transmitter Buffer Empty** — This flag indicates that the associated transmit message buffer is empty, and thus not scheduled for transmission. The CPU must clear the flag after a message is set up in the transmit buffer and is due for transmission. The MSCAN sets the flag after the message is sent successfully. The flag is also set by the MSCAN when the transmission request is successfully aborted due to a pending abort request (see Section 21.3.2.9, "MSCAN Transmitter Message Abort Request Register (CANTARQ)"). If not masked, a transmit interrupt is pending while this flag is set. Clearing a TXEx flag also clears the corresponding ABTAKx (see Section 21.3.2.10, "MSCAN Transmitter Message Abort Acknowledge Register (CANTAAK)"). When a TXEx flag is set, the corresponding ABTRQx bit is cleared (see Section 21.3.2.9, "MSCAN Transmitter Message Abort Request Register (CANTARQ)"). When listen-mode is active (see Section 21.3.2.2, "MSCAN Control Register 1 (CANCTL1)") the TXEx flags cannot be cleared and no transmission is started. Read and write accesses to the transmit buffer will be blocked, if the corresponding TXEx bit is cleared (TXEx = 0) and the buffer is scheduled for transmission. 0 The associated message buffer is full (loaded with a message due for transmission) 1 The associated message buffer is empty (not scheduled)

21.3.2.8 MSCAN Transmitter Interrupt Enable Register (CANTIER)

This register contains the interrupt enable bits for the transmit buffer empty interrupt flags.

Module Base + 0x0007 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	TXEIE2	TXEIE1	TXEIE0
W								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 21-11. MSCAN Transmitter Interrupt Enable Register (CANTIER)

1. Read: Anytime
   Write: Anytime when not in initialization mode

NOTE

The CANTIER register is held in the reset state when the initialization mode is active (INITRQ = 1 and INITAK = 1). This register is writable when not in initialization mode (INITRQ = 0 and INITAK = 0).

Table 21-14. CANTIER Register Field Descriptions

Field	Description
2-0 TXEIE[2:0]	**Transmitter Empty Interrupt Enable** 0 No interrupt request is generated from this event. 1 A transmitter empty (transmit buffer available for transmission) event causes a transmitter empty interrupt request.

21.3.2.9 MSCAN Transmitter Message Abort Request Register (CANTARQ)

The CANTARQ register allows abort request of queued messages as described below.

Module Base + 0x0008 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	ABTRQ2	ABTRQ1	ABTRQ0
W								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 21-12. MSCAN Transmitter Message Abort Request Register (CANTARQ)

1. Read: Anytime
   Write: Anytime when not in initialization mode

NOTE

The CANTARQ register is held in the reset state when the initialization mode is active (INITRQ = 1 and INITAK = 1). This register is writable when not in initialization mode (INITRQ = 0 and INITAK = 0).

Table 21-15. CANTARQ Register Field Descriptions

Field	Description
2-0 ABTRQ[2:0]	**Abort Request** — The CPU sets the ABTRQx bit to request that a scheduled message buffer (TXEx = 0) be aborted. The MSCAN grants the request if the message has not already started transmission, or if the transmission is not successful (lost arbitration or error). When a message is aborted, the associated TXE (see Section 21.3.2.7, "MSCAN Transmitter Flag Register (CANTFLG)") and abort acknowledge flags (ABTAK, see Section 21.3.2.10, "MSCAN Transmitter Message Abort Acknowledge Register (CANTAAK)") are set and a transmit interrupt occurs if enabled. The CPU cannot reset ABTRQx. ABTRQx is reset whenever the associated TXE flag is set. 0 No abort request 1 Abort request pending

21.3.2.10 MSCAN Transmitter Message Abort Acknowledge Register (CANTAAK)

The CANTAAK register indicates the successful abort of a queued message, if requested by the appropriate bits in the CANTARQ register.

Module Base + 0x0009 Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	ABTAK2	ABTAK1	ABTAK0
W								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 21-13. MSCAN Transmitter Message Abort Acknowledge Register (CANTAAK)

1. Read: Anytime
   Write: Unimplemented

NOTE

The CANTAAK register is held in the reset state when the initialization mode is active (INITRQ = 1 and INITAK = 1).

Table 21-16. CANTAAK Register Field Descriptions

Field	Description
2-0 ABTAK[2:0]	**Abort Acknowledge** — This flag acknowledges that a message was aborted due to a pending abort request from the CPU. After a particular message buffer is flagged empty, this flag can be used by the application software to identify whether the message was aborted successfully or was sent anyway. The ABTAKx flag is cleared whenever the corresponding TXE flag is cleared. 0 The message was not aborted. 1 The message was aborted.

21.3.2.11 MSCAN Transmit Buffer Selection Register (CANTBSEL)

The CANTBSEL register allows the selection of the actual transmit message buffer, which then will be accessible in the CANTXFG register space.

Module Base + 0x000A Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	TX2	TX1	TX0
W								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 21-14. MSCAN Transmit Buffer Selection Register (CANTBSEL)

1. Read: Find the lowest ordered bit set to 1, all other bits will be read as 0
   Write: Anytime when not in initialization mode

NOTE

The CANTBSEL register is held in the reset state when the initialization mode is active (INITRQ = 1 and INITAK=1). This register is writable when not in initialization mode (INITRQ = 0 and INITAK = 0).

Table 21-17. CANTBSEL Register Field Descriptions

Field	Description
2-0 TX[2:0]	**Transmit Buffer Select** — The lowest numbered bit places the respective transmit buffer in the CANTXFG register space (e.g., TX1 = 1 and TX0 = 1 selects transmit buffer TX0; TX1 = 1 and TX0 = 0 selects transmit buffer TX1). Read and write accesses to the selected transmit buffer will be blocked, if the corresponding TXEx bit is cleared and the buffer is scheduled for transmission (see Section 21.3.2.7, "MSCAN Transmitter Flag Register (CANTFLG)"). 0 The associated message buffer is deselected 1 The associated message buffer is selected, if lowest numbered bit

The following gives a short programming example of the usage of the CANTBSEL register:

To get the next available transmit buffer, application software must read the CANTFLG register and write this value back into the CANTBSEL register. In this example Tx buffers TX1 and TX2 are available. The value read from CANTFLG is therefore 0b0000_0110. When writing this value back to CANTBSEL, the Tx buffer TX1 is selected in the CANTXFG because the lowest numbered bit set to 1 is at bit position 1. Reading back this value out of CANTBSEL results in 0b0000_0010, because only the lowest numbered bit position set to 1 is presented. This mechanism eases the application software's selection of the next available Tx buffer.

- LDAA CANTFLG; value read is 0b0000_0110
- STAA CANTBSEL; value written is 0b0000_0110
- LDAA CANTBSEL; value read is 0b0000_0010

If all transmit message buffers are deselected, no accesses are allowed to the CANTXFG registers.

21.3.2.12 MSCAN Identifier Acceptance Control Register (CANIDAC)

The CANIDAC register is used for identifier acceptance control as described below.

Module Base + 0x000B — Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	0	0	IDAM1	IDAM0	0	IDHIT2	IDHIT1	IDHIT0
W								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 21-15. MSCAN Identifier Acceptance Control Register (CANIDAC)

1. Read: Anytime
   Write: Anytime in initialization mode (INITRQ = 1 and INITAK = 1), except bits IDHITx, which are read-only

Table 21-18. CANIDAC Register Field Descriptions

Field	Description
5-4 IDAM[1:0]	**Identifier Acceptance Mode** — The CPU sets these flags to define the identifier acceptance filter organization (see Section 21.4.3, "Identifier Acceptance Filter"). Table 21-19 summarizes the different settings. In filter closed mode, no message is accepted such that the foreground buffer is never reloaded.
2-0 IDHIT[2:0]	Identifier Acceptance Hit Indicator — The MSCAN sets these flags to indicate an identifier acceptance hit (see Section 21.4.3, "Identifier Acceptance Filter"). Table 21-20 summarizes the different settings.

Table 21-19. Identifier Acceptance Mode Settings

IDAM1	IDAM0	Identifier Acceptance Mode
0	0	Two 32-bit acceptance filters
0	1	Four 16-bit acceptance filters
1	0	Eight 8-bit acceptance filters
1	1	Filter closed

Table 21-20. Identifier Acceptance Hit Indication

IDHIT2	IDHIT1	IDHIT0	Identifier Acceptance Hit
0	0	0	Filter 0 hit
0	0	1	Filter 1 hit
0	1	0	Filter 2 hit
0	1	1	Filter 3 hit
1	0	0	Filter 4 hit
1	0	1	Filter 5 hit
1	1	0	Filter 6 hit
1	1	1	Filter 7 hit

The IDHITx indicators are always related to the message in the foreground buffer (RxFG). When a message gets shifted into the foreground buffer of the receiver FIFO the indicators are updated as well.

21.3.2.13 MSCAN Reserved Registers

These registers are reserved for factory testing of the MSCAN module and is not available in normal system operating modes.

Module Base + 0x000C to Module Base + 0x000D — Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 21-16. MSCAN Reserved Registers

1. Read: Always reads zero in normal system operation modes
   Write: Unimplemented in normal system operation modes

NOTE

Writing to this register when in special system operating modes can alter the MSCAN functionality.

21.3.2.14 MSCAN Receive Error Counter (CANRXERR)

This register reflects the status of the MSCAN receive error counter.

Module Base + 0x000E — Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	RXERR7	RXERR6	RXERR5	RXERR4	RXERR3	RXERR2	RXERR1	RXERR0
W								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 21-17. MSCAN Receive Error Counter (CANRXERR)

1. Read: Only when in sleep mode (SLPRQ = 1 and SLPAK = 1) or initialization mode (INITRQ = 1 and INITAK = 1)
   Write: Unimplemented

NOTE

Reading this register when in any other mode other than sleep or initialization mode may return an incorrect value. For MCUs with dual CPUs, this may result in a CPU fault condition.

Writing to this register when in special modes can alter the MSCAN functionality.

21.3.2.15 MSCAN Transmit Error Counter (CANTXERR)

This register reflects the status of the MSCAN transmit error counter.

Module Base + 0x000F — Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	TXERR7	TXERR6	TXERR5	TXERR4	TXERR3	TXERR2	TXERR1	TXERR0
W								
Reset:	0	0	0	0	0	0	0	0

= Unimplemented

Figure 21-18. MSCAN Transmit Error Counter (CANTXERR)

1. Read: Only when in sleep mode (SLPRQ = 1 and SLPAK = 1) or initialization mode (INITRQ = 1 and INITAK = 1)
   Write: Unimplemented

NOTE

Reading this register when in any other mode other than sleep or initialization mode, may return an incorrect value. For MCUs with dual CPUs, this may result in a CPU fault condition.

Writing to this register when in special modes can alter the MSCAN functionality.

21.3.2.16 MSCAN Identifier Acceptance Registers (CANIDAR0-7)

On reception, each message is written into the background receive buffer. The CPU is only signalled to read the message if it passes the criteria in the identifier acceptance and identifier mask registers (accepted); otherwise, the message is overwritten by the next message (dropped).

The acceptance registers of the MSCAN are applied on the IDR0–IDR3 registers (see Section 21.3.3.1, "Identifier Registers (IDR0–IDR3)") of incoming messages in a bit by bit manner (see Section 21.4.3, "Identifier Acceptance Filter").

For extended identifiers, all four acceptance and mask registers are applied. For standard identifiers, only the first two (CANIDAR0/1, CANIDMR0/1) are applied.

Module Base + 0x0010 to Module Base + 0x0013 — Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
Reset	0	0	0	0	0	0	0	0

Figure 21-19. MSCAN Identifier Acceptance Registers (First Bank) — CANIDAR0–CANIDAR3

1. Read: Anytime
   Write: Anytime in initialization mode (INITRQ = 1 and INITAK = 1)

Table 21-21. CANIDAR0–CANIDAR3 Register Field Descriptions

Field	Description
7-0 AC[7:0]	**Acceptance Code Bits** — AC[7:0] comprise a user-defined sequence of bits with which the corresponding bits of the related identifier register (IDRn) of the receive message buffer are compared. The result of this comparison is then masked with the corresponding identifier mask register.

Module Base + 0x0018 to Module Base + 0x001B — Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
Reset	0	0	0	0	0	0	0	0

Figure 21-20. MSCAN Identifier Acceptance Registers (Second Bank) — CANIDAR4–CANIDAR7

1. Read: Anytime
   Write: Anytime in initialization mode (INITRQ = 1 and INITAK = 1)

Table 21-22. CANIDAR4–CANIDAR7 Register Field Descriptions

Field	Description
7-0 AC[7:0]	**Acceptance Code Bits** — AC[7:0] comprise a user-defined sequence of bits with which the corresponding bits of the related identifier register (IDRn) of the receive message buffer are compared. The result of this comparison is then masked with the corresponding identifier mask register.

21.3.2.17 MSCAN Identifier Mask Registers (CANIDMR0–CANIDMR7)

The identifier mask register specifies which of the corresponding bits in the identifier acceptance register are relevant for acceptance filtering. To receive standard identifiers in 32 bit filter mode, it is required to program the last three bits (AM[2:0]) in the mask registers CANIDMR1 and CANIDMR5 to "don't care." To receive standard identifiers in 16 bit filter mode, it is required to program the last three bits (AM[2:0]) in the mask registers CANIDMR1, CANIDMR3, CANIDMR5, and CANIDMR7 to "don't care."

Module Base + 0x0014 to Module Base + 0x0017 — Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
Reset	0	0	0	0	0	0	0	0

Figure 21-21. MSCAN Identifier Mask Registers (First Bank) — CANIDMR0–CANIDMR3

1. Read: Anytime
   Write: Anytime in initialization mode (INITRQ = 1 and INITAK = 1)

Table 21-23. CANIDMR0–CANIDMR3 Register Field Descriptions

Field	Description
7-0 AM[7:0]	**Acceptance Mask Bits** — If a particular bit in this register is cleared, this indicates that the corresponding bit in the identifier acceptance register must be the same as its identifier bit before a match is detected. The message is accepted if all such bits match. If a bit is set, it indicates that the state of the corresponding bit in the identifier acceptance register does not affect whether or not the message is accepted. 0 Match corresponding acceptance code register and identifier bits 1 Ignore corresponding acceptance code register bit

Module Base + 0x001C to Module Base + 0x001F — Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
Reset	0	0	0	0	0	0	0	0

Figure 21-22. MSCAN Identifier Mask Registers (Second Bank) — CANIDMR4–CANIDMR7

1. Read: Anytime
   Write: Anytime in initialization mode (INITRQ = 1 and INITAK = 1)

Table 21-24. CANIDMR4–CANIDMR7 Register Field Descriptions

Field	Description
7-0 AM[7:0]	**Acceptance Mask Bits** — If a particular bit in this register is cleared, this indicates that the corresponding bit in the identifier acceptance register must be the same as its identifier bit before a match is detected. The message is accepted if all such bits match. If a bit is set, it indicates that the state of the corresponding bit in the identifier acceptance register does not affect whether or not the message is accepted. 0 Match corresponding acceptance code register and identifier bits 1 Ignore corresponding acceptance code register bit

21.3.3 Programmer's Model of Message Storage

The following section details the organization of the receive and transmit message buffers and the associated control registers.

To simplify the programmer interface, the receive and transmit message buffers have the same outline. Each message buffer allocates 16 bytes in the memory map containing a 13 byte data structure.

An additional transmit buffer priority register (TBPR) is defined for the transmit buffers. Within the last two bytes of this memory map, the MSCAN stores a special 16-bit time stamp, which is sampled from an internal timer after successful transmission or reception of a message. This feature is only available for transmit and receiver buffers, if the TIME bit is set (see Section 21.3.2.1, "MSCAN Control Register 0 (CANCTL0)").

The time stamp register is written by the MSCAN. The CPU can only read these registers.

Table 21-25. Message Buffer Organization

Offset Address	Register	Access
0x00X0	Identifier Register 0	R/W
0x00X1	Identifier Register 1	R/W
0x00X2	Identifier Register 2	R/W
0x00X3	Identifier Register 3	R/W
0x00X4	Data Segment Register 0	R/W
0x00X5	Data Segment Register 1	R/W
0x00X6	Data Segment Register 2	R/W
0x00X7	Data Segment Register 3	R/W
0x00X8	Data Segment Register 4	R/W
0x00X9	Data Segment Register 5	R/W
0x00XA	Data Segment Register 6	R/W
0x00XB	Data Segment Register 7	R/W
0x00XC	Data Length Register	R/W
0x00XD	Transmit Buffer Priority Register(1)	R/W
0x00XE	Time Stamp Register (High Byte)	R
0x00XF	Time Stamp Register (Low Byte)	R

1. Not applicable for receive buffers

Figure 21-23 shows the common 13-byte data structure of receive and transmit buffers for extended identifiers. The mapping of standard identifiers into the IDR registers is shown in Figure 21-24.

All bits of the receive and transmit buffers are 'x' out of reset because of RAM-based implementation[1]. All reserved or unused bits of the receive and transmit buffers always read 'x'.

1. Exception: The transmit buffer priority registers are 0 out of reset.

Figure 21-23. Receive/Transmit Message Buffer — Extended Identifier Mapping

Register Name		Bit 7	6	5	4	3	2	1	Bit0
0x00X0 IDR0	R	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21
	W								
0x00X1 IDR1	R	ID20	ID19	ID18	SRR (=1)	IDE (=1)	ID17	ID16	ID15
	W								
0x00X2 IDR2	R	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7
	W								
0x00X3 IDR3	R	ID6	ID5	ID4	ID3	ID2	ID1	ID0	RTR
	W								
0x00X4 DSR0	R	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	W								
0x00X5 DSR1	R	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	W								
0x00X6 DSR2	R	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	W								
0x00X7 DSR3	R	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	W								
0x00X8 DSR4	R	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	W								
0x00X9 DSR5	R	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	W								
0x00XA DSR6	R	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	W								
0x00XB DSR7	R	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
	W								
0x00XC DLR	R					DLC3	DLC2	DLC1	DLC0
	W								

Figure 21-23. Receive/Transmit Message Buffer — Extended Identifier Mapping (continued)

Register Name	Bit 7	6	5	4	3	2	1	Bit0

= Unused, always read 'x'

Read:

- For transmit buffers, anytime when TXEx flag is set (see Section 21.3.2.7, "MSCAN Transmitter Flag Register (CANTFLG)") and the corresponding transmit buffer is selected in CANTBSEL (see Section 21.3.2.11, "MSCAN Transmit Buffer Selection Register (CANTBSEL)").
- For receive buffers, only when RXF flag is set (see Section 21.3.2.5, "MSCAN Receiver Flag Register (CANRFLG)").

Write:

- For transmit buffers, anytime when TXEx flag is set (see Section 21.3.2.7, "MSCAN Transmitter Flag Register (CANTFLG)") and the corresponding transmit buffer is selected in CANTBSEL (see Section 21.3.2.11, "MSCAN Transmit Buffer Selection Register (CANTBSEL)").
- Unimplemented for receive buffers.

Reset: Undefined because of RAM-based implementation

Figure 21-24. Receive/Transmit Message Buffer — Standard Identifier Mapping

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
IDR0 0x00X0	R	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3
	W								
IDR1 0x00X1	R	ID2	ID1	ID0	RTR	IDE (=0)			
	W								
IDR2 0x00X2	R								
	W								
IDR3 0x00X3	R								
	W								

= Unused, always read 'x'

21.3.3.1 Identifier Registers (IDR0–IDR3)

The identifier registers for an extended format identifier consist of a total of 32 bits: ID[28:0], SRR, IDE, and RTR. The identifier registers for a standard format identifier consist of a total of 13 bits: ID[10:0], RTR, and IDE.

21.3.3.1.1 IDR0–IDR3 for Extended Identifier Mapping

Module Base + 0x00X0

	7	6	5	4	3	2	1	0
R W	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21
Reset:	x	x	x	x	x	x	x	x

Figure 21-25. Identifier Register 0 (IDR0) — Extended Identifier Mapping

Table 21-26. IDR0 Register Field Descriptions — Extended

Field	Description
7-0 ID[28:21]	**Extended Format Identifier —** The identifiers consist of 29 bits (ID[28:0]) for the extended format. ID28 is the most significant bit and is transmitted first on the CAN bus during the arbitration procedure. The priority of an identifier is defined to be highest for the smallest binary number.

Module Base + 0x00X1

	7	6	5	4	3	2	1	0
R W	ID20	ID19	ID18	SRR (=1)	IDE (=1)	ID17	ID16	ID15
Reset:	x	x	x	x	x	x	x	x

Figure 21-26. Identifier Register 1 (IDR1) — Extended Identifier Mapping

Table 21-27. IDR1 Register Field Descriptions — Extended

Field	Description
7-5 ID[20:18]	**Extended Format Identifier —** The identifiers consist of 29 bits (ID[28:0]) for the extended format. ID28 is the most significant bit and is transmitted first on the CAN bus during the arbitration procedure. The priority of an identifier is defined to be highest for the smallest binary number.
4 SRR	**Substitute Remote Request** — This fixed recessive bit is used only in extended format. It must be set to 1 by the user for transmission buffers and is stored as received on the CAN bus for receive buffers.
3 IDE	**ID Extended** — This flag indicates whether the extended or standard identifier format is applied in this buffer. In the case of a receive buffer, the flag is set as received and indicates to the CPU how to process the buffer identifier registers. In the case of a transmit buffer, the flag indicates to the MSCAN what type of identifier to send. 0 Standard format (11 bit) 1 Extended format (29 bit)
2-0 ID[17:15]	**Extended Format Identifier —** The identifiers consist of 29 bits (ID[28:0]) for the extended format. ID28 is the most significant bit and is transmitted first on the CAN bus during the arbitration procedure. The priority of an identifier is defined to be highest for the smallest binary number.

Module Base + 0x00X2

	7	6	5	4	3	2	1	0
R W	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7
Reset:	x	x	x	x	x	x	x	x

Figure 21-27. Identifier Register 2 (IDR2) — Extended Identifier Mapping

Table 21-28. IDR2 Register Field Descriptions — Extended

Field	Description
7-0 ID[14:7]	**Extended Format Identifier —** The identifiers consist of 29 bits (ID[28:0]) for the extended format. ID28 is the most significant bit and is transmitted first on the CAN bus during the arbitration procedure. The priority of an identifier is defined to be highest for the smallest binary number.

Module Base + 0x00X3

	7	6	5	4	3	2	1	0
R W	ID6	ID5	ID4	ID3	ID2	ID1	ID0	RTR
Reset:	x	x	x	x	x	x	x	x

Figure 21-28. Identifier Register 3 (IDR3) — Extended Identifier Mapping

Table 21-29. IDR3 Register Field Descriptions — Extended

Field	Description
7-1 ID[6:0]	**Extended Format Identifier —** The identifiers consist of 29 bits (ID[28:0]) for the extended format. ID28 is the most significant bit and is transmitted first on the CAN bus during the arbitration procedure. The priority of an identifier is defined to be highest for the smallest binary number.
0 RTR	**Remote Transmission Request** — This flag reflects the status of the remote transmission request bit in the CAN frame. In the case of a receive buffer, it indicates the status of the received frame and supports the transmission of an answering frame in software. In the case of a transmit buffer, this flag defines the setting of the RTR bit to be sent. 0 Data frame 1 Remote frame

21.3.3.1.2 IDR0–IDR3 for Standard Identifier Mapping

Module Base + 0x00X0

	7	6	5	4	3	2	1	0
R W	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3
Reset:	x	x	x	x	x	x	x	x

Figure 21-29. Identifier Register 0 — Standard Mapping

Table 21-30. IDR0 Register Field Descriptions — Standard

Field	Description
7-0 ID[10:3]	**Standard Format Identifier —** The identifiers consist of 11 bits (ID[10:0]) for the standard format. ID10 is the most significant bit and is transmitted first on the CAN bus during the arbitration procedure. The priority of an identifier is defined to be highest for the smallest binary number. See also ID bits in Table 21-31.

Module Base + 0x00X1

	7	6	5	4	3	2	1	0
R W	ID2	ID1	ID0	RTR	IDE (=0)			
Reset:	x	x	x	x	x	x	x	x

= Unused; always read 'x'

Figure 21-30. Identifier Register 1 — Standard Mapping

Table 21-31. IDR1 Register Field Descriptions

Field	Description
7-5 ID[2:0]	**Standard Format Identifier —** The identifiers consist of 11 bits (ID[10:0]) for the standard format. ID10 is the most significant bit and is transmitted first on the CAN bus during the arbitration procedure. The priority of an identifier is defined to be highest for the smallest binary number. See also ID bits in Table 21-30.
4 RTR	**Remote Transmission Request** — This flag reflects the status of the Remote Transmission Request bit in the CAN frame. In the case of a receive buffer, it indicates the status of the received frame and supports the transmission of an answering frame in software. In the case of a transmit buffer, this flag defines the setting of the RTR bit to be sent. 0 Data frame 1 Remote frame
3 IDE	**ID Extended** — This flag indicates whether the extended or standard identifier format is applied in this buffer. In the case of a receive buffer, the flag is set as received and indicates to the CPU how to process the buffer identifier registers. In the case of a transmit buffer, the flag indicates to the MSCAN what type of identifier to send. 0 Standard format (11 bit) 1 Extended format (29 bit)

Module Base + 0x00X2

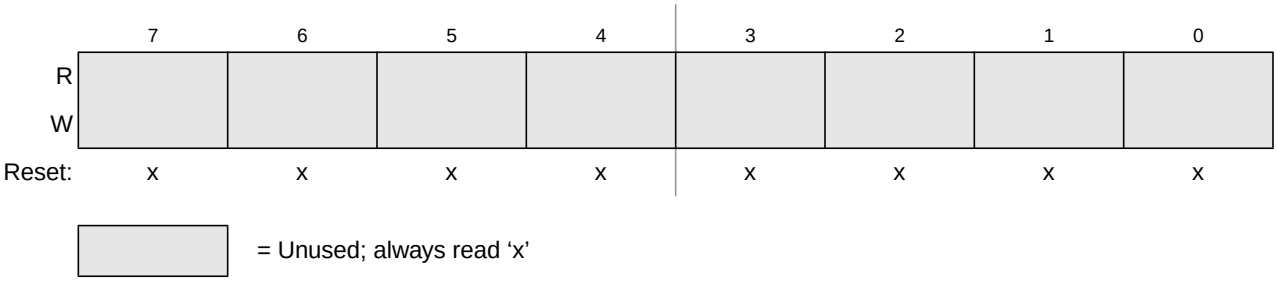


Figure 21-31. Identifier Register 2 — Standard Mapping

Module Base + 0x00X3

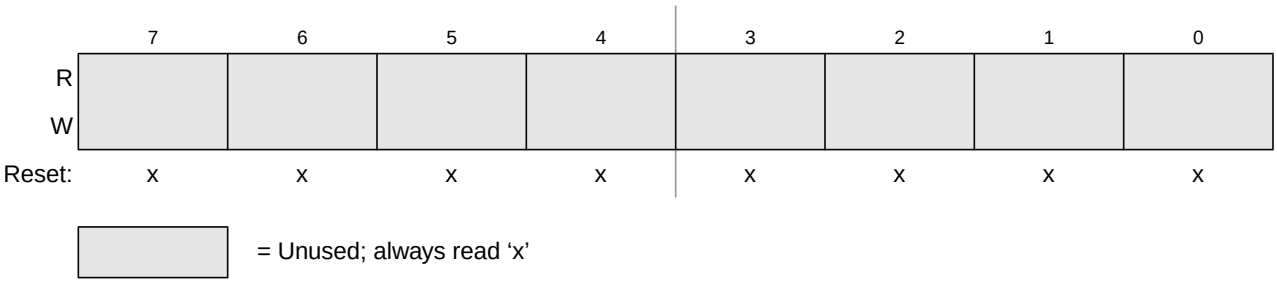


Figure 21-32. Identifier Register 3 — Standard Mapping

21.3.3.2 Data Segment Registers (DSR0-7)

The eight data segment registers, each with bits DB[7:0], contain the data to be transmitted or received. The number of bytes to be transmitted or received is determined by the data length code in the corresponding DLR register.

Module Base + 0x00X4 to Module Base + 0x00XB

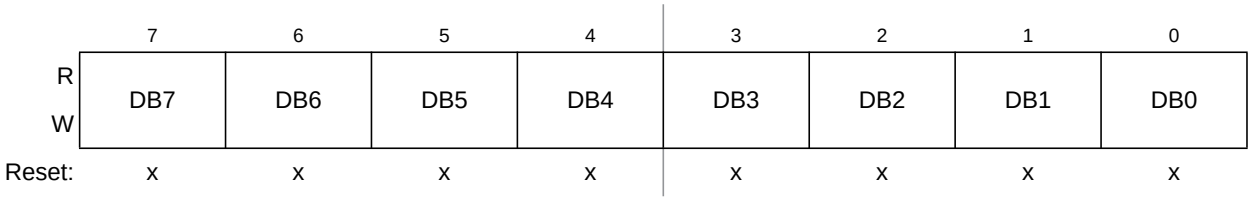


Figure 21-33. Data Segment Registers (DSR0–DSR7) — Extended Identifier Mapping

Table 21-32. DSR0–DSR7 Register Field Descriptions

Field	Description
7-0 DB[7:0]	**Data bits 7-0**

21.3.3.3 Data Length Register (DLR)

This register keeps the data length field of the CAN frame.

Module Base + 0x00XC

	7	6	5	4	3	2	1	0
R					DLC3	DLC2	DLC1	DLC0
W								
Reset:	x	x	x	x	x	x	x	x

= Unused; always read "x"

Figure 21-34. Data Length Register (DLR) — Extended Identifier Mapping

Table 21-33. DLR Register Field Descriptions

Field	Description
3-0 DLC[3:0]	**Data Length Code Bits** — The data length code contains the number of bytes (data byte count) of the respective message. During the transmission of a remote frame, the data length code is transmitted as programmed while the number of transmitted data bytes is always 0. The data byte count ranges from 0 to 8 for a data frame. Table 21-34 shows the effect of setting the DLC bits.

Table 21-34. Data Length Codes

Data Length Code				Data Byte Count
DLC3	DLC2	DLC1	DLC0	
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8

21.3.3.4 Transmit Buffer Priority Register (TBPR)

This register defines the local priority of the associated message buffer. The local priority is used for the internal prioritization process of the MSCAN and is defined to be highest for the smallest binary number. The MSCAN implements the following internal prioritization mechanisms:

- All transmission buffers with a cleared TXEx flag participate in the prioritization immediately before the SOF (start of frame) is sent.

- The transmission buffer with the lowest local priority field wins the prioritization.

In cases of more than one buffer having the same lowest priority, the message buffer with the lower index number wins.

Module Base + 0x00XD — Access: User read/write(1)

	7	6	5	4	3	2	1	0
R W	PRIO7	PRIO6	PRIO5	PRIO4	PRIO3	PRIO2	PRIO1	PRIO0
Reset:	0	0	0	0	0	0	0	0

Figure 21-35. Transmit Buffer Priority Register (TBPR)

1. Read: Anytime when TXEx flag is set (see Section 21.3.2.7, "MSCAN Transmitter Flag Register (CANTFLG)") and the corresponding transmit buffer is selected in CANTBSEL (see Section 21.3.2.11, "MSCAN Transmit Buffer Selection Register (CANTBSEL)")
   Write: Anytime when TXEx flag is set (see Section 21.3.2.7, "MSCAN Transmitter Flag Register (CANTFLG)") and the corresponding transmit buffer is selected in CANTBSEL (see Section 21.3.2.11, "MSCAN Transmit Buffer Selection Register (CANTBSEL)")

21.3.3.5 Time Stamp Register (TSRH–TSRL)

If the TIME bit is enabled, the MSCAN will write a time stamp to the respective registers in the active transmit or receive buffer right after the EOF of a valid message on the CAN bus (see Section 21.3.2.1, "MSCAN Control Register 0 (CANCTL0)"). In case of a transmission, the CPU can only read the time stamp after the respective transmit buffer has been flagged empty.

The timer value, which is used for stamping, is taken from a free running internal CAN bit clock. A timer overrun is not indicated by the MSCAN. The timer is reset (all bits set to 0) during initialization mode. The CPU can only read the time stamp registers.

Module Base + 0x00XE — Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	TSR15	TSR14	TSR13	TSR12	TSR11	TSR10	TSR9	TSR8
W								
Reset:	x	x	x	x	x	x	x	x

Figure 21-36. Time Stamp Register — High Byte (TSRH)

1. Read: Anytime when TXEx flag is set (see Section 21.3.2.7, "MSCAN Transmitter Flag Register (CANTFLG)") and the corresponding transmit buffer is selected in CANTBSEL (see Section 21.3.2.11, "MSCAN Transmit Buffer Selection Register (CANTBSEL)")
   Write: Unimplemented

Module Base + 0x00XF

Access: User read/write(1)

	7	6	5	4	3	2	1	0
R	TSR7	TSR6	TSR5	TSR4	TSR3	TSR2	TSR1	TSR0
W								
Reset:	x	x	x	x	x	x	x	x

Figure 21-37. Time Stamp Register — Low Byte (TSRL)

1. Read: Anytime when TXEx flag is set (see Section 21.3.2.7, "MSCAN Transmitter Flag Register (CANTFLG)") and the corresponding transmit buffer is selected in CANTBSEL (see Section 21.3.2.11, "MSCAN Transmit Buffer Selection Register (CANTBSEL)")
   Write: Unimplemented

21.4 Functional Description

21.4.1 General

This section provides a complete functional description of the MSCAN.

21.4.2 Message Storage

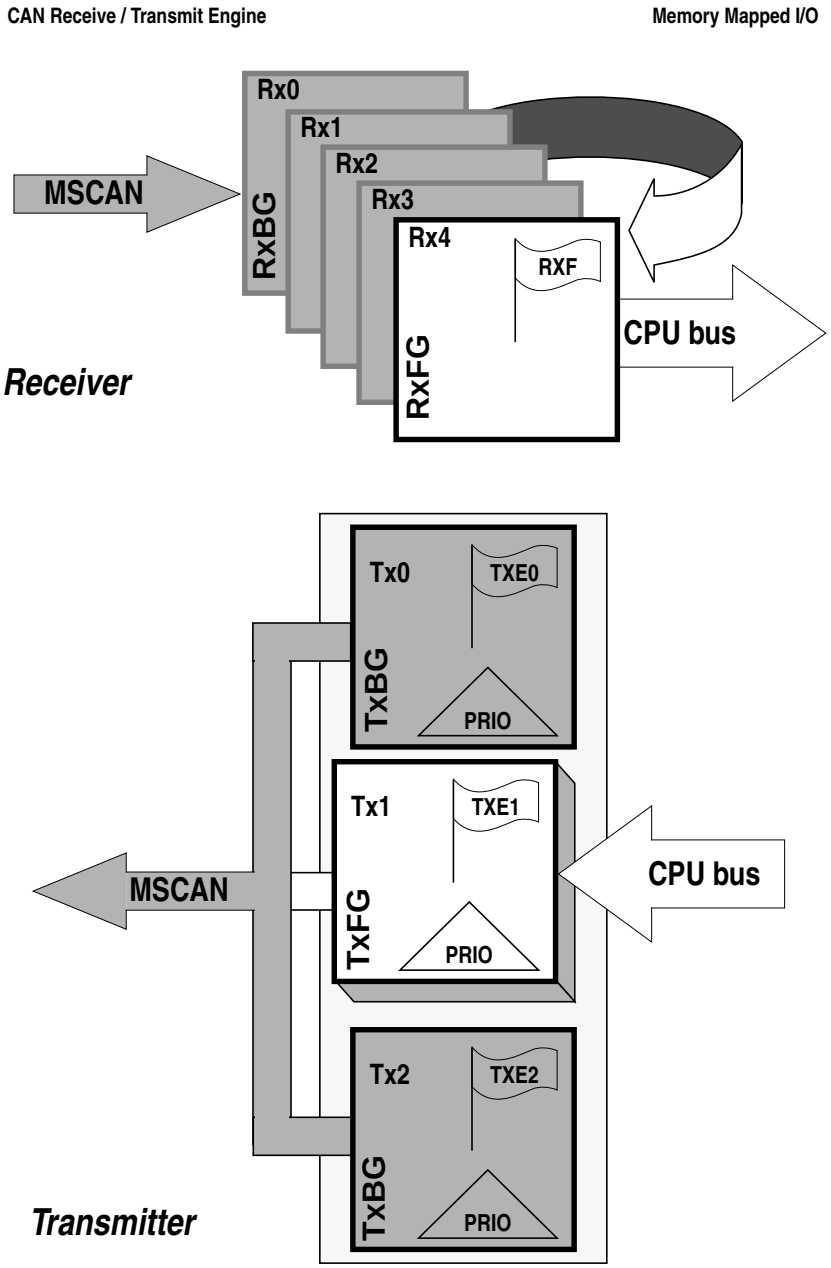


Figure 21-38. User Model for Message Buffer Organization

The MSCAN facilitates a sophisticated message storage system which addresses the requirements of a broad range of network applications.

21.4.2.1 Message Transmit Background

Modern application layer software is built upon two fundamental assumptions:

- Any CAN node is able to send out a stream of scheduled messages without releasing the CAN bus between the two messages. Such nodes arbitrate for the CAN bus immediately after sending the previous message and only release the CAN bus in case of lost arbitration.
- The internal message queue within any CAN node is organized such that the highest priority message is sent out first, if more than one message is ready to be sent.

The behavior described in the bullets above cannot be achieved with a single transmit buffer. That buffer must be reloaded immediately after the previous message is sent. This loading process lasts a finite amount of time and must be completed within the inter-frame sequence (IFS) to be able to send an uninterrupted stream of messages. Even if this is feasible for limited CAN bus speeds, it requires that the CPU reacts with short latencies to the transmit interrupt.

A double buffer scheme de-couples the reloading of the transmit buffer from the actual message sending and, therefore, reduces the reactiveness requirements of the CPU. Problems can arise if the sending of a message is finished while the CPU re-loads the second buffer. No buffer would then be ready for transmission, and the CAN bus would be released.

At least three transmit buffers are required to meet the first of the above requirements under all circumstances. The MSCAN has three transmit buffers.

The second requirement calls for some sort of internal prioritization which the MSCAN implements with the "local priority" concept described in Section 21.4.2.2, "Transmit Structures."

21.4.2.2 Transmit Structures

The MSCAN triple transmit buffer scheme optimizes real-time performance by allowing multiple messages to be set up in advance. The three buffers are arranged as shown in Figure 21-38.

All three buffers have a 13-byte data structure similar to the outline of the receive buffers (see Section 21.3.3, "Programmer's Model of Message Storage"). An additional Transmit Buffer Priority Register (TBPR) contains an 8-bit local priority field (PRIO) (see Section 21.3.3.4, "Transmit Buffer Priority Register (TBPR)"). The remaining two bytes are used for time stamping of a message, if required (see Section 21.3.3.5, "Time Stamp Register (TSRH–TSRL)").

To transmit a message, the CPU must identify an available transmit buffer, which is indicated by a set transmitter buffer empty (TXEx) flag (see Section 21.3.2.7, "MSCAN Transmitter Flag Register (CANTFLG)"). If a transmit buffer is available, the CPU must set a pointer to this buffer by writing to the CANTBSEL register (see Section 21.3.2.11, "MSCAN Transmit Buffer Selection Register (CANTBSEL)"). This makes the respective buffer accessible within the CANTXFG address space (see Section 21.3.3, "Programmer's Model of Message Storage"). The algorithmic feature associated with the CANTBSEL register simplifies the transmit buffer selection. In addition, this scheme makes the handler

software simpler because only one address area is applicable for the transmit process, and the required address space is minimized.

The CPU then stores the identifier, the control bits, and the data content into one of the transmit buffers. Finally, the buffer is flagged as ready for transmission by clearing the associated TXE flag.

The MSCAN then schedules the message for transmission and signals the successful transmission of the buffer by setting the associated TXE flag. A transmit interrupt (see Section 21.4.7.2, "Transmit Interrupt") is generated[1] when TXEx is set and can be used to drive the application software to re-load the buffer.

If more than one buffer is scheduled for transmission when the CAN bus becomes available for arbitration, the MSCAN uses the local priority setting of the three buffers to determine the prioritization. For this purpose, every transmit buffer has an 8-bit local priority field (PRIO). The application software programs this field when the message is set up. The local priority reflects the priority of this particular message relative to the set of messages being transmitted from this node. The lowest binary value of the PRIO field is defined to be the highest priority. The internal scheduling process takes place whenever the MSCAN arbitrates for the CAN bus. This is also the case after the occurrence of a transmission error.

When a high priority message is scheduled by the application software, it may become necessary to abort a lower priority message in one of the three transmit buffers. Because messages that are already in transmission cannot be aborted, the user must request the abort by setting the corresponding abort request bit (ABTRQ) (see Section 21.3.2.9, "MSCAN Transmitter Message Abort Request Register (CANTARQ)".) The MSCAN then grants the request, if possible, by:

1. Setting the corresponding abort acknowledge flag (ABTAK) in the CANTAAK register.
2. Setting the associated TXE flag to release the buffer.
3. Generating a transmit interrupt. The transmit interrupt handler software can determine from the setting of the ABTAK flag whether the message was aborted (ABTAK = 1) or sent (ABTAK = 0).

21.4.2.3 Receive Structures

The received messages are stored in a five stage input FIFO. The five message buffers are alternately mapped into a single memory area (see Figure 21-38). The background receive buffer (RxBG) is exclusively associated with the MSCAN, but the foreground receive buffer (RxFG) is addressable by the CPU (see Figure 21-38). This scheme simplifies the handler software because only one address area is applicable for the receive process.

All receive buffers have a size of 15 bytes to store the CAN control bits, the identifier (standard or extended), the data contents, and a time stamp, if enabled (see Section 21.3.3, "Programmer's Model of Message Storage").

The receiver full flag (RXF) (see Section 21.3.2.5, "MSCAN Receiver Flag Register (CANRFLG)") signals the status of the foreground receive buffer. When the buffer contains a correctly received message with a matching identifier, this flag is set.

On reception, each message is checked to see whether it passes the filter (see Section 21.4.3, "Identifier Acceptance Filter") and simultaneously is written into the active RxBG. After successful reception of a valid message, the MSCAN shifts the content of RxBG into the receiver FIFO, sets the RXF flag, and

1. The transmit interrupt occurs only if not masked. A polling scheme can be applied on TXEx also.

generates a receive interrupt[1] (see Section 21.4.7.3, "Receive Interrupt") to the CPU. The user's receive handler must read the received message from the RxFG and then reset the RXF flag to acknowledge the interrupt and to release the foreground buffer. A new message, which can follow immediately after the IFS field of the CAN frame, is received into the next available RxBG. If the MSCAN receives an invalid message in its RxBG (wrong identifier, transmission errors, etc.) the actual contents of the buffer will be over-written by the next message. The buffer will then not be shifted into the FIFO.

When the MSCAN module is transmitting, the MSCAN receives its own transmitted messages into the background receive buffer, RxBG, but does not shift it into the receiver FIFO, generate a receive interrupt, or acknowledge its own messages on the CAN bus. The exception to this rule is in loopback mode (see Section 21.3.2.2, "MSCAN Control Register 1 (CANCTL1)") where the MSCAN treats its own messages exactly like all other incoming messages. The MSCAN receives its own transmitted messages in the event that it loses arbitration. If arbitration is lost, the MSCAN must be prepared to become a receiver.

An overrun condition occurs when all receive message buffers in the FIFO are filled with correctly received messages with accepted identifiers and another message is correctly received from the CAN bus with an accepted identifier. The latter message is discarded and an error interrupt with overrun indication is generated if enabled (see Section 21.4.7.5, "Error Interrupt"). The MSCAN remains able to transmit messages while the receiver FIFO is being filled, but all incoming messages are discarded. As soon as a receive buffer in the FIFO is available again, new valid messages will be accepted.

21.4.3 Identifier Acceptance Filter

The MSCAN identifier acceptance registers (see Section 21.3.2.12, "MSCAN Identifier Acceptance Control Register (CANIDAC)") define the acceptable patterns of the standard or extended identifier (ID[10:0] or ID[28:0]). Any of these bits can be marked 'don't care' in the MSCAN identifier mask registers (see Section 21.3.2.17, "MSCAN Identifier Mask Registers (CANIDMR0–CANIDMR7)").

A filter hit is indicated to the application software by a set receive buffer full flag (RXF = 1) and three bits in the CANIDAC register (see Section 21.3.2.12, "MSCAN Identifier Acceptance Control Register (CANIDAC)"). These identifier hit flags (IDHIT[2:0]) clearly identify the filter section that caused the acceptance. They simplify the application software's task to identify the cause of the receiver interrupt. If more than one hit occurs (two or more filters match), the lower hit has priority.

A very flexible programmable generic identifier acceptance filter has been introduced to reduce the CPU interrupt loading. The filter is programmable to operate in four different modes:

- Two identifier acceptance filters, each to be applied to:
  - The full 29 bits of the extended identifier and to the following bits of the CAN 2.0B frame:
    - Remote transmission request (RTR)
    - Identifier extension (IDE)
    - Substitute remote request (SRR)
  - The 11 bits of the standard identifier plus the RTR and IDE bits of the CAN 2.0A/B messages. This mode implements two filters for a full length CAN 2.0B compliant extended identifier. Although this mode can be used for standard identifiers, it is recommended to use the four or eight identifier acceptance filters.

1. The receive interrupt occurs only if not masked. A polling scheme can be applied on RXF also.

Figure 21-39 shows how the first 32-bit filter bank (CANIDAR0–CANIDAR3, CANIDMR0–CANIDMR3) produces a filter 0 hit. Similarly, the second filter bank (CANIDAR4–CANIDAR7, CANIDMR4–CANIDMR7) produces a filter 1 hit.

- Four identifier acceptance filters, each to be applied to:
  - The 14 most significant bits of the extended identifier plus the SRR and IDE bits of CAN 2.0B messages.
  - The 11 bits of the standard identifier, the RTR and IDE bits of CAN 2.0A/B messages. Figure 21-40 shows how the first 32-bit filter bank (CANIDAR0–CANIDA3, CANIDMR0–3CANIDMR) produces filter 0 and 1 hits. Similarly, the second filter bank (CANIDAR4–CANIDAR7, CANIDMR4–CANIDMR7) produces filter 2 and 3 hits.
- Eight identifier acceptance filters, each to be applied to the first 8 bits of the identifier. This mode implements eight independent filters for the first 8 bits of a CAN 2.0A/B compliant standard identifier or a CAN 2.0B compliant extended identifier.
  Figure 21-41 shows how the first 32-bit filter bank (CANIDAR0–CANIDAR3, CANIDMR0–CANIDMR3) produces filter 0 to 3 hits. Similarly, the second filter bank (CANIDAR4–CANIDAR7, CANIDMR4–CANIDMR7) produces filter 4 to 7 hits.
- Closed filter. No CAN message is copied into the foreground buffer RxFG, and the RXF flag is never set.

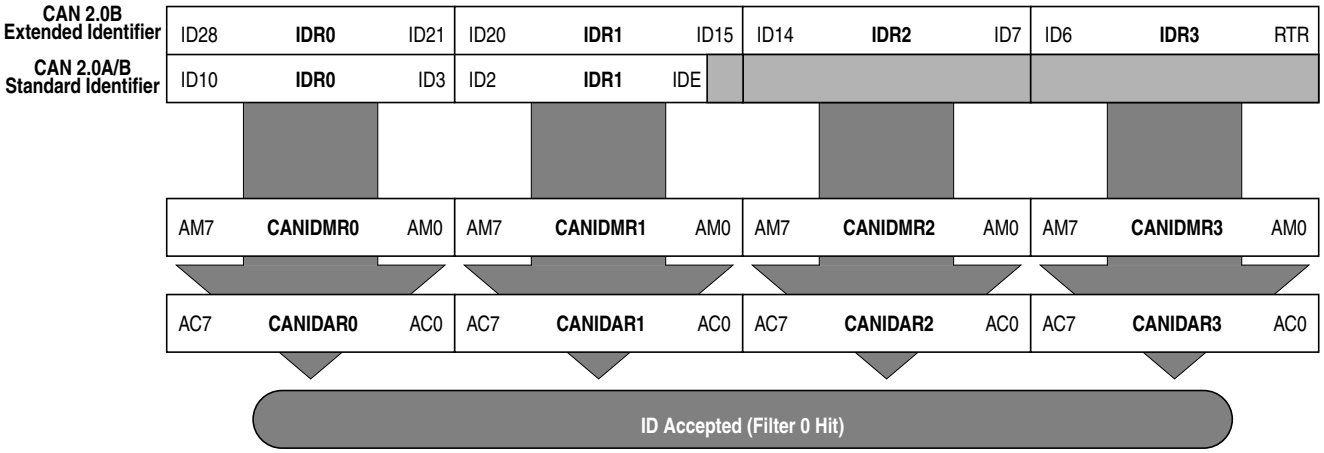


Figure 21-39. 32-bit Maskable Identifier Acceptance Filter

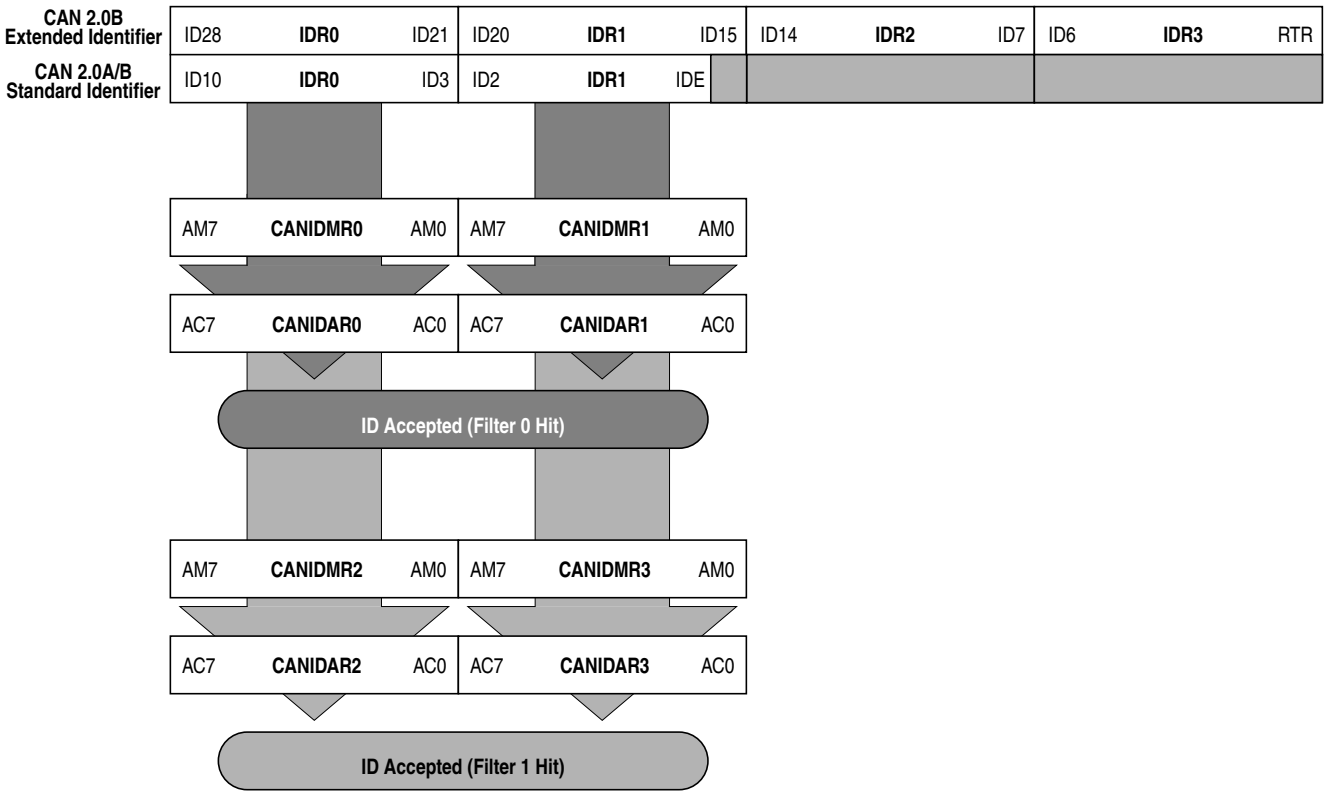


Figure 21-40. 16-bit Maskable Identifier Acceptance Filters

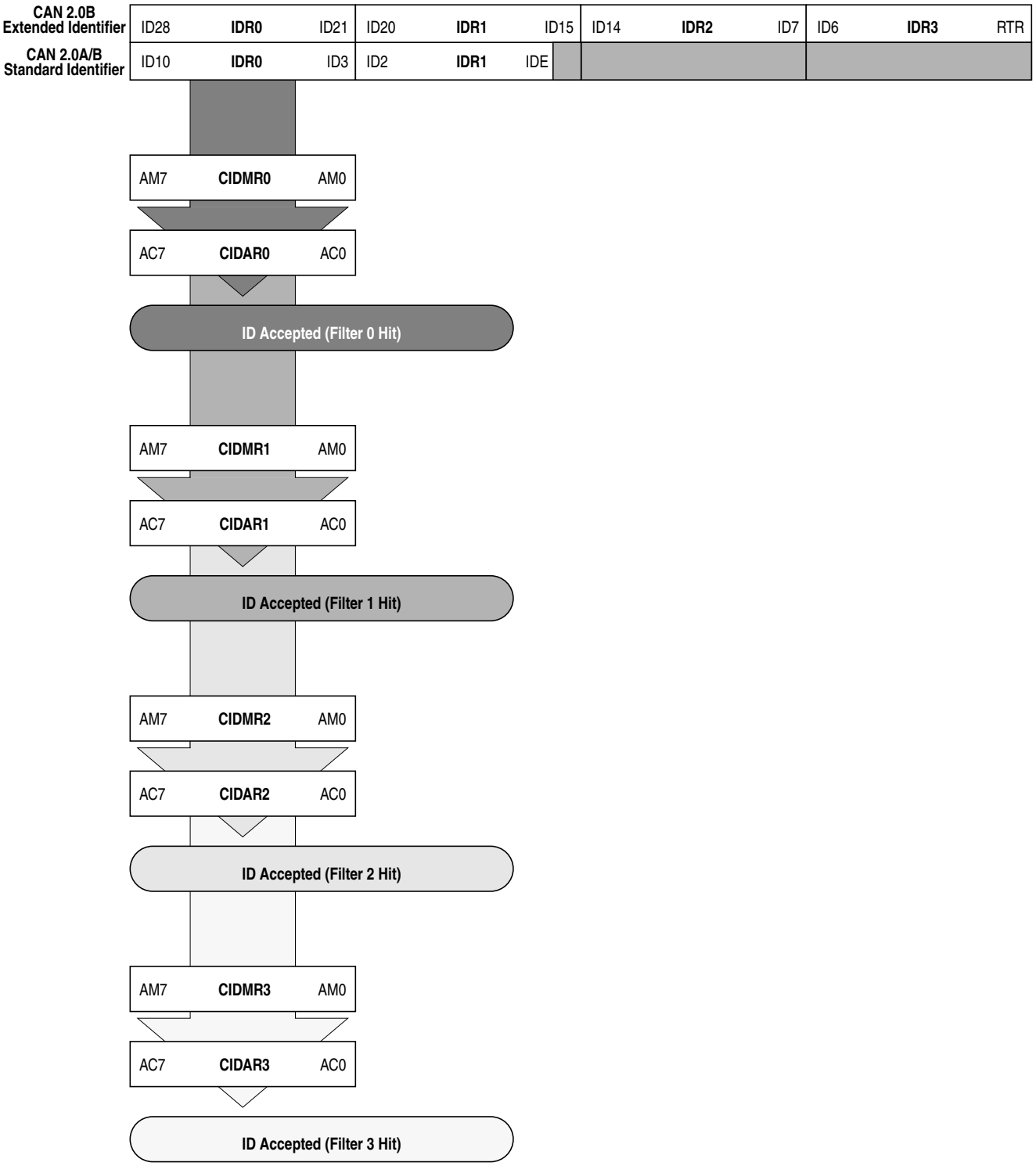


Figure 21-41. 8-bit Maskable Identifier Acceptance Filters

21.4.3.1 Protocol Violation Protection

The MSCAN protects the user from accidentally violating the CAN protocol through programming errors. The protection logic implements the following features:

- The receive and transmit error counters cannot be written or otherwise manipulated.
- All registers which control the configuration of the MSCAN cannot be modified while the MSCAN is on-line. The MSCAN has to be in Initialization Mode. The corresponding INITRQ/INITAK handshake bits in the CANCTL0/CANCTL1 registers (see Section 21.3.2.1, "MSCAN Control Register 0 (CANCTL0)") serve as a lock to protect the following registers:
  - MSCAN control 1 register (CANCTL1)
  - MSCAN bus timing registers 0 and 1 (CANBTR0, CANBTR1)
  - MSCAN identifier acceptance control register (CANIDAC)
  - MSCAN identifier acceptance registers (CANIDAR0–CANIDAR7)
  - MSCAN identifier mask registers (CANIDMR0–CANIDMR7)
- The TXCAN is immediately forced to a recessive state when the MSCAN goes into the power down mode or initialization mode (see Section 21.4.5.6, "MSCAN Power Down Mode," and Section 21.4.4.5, "MSCAN Initialization Mode").
- The MSCAN enable bit (CANE) is writable only once in normal system operation modes, which provides further protection against inadvertently disabling the MSCAN.

21.4.3.2 Clock System

Figure 21-42 shows the structure of the MSCAN clock generation circuitry.

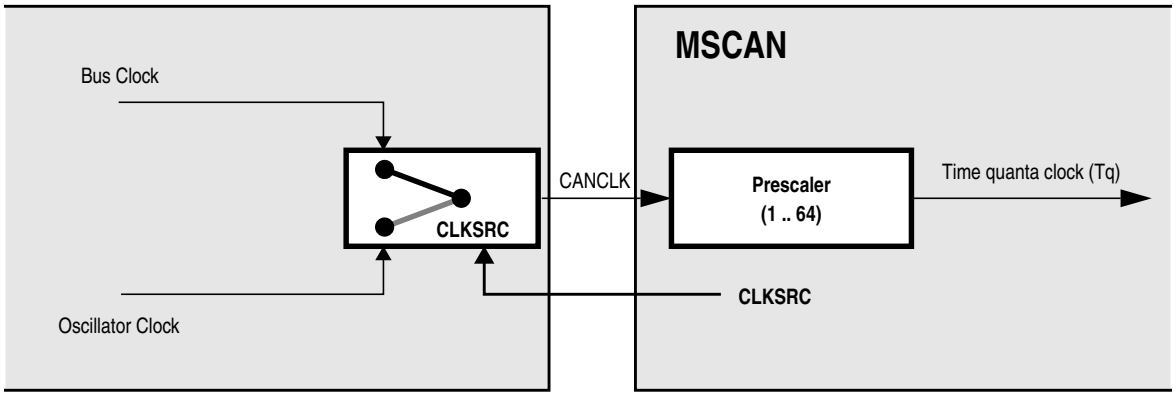


Figure 21-42. MSCAN Clocking Scheme

The clock source bit (CLKSRC) in the CANCTL1 register (21.3.2.2/21-965) defines whether the internal CANCLK is connected to the output of a crystal oscillator (oscillator clock) or to the bus clock.

The clock source has to be chosen such that the tight oscillator tolerance requirements (up to 0.4%) of the CAN protocol are met. Additionally, for high CAN bus rates (1 Mbps), a 45% to 55% duty cycle of the clock is required.

If the bus clock is generated from a PLL, it is recommended to select the oscillator clock rather than the bus clock due to jitter considerations, especially at the faster CAN bus rates.

For microcontrollers without a clock and reset generator (CRG), CANCLK is driven from the crystal oscillator (oscillator clock).

A programmable prescaler generates the time quanta (Tq) clock from CANCLK. A time quantum is the atomic unit of time handled by the MSCAN.

Eqn. 21-2

$$Tq = \frac{f_{CANCLK}}{(\text{Prescaler value})}$$

A bit time is subdivided into three segments as described in the Bosch CAN specification. (see Figure 21-43):

- SYNC_SEG: This segment has a fixed length of one time quantum. Signal edges are expected to happen within this section.
- Time Segment 1: This segment includes the PROP_SEG and the PHASE_SEG1 of the CAN standard. It can be programmed by setting the parameter TSEG1 to consist of 4 to 16 time quanta.
- Time Segment 2: This segment represents the PHASE_SEG2 of the CAN standard. It can be programmed by setting the TSEG2 parameter to be 2 to 8 time quanta long.

Eqn. 21-3

$$\text{Bit Rate} = \frac{f_{Tq}}{(\text{number of Time Quanta})}$$

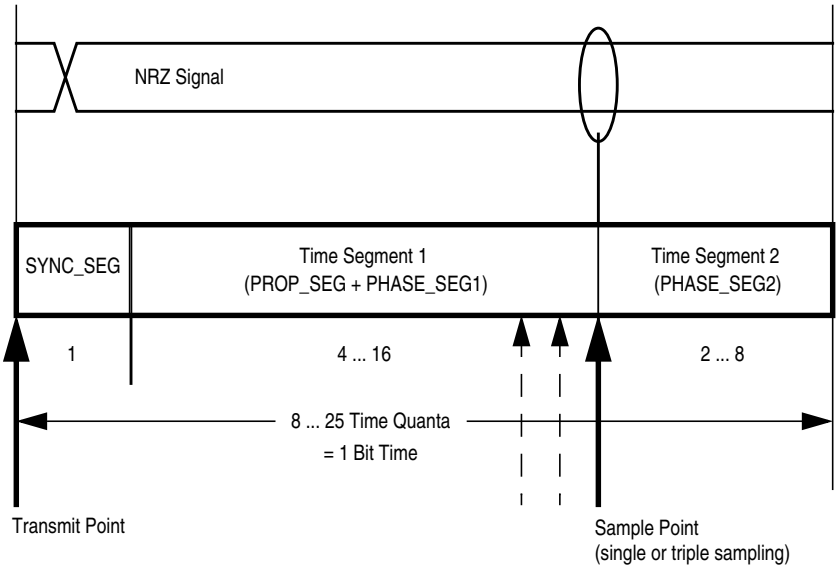


Figure 21-43. Segments within the Bit Time

Table 21-35. Time Segment Syntax

Syntax	Description
SYNC_SEG	System expects transitions to occur on the CAN bus during this period.
Transmit Point	A node in transmit mode transfers a new value to the CAN bus at this point.
Sample Point	A node in receive mode samples the CAN bus at this point. If the three samples per bit option is selected, then this point marks the position of the third sample.

The synchronization jump width (see the Bosch CAN specification for details) can be programmed in a range of 1 to 4 time quanta by setting the SJW parameter.

The SYNC_SEG, TSEG1, TSEG2, and SJW parameters are set by programming the MSCAN bus timing registers (CANBTR0, CANBTR1) (see Section 21.3.2.3, "MSCAN Bus Timing Register 0 (CANBTR0)" and Section 21.3.2.4, "MSCAN Bus Timing Register 1 (CANBTR1)").

Table 21-36 gives an overview of the CAN compliant segment settings and the related parameter values.

NOTE

It is the user's responsibility to ensure the bit time settings are in compliance with the CAN standard.

Table 21-36. CAN Standard Compliant Bit Time Segment Settings

Time Segment 1	TSEG1	Time Segment 2	TSEG2	Synchronization Jump Width	SJW
5 .. 10	4 .. 9	2	1	1 .. 2	0 .. 1
4 .. 11	3 .. 10	3	2	1 .. 3	0 .. 2
5 .. 12	4 .. 11	4	3	1 .. 4	0 .. 3
6 .. 13	5 .. 12	5	4	1 .. 4	0 .. 3
7 .. 14	6 .. 13	6	5	1 .. 4	0 .. 3
8 .. 15	7 .. 14	7	6	1 .. 4	0 .. 3
9 .. 16	8 .. 15	8	7	1 .. 4	0 .. 3

21.4.4 Modes of Operation

21.4.4.1 Normal System Operating Modes

The MSCAN module behaves as described within this specification in all normal system operating modes. Write restrictions exist for some registers.

21.4.4.2 Special System Operating Modes

The MSCAN module behaves as described within this specification in all special system operating modes. Write restrictions which exist on specific registers in normal modes are lifted for test purposes in special modes.

21.4.4.3 Emulation Modes

In all emulation modes, the MSCAN module behaves just like in normal system operating modes as described within this specification.

21.4.4.4 Listen-Only Mode

In an optional CAN bus monitoring mode (listen-only), the CAN node is able to receive valid data frames and valid remote frames, but it sends only "recessive" bits on the CAN bus. In addition, it cannot start a transmission.

If the MAC sub-layer is required to send a "dominant" bit (ACK bit, overload flag, or active error flag), the bit is rerouted internally so that the MAC sub-layer monitors this "dominant" bit, although the CAN bus may remain in recessive state externally.

21.4.4.5 MSCAN Initialization Mode

The MSCAN enters initialization mode when it is enabled (CANE=1).

When entering initialization mode during operation, any on-going transmission or reception is immediately aborted and synchronization to the CAN bus is lost, potentially causing CAN protocol violations. To protect the CAN bus system from fatal consequences of violations, the MSCAN immediately drives TXCAN into a recessive state.

NOTE

The user is responsible for ensuring that the MSCAN is not active when initialization mode is entered. The recommended procedure is to bring the MSCAN into sleep mode (SLPRQ = 1 and SLPAK = 1) before setting the INITRQ bit in the CANCTL0 register. Otherwise, the abort of an on-going message can cause an error condition and can impact other CAN bus devices.

In initialization mode, the MSCAN is stopped. However, interface registers remain accessible. This mode is used to reset the CANCTL0, CANRFLG, CANRIER, CANTFLG, CANTIER, CANTARQ, CANTAAK, and CANTBSEL registers to their default values. In addition, the MSCAN enables the configuration of the CANBTR0, CANBTR1 bit timing registers; CANIDAC; and the CANIDAR, CANIDMR message filters. See Section 21.3.2.1, "MSCAN Control Register 0 (CANCTL0)," for a detailed description of the initialization mode.

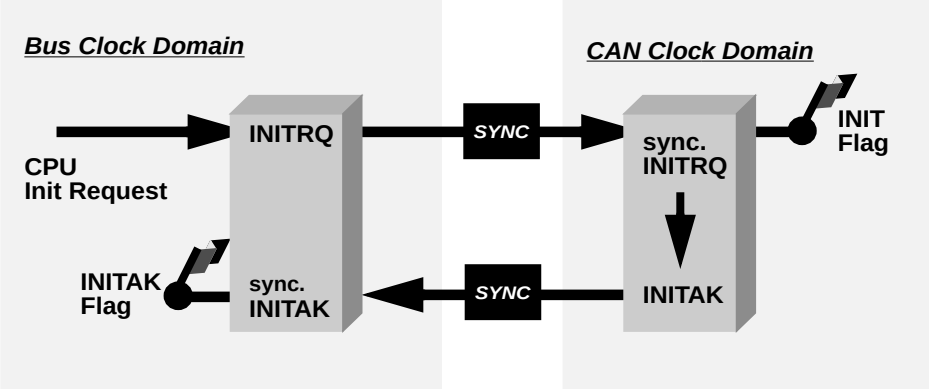


Figure 21-44. Initialization Request/Acknowledge Cycle

Due to independent clock domains within the MSCAN, INITRQ must be synchronized to all domains by using a special handshake mechanism. This handshake causes additional synchronization delay (see Figure 21-44).

If there is no message transfer ongoing on the CAN bus, the minimum delay will be two additional bus clocks and three additional CAN clocks. When all parts of the MSCAN are in initialization mode, the INITAK flag is set. The application software must use INITAK as a handshake indication for the request (INITRQ) to go into initialization mode.

NOTE

The CPU cannot clear INITRQ before initialization mode (INITRQ = 1 and INITAK = 1) is active.

21.4.5 Low-Power Options

If the MSCAN is disabled (CANE = 0), the MSCAN clocks are stopped for power saving.

If the MSCAN is enabled (CANE = 1), the MSCAN has two additional modes with reduced power consumption, compared to normal mode: sleep and power down mode. In sleep mode, power consumption is reduced by stopping all clocks except those to access the registers from the CPU side. In power down mode, all clocks are stopped and no power is consumed.

Table 21-37 summarizes the combinations of MSCAN and CPU modes. A particular combination of modes is entered by the given settings on the CSWAI and SLPRQ/SLPAK bits.

Table 21-37. CPU vs. MSCAN Operating Modes

CPU Mode	MSCAN Mode			
	Normal	Reduced Power Consumption		
		Sleep	Power Down	Disabled (CANE=0)
RUN	CSWAI = X[1] SLPRQ = 0 SLPAK = 0	CSWAI = X SLPRQ = 1 SLPAK = 1		CSWAI = X SLPRQ = X SLPAK = X
WAIT	CSWAI = 0 SLPRQ = 0 SLPAK = 0	CSWAI = 0 SLPRQ = 1 SLPAK = 1	CSWAI = 1 SLPRQ = X SLPAK = X	CSWAI = X SLPRQ = X SLPAK = X
STOP			CSWAI = X SLPRQ = X SLPAK = X	CSWAI = X SLPRQ = X SLPAK = X

1. 'X' means don't care.

21.4.5.1 Operation in Run Mode

As shown in Table 21-37, only MSCAN sleep mode is available as low power option when the CPU is in run mode.

21.4.5.2 Operation in Wait Mode

The WAI instruction puts the MCU in a low power consumption stand-by mode. If the CSWAI bit is set, additional power can be saved in power down mode because the CPU clocks are stopped. After leaving this power down mode, the MSCAN restarts and enters normal mode again.

While the CPU is in wait mode, the MSCAN can be operated in normal mode and generate interrupts (registers can be accessed via background debug mode).

21.4.5.3 Operation in Stop Mode

The STOP instruction puts the MCU in a low power consumption stand-by mode. In stop mode, the MSCAN is set in power down mode regardless of the value of the SLPRQ/SLPAK and CSWAI bits (Table 21-37).

21.4.5.4 MSCAN Normal Mode

This is a non-power-saving mode. Enabling the MSCAN puts the module from disabled mode into normal mode. In this mode the module can either be in initialization mode or out of initialization mode. See Section 21.4.4.5, "MSCAN Initialization Mode".

21.4.5.5 MSCAN Sleep Mode

The CPU can request the MSCAN to enter this low power mode by asserting the SLPRQ bit in the CANCTL0 register. The time when the MSCAN enters sleep mode depends on a fixed synchronization delay and its current activity:

- If there are one or more message buffers scheduled for transmission (TXEx = 0), the MSCAN will continue to transmit until all transmit message buffers are empty (TXEx = 1, transmitted successfully or aborted) and then goes into sleep mode.
- If the MSCAN is receiving, it continues to receive and goes into sleep mode as soon as the CAN bus next becomes idle.
- If the MSCAN is neither transmitting nor receiving, it immediately goes into sleep mode.

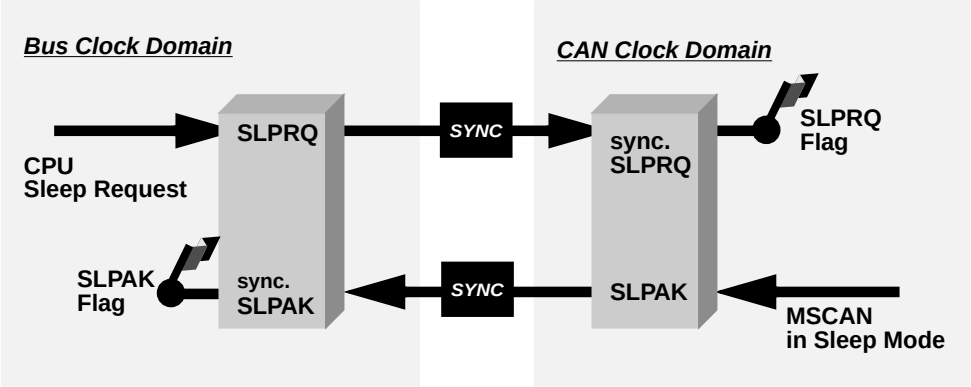


Figure 21-45. Sleep Request / Acknowledge Cycle

NOTE

The application software must avoid setting up a transmission (by clearing one or more TXEx flag(s)) and immediately request sleep mode (by setting SLPRQ). Whether the MSCAN starts transmitting or goes into sleep mode directly depends on the exact sequence of operations.

If sleep mode is active, the SLPRQ and SLPAK bits are set (Figure 21-45). The application software must use SLPAK as a handshake indication for the request (SLPRQ) to go into sleep mode.

When in sleep mode (SLPRQ = 1 and SLPAK = 1), the MSCAN stops its internal clocks. However, clocks that allow register accesses from the CPU side continue to run.

If the MSCAN is in bus-off state, it stops counting the 128 occurrences of 11 consecutive recessive bits due to the stopped clocks. TXCAN remains in a recessive state. If RXF = 1, the message can be read and RXF can be cleared. Shifting a new message into the foreground buffer of the receiver FIFO (RxFG) does not take place while in sleep mode.

It is possible to access the transmit buffers and to clear the associated TXE flags. No message abort takes place while in sleep mode.

If the WUPE bit in CANCTL0 is not asserted, the MSCAN will mask any activity it detects on CAN. RXCAN is therefore held internally in a recessive state. This locks the MSCAN in sleep mode. WUPE must be set before entering sleep mode to take effect.

The MSCAN is able to leave sleep mode (wake up) only when:

- CAN bus activity occurs and WUPE = 1

  or
- the CPU clears the SLPRQ bit

NOTE

The CPU cannot clear the SLPRQ bit before sleep mode (SLPRQ = 1 and SLPAK = 1) is active.

After wake-up, the MSCAN waits for 11 consecutive recessive bits to synchronize to the CAN bus. As a consequence, if the MSCAN is woken-up by a CAN frame, this frame is not received.

The receive message buffers (RxFG and RxBG) contain messages if they were received before sleep mode was entered. All pending actions will be executed upon wake-up; copying of RxBG into RxFG, message aborts and message transmissions. If the MSCAN remains in bus-off state after sleep mode was exited, it continues counting the 128 occurrences of 11 consecutive recessive bits.

21.4.5.6 MSCAN Power Down Mode

The MSCAN is in power down mode (Table 21-37) when

- CPU is in stop mode

  or
- CPU is in wait mode and the CSWAI bit is set

When entering the power down mode, the MSCAN immediately stops all ongoing transmissions and receptions, potentially causing CAN protocol violations. To protect the CAN bus system from fatal consequences of violations to the above rule, the MSCAN immediately drives TXCAN into a recessive state.

NOTE

The user is responsible for ensuring that the MSCAN is not active when power down mode is entered. The recommended procedure is to bring the MSCAN into Sleep mode before the STOP or WAI instruction (if CSWAI is set) is executed. Otherwise, the abort of an ongoing message can cause an error condition and impact other CAN bus devices.

In power down mode, all clocks are stopped and no registers can be accessed. If the MSCAN was not in sleep mode before power down mode became active, the module performs an internal recovery cycle after powering up. This causes some fixed delay before the module enters normal mode again.

21.4.5.7 Disabled Mode

The MSCAN is in disabled mode out of reset (CANE=0). All module clocks are stopped for power saving, however the register map can still be accessed as specified.

21.4.5.8 Programmable Wake-Up Function

The MSCAN can be programmed to wake up from sleep or power down mode as soon as CAN bus activity is detected (see control bit WUPE in MSCAN Control Register 0 (CANCTL0). The sensitivity to existing CAN bus action can be modified by applying a low-pass filter function to the RXCAN input line (see control bit WUPM in Section 21.3.2.2, "MSCAN Control Register 1 (CANCTL1)").

This feature can be used to protect the MSCAN from wake-up due to short glitches on the CAN bus lines. Such glitches can result from—for example—electromagnetic interference within noisy environments.

21.4.6 Reset Initialization

The reset state of each individual bit is listed in Section 21.3.2, "Register Descriptions," which details all the registers and their bit-fields.

21.4.7 Interrupts

This section describes all interrupts originated by the MSCAN. It documents the enable bits and generated flags. Each interrupt is listed and described separately.

21.4.7.1 Description of Interrupt Operation

The MSCAN supports four interrupt vectors (see Table 21-38), any of which can be individually masked (for details see Section 21.3.2.6, "MSCAN Receiver Interrupt Enable Register (CANRIER)" to Section 21.3.2.8, "MSCAN Transmitter Interrupt Enable Register (CANTIER)").

Refer to the device overview section to determine the dedicated interrupt vector addresses.

Table 21-38. Interrupt Vectors

Interrupt Source	CCR Mask	Local Enable
Wake-Up Interrupt (WUPIF)	I bit	CANRIER (WUPIE)
Error Interrupts Interrupt (CSCIF, OVRIF)	I bit	CANRIER (CSCIE, OVRIE)
Receive Interrupt (RXF)	I bit	CANRIER (RXFIE)
Transmit Interrupts (TXE[2:0])	I bit	CANTIER (TXEIE[2:0])

21.4.7.2 Transmit Interrupt

At least one of the three transmit buffers is empty (not scheduled) and can be loaded to schedule a message for transmission. The TXEx flag of the empty message buffer is set.

21.4.7.3 Receive Interrupt

A message is successfully received and shifted into the foreground buffer (RxFG) of the receiver FIFO. This interrupt is generated immediately after receiving the EOF symbol. The RXF flag is set. If there are multiple messages in the receiver FIFO, the RXF flag is set as soon as the next message is shifted to the foreground buffer.

21.4.7.4 Wake-Up Interrupt

A wake-up interrupt is generated if activity on the CAN bus occurs during MSCAN sleep or power-down mode.

NOTE

This interrupt can only occur if the MSCAN was in sleep mode (SLPRQ = 1 and SLPAK = 1) before entering power down mode, the wake-up option is enabled (WUPE = 1), and the wake-up interrupt is enabled (WUPIE = 1).

21.4.7.5 Error Interrupt

An error interrupt is generated if an overrun of the receiver FIFO, error, warning, or bus-off condition occurrs. MSCAN Receiver Flag Register (CANRFLG) indicates one of the following conditions:

- **Overrun** — An overrun condition of the receiver FIFO as described in Section 21.4.2.3, "Receive Structures," occurred.
- **CAN Status Change** — The actual value of the transmit and receive error counters control the CAN bus state of the MSCAN. As soon as the error counters skip into a critical range (Tx/Rx-warning, Tx/Rx-error, bus-off) the MSCAN flags an error condition. The status change, which caused the error condition, is indicated by the TSTAT and RSTAT flags (see Section 21.3.2.5, "MSCAN Receiver Flag Register (CANRFLG)" and Section 21.3.2.6, "MSCAN Receiver Interrupt Enable Register (CANRIER)").

21.4.7.6 Interrupt Acknowledge

Interrupts are directly associated with one or more status flags in either the MSCAN Receiver Flag Register (CANRFLG) or the MSCAN Transmitter Flag Register (CANTFLG). Interrupts are pending as long as one of the corresponding flags is set. The flags in CANRFLG and CANTFLG must be reset within the interrupt handler to handshake the interrupt. The flags are reset by writing a 1 to the corresponding bit position. A flag cannot be cleared if the respective condition prevails.

NOTE

It must be guaranteed that the CPU clears only the bit causing the current interrupt. For this reason, bit manipulation instructions (BSET) must not be used to clear interrupt flags. These instructions may cause accidental clearing of interrupt flags which are set after entering the current interrupt service routine.

21.5 Initialization/Application Information

21.5.1 MSCAN initialization

The procedure to initially start up the MSCAN module out of reset is as follows:

1. Assert CANE
2. Write to the configuration registers in initialization mode
3. Clear INITRQ to leave initialization mode

If the configuration of registers which are only writable in initialization mode shall be changed:

1. Bring the module into sleep mode by setting SLPRQ and awaiting SLPAK to assert after the CAN bus becomes idle.
2. Enter initialization mode: assert INITRQ and await INITAK
3. Write to the configuration registers in initialization mode
4. Clear INITRQ to leave initialization mode and continue

Chapter 22
Enhanced Programmable Interrupt Timer (S12XEPIT24B8CV1)

22.1 Revision History

Version Number	Revision Date	Effective Date	Author	Description of Changes
00.01	18-Aug-06	18-Aug-06		Initial Draft, based on PIT V01.01
00.02	21-Aug-06	21-Aug-06		- reworked channel restart register “PITCSTP”, reordered new registers
00.03	15-Sep-06	20-Sep-06		- added PITTRIGEDGE config bits to PITTRIGCTL register - clarified channels in external trigger mode are bypassing the microtimers, thus running at bus-speed - cleaned up register description based on feedback by Joachim - renamed PITTRIGIN register to PITTRIGE (PIT trigger channel enable register) - added timing diagram for externally triggered mode - updated block diagram in Introduction section (block diagram in Functional Description section is not complete)
00.04	13-Oct-06	13-Oct-06		- renamed PITTRIGOUT register bits: PITTRIGO -> PITCOTEn - fixed register offsets in register descriptions (AS) - clarified PITCSTP register function (AS, JK) - added NOTE to disable channel before changing the time-base for the channel by writing to PITMUX or PITTRIGE (AS) - removed wrong visible borders from PITLDn and PITCNTn register description (JK)

22.2 Introduction

The enhanced period interrupt timer (EPIT) is an array of 24-bit timers that can be used to trigger peripheral modules or raise periodic interrupts. Refer to Figure 22-1 for a simplified block diagram.

22.2.1 Glossary

Acronyms and Abbreviations	
EPIT	Enhanced Programmable Interrupt Timer
ISR	Interrupt Service Routine
CCR	Condition Code Register
SoC	System on Chip

Acronyms and Abbreviations	
micro time bases	clock periods of the 16-bit timer modulus down-counters, which are generated by the 8-bit modulus down-counters.

22.2.2 Features

The EPIT includes these features:

- Eight timers implemented as modulus down-counters with independent time-out periods.
- Time-out periods selectable between 1 and 2^{24} bus clock cycles. Time-out equals m*n bus clock cycles with 1 <= m <= 256 and 1 <= n <= 65536.
- Timers that can be enabled individually.
- Eight time-out interrupts.
- Eight time-out trigger output signals available to trigger peripheral modules.
- Start of timer channels can be aligned to each other.
- Start of timer channels can be aligned to an external trigger event.

22.2.3 Modes of Operation

Refer to the Device Reference Manual for a detailed explanation of the chip modes.

- Run mode

  This is the basic mode of operation.
- Wait mode

  EPIT operation in wait mode is controlled by the PITSWAI bit located in the PITCFLMT register. In wait mode, if the bus clock is globally enabled and if the PITSWAI bit is clear, the EPIT operates like in run mode. In wait mode, if the PITSWAI bit is set, the EPIT module is stalled.
- Stop mode

  In full stop mode or pseudo stop mode, the EPIT module is stalled.
- Freeze mode

  EPIT operation in freeze mode is controlled by the PITFRZ bit located in the PITCFLMT register. In freeze mode, if the PITFRZ bit is clear, the EPIT operates like in run mode. In freeze mode, if the PITFRZ bit is set, the EPIT module is stalled.

22.2.4 Block Diagram

Figure 22-1 shows a block diagram of the EPIT.

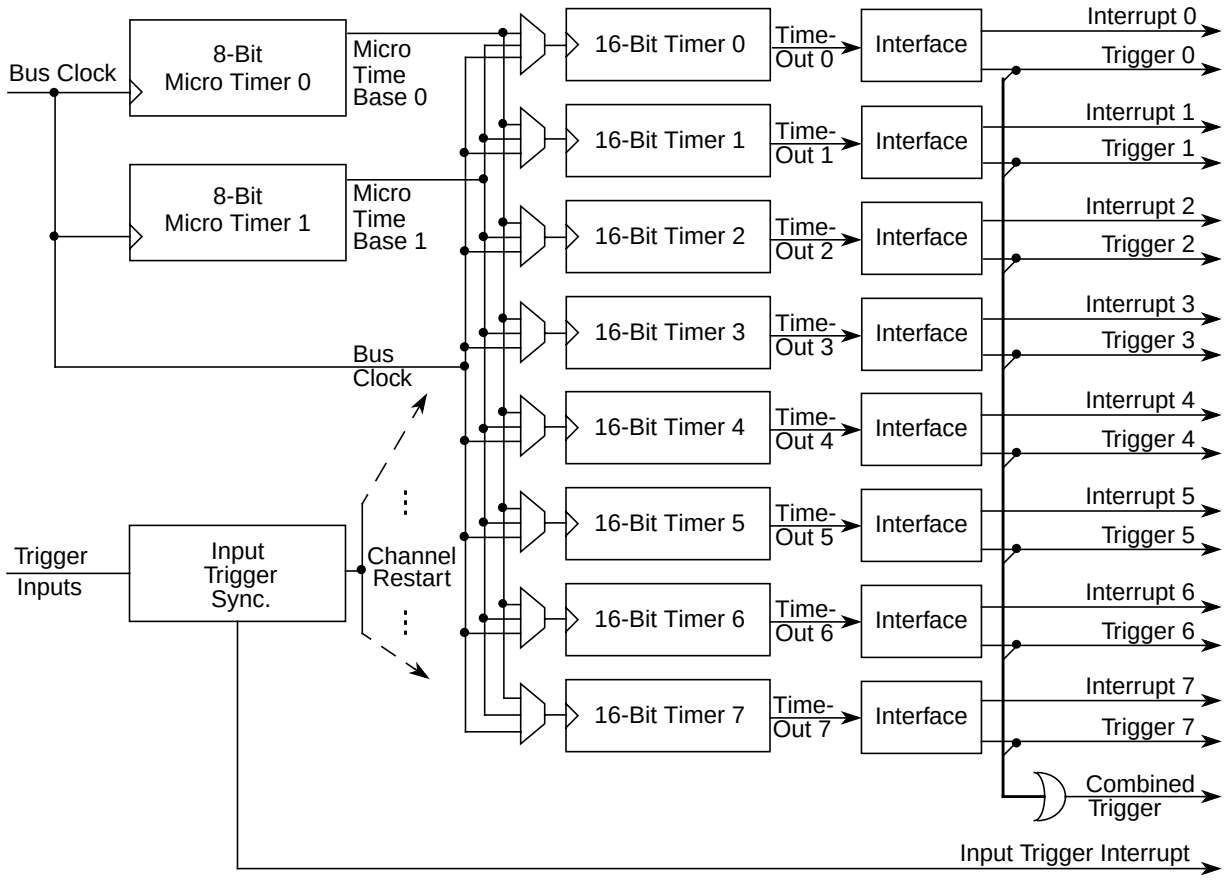


Figure 22-1. EPIT Block Diagram

22.3 External Signal Description

The EPIT module has no external pins.

22.4 Register Definition

This section consists of register descriptions in address order of the EPIT. Each description includes a standard register diagram with an associated figure number. Details of register bit and field function follow the register diagrams, in bit order.

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000 PITCFLMT	R	PITE	PITSWAI	PITFRZ	0	0	0	0	0
	W							PITFLMT1	PITFLMT0

= Unimplemented or Reserved

Figure 22-2. EPIT Register Summary (Sheet 1 of 4)

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0001 PITFLT	R	0	0	0	0	0	0	0	0
	W	PITFLT7	PITFLT6	PITFLT5	PITFLT4	PITFLT3	PITFLT2	PITFLT1	PITFLT0
0x0002 PITCE	R W	PITCE7	PITCE6	PITCE5	PITCE4	PITCE3	PITCE2	PITCE1	PITCE0
0x0003 PITMUX	R W	PITMUX7	PITMUX6	PITMUX5	PITMUX4	PITMUX3	PITMUX2	PITMUX1	PITMUX0
0x0004 PITINTE	R W	PITINTE7	PITINTE6	PITINTE5	PITINTE4	PITINTE3	PITINTE2	PITINTE1	PITINTE0
0x0005 PITTF	R W	PITTF7	PITTF6	PITTF5	PITTF4	PITTF3	PITTF2	PITTF1	PITTF0
0x0006 PITMTLD0	R W	PITMTLD0[7:0]							
0x0007 PITMTLD1	R W	PITMTLD1[7:0]							
0x0008 PITLD0 (High)	R W	PITLD0[15:8]							
0x0009 PITLD0 (Low)	R W	PITLD0[7:0]							
0x000A PITCNT0 (High)	R	PITCNT0[15:8]							
	W								
0x000B PITCNT0 (Low)	R	PITCNT0[7:0]							
	W								
0x000C PITLD1 (High)	R W	PITLD1[15:8]							
0x000D PITLD1 (Low)	R W	PITLD1[7:0]							
0x000E PITCNT1 (High)	R	PITCNT1[15:8]							
	W								
0x000F PITCNT1 (Low)	R	PITCNT1[7:0]							
	W								

= Unimplemented or Reserved

Figure 22-2. EPIT Register Summary (Sheet 2 of 4)

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0010 PITLD2 (High)	R	PITLD2[15:8]							
	W								
0x0011 PITLD2 (Low)	R	PITLD2[7:0]							
	W								
0x0012 PITCNT2 (High)	R	PITCNT2[15:8]							
	W								
0x0013 PITCNT2 (Low)	R	PITCNT2[7:0]							
	W								
0x0014 PITLD3 (High)	R	PITLD3[15:8]							
	W								
0x0015 PITLD3 (Low)	R	PITLD3[7:0]							
	W								
0x0016 PITCNT3 (High)	R	PITCNT3[15:8]							
	W								
0x0017 PITCNT3 (Low)	R	PITCNT3[7:0]							
	W								
0x0018 PITLD4 (High)	R	PITLD4[15:8]							
	W								
0x0019 PITLD4 (Low)	R	PITLD4[7:0]							
	W								
0x001A PITCNT4 (High)	R	PITCNT4[15:8]							
	W								
0x001B PITCNT4 (Low)	R	PITCNT4[7:0]							
	W								
0x001C PITLD5 (High)	R	PITLD5[15:8]							
	W								
0x001D PITLD5 (Low)	R	PITLD5[7:0]							
	W								
0x001E PITCNT5 (High)	R	PITCNT5[15:8]							
	W								

= Unimplemented or Reserved

Figure 22-2. EPIT Register Summary (Sheet 3 of 4)

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x001F PITCNT5 (Low)	R	PITCNT5[7:0]							
	W								
0x0020 PITLD6 (High)	R	PITLD6[15:8]							
	W								
0x0021 PITLD6 (Low)	R	PITLD6[7:0]							
	W								
0x0022 PITCNT6 (High)	R	PITCNT6[15:8]							
	W								
0x0023 PITCNT6 (Low)	R	PITCNT6[7:0]							
	W								
0x0024 PITLD7 (High)	R	PITLD7[15:8]							
	W								
0x0025 PITLD7 (Low)	R	PITLD7[7:0]							
	W								
0x0026 PITCNT7 (High)	R	PITCNT7[15:8]							
	W								
0x0027 PITCNT7 (Low)	R	PITCNT7[7:0]							
	W								
0x0028 PITCSTP	R	PITCSTP7	PITCSTP6	PITCSTP5	PITCSTP4	PITCSTP3	PITCSTP2	PITCSTP1	PITCSTP0
	W								
0x0029 PITTRIGOUT	R	PITCOTE7	PITCOTE6	PITCOTE5	PITCOTE4	PITCOTE3	PITCOTE2	PITCOTE1	PITCOTE0
	W								
0x002A PITTRIGCTL	R	PITTRIGIE	0	PITTRIGEDGE[1:0]		0	0	PITTRIGSRC[1:0]	
	W								
0x002B PITTRIGSTAT	R	PITTRIGIF	0	0	0	0	0	0	0
	W								
0x002C PITTRIGE	R	PITTRIGE7	PITTRIGE6	PITTRIGE5	PITTRICE4	PITTRIGE3	PITTRIGE2	PITTRIGE1	PITTRIGE0
	W								

= Unimplemented or Reserved

Figure 22-2. EPIT Register Summary (Sheet 4 of 4)

22.4.0.1 PIT Control and Force Load Micro Timer Register (PITCFLMT)

Module Base + 0x0000

	7	6	5	4	3	2	1	0
R	PITE	PITSWAI	PITFRZ	0	0	0	0	0
W							PITFLMT1	PITFLMT0
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 22-3. PIT Control and Force Load Micro Timer Register (PITCFLMT)

Read: Anytime

Write: Anytime; writes to the reserved bits have no effect

Table 22-1. PITCFLMT Field Descriptions

Field	Description
7 PITE	**PIT Module Enable Bit** — This bit enables the PIT module. If PITE is cleared, the PIT module is disabled and flag bits in the PITTF register are cleared. When PITE is set, individually enabled timers (PCE set) start down-counting with the corresponding load register values. 0 PIT disabled (lower power consumption). 1 PIT is enabled.
6 PITSWAI	**PIT Stop in Wait Mode Bit** — This bit is used for power conservation while in wait mode. 0 PIT operates normally in wait mode 1 PIT clock generation stops and freezes the PIT module when in wait mode
5 PITFRZ	**PIT Counter Freeze while in Freeze Mode Bit** — When during debugging a breakpoint (freeze mode) is encountered it is useful in many cases to freeze the PIT counters to avoid e.g. interrupt generation. The PITFRZ bit controls the PIT operation while in freeze mode. 0 PIT operates normally in freeze mode 1 PIT counters are stalled when in freeze mode
1:0 PITFLMT [1:0]	**PIT Force Load Bits for Micro Timer 1:0** — These bits have only an effect if the corresponding micro timer is active and if the PIT module is enabled (PITE set). Writing a one into a PITFLMT bit loads the corresponding 8-bit micro timer load register into the 8-bit micro timer down-counter. Writing a zero has no effect. Reading these bits will always return zero. **Note:** A micro timer force load affects all timer channels that use the corresponding micro time base.

22.4.0.2 PIT Force Load Timer Register (PITFLT)

Module Base + 0x0001

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W	PITFLT7	PITFLT6	PITFLT5	PITFLT4	PITFLT3	PITFLT2	PITFLT1	PITFLT0
Reset	0	0	0	0	0	0	0	0

Figure 22-4. PIT Force Load Timer Register (PITFLT)

Read: Anytime

Write: Anytime

Table 22-2. PITFLT Field Descriptions

Field	Description
7:0 PITFLT[7:0]	**PIT Force Load Bits for Timer 7-0** — These bits have only an effect if the corresponding timer channel (PCE set) is enabled and if the PIT module is enabled (PITE set). Writing a one into a PITFLT bit loads the corresponding 16-bit timer load register into the 16-bit timer down-counter. Additionally, the corresponding PIT Channel Stop Bit (PTCSTP register) is loaded. Force load also affects PIT channels which are configured to externally triggered mode (see PITTRIGE register). Writing a zero has no effect. Reading this register returns zero.

22.4.0.3 PIT Channel Enable Register (PITCE)

Module Base + 0x0002

	7	6	5	4	3	2	1	0
R W	PITCE7	PITCE6	PITCE5	PITCE4	PITCE3	PITCE2	PITCE1	PITCE0
Reset	0	0	0	0	0	0	0	0

Figure 22-5. PIT Channel Enable Register (PITCE)

Read: Anytime

Write: Anytime

Table 22-3. PITCE Field Descriptions

Field	Description
7:0 PITCE[7:0]	**PIT Enable Bits for Timer Channel 7:0** — These bits enable the PIT channels 7-0. If PITCE is cleared, the PIT channel is disabled and the corresponding flag bit in the PITTF register is cleared. When PITCE is set, and if the PIT module is enabled (PITE = 1) the 16-bit timer counter is loaded with the start count value and starts down-counting. 0 The corresponding PIT channel is disabled. 1 The corresponding PIT channel is enabled.

22.4.0.4 PIT Multiplex Register (PITMUX)

Module Base + 0x0003

	7	6	5	4	3	2	1	0
R W	PITMUX7	PITMUX6	PITMUX5	PITMUX4	PITMUX3	PITMUX2	PITMUX1	PITMUX0
Reset	0	0	0	0	0	0	0	0

Figure 22-6. PIT Multiplex Register (PITMUX)

Read: Anytime

Write: Anytime

Table 22-4. PITMUX Field Descriptions

Field	Description
7:0 PITMUX[7:0]	**PIT Multiplex Bits for Timer Channel 7:0** — These bits select if the corresponding 16-bit timer is connected to micro time base 1 or 0. If PITMUX is modified, the corresponding 16-bit timer is immediately switched to the other micro time base. If a PIT channel is in external trigger mode (i.e. the corresponding bit in the PITTRIGE register is set), the corresponding PITMUX bit for this channel is ignored. The time base for a PIT channel in external trigger mode is the bus clock. 0 The corresponding 16-bit timer counts with micro time base 0. 1 The corresponding 16-bit timer counts with micro time base 1.

22.4.0.5 PIT Interrupt Enable Register (PITINTE)

Module Base + 0x0004

	7	6	5	4	3	2	1	0
R W	PITINTE7	PITINTE6	PITINTE5	PITINTE4	PITINTE3	PITINTE2	PITINTE1	PITINTE0
Reset	0	0	0	0	0	0	0	0

Figure 22-7. PIT Interrupt Enable Register (PITINTE)

Read: Anytime

Write: Anytime

Table 22-5. PITINTE Field Descriptions

Field	Description
7:0 PITINTE [7:0]	**PIT Time-out Interrupt Enable Bits for Timer Channel 7:0** — These bits enable an interrupt service request whenever the time-out flag PITTF of the corresponding PIT channel is set. When an interrupt is pending (PITTF set) setting this bit will immediately cause an interrupt. To avoid this, the corresponding PITTF flag has to be cleared first. 0 Interrupt of the corresponding PIT channel is disabled. 1 Interrupt of the corresponding PIT channel is enabled.

22.4.0.6 PIT Time-Out Flag Register (PITTF)

Module Base + 0x0005

	7	6	5	4	3	2	1	0
R W	PITTF7	PITTF6	PITTF5	PITTF4	PITTF3	PITTF2	PITTF1	PITTF0
Reset	0	0	0	0	0	0	0	0

Figure 22-8. PIT Time-Out Flag Register (PITTF)

Read: Anytime

Write: Anytime (write one to clear)

Table 22-6. PITTF Field Descriptions

Field	Description
7:0 PITTF[7:0]	**PIT Time-out Flag Bits for Timer Channel 7:0** — PITTF is set when the corresponding 16-bit timer modulus down-counter and the selected 8-bit micro timer modulus down-counter have counted to zero. The flag can be cleared by writing a one to the flag bit. Writing a zero has no effect. If flag clearing by writing a one and flag setting happen in the same bus clock cycle, the flag remains set. The flag bits are cleared if the PIT module is disabled or if the corresponding timer channel is disabled. 0 Time-out of the corresponding PIT channel has not yet occurred. 1 Time-out of the corresponding PIT channel has occurred.

22.4.0.7 PIT Micro Timer Load Register 0 to 1 (PITMTLD0–1)

Module Base + 0x0006

	7	6	5	4	3	2	1	0
R W	PITMTLD0[7:0]							
Reset	0	0	0	0	0	0	0	0

Figure 22-9. PIT Micro Timer Load Register 0 (PITMTLD0)

Module Base + 0x0007

	7	6	5	4	3	2	1	0
R W	PITMTLD1[7:0]							
Reset	0	0	0	0	0	0	0	0

Figure 22-10. PIT Micro Timer Load Register 1 (PITMTLD1)

Read: Anytime

Write: Anytime

Table 22-7. PITMTLD0–1 Field Descriptions

Field	Description
7:0 PITMTLDn [7:0]	**PIT Micro Timer Load Bits 7:0** — These bits set the 8-bit modulus down-counter load value of the micro timers. Writing a new value into the PITMTLD register will not restart the timer. When the micro timer has counted down to zero, the PMTLD register value will be loaded. The PITFLMT bits in the PITCFLMT register can be used to immediately update the count register with the new value if an immediate load is desired.

22.4.0.8 PIT Load Register 0 to 7 (PITLD0–7)

Module Base + 0x0008, 0x0009

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PITLD0[15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 22-11. PIT Load Register 0 (PITLD0)

Module Base + 0x000C, 0x000D

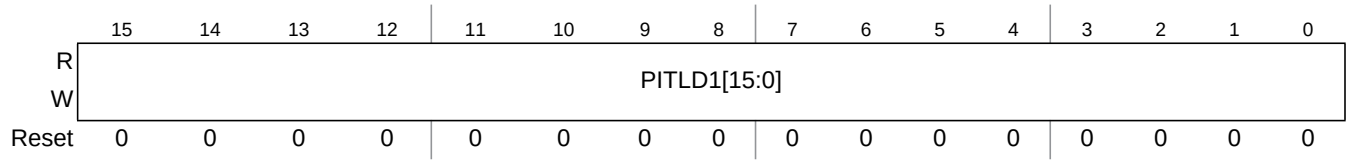


Figure 22-12. PIT Load Register 1 (PITLD1)

Module Base + 0x0010, 0x0011

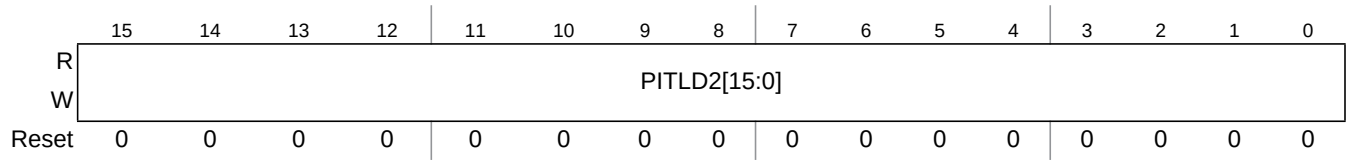


Figure 22-13. PIT Load Register 2 (PITLD2)

Module Base + 0x0014, 0x0015

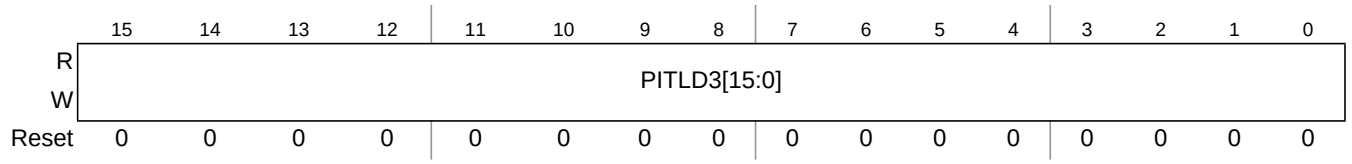


Figure 22-14. PIT Load Register 3 (PITLD3)

Module Base + 0x0018, 0x0019

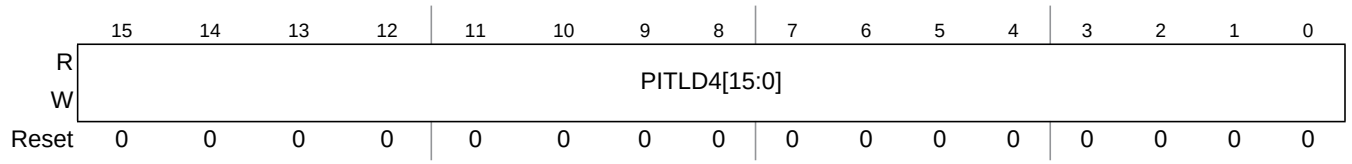


Figure 22-15. PIT Load Register 4 (PITLD4)

Module Base + 0x001C, 0x001D

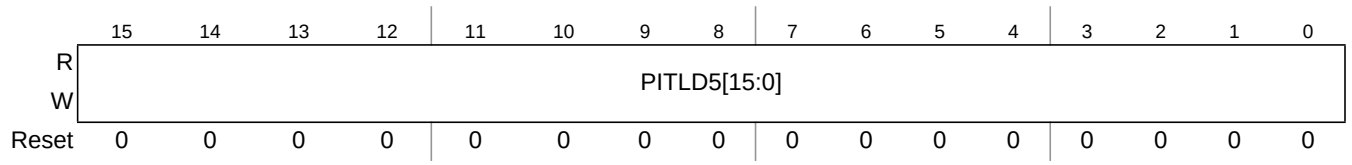


Figure 22-16. PIT Load Register 5 (PITLD5)

Module Base + 0x0020, 0x0021

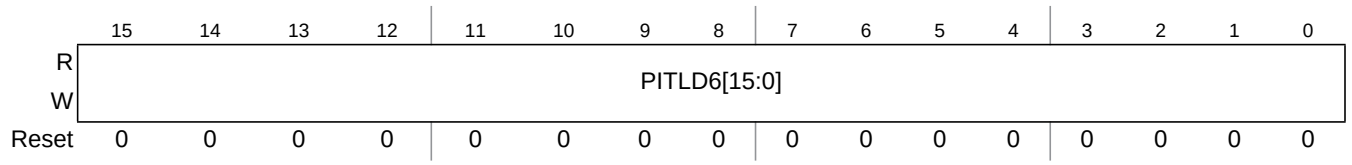


Figure 22-17. PIT Load Register 6 (PITLD6)

Module Base + 0x0024, 0x0025

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	PITLD7[15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 22-18. PIT Load Register 7 (PITLD7)

Read: Anytime

Write: Anytime

Table 22-8. PITLD0–7 Field Descriptions

Field	Description
15:0 PITLDn [15:0]	**PIT Load Bits 15:0** — These bits set the 16-bit modulus down-counter load value. Writing a new value into the PITLD register must be a 16-bit access, to ensure data consistency. It will not restart the timer. When the timer has counted down to zero the PITTF time-out flag will be set and the register value will be loaded. The PITFLT bits in the PITFLT register can be used to immediately update the count register with the new value if an immediate load is desired.

22.4.0.9 PIT Count Register 0 to 7 (PITCNT0–7)

Module Base + 0x000A, 0x000B

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PITCNT0[15:0]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 22-19. PIT Count Register 0 (PITCNT0)

Module Base + 0x000E, 0x000F

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PITCNT1[15:0]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 22-20. PIT Count Register 1 (PITCNT1)

Module Base + 0x0012, 0x0013

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PITCNT2[15:0]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 22-21. PIT Count Register 2 (PITCNT2)

Module Base + 0x0016, 0x0017

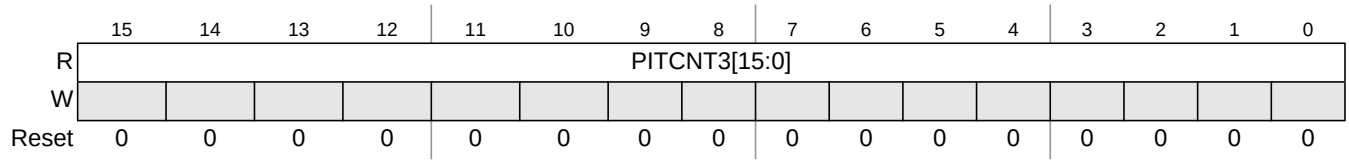


Figure 22-22. PIT Count Register 3 (PITCNT3)

Module Base + 0x001A, 0x001B

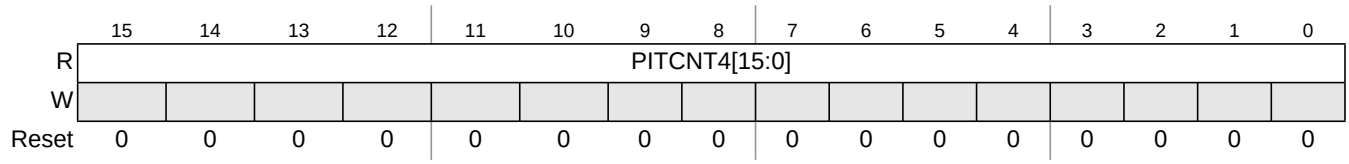


Figure 22-23. PIT Count Register 4 (PITCNT4)

Module Base + 0x001E, 0x001F

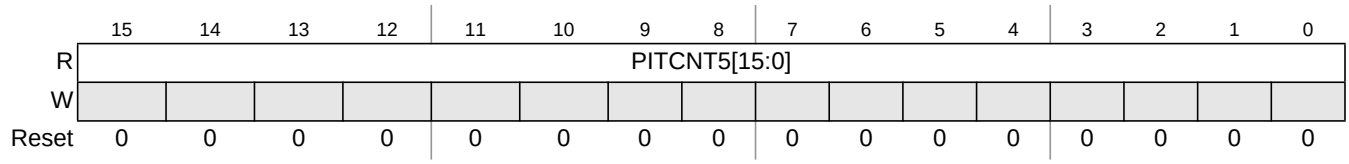


Figure 22-24. PIT Count Register 5 (PITCNT5)

Module Base + 0x0022, 0x0023

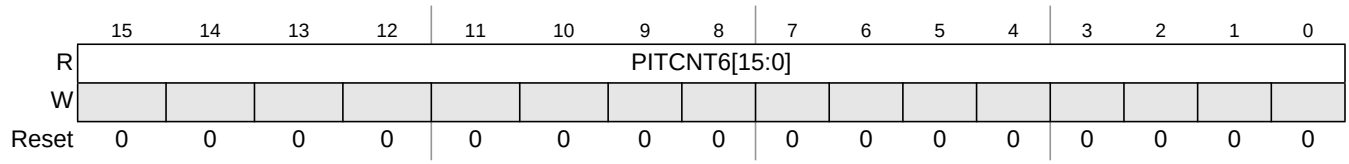


Figure 22-25. PIT Count Register 6 (PITCNT6)

Module Base + 0x0026, 0x0027

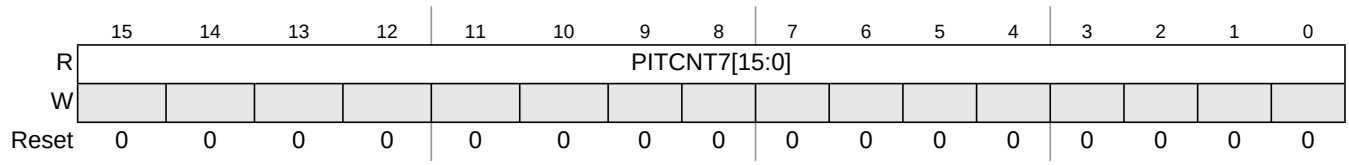


Figure 22-26. PIT Count Register 7 (PITCNT7)

Read: Anytime

Write: Has no meaning or effect

Table 22-9. PITCNT0–7 Field Descriptions

Field	Description
15:0 PITCNTn [15:0]	**PIT Count Bits 15-0** — These bits represent the current 16-bit modulus down-counter value. The read access for the count register must take place in one clock cycle as a 16-bit access.

22.4.0.10 PIT Channel Stop Register (PITCSTP)

Module Base + 0x0028

	7	6	5	4	3	2	1	0
R W	PITCSTP7	PITCSTP6	PITCSTP5	PITCSTP4	PITCSTP3	PITCSTP2	PITCSTP1	PITCSTP0
Reset	0	0	0	0	0	0	0	0

Figure 22-27. PIT Channel Stop Register (PITCSTP)

Read: Anytime

Write: Anytime

Table 22-10. PITCSTP Field Descriptions

Field	Description
7:0 PITCSTP	**PIT Channel Stop** — If cleared, the corresponding PIT channel normally restarts after a time-out event. If set, the corresponding PIT channel stops after the next time-out event. Once a channel has been stopped, the channels waits for a force-load event (i.e. writing a one to the corresponding PITFLT bit) to restart. Additionally, if the channel is in externally triggered mode, an input trigger event also restarts the channel. The information in this register is “pipe-lined”, i.e. it is transferred to internal configuration bits every time the counter of the corresponding PIT channel restarts. This way, the current state of the PITCSTP bit (together with the corresponding counter value in the PITLD register) always reflects the configuration of the PIT channel *after* a restart (and defines the state of the PIT channel after the time-out event following the restart). 0 The PIT channel restarts after the next time-out event. 1 The PIT channel stops after the next time-out event.

22.4.0.11 PIT Combined Output Trigger Configuration Register (PITTRIGOUT)

Module Base + 0x0029

	7	6	5	4	3	2	1	0
R W	PITCOTE7	PITCOTE6	PITCOTE5	PITCOTE4	PITCOTE3	PITCOTE2	PITCOTE1	PITCOTE0
Reset	0	0	0	0	0	0	0	0

Figure 22-28. PIT Combined Output Trigger Configuration Register (PITTRIGOUT)

Read: Anytime

Write: Anytime

Table 22-11. PITTRIGOUT Field Descriptions

Field	Description
7:0 PITCOTE	**PIT Combined Output Trigger Enable** — If set, the time-out event of the corresponding PIT channel contributes to the generation of the combined trigger output. This is used to trigger other modules on programmable timing event sequences. 0 The time-out event of the corresponding PIT channel does not contribute to the combined trigger output. 1 The time-out event of the corresponding PIT channel contributes to the combined trigger output.

22.4.0.12 PIT Input Trigger Control Register (PITTRIGCTL)

Module Base + 0x002A

	7	6	5	4	3	2	1	0
R	PITTRIGIE	0	PITTRIGEDGE[1:0]		0	0	PITTRIGSRC[1:0]	
W								
Reset	0	0	0	0	0	0	0	0

Figure 22-29. PIT Input Trigger Control Register (PITTRIGCTL)

Read: Anytime

Write: Anytime

Table 22-12. PITTRIGCTL Field Descriptions

Field	Description
7 PITTRIGIE	**PIT Input Trigger Interrupt Enable** — This bit enables an interrupt service request whenever the input trigger flag (PITTRIGIF) is set. When the interrupt flag is already set, setting the PITTRIGIE bit will immediately cause an interrupt. To avoid this, the interrupt flag (PITTRIGIF) has to be cleared first. 0 PIT input trigger interrupt is disabled. 1 PIT input trigger interrupt is enabled.
5:4 PITTRIG-EDGE [1:0]	**PIT Input Trigger Edge Select** — These bits select one of 3 possible kinds of events on the PIT trigger input to be used to generate an input trigger event. If the selected kind of event occurs on the selected trigger input an input trigger event is generated for PIT channels configured to be externally triggered, i.e. for channels which have the associated PITTRIGE bit set in the input trigger channel enable register (PITTRIGE). Also, on an input trigger event, the input trigger interrupt flag (PITTRIGIF) is set. 00 a rising edge on the selected trigger input causes a trigger event to be generated 01 a falling edge on the selected trigger input causes a trigger event to generated 10 both a rising or a falling edge on the selected trigger input cause a trigger event to be generated 11 same as ‘10’
1:0 PITTRIGSRC [1:0]	**PIT Input Trigger Source Select** — These bits select one of the four possible PIT input trigger sources. The selected input trigger source is used to generate an input trigger event for PIT channels configured to be externally triggered, i.e. for channels which have the associated PITTRIGE bit set in the input trigger channel enable register (PITTRIGE). Also, on an input trigger event, the input trigger interrupt flag (PITTRIGIF) is set. 00 source 0 is selected 01 source 1 is selected 10 source 2 is selected 11 source 3 is selected

22.4.0.13 PIT Input Trigger Status Register (PITTRIGSTAT)

Module Base + 0x002B

	7	6	5	4	3	2	1	0
R	PITTRIGIF	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

Figure 22-30. PIT Input Trigger Status Register (PITTRIGSTAT)

Read: Anytime

Write: Anytime (write one to clear)

Table 22-13. PITTRIGSTAT Field Descriptions

Field	Description
7 PITTRIGIF	**PIT Input Trigger Interrupt Flag** — PITTRIGIF is set when an input trigger event has occurred. The flag can be cleared by writing a one to the flag bit. Writing a zero has no effect. If flag clearing by writing a one and flag setting happen in the same bus clock cycle, the flag remains set. The flag bit is cleared if the PIT module is disabled. 0 PIT input trigger event has not yet occurred. 1 PIT input trigger event has occurred.

22.4.0.14 PIT Input Trigger Channel Enable Register (PITTRIGE)

Module Base + 0x002C

	7	6	5	4	3	2	1	0
R W	PITTRIGE7	PITTRIGE6	PITTRIGE5	PITTRIGE4	PITTRIGE3	PITTRIGE2	PITTRIGE1	PITTRIGE0
Reset	0	0	0	0	0	0	0	0

Figure 22-31. PIT Input Trigger Channel Enable Register (PITTRIGE)

Read: Anytime

Write: Anytime

Table 22-14. PITTRIGE Field Descriptions

Field	Description
7:0 PITTRIGE	**PIT Input Trigger Channel Enable** — If set, the start of the corresponding PIT channel is configured to be aligned to the external input trigger event (“externally triggered mode”). 0 PIT channel start is not aligned to the external input trigger event. 1 PIT channel start is aligned to the external input trigger event.

NOTE

Before changing the time base for an EPIT channel (i.e. by changing the associated PITRIGE or PITMUX bit), the channel should be disabled first by clearing the appropriate PITCE bit (or by disabling the entire module). Changing the time base of an already enabled EPIT channel can yield unexpected results.

22.5 Functional Description

Figure 22-32 shows a detailed block diagram of the EPIT module. The main parts of the EPIT are status, control and data registers, two 8-bit down-counters, eight 16-bit down-counters and an interrupt/trigger interface.

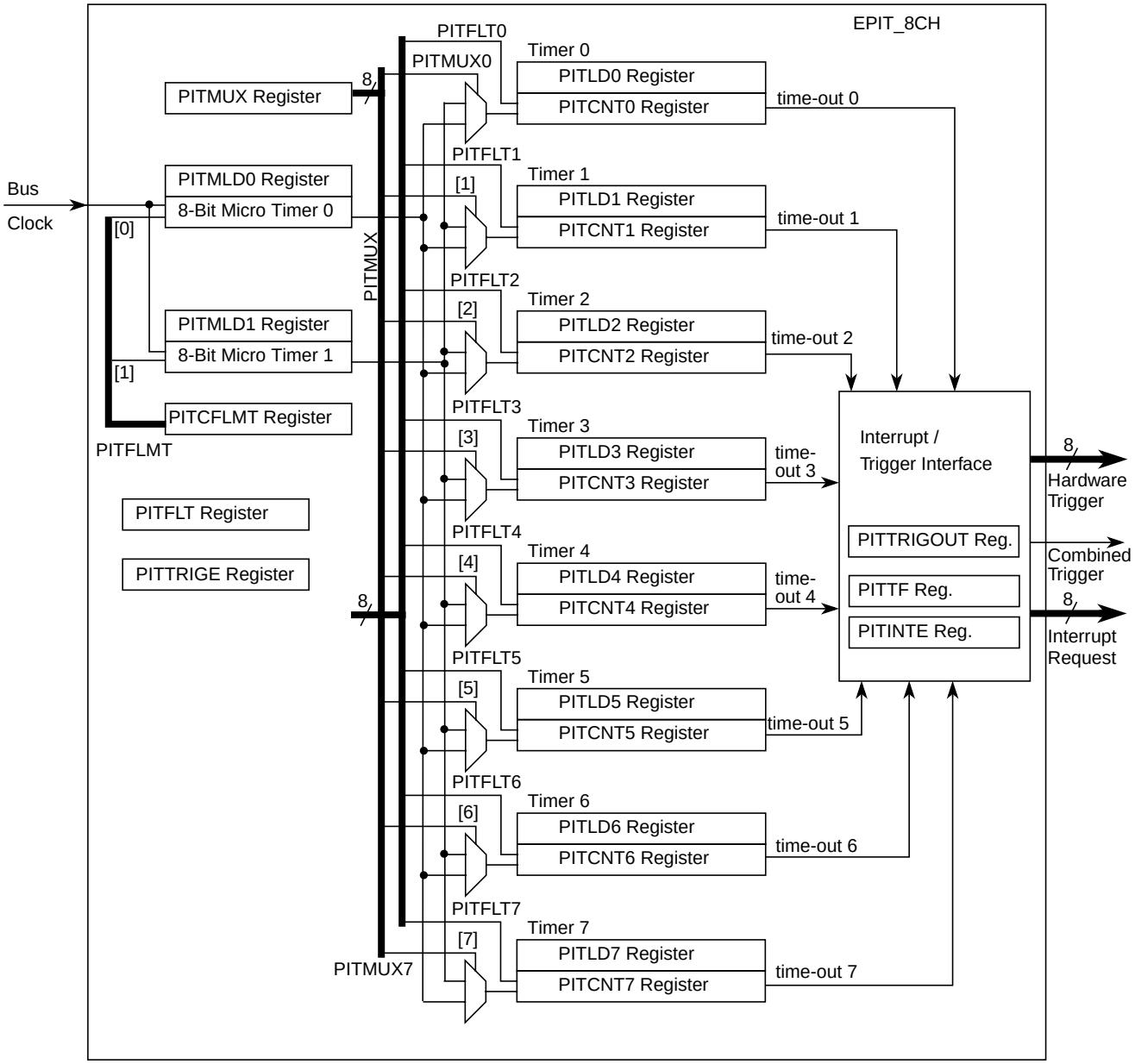


Figure 22-32. EPIT Detailed Block Diagram (tbc)

22.5.1 Timer

As shown in Figure 22-1and Figure 22-32, the 24-bit timers are built in a two-stage architecture with eight 16-bit modulus down-counters and two 8-bit modulus down-counters. The 16-bit timers are clocked with two selectable micro time bases which are generated with 8-bit modulus down-counters. Each 16-bit timer is connected to micro time base 0 or 1 via the PITMUX[7:0] bit setting in the PIT Multiplex (PITMUX) register.

A timer channel is enabled if the module enable bit PITE in the PIT control and force load micro timer (PITCFLMT) register is set and if the corresponding PITCE bit in the PIT channel enable (PITCE) register is set. Two 8-bit modulus down-counters are used to generate two micro time bases. As soon as a micro time base is selected for an enabled timer channel, the corresponding micro timer modulus down-counter will load its start value as specified in the PITMTLD0 or PITMTLD1 register and will start down-counting. Whenever the micro timer down-counter has counted to zero the PITMTLD register is reloaded and the connected 16-bit modulus down-counters count one cycle.

Whenever a 16-bit timer counter and the connected 8-bit micro timer counter have counted to zero, the PITLD register is reloaded and the corresponding time-out flag PITTF in the PIT time-out flag (PITTF) register is set, as shown in Figure 22-33. The time-out period is a function of the timer load (PITLD) and micro timer load (PITMTLD) registers and the bus clock f_{BUS}:

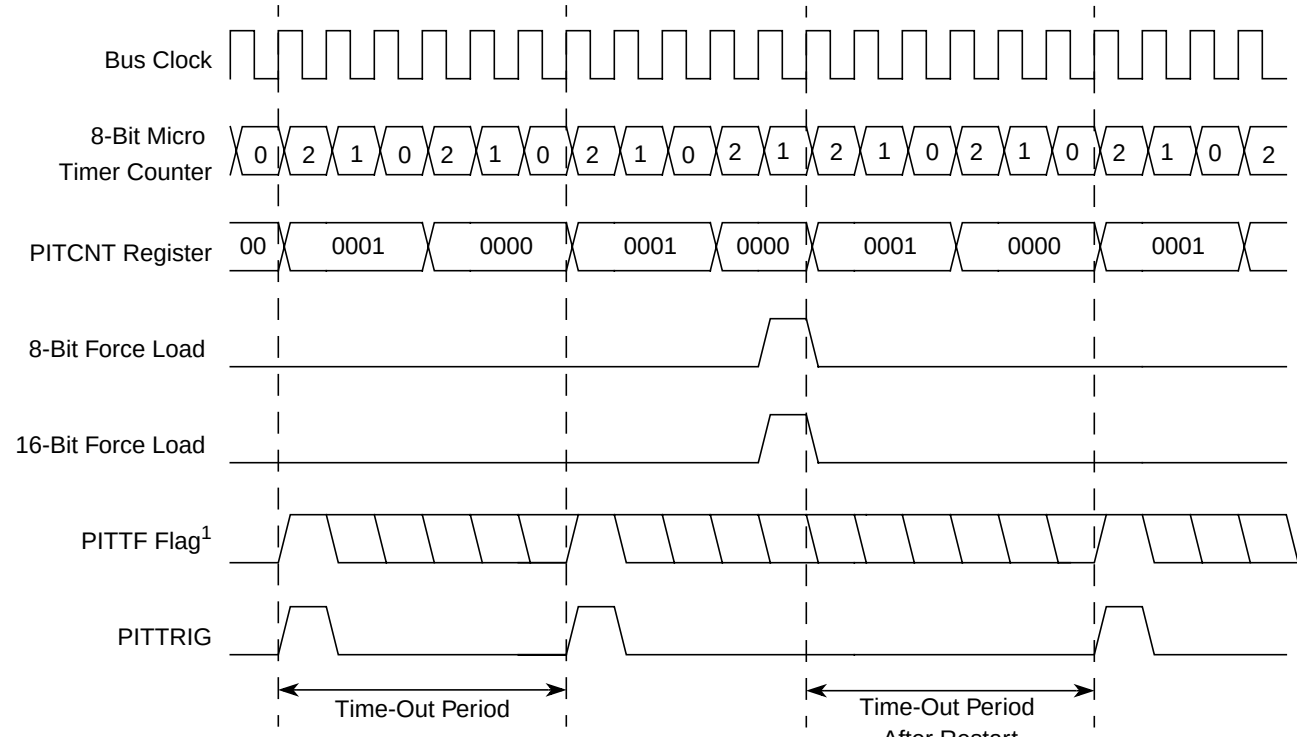
time-out period = (PITMTLD + 1) * (PITLD + 1) / f_{BUS}.

For example, for a 40 MHz bus clock, the maximum time-out period equals:

256 * 65536 * 25 ns = 419.43 ms.

The current 16-bit modulus down-counter value can be read via the PITCNT register. The micro timer down-counter values cannot be read.

The 8-bit micro timers can individually be restarted by writing a one to the corresponding force load micro timer PITFLMT bits in the PIT control and force load micro timer (PITCFLMT) register. The 16-bit timers can individually be restarted by writing a one to the corresponding force load timer PITFLT bits in the PIT force-load timer (PITFLT) register. If desired, any group of timers and micro timers can be restarted at the same time by using one 16-bit write to the adjacent PITCFLMT and PITFLT registers with the relevant bits set, as shown in Figure 22-33.



Note 1. The PITTF flag clearing depends on the software

Figure 22-33. PIT Trigger and Flag Signal Timing

22.5.2 Interrupt Interface

Each time-out event can be used to trigger an interrupt service request. For each timer channel, an individual bit PITINTE in the PIT interrupt enable (PITINTE) register exists to enable this feature. If PITINTE is set, an interrupt service is requested whenever the corresponding time-out flag PITTF in the PIT time-out flag (PITTF) register is set. The flag can be cleared by writing a one to the flag bit.

NOTE

Be careful when resetting the PITE, PITINTE, PITCE or PITTRIGIE bits in case of pending PIT interrupts, to avoid spurious interrupt requests.

22.5.3 Hardware Trigger

The EPIT module contains eight hardware trigger signal lines PITTRIG[7:0], one for each timer channel. These signals can be connected on SoC level to peripheral modules enabling e.g. periodic ATD conversion (please refer to the top-level chapter in the Device Reference Manual for the mapping of PITTRIG[7:0] signals to peripheral modules).

Whenever a timer channel time-out is reached, the corresponding PITTF flag is set and the corresponding trigger signal PITTRIG triggers a rising edge. The trigger feature requires a minimum time-out period of two bus clock cycles because the trigger is asserted high for at least one bus clock cycle. For load register values PITLD = 0x0001 and PITMTLD = 0x0002 the flag setting, trigger timing and a restart with force load is shown in Figure 22-33.

Additionally, each PIT channel can be configured to contribute to the combined output trigger signal by setting the associated bit in the PITTRIGOUT register. By combining the output of multiple PIT channels, it is possible to generate more complex trigger sequences for other peripheral modules, like an ADC (please refer to the top-level chapter in the Device Reference Manual for information about which module the combined trigger output is connected to).

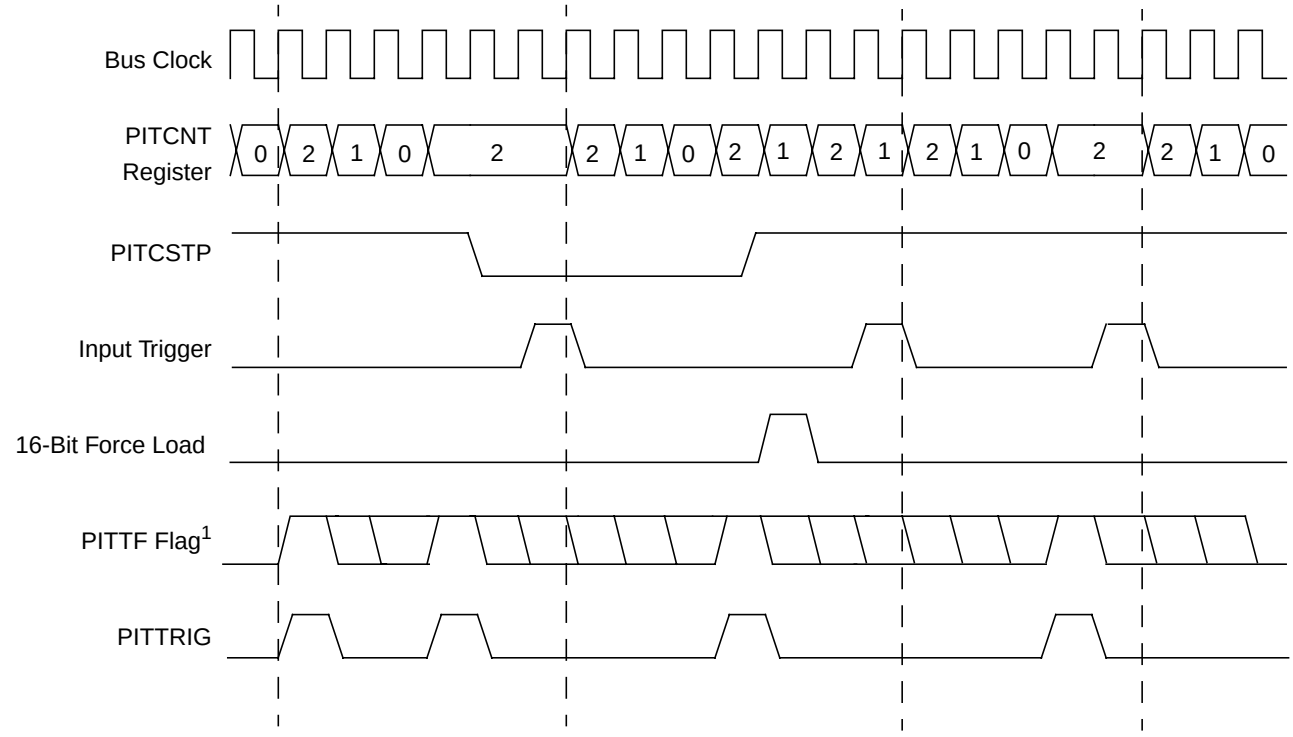
22.5.4 External Input Trigger

Each of the eight EPIT channels can be configured to (re-)start aligned to an external input trigger event (“externally triggered mode”, see PITTRIGE register).

The external input trigger signal has the same effect like a “force load”, i.e. all EPIT channels configured to start aligned to the external input trigger (externally triggered mode) reload the counter value (along with the corresponding PITCSTPn configuration bit in PITCSTP) and start counting when the input trigger event occurs. The re-start takes place for all channels in externally triggered mode, regardless of the current state of the affected channel. This means that additionally to all stopped channels, not yet expired channels are restarted as well.

EPIT channels in externally triggered mode can also be restarted by writing to the PIT Force Load Timer register (PITFLT).

For channels configured to externally triggered mode, the micro-timers are bypassed, i.e. the effective time-base for these channels is the bus-clock.



Note 1. The PITTF flag clearing depends on the software

Figure 22-34. Externally Triggered EPIT timing

22.6 Initialization

22.6.1 Startup

Set the configuration registers before the PITE bit in the PITCFLMT register is set. Before PITE is set, the configuration registers can be written in arbitrary order.

22.6.2 Shutdown

When the PITCE register bits, the PITINTE register bits or the PITE bit in the PITCFLMT register are cleared, the corresponding PIT interrupt flags are cleared. In case of a pending PIT interrupt request, a spurious interrupt can be generated. Two strategies, which avoid spurious interrupts, are recommended:

1. Reset the PIT interrupt flags only in an ISR. When entering the ISR, the I mask bit in the CCR is set automatically. The I mask bit must not be cleared before the PIT interrupt flags are cleared.
2. After setting the I mask bit with the SEI instruction, the PIT interrupt flags can be cleared. Then clear the I mask bit with the CLI instruction to re-enable interrupts.

22.6.3 Flag Clearing

A flag is cleared by writing a one to the flag bit. Always use store or move instructions to write a one in certain bit positions. Do not use the BSET instructions. Do not use any C-constructs that compile to BSET instructions. “BSET flag_register, #mask” must not be used for flag clearing because BSET is a read-modify-write instruction which writes back the “bit-wise or” of the flag_register and the mask into the flag_register. BSET would clear all flag bits that were set, independent from the mask.

For example, to clear flag bit 0 use: MOVB #$01,PITTF.

22.7 Application Information

To get started quickly with the PIT24B8C module this section provides a small code example how to use the block. Please note that the example provided is only one specific case out of the possible configurations and implementations.

Functionality: Generate an PIT interrupt on channel 0 every 500 PIT clock cycles.

```
                ORG     CODESTART               ; place the program into specific
                                                ; range (to be selected)
                LDS     RAMEND                  ; load stack pointer to top of RAM
                MOVW    #CH0_ISR,VEC_PIT_CH0    ; Change value of channel 0 ISR adr

; ******************** Start PIT Initialization ***************************************************

                CLR     PITCFLMT                ; disable PIT
                MOVB    #$01,PITCE              ; enable timer channel 0
                CLR     PITMUX                  ; ch0 connected to micro timer 0
                MOVB    #$63,PITMTLD0           ; micro time base 0 equals 100 clock cycles
                MOVW    #$0004,PITLD0           ; time base 0 eq. 5 micro time bases 0 =5*100 = 500
```

```
                MOVB    #$01,PITINTE            ; enable interrupt channel 0
                MOVB    #$80,PITCFLMT           ; enable PIT
                CLI                             ; clear Interrupt disable Mask bit

;******************** Main Program ************************************************************

MAIN:           BRA *                           ; loop until interrupt

;******************** Channel 0 Interrupt Routine *********************************************

CH0_ISR:        LDAA    PITTF                   ; 8 bit read of PIT time out flags
                MOVB    #$01,PITTF              ; clear PIT channel 0 time out flag
                RTI                             ; return to MAIN
```

Chapter 23
Serial Communication Interface (S12SCIV5)

Table 23-1. Revision History

Version Number	Revision Date	Effective Date	Author	Description of Changes
05.03	12/25/2008			remove redundancy comments in Figure 23-2
05.04	08/05/2009			fix typo, SCIBDL reset value be 0x04, not 0x00
05.05	06/03/2010			fix typo, Table 23-4,SCICR1 Even parity should be PT=0 fix typo, on page 23-1055,should be BKDIF,not BLDIF

23.1 Introduction

This block guide provides an overview of the serial communication interface (SCI) module.

The SCI allows asynchronous serial communications with peripheral devices and other CPUs.

23.1.1 Glossary

IR: InfraRed

IrDA: Infrared Design Associate

IRQ: Interrupt Request

LIN: Local Interconnect Network

LSB: Least Significant Bit

MSB: Most Significant Bit

NRZ: Non-Return-to-Zero

RZI: Return-to-Zero-Inverted

RXD: Receive Pin

SCI : Serial Communication Interface

TXD: Transmit Pin

23.1.2 Features

The SCI includes these distinctive features:

- Full-duplex or single-wire operation
- Standard mark/space non-return-to-zero (NRZ) format
- Selectable IrDA 1.4 return-to-zero-inverted (RZI) format with programmable pulse widths
- 13-bit baud rate selection
- Programmable 8-bit or 9-bit data format
- Separately enabled transmitter and receiver
- Programmable polarity for transmitter and receiver
- Programmable transmitter output parity
- Two receiver wakeup methods:
  - Idle line wakeup
  - Address mark wakeup
- Interrupt-driven operation with eight flags:
  - Transmitter empty
  - Transmission complete
  - Receiver full
  - Idle receiver input
  - Receiver overrun
  - Noise error
  - Framing error
  - Parity error
  - Receive wakeup on active edge
  - Transmit collision detect supporting LIN
  - Break Detect supporting LIN
- Receiver framing error detection
- Hardware parity checking
- 1/16 bit-time noise detection

23.1.3 Modes of Operation

The SCI functions the same in normal, special, and emulation modes. It has two low power modes, wait and stop modes.

- Run mode
- Wait mode
- Stop mode

23.1.4 Block Diagram

Figure 23-1 is a high level block diagram of the SCI module, showing the interaction of various function blocks.

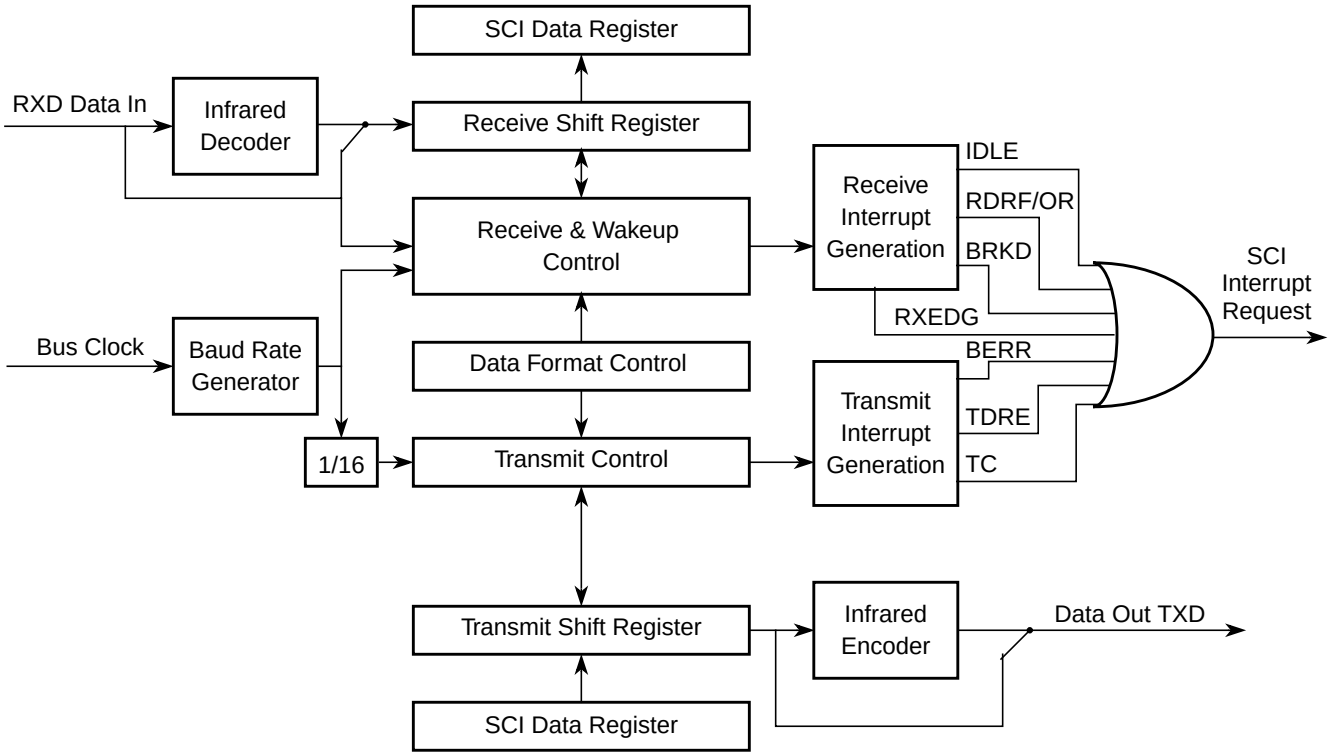


Figure 23-1. SCI Block Diagram

23.2 External Signal Description

The SCI module has a total of two external pins.

23.2.1 TXD — Transmit Pin

The TXD pin transmits SCI (standard or infrared) data. It will idle high in either mode and is high impedance anytime the transmitter is disabled.

23.2.2 RXD — Receive Pin

The RXD pin receives SCI (standard or infrared) data. An idle line is detected as a line high. This input is ignored when the receiver is disabled and should be terminated to a known voltage.

23.3 Memory Map and Register Definition

This section provides a detailed description of all the SCI registers.

23.3.1 Module Memory Map and Register Definition

The memory map for the SCI module is given below in Figure 23-2. The address listed for each register is the address offset. The total address for each register is the sum of the base address for the SCI module and the address offset for each register.

23.3.2 Register Descriptions

This section consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number. Writes to a reserved register locations do not have any effect and reads of these locations return a zero. Details of register bit and field function follow the register diagrams, in bit order.

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000 SCIBDH[1]	R	IREN	TNP1	TNP0	SBR12	SBR11	SBR10	SBR9	SBR8
	W								
0x0001 SCIBDL[1]	R	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
	W								
0x0002 SCICR1[1]	R	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
	W								
0x0000 SCIASR1[2]	R	RXEDGIF	0	0	0	0	BERRV	BERRIF	BKDIF
	W								
0x0001 SCIACR1[2]	R	RXEDGIE	0	0	0	0	0	BERRIE	BKDIE
	W								
0x0002 SCIACR2[2]	R	0	0	0	0	0	BERRM1	BERRM0	BKDFE
	W								
0x0003 SCICR2	R	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
	W								
0x0004 SCISR1	R	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
	W								
0x0005 SCISR2	R	AMAP	0	0	TXPOL	RXPOL	BRK13	TXDIR	RAF
	W								
0x0006 SCIDRH	R	R8	T8	0	0	0	0	0	0
	W								
0x0007 SCIDRL	R	R7	R6	R5	R4	R3	R2	R1	R0
	W	T7	T6	T5	T4	T3	T2	T1	T0

1.These registers are accessible if the AMAP bit in the SCISR2 register is set to zero.

2,These registers are accessible if the AMAP bit in the SCISR2 register is set to one.

= Unimplemented or Reserved

Figure 23-2. SCI Register Summary

23.3.2.1 SCI Baud Rate Registers (SCIBDH, SCIBDL)

Module Base + 0x0000

	7	6	5	4	3	2	1	0
R W	IREN	TNP1	TNP0	SBR12	SBR11	SBR10	SBR9	SBR8
Reset	0	0	0	0	0	0	0	0

Figure 23-3. SCI Baud Rate Register (SCIBDH)

Module Base + 0x0001

	7	6	5	4	3	2	1	0
R W	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
Reset	0	0	0	0	0	1	0	0

Figure 23-4. SCI Baud Rate Register (SCIBDL)

Read: Anytime, if AMAP = 0. If only SCIBDH is written to, a read will not return the correct data until SCIBDL is written to as well, following a write to SCIBDH.

Write: Anytime, if AMAP = 0.

NOTE

Those two registers are only visible in the memory map if AMAP = 0 (reset condition).

The SCI baud rate register is used by to determine the baud rate of the SCI, and to control the infrared modulation/demodulation submodule.

Table 23-2. SCIBDH and SCIBDL Field Descriptions

Field	Description
7 IREN	**Infrared Enable Bit** — This bit enables/disables the infrared modulation/demodulation submodule. 0 IR disabled 1 IR enabled
6:5 TNP[1:0]	**Transmitter Narrow Pulse Bits** — These bits enable whether the SCI transmits a 1/16, 3/16, 1/32 or 1/4 narrow pulse. See Table 23-3.
4:0 7:0 SBR[12:0]	**SCI Baud Rate Bits** — The baud rate for the SCI is determined by the bits in this register. The baud rate is calculated two different ways depending on the state of the IREN bit. The formulas for calculating the baud rate are: When IREN = 0 then, SCI baud rate = SCI bus clock / (16 x SBR[12:0]) When IREN = 1 then, SCI baud rate = SCI bus clock / (32 x SBR[12:1]) **Note:** The baud rate generator is disabled after reset and not started until the TE bit or the RE bit is set for the first time. The baud rate generator is disabled when (SBR[12:0] = 0 and IREN = 0) or (SBR[12:1] = 0 and IREN = 1). **Note:** Writing to SCIBDH has no effect without writing to SCIBDL, because writing to SCIBDH puts the data in a temporary location until SCIBDL is written to.

Table 23-3. IRSCI Transmit Pulse Width

TNP[1:0]	Narrow Pulse Width
11	1/4
10	1/32
01	1/16
00	3/16

23.3.2.2 SCI Control Register 1 (SCICR1)

Module Base + 0x0002

	7	6	5	4	3	2	1	0
R W	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
Reset	0	0	0	0	0	0	0	0

Figure 23-5. SCI Control Register 1 (SCICR1)

Read: Anytime, if AMAP = 0.

Write: Anytime, if AMAP = 0.

NOTE

This register is only visible in the memory map if AMAP = 0 (reset condition).

Table 23-4. SCICR1 Field Descriptions

Field	Description
7 LOOPS	**Loop Select Bit** — LOOPS enables loop operation. In loop operation, the RXD pin is disconnected from the SCI and the transmitter output is internally connected to the receiver input. Both the transmitter and the receiver must be enabled to use the loop function. 0 Normal operation enabled 1 Loop operation enabled The receiver input is determined by the RSRC bit.
6 SCISWAI	**SCI Stop in Wait Mode Bit** — SCISWAI disables the SCI in wait mode. 0 SCI enabled in wait mode 1 SCI disabled in wait mode
5 RSRC	**Receiver Source Bit** — When LOOPS = 1, the RSRC bit determines the source for the receiver shift register input. See Table 23-5. 0 Receiver input internally connected to transmitter output 1 Receiver input connected externally to transmitter
4 M	**Data Format Mode Bit** — MODE determines whether data characters are eight or nine bits long. 0 One start bit, eight data bits, one stop bit 1 One start bit, nine data bits, one stop bit
3 WAKE	**Wakeup Condition Bit** — WAKE determines which condition wakes up the SCI: a logic 1 (address mark) in the most significant bit position of a received data character or an idle condition on the RXD pin. 0 Idle line wakeup 1 Address mark wakeup

Table 23-4. SCICR1 Field Descriptions (continued)

Field	Description
2 ILT	**Idle Line Type Bit** — ILT determines when the receiver starts counting logic 1s as idle character bits. The counting begins either after the start bit or after the stop bit. If the count begins after the start bit, then a string of logic 1s preceding the stop bit may cause false recognition of an idle character. Beginning the count after the stop bit avoids false idle character recognition, but requires properly synchronized transmissions. 0 Idle character bit count begins after start bit 1 Idle character bit count begins after stop bit
1 PE	**Parity Enable Bit** — PE enables the parity function. When enabled, the parity function inserts a parity bit in the most significant bit position. 0 Parity function disabled 1 Parity function enabled
0 PT	**Parity Type Bit** — PT determines whether the SCI generates and checks for even parity or odd parity. With even parity, an even number of 1s clears the parity bit and an odd number of 1s sets the parity bit. With odd parity, an odd number of 1s clears the parity bit and an even number of 1s sets the parity bit. 0 Even parity 1 Odd parity

Table 23-5. Loop Functions

LOOPS	RSRC	Function
0	x	Normal operation
1	0	Loop mode with transmitter output internally connected to receiver input
1	1	Single-wire mode with TXD pin connected to receiver input

23.3.2.3 SCI Alternative Status Register 1 (SCIASR1)

Module Base + 0x0000

	7	6	5	4	3	2	1	0
R	RXEDGIF	0	0	0	0	BERRV	BERRIF	BKDIF
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 23-6. SCI Alternative Status Register 1 (SCIASR1)

Read: Anytime, if AMAP = 1

Write: Anytime, if AMAP = 1

Table 23-6. SCIASR1 Field Descriptions

Field	Description
7 RXEDGIF	**Receive Input Active Edge Interrupt Flag** — RXEDGIF is asserted, if an active edge (falling if RXPOL = 0, rising if RXPOL = 1) on the RXD input occurs. RXEDGIF bit is cleared by writing a “1” to it. 0 No active receive on the receive input has occurred 1 An active edge on the receive input has occurred
2 BERRV	**Bit Error Value** — BERRV reflects the state of the RXD input when the bit error detect circuitry is enabled and a mismatch to the expected value happened. The value is only meaningful, if BERRIF = 1. 0 A low input was sampled, when a high was expected 1 A high input reassembled, when a low was expected
1 BERRIF	**Bit Error Interrupt Flag** — BERRIF is asserted, when the bit error detect circuitry is enabled and if the value sampled at the RXD input does not match the transmitted value. If the BERRIE interrupt enable bit is set an interrupt will be generated. The BERRIF bit is cleared by writing a “1” to it. 0 No mismatch detected 1 A mismatch has occurred
0 BKDIF	**Break Detect Interrupt Flag** — BKDIF is asserted, if the break detect circuitry is enabled and a break signal is received. If the BKDIE interrupt enable bit is set an interrupt will be generated. The BKDIF bit is cleared by writing a “1” to it. 0 No break signal was received 1 A break signal was received

23.3.2.4 SCI Alternative Control Register 1 (SCIACR1)

Module Base + 0x0001

	7	6	5	4	3	2	1	0
R	RXEDGIE	0	0	0	0	0	BERRIE	BKDIE
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 23-7. SCI Alternative Control Register 1 (SCIACR1)

Read: Anytime, if AMAP = 1

Write: Anytime, if AMAP = 1

Table 23-7. SCIACR1 Field Descriptions

Field	Description
7 RSEDGIE	**Receive Input Active Edge Interrupt Enable** — RXEDGIE enables the receive input active edge interrupt flag, RXEDGIF, to generate interrupt requests. 0 RXEDGIF interrupt requests disabled 1 RXEDGIF interrupt requests enabled
1 BERRIE	**Bit Error Interrupt Enable** — BERRIE enables the bit error interrupt flag, BERRIF, to generate interrupt requests. 0 BERRIF interrupt requests disabled 1 BERRIF interrupt requests enabled
0 BKDIE	**Break Detect Interrupt Enable** — BKDIE enables the break detect interrupt flag, BKDIF, to generate interrupt requests. 0 BKDIF interrupt requests disabled 1 BKDIF interrupt requests enabled

23.3.2.5 SCI Alternative Control Register 2 (SCIACR2)

Module Base + 0x0002

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	BERRM1	BERRM0	BKDFE
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 23-8. SCI Alternative Control Register 2 (SCIACR2)

Read: Anytime, if AMAP = 1

Write: Anytime, if AMAP = 1

Table 23-8. SCIACR2 Field Descriptions

Field	Description
2:1 BERRM[1:0]	**Bit Error Mode** — Those two bits determines the functionality of the bit error detect feature. See Table 23-9.
0 BKDFE	**Break Detect Feature Enable** — BKDFE enables the break detect circuitry. 0 Break detect circuit disabled 1 Break detect circuit enabled

Table 23-9. Bit Error Mode Coding

BERRM1	BERRM0	Function
0	0	Bit error detect circuit is disabled
0	1	Receive input sampling occurs during the 9th time tick of a transmitted bit (refer to Figure 23-19)
1	0	Receive input sampling occurs during the 13th time tick of a transmitted bit (refer to Figure 23-19)
1	1	Reserved

23.3.2.6 SCI Control Register 2 (SCICR2)

Module Base + 0x0003

	7	6	5	4	3	2	1	0
R W	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
Reset	0	0	0	0	0	0	0	0

Figure 23-9. SCI Control Register 2 (SCICR2)

Read: Anytime

Write: Anytime

Table 23-10. SCICR2 Field Descriptions

Field	Description
7 TIE	**Transmitter Interrupt Enable Bit** — TIE enables the transmit data register empty flag, TDRE, to generate interrupt requests. 0 TDRE interrupt requests disabled 1 TDRE interrupt requests enabled
6 TCIE	**Transmission Complete Interrupt Enable Bit** — TCIE enables the transmission complete flag, TC, to generate interrupt requests. 0 TC interrupt requests disabled 1 TC interrupt requests enabled
5 RIE	**Receiver Full Interrupt Enable Bit** — RIE enables the receive data register full flag, RDRF, or the overrun flag, OR, to generate interrupt requests. 0 RDRF and OR interrupt requests disabled 1 RDRF and OR interrupt requests enabled
4 ILIE	**Idle Line Interrupt Enable Bit** — ILIE enables the idle line flag, IDLE, to generate interrupt requests. 0 IDLE interrupt requests disabled 1 IDLE interrupt requests enabled
3 TE	**Transmitter Enable Bit** — TE enables the SCI transmitter and configures the TXD pin as being controlled by the SCI. The TE bit can be used to queue an idle preamble. 0 Transmitter disabled 1 Transmitter enabled
2 RE	**Receiver Enable Bit** — RE enables the SCI receiver. 0 Receiver disabled 1 Receiver enabled
1 RWU	**Receiver Wakeup Bit** — Standby state 0 Normal operation. 1 RWU enables the wakeup function and inhibits further receiver interrupt requests. Normally, hardware wakes the receiver by automatically clearing RWU.
0 SBK	**Send Break Bit** — Toggling SBK sends one break character (10 or 11 logic 0s, respectively 13 or 14 logics 0s if BRK13 is set). Toggling implies clearing the SBK bit before the break character has finished transmitting. As long as SBK is set, the transmitter continues to send complete break characters (10 or 11 bits, respectively 13 or 14 bits). 0 No break characters 1 Transmit break characters

23.3.2.7 SCI Status Register 1 (SCISR1)

The SCISR1 and SCISR2 registers provides inputs to the MCU for generation of SCI interrupts. Also, these registers can be polled by the MCU to check the status of these bits. The flag-clearing procedures require that the status register be read followed by a read or write to the SCI data register.It is permissible to execute other instructions between the two steps as long as it does not compromise the handling of I/O, but the order of operations is important for flag clearing.

Module Base + 0x0004

	7	6	5	4	3	2	1	0
R	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
W								
Reset	1	1	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 23-10. SCI Status Register 1 (SCISR1)

Read: Anytime

Write: Has no meaning or effect

Table 23-11. SCISR1 Field Descriptions

Field	Description
7 TDRE	**Transmit Data Register Empty Flag** — TDRE is set when the transmit shift register receives a byte from the SCI data register. When TDRE is 1, the transmit data register (SCIDRH/L) is empty and can receive a new value to transmit.Clear TDRE by reading SCI status register 1 (SCISR1), with TDRE set and then writing to SCI data register low (SCIDRL). 0 No byte transferred to transmit shift register 1 Byte transferred to transmit shift register; transmit data register empty
6 TC	**Transmit Complete Flag** — TC is set low when there is a transmission in progress or when a preamble or break character is loaded. TC is set high when the TDRE flag is set and no data, preamble, or break character is being transmitted.When TC is set, the TXD pin becomes idle (logic 1). Clear TC by reading SCI status register 1 (SCISR1) with TC set and then writing to SCI data register low (SCIDRL). TC is cleared automatically when data, preamble, or break is queued and ready to be sent. TC is cleared in the event of a simultaneous set and clear of the TC flag (transmission not complete). 0 Transmission in progress 1 No transmission in progress
5 RDRF	**Receive Data Register Full Flag** — RDRF is set when the data in the receive shift register transfers to the SCI data register. Clear RDRF by reading SCI status register 1 (SCISR1) with RDRF set and then reading SCI data register low (SCIDRL). 0 Data not available in SCI data register 1 Received data available in SCI data register
4 IDLE	**Idle Line Flag** — IDLE is set when 10 consecutive logic 1s (if M = 0) or 11 consecutive logic 1s (if M =1) appear on the receiver input. Once the IDLE flag is cleared, a valid frame must again set the RDRF flag before an idle condition can set the IDLE flag.Clear IDLE by reading SCI status register 1 (SCISR1) with IDLE set and then reading SCI data register low (SCIDRL). 0 Receiver input is either active now or has never become active since the IDLE flag was last cleared 1 Receiver input has become idle **Note:** When the receiver wakeup bit (RWU) is set, an idle line condition does not set the IDLE flag.

Table 23-11. SCISR1 Field Descriptions (continued)

Field	Description
3 OR	**Overrun Flag** — OR is set when software fails to read the SCI data register before the receive shift register receives the next frame. The OR bit is set immediately after the stop bit has been completely received for the second frame. The data in the shift register is lost, but the data already in the SCI data registers is not affected. Clear OR by reading SCI status register 1 (SCISR1) with OR set and then reading SCI data register low (SCIDRL). 0 No overrun 1 Overrun **Note:** OR flag may read back as set when RDRF flag is clear. This may happen if the following sequence of events occurs: 1. After the first frame is received, read status register SCISR1 (returns RDRF set and OR flag clear); 2. Receive second frame without reading the first frame in the data register (the second frame is not received and OR flag is set); 3. Read data register SCIDRL (returns first frame and clears RDRF flag in the status register); 4. Read status register SCISR1 (returns RDRF clear and OR set). Event 3 may be at exactly the same time as event 2 or any time after. When this happens, a dummy SCIDRL read following event 4 will be required to clear the OR flag if further frames are to be received.
2 NF	**Noise Flag** — NF is set when the SCI detects noise on the receiver input. NF bit is set during the same cycle as the RDRF flag but does not get set in the case of an overrun. Clear NF by reading SCI status register 1(SCISR1), and then reading SCI data register low (SCIDRL). 0 No noise 1 Noise
1 FE	**Framing Error Flag** — FE is set when a logic 0 is accepted as the stop bit. FE bit is set during the same cycle as the RDRF flag but does not get set in the case of an overrun. FE inhibits further data reception until it is cleared. Clear FE by reading SCI status register 1 (SCISR1) with FE set and then reading the SCI data register low (SCIDRL). 0 No framing error 1 Framing error
0 PF	**Parity Error Flag** — PF is set when the parity enable bit (PE) is set and the parity of the received data does not match the parity type bit (PT). PF bit is set during the same cycle as the RDRF flag but does not get set in the case of an overrun. Clear PF by reading SCI status register 1 (SCISR1), and then reading SCI data register low (SCIDRL). 0 No parity error 1 Parity error

23.3.2.8 SCI Status Register 2 (SCISR2)

Module Base + 0x0005

	7	6	5	4	3	2	1	0
R	AMAP	0	0	TXPOL	RXPOL	BRK13	TXDIR	RAF
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 23-11. SCI Status Register 2 (SCISR2)

Read: Anytime

Write: Anytime

Table 23-12. SCISR2 Field Descriptions

Field	Description
7 AMAP	**Alternative Map** — This bit controls which registers sharing the same address space are accessible. In the reset condition the SCI behaves as previous versions. Setting AMAP=1 allows the access to another set of control and status registers and hides the baud rate and SCI control Register 1. 0 The registers labelled SCIBDH (0x0000),SCIBDL (0x0001), SCICR1 (0x0002) are accessible 1 The registers labelled SCIASR1 (0x0000),SCIACR1 (0x0001), SCIACR2 (0x00002) are accessible
4 TXPOL	**Transmit Polarity** — This bit control the polarity of the transmitted data. In NRZ format, a one is represented by a mark and a zero is represented by a space for normal polarity, and the opposite for inverted polarity. In IrDA format, a zero is represented by short high pulse in the middle of a bit time remaining idle low for a one for normal polarity, and a zero is represented by short low pulse in the middle of a bit time remaining idle high for a one for inverted polarity. 0 Normal polarity 1 Inverted polarity
3 RXPOL	**Receive Polarity** — This bit control the polarity of the received data. In NRZ format, a one is represented by a mark and a zero is represented by a space for normal polarity, and the opposite for inverted polarity. In IrDA format, a zero is represented by short high pulse in the middle of a bit time remaining idle low for a one for normal polarity, and a zero is represented by short low pulse in the middle of a bit time remaining idle high for a one for inverted polarity. 0 Normal polarity 1 Inverted polarity
2 BRK13	**Break Transmit Character Length** — This bit determines whether the transmit break character is 10 or 11 bit respectively 13 or 14 bits long. The detection of a framing error is not affected by this bit. 0 Break character is 10 or 11 bit long 1 Break character is 13 or 14 bit long
1 TXDIR	**Transmitter Pin Data Direction in Single-Wire Mode** — This bit determines whether the TXD pin is going to be used as an input or output, in the single-wire mode of operation. This bit is only relevant in the single-wire mode of operation. 0 TXD pin to be used as an input in single-wire mode 1 TXD pin to be used as an output in single-wire mode
0 RAF	**Receiver Active Flag** — RAF is set when the receiver detects a logic 0 during the RT1 time period of the start bit search. RAF is cleared when the receiver detects an idle character. 0 No reception in progress 1 Reception in progress

23.3.2.9 SCI Data Registers (SCIDRH, SCIDRL)

Module Base + 0x0006

	7	6	5	4	3	2	1	0
R	R8	T8	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 23-12. SCI Data Registers (SCIDRH)

Module Base + 0x0007

	7	6	5	4	3	2	1	0
R	R7	R6	R5	R4	R3	R2	R1	R0
W	T7	T6	T5	T4	T3	T2	T1	T0
Reset	0	0	0	0	0	0	0	0

Figure 23-13. SCI Data Registers (SCIDRL)

Read: Anytime; reading accesses SCI receive data register

Write: Anytime; writing accesses SCI transmit data register; writing to R8 has no effect

Table 23-13. SCIDRH and SCIDRL Field Descriptions

Field	Description
SCIDRH 7 R8	**Received Bit 8** — R8 is the ninth data bit received when the SCI is configured for 9-bit data format (M = 1).
SCIDRH 6 T8	**Transmit Bit 8** — T8 is the ninth data bit transmitted when the SCI is configured for 9-bit data format (M = 1).
SCIDRL 7:0 R[7:0] T[7:0]	**R7:R0** — Received bits seven through zero for 9-bit or 8-bit data formats **T7:T0** — Transmit bits seven through zero for 9-bit or 8-bit formats

NOTE

If the value of T8 is the same as in the previous transmission, T8 does not have to be rewritten.The same value is transmitted until T8 is rewritten

In 8-bit data format, only SCI data register low (SCIDRL) needs to be accessed.

When transmitting in 9-bit data format and using 8-bit write instructions, write first to SCI data register high (SCIDRH), then SCIDRL.

23.4 Functional Description

This section provides a complete functional description of the SCI block, detailing the operation of the design from the end user perspective in a number of subsections.

Figure 23-14 shows the structure of the SCI module. The SCI allows full duplex, asynchronous, serial communication between the CPU and remote devices, including other CPUs. The SCI transmitter and receiver operate independently, although they use the same baud rate generator. The CPU monitors the status of the SCI, writes the data to be transmitted, and processes received data.

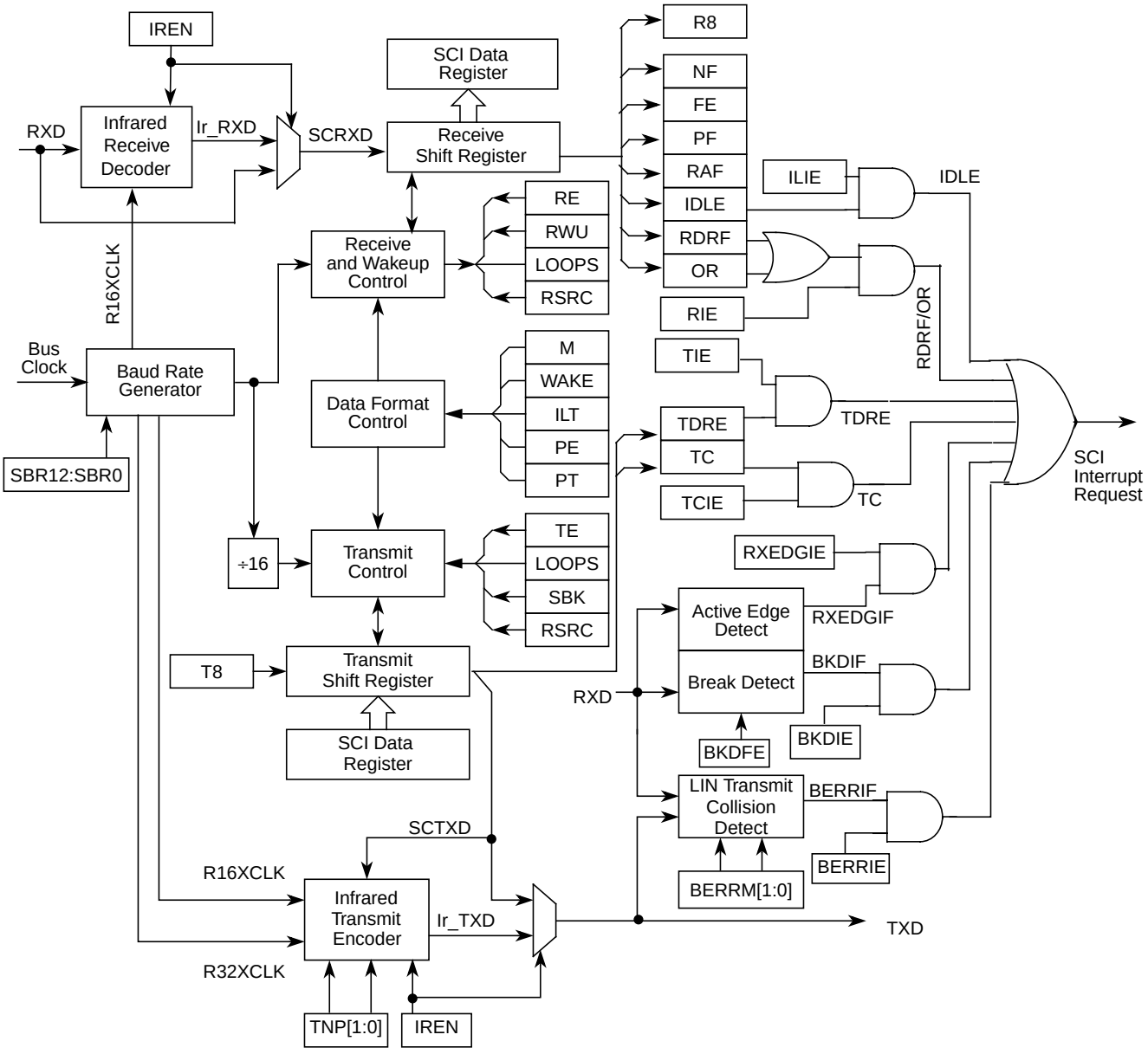


Figure 23-14. Detailed SCI Block Diagram

23.4.1 Infrared Interface Submodule

This module provides the capability of transmitting narrow pulses to an IR LED and receiving narrow pulses and transforming them to serial bits, which are sent to the SCI. The IrDA physical layer specification defines a half-duplex infrared communication link for exchange data. The full standard includes data rates up to 16 Mbits/s. This design covers only data rates between 2.4 Kbits/s and 115.2 Kbits/s.

The infrared submodule consists of two major blocks: the transmit encoder and the receive decoder. The SCI transmits serial bits of data which are encoded by the infrared submodule to transmit a narrow pulse for every zero bit. No pulse is transmitted for every one bit. When receiving data, the IR pulses should be detected using an IR photo diode and transformed to CMOS levels by the IR receive decoder (external from the MCU). The narrow pulses are then stretched by the infrared submodule to get back to a serial bit stream to be received by the SCI.The polarity of transmitted pulses and expected receive pulses can be inverted so that a direct connection can be made to external IrDA transceiver modules that uses active low pulses.

The infrared submodule receives its clock sources from the SCI. One of these two clocks are selected in the infrared submodule in order to generate either 3/16, 1/16, 1/32 or 1/4 narrow pulses during transmission. The infrared block receives two clock sources from the SCI, R16XCLK and R32XCLK, which are configured to generate the narrow pulse width during transmission. The R16XCLK and R32XCLK are internal clocks with frequencies 16 and 32 times the baud rate respectively. Both R16XCLK and R32XCLK clocks are used for transmitting data. The receive decoder uses only the R16XCLK clock.

23.4.1.1 Infrared Transmit Encoder

The infrared transmit encoder converts serial bits of data from transmit shift register to the TXD pin. A narrow pulse is transmitted for a zero bit and no pulse for a one bit. The narrow pulse is sent in the middle of the bit with a duration of 1/32, 1/16, 3/16 or 1/4 of a bit time. A narrow high pulse is transmitted for a zero bit when TXPOL is cleared, while a narrow low pulse is transmitted for a zero bit when TXPOL is set.

23.4.1.2 Infrared Receive Decoder

The infrared receive block converts data from the RXD pin to the receive shift register. A narrow pulse is expected for each zero received and no pulse is expected for each one received. A narrow high pulse is expected for a zero bit when RXPOL is cleared, while a narrow low pulse is expected for a zero bit when RXPOL is set. This receive decoder meets the edge jitter requirement as defined by the IrDA serial infrared physical layer specification.

23.4.2 LIN Support

This module provides some basic support for the LIN protocol. At first this is a break detect circuitry making it easier for the LIN software to distinguish a break character from an incoming data stream. As a further addition is supports a collision detection at the bit level as well as cancelling pending transmissions.

23.4.3 Data Format

The SCI uses the standard NRZ mark/space data format. When Infrared is enabled, the SCI uses RZI data format where zeroes are represented by light pulses and ones remain low. See Figure 23-15 below.

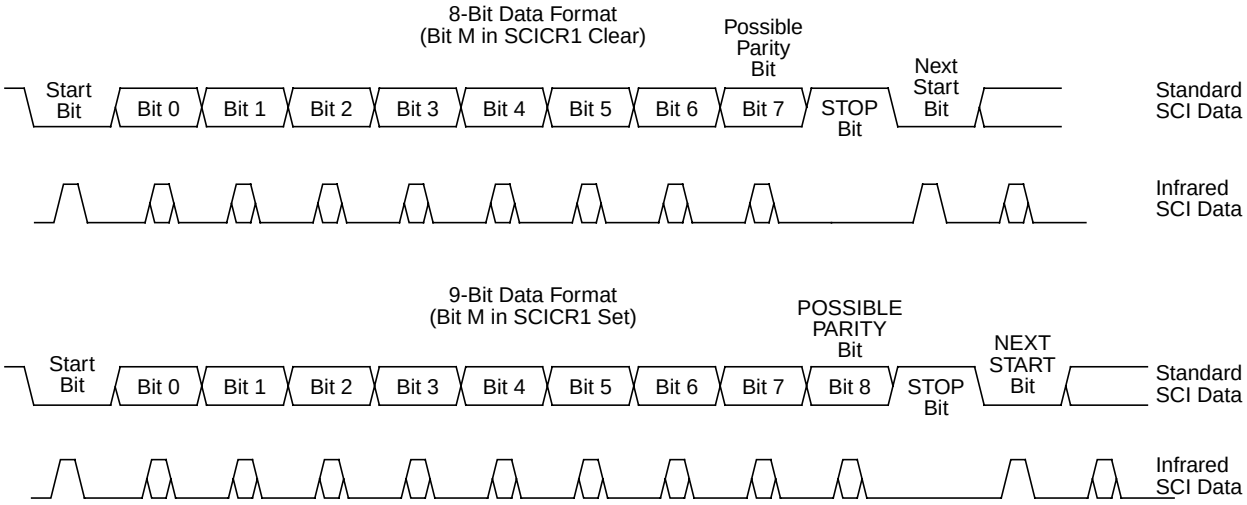


Figure 23-15. SCI Data Formats

Each data character is contained in a frame that includes a start bit, eight or nine data bits, and a stop bit. Clearing the M bit in SCI control register 1 configures the SCI for 8-bit data characters. A frame with eight data bits has a total of 10 bits. Setting the M bit configures the SCI for nine-bit data characters. A frame with nine data bits has a total of 11 bits.

Table 23-14. Example of 8-Bit Data Formats

Start Bit	Data Bits	Address Bits	Parity Bits	Stop Bit
1	8	0	0	1
1	7	0	1	1
1	7	1(1)	0	1

1. The address bit identifies the frame as an address character. See Section 23.4.6.6, “Receiver Wakeup”.

When the SCI is configured for 9-bit data characters, the ninth data bit is the T8 bit in SCI data register high (SCIDRH). It remains unchanged after transmission and can be used repeatedly without rewriting it. A frame with nine data bits has a total of 11 bits.

Table 23-15. Example of 9-Bit Data Formats

Start Bit	Data Bits	Address Bits	Parity Bits	Stop Bit
1	9	0	0	1
1	8	0	1	1
1	8	1(1)	0	1

1. The address bit identifies the frame as an address character. See Section 23.4.6.6, “Receiver Wakeup”.

23.4.4 Baud Rate Generation

A 13-bit modulus counter in the baud rate generator derives the baud rate for both the receiver and the transmitter. The value from 0 to 8191 written to the SBR12:SBR0 bits determines the bus clock divisor. The SBR bits are in the SCI baud rate registers (SCIBDH and SCIBDL). The baud rate clock is synchronized with the bus clock and drives the receiver. The baud rate clock divided by 16 drives the transmitter. The receiver has an acquisition rate of 16 samples per bit time.

Baud rate generation is subject to one source of error:

- Integer division of the bus clock may not give the exact target frequency.

Table 23-16 lists some examples of achieving target baud rates with a bus clock frequency of 25 MHz.

When IREN = 0 then,

SCI baud rate = SCI bus clock / (16 * SCIBR[12:0])

Table 23-16. Baud Rates (Example: Bus Clock = 25 MHz)

Bits SBR[12:0]	Receiver Clock (Hz)	Transmitter Clock (Hz)	Target Baud Rate	Error (%)
41	609,756.1	38,109.8	38,400	.76
81	308,642.0	19,290.1	19,200	.47
163	153,374.2	9585.9	9,600	.16
326	76,687.1	4792.9	4,800	.15
651	38,402.5	2400.2	2,400	.01
1302	19,201.2	1200.1	1,200	.01
2604	9600.6	600.0	600	.00
5208	4800.0	300.0	300	.00

23.4.5 Transmitter

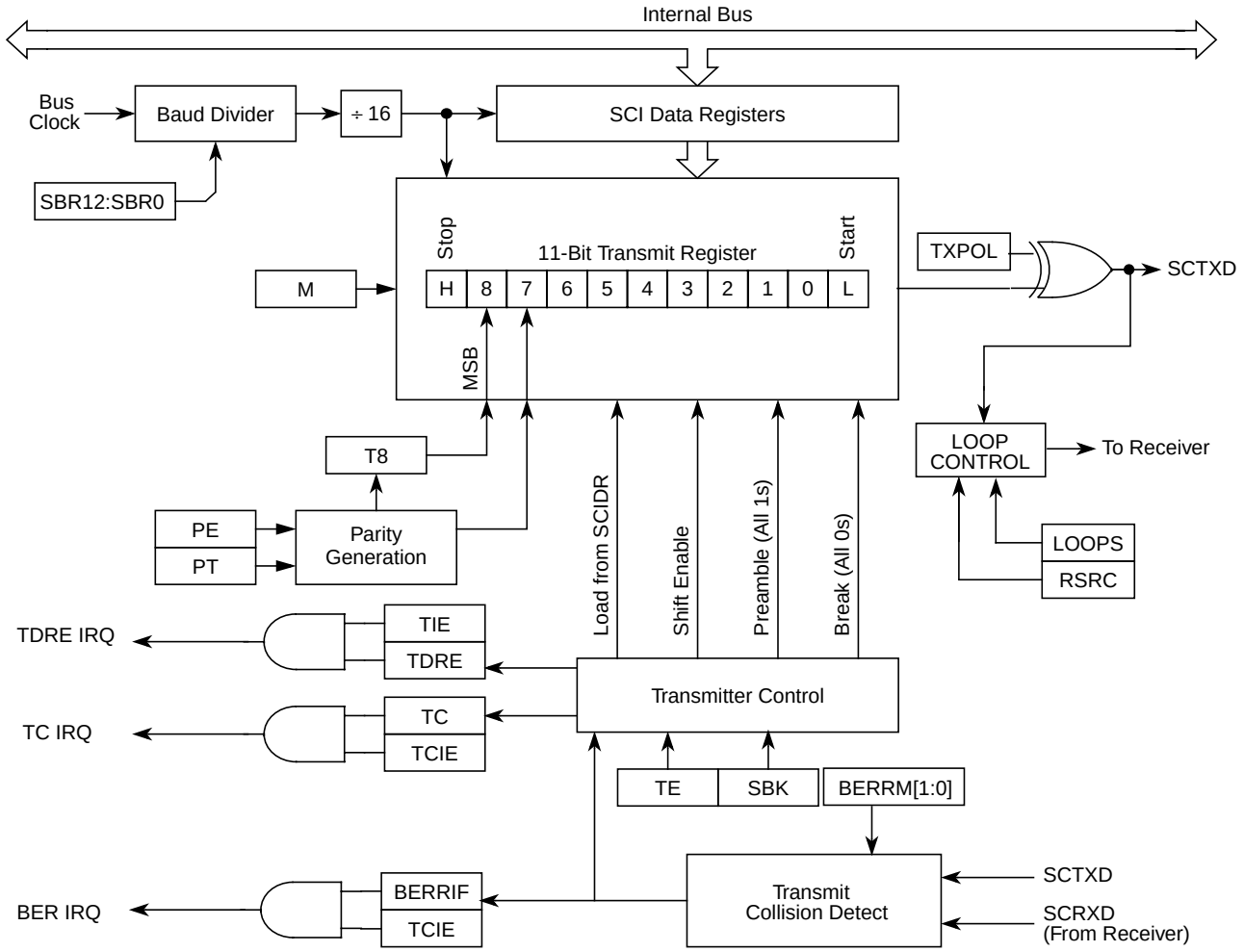


Figure 23-16. Transmitter Block Diagram

23.4.5.1 Transmitter Character Length

The SCI transmitter can accommodate either 8-bit or 9-bit data characters. The state of the M bit in SCI control register 1 (SCICR1) determines the length of data characters. When transmitting 9-bit data, bit T8 in SCI data register high (SCIDRH) is the ninth bit (bit 8).

23.4.5.2 Character Transmission

To transmit data, the MCU writes the data bits to the SCI data registers (SCIDRH/SCIDRL), which in turn are transferred to the transmitter shift register. The transmit shift register then shifts a frame out through the TXD pin, after it has prefaced them with a start bit and appended them with a stop bit. The SCI data registers (SCIDRH and SCIDRL) are the write-only buffers between the internal data bus and the transmit shift register.

The SCI also sets a flag, the transmit data register empty flag (TDRE), every time it transfers data from the buffer (SCIDRH/L) to the transmitter shift register.The transmit driver routine may respond to this flag by writing another byte to the Transmitter buffer (SCIDRH/SCIDRL), while the shift register is still shifting out the first byte.

To initiate an SCI transmission:

1. Configure the SCI:
   a) Select a baud rate. Write this value to the SCI baud registers (SCIBDH/L) to begin the baud rate generator. Remember that the baud rate generator is disabled when the baud rate is zero. Writing to the SCIBDH has no effect without also writing to SCIBDL.
   b) Write to SCICR1 to configure word length, parity, and other configuration bits (LOOPS,RSRC,M,WAKE,ILT,PE,PT).
   c) Enable the transmitter, interrupts, receive, and wake up as required, by writing to the SCICR2 register bits (TIE,TCIE,RIE,ILIE,TE,RE,RWU,SBK). A preamble or idle character will now be shifted out of the transmitter shift register.
2. Transmit Procedure for each byte:
   a) Poll the TDRE flag by reading the SCISR1 or responding to the TDRE interrupt. Keep in mind that the TDRE bit resets to one.
   b) If the TDRE flag is set, write the data to be transmitted to SCIDRH/L, where the ninth bit is written to the T8 bit in SCIDRH if the SCI is in 9-bit data format. A new transmission will not result until the TDRE flag has been cleared.
3. Repeat step 2 for each subsequent transmission.

NOTE

The TDRE flag is set when the shift register is loaded with the next data to be transmitted from SCIDRH/L, which happens, generally speaking, a little over half-way through the stop bit of the previous frame. Specifically, this transfer occurs 9/16ths of a bit time AFTER the start of the stop bit of the previous frame.

Writing the TE bit from 0 to a 1 automatically loads the transmit shift register with a preamble of 10 logic 1s (if M = 0) or 11 logic 1s (if M = 1). After the preamble shifts out, control logic transfers the data from the SCI data register into the transmit shift register. A logic 0 start bit automatically goes into the least significant bit position of the transmit shift register. A logic 1 stop bit goes into the most significant bit position.

Hardware supports odd or even parity. When parity is enabled, the most significant bit (MSB) of the data character is the parity bit.

The transmit data register empty flag, TDRE, in SCI status register 1 (SCISR1) becomes set when the SCI data register transfers a byte to the transmit shift register. The TDRE flag indicates that the SCI data register can accept new data from the internal data bus. If the transmit interrupt enable bit, TIE, in SCI control register 2 (SCICR2) is also set, the TDRE flag generates a transmitter interrupt request.

When the transmit shift register is not transmitting a frame, the TXD pin goes to the idle condition, logic 1. If at any time software clears the TE bit in SCI control register 2 (SCICR2), the transmitter enable signal goes low and the transmit signal goes idle.

If software clears TE while a transmission is in progress (TC = 0), the frame in the transmit shift register continues to shift out. To avoid accidentally cutting off the last frame in a message, always wait for TDRE to go high after the last frame before clearing TE.

To separate messages with preambles with minimum idle line time, use this sequence between messages:

1. Write the last byte of the first message to SCIDRH/L.
2. Wait for the TDRE flag to go high, indicating the transfer of the last frame to the transmit shift register.
3. Queue a preamble by clearing and then setting the TE bit.
4. Write the first byte of the second message to SCIDRH/L.

23.4.5.3 Break Characters

Writing a logic 1 to the send break bit, SBK, in SCI control register 2 (SCICR2) loads the transmit shift register with a break character. A break character contains all logic 0s and has no start, stop, or parity bit. Break character length depends on the M bit in SCI control register 1 (SCICR1). As long as SBK is at logic 1, transmitter logic continuously loads break characters into the transmit shift register. After software clears the SBK bit, the shift register finishes transmitting the last break character and then transmits at least one logic 1. The automatic logic 1 at the end of a break character guarantees the recognition of the start bit of the next frame.

The SCI recognizes a break character when there are 10 or 11(M = 0 or M = 1) consecutive zero received. Depending if the break detect feature is enabled or not receiving a break character has these effects on SCI registers.

If the break detect feature is disabled (BKDFE = 0):

- Sets the framing error flag, FE
- Sets the receive data register full flag, RDRF
- Clears the SCI data registers (SCIDRH/L)
- May set the overrun flag, OR, noise flag, NF, parity error flag, PE, or the receiver active flag, RAF (see 3.4.4 and 3.4.5 SCI Status Register 1 and 2)

If the break detect feature is enabled (BKDFE = 1) there are two scenarios[1]

The break is detected right from a start bit or is detected during a byte reception.

- Sets the break detect interrupt flag, BKDIF
- Does not change the data register full flag, RDRF or overrun flag OR
- Does not change the framing error flag FE, parity error flag PE.
- Does not clear the SCI data registers (SCIDRH/L)
- May set noise flag NF, or receiver active flag RAF.

1. A Break character in this context are either 10 or 11 consecutive zero received bits

Figure 23-17 shows two cases of break detect. In trace RXD_1 the break symbol starts with the start bit, while in RXD_2 the break starts in the middle of a transmission. If BRKDFE = 1, in RXD_1 case there will be no byte transferred to the receive buffer and the RDRF flag will not be modified. Also no framing error or parity error will be flagged from this transfer. In RXD_2 case, however the break signal starts later during the transmission. At the expected stop bit position the byte received so far will be transferred to the receive buffer, the receive data register full flag will be set, a framing error and if enabled and appropriate a parity error will be set. Once the break is detected the BRKDIF flag will be set.

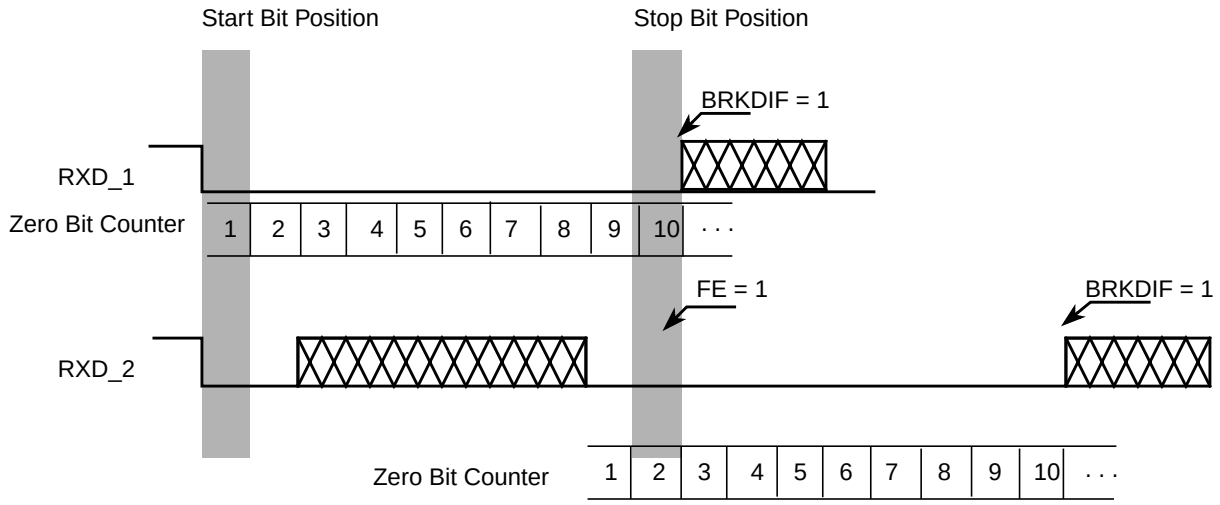


Figure 23-17. Break Detection if BRKDFE = 1 (M = 0)

23.4.5.4 Idle Characters

An idle character (or preamble) contains all logic 1s and has no start, stop, or parity bit. Idle character length depends on the M bit in SCI control register 1 (SCICR1). The preamble is a synchronizing idle character that begins the first transmission initiated after writing the TE bit from 0 to 1.

If the TE bit is cleared during a transmission, the TXD pin becomes idle after completion of the transmission in progress. Clearing and then setting the TE bit during a transmission queues an idle character to be sent after the frame currently being transmitted.

NOTE

When queueing an idle character, return the TE bit to logic 1 before the stop bit of the current frame shifts out through the TXD pin. Setting TE after the stop bit appears on TXD causes data previously written to the SCI data register to be lost. Toggle the TE bit for a queued idle character while the TDRE flag is set and immediately before writing the next byte to the SCI data register.

If the TE bit is clear and the transmission is complete, the SCI is not the master of the TXD pin

23.4.5.5 LIN Transmit Collision Detection

This module allows to check for collisions on the LIN bus.

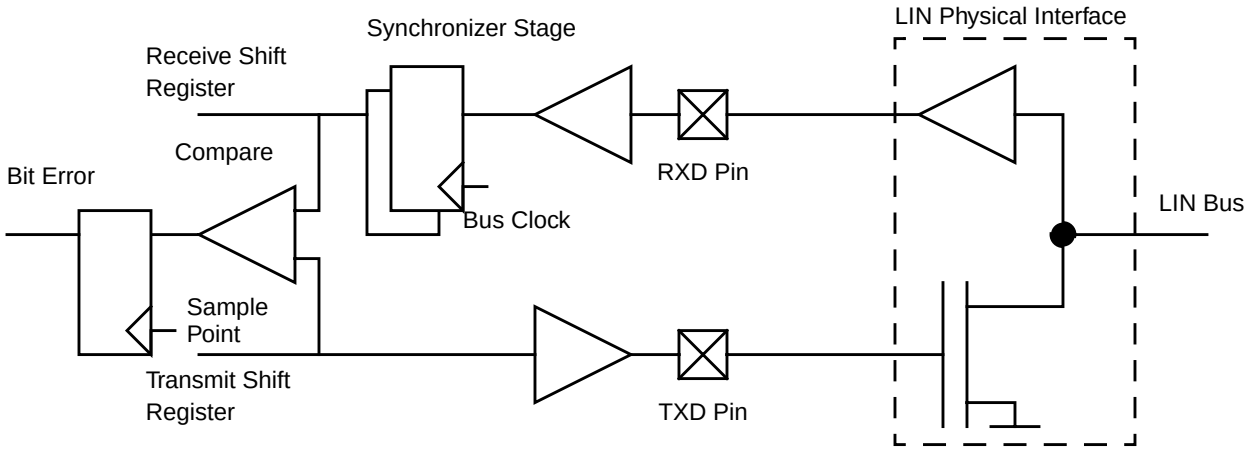


Figure 23-18. Collision Detect Principle

If the bit error circuit is enabled (BERRM[1:0] = 0:1 or = 1:0]), the error detect circuit will compare the transmitted and the received data stream at a point in time and flag any mismatch. The timing checks run when transmitter is active (not idle). As soon as a mismatch between the transmitted data and the received data is detected the following happens:

- The next bit transmitted will have a high level (TXPOL = 0) or low level (TXPOL = 1)
- The transmission is aborted and the byte in transmit buffer is discarded.
- the transmit data register empty and the transmission complete flag will be set
- The bit error interrupt flag, BERRIF, will be set.
- No further transmissions will take place until the BERRIF is cleared.

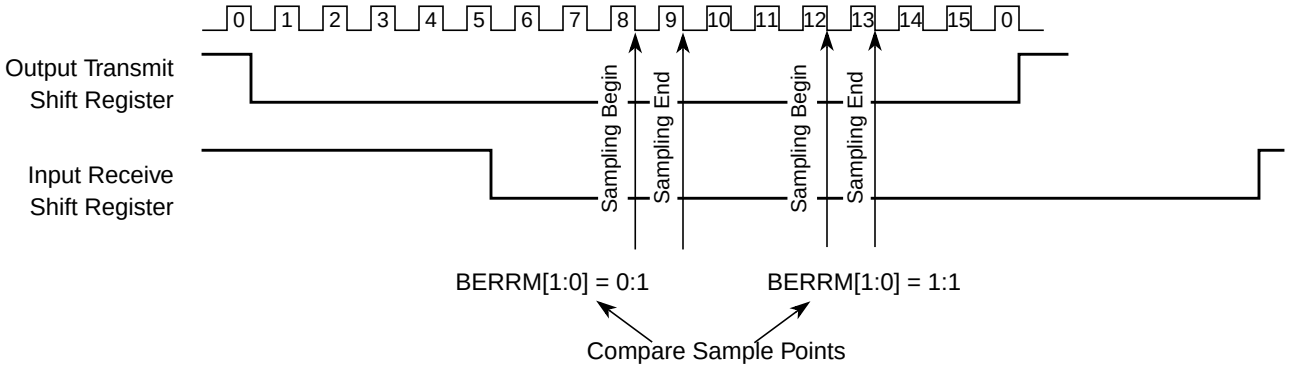


Figure 23-19. Timing Diagram Bit Error Detection

If the bit error detect feature is disabled, the bit error interrupt flag is cleared.

NOTE

The RXPOL and TXPOL bit should be set the same when transmission collision detect feature is enabled, otherwise the bit error interrupt flag may be set incorrectly.

23.4.6 Receiver

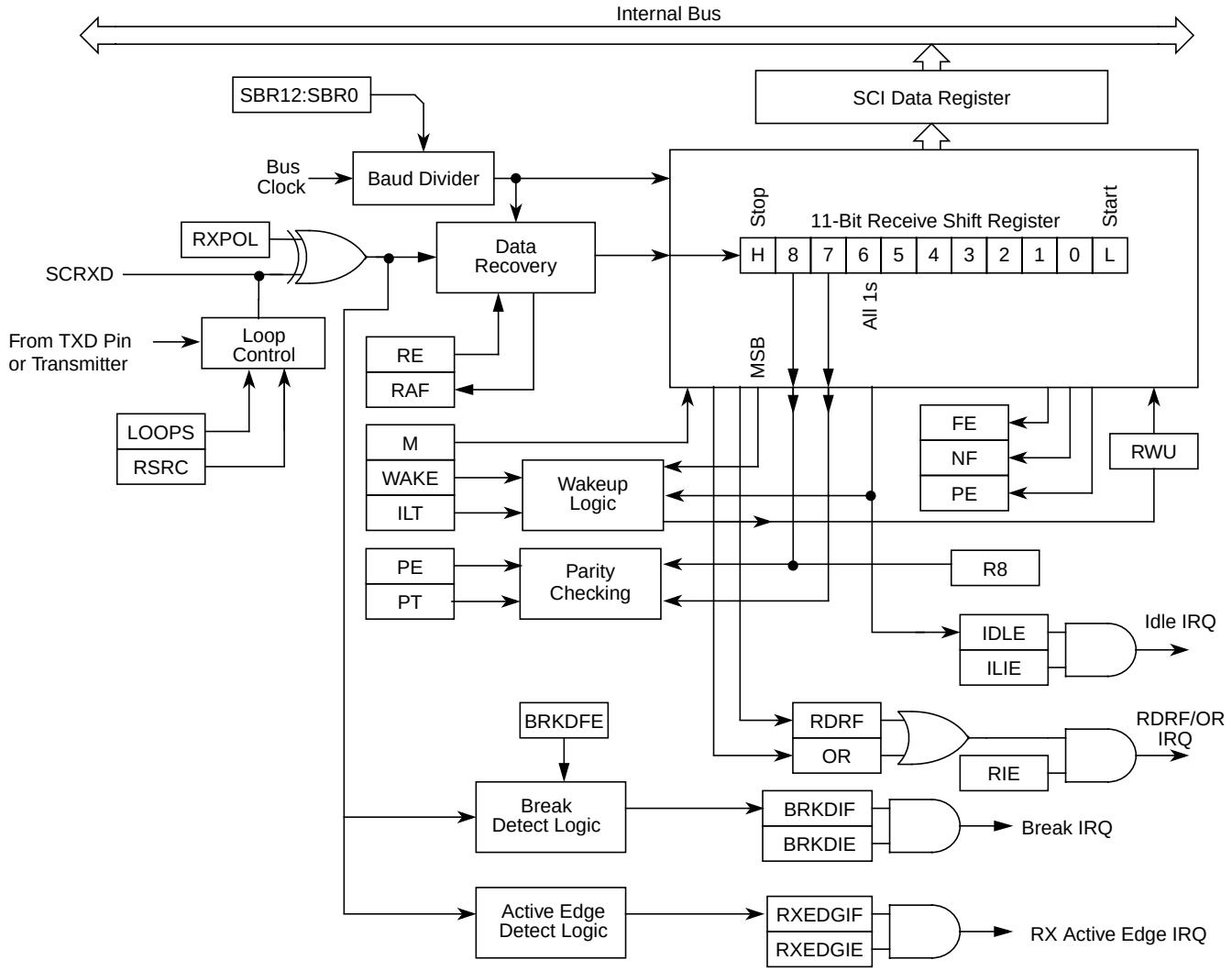


Figure 23-20. SCI Receiver Block Diagram

23.4.6.1 Receiver Character Length

The SCI receiver can accommodate either 8-bit or 9-bit data characters. The state of the M bit in SCI control register 1 (SCICR1) determines the length of data characters. When receiving 9-bit data, bit R8 in SCI data register high (SCIDRH) is the ninth bit (bit 8).

23.4.6.2 Character Reception

During an SCI reception, the receive shift register shifts a frame in from the RXD pin. The SCI data register is the read-only buffer between the internal data bus and the receive shift register.

After a complete frame shifts into the receive shift register, the data portion of the frame transfers to the SCI data register. The receive data register full flag, RDRF, in SCI status register 1 (SCISR1) becomes set,

indicating that the received byte can be read. If the receive interrupt enable bit, RIE, in SCI control register 2 (SCICR2) is also set, the RDRF flag generates an RDRF interrupt request.

23.4.6.3 Data Sampling

The RT clock rate. The RT clock is an internal signal with a frequency 16 times the baud rate. To adjust for baud rate mismatch, the RT clock (see Figure 23-21) is re-synchronized:

- After every start bit
- After the receiver detects a data bit change from logic 1 to logic 0 (after the majority of data bit samples at RT8, RT9, and RT10 returns a valid logic 1 and the majority of the next RT8, RT9, and RT10 samples returns a valid logic 0)

To locate the start bit, data recovery logic does an asynchronous search for a logic 0 preceded by three logic 1s.When the falling edge of a possible start bit occurs, the RT clock begins to count to 16.

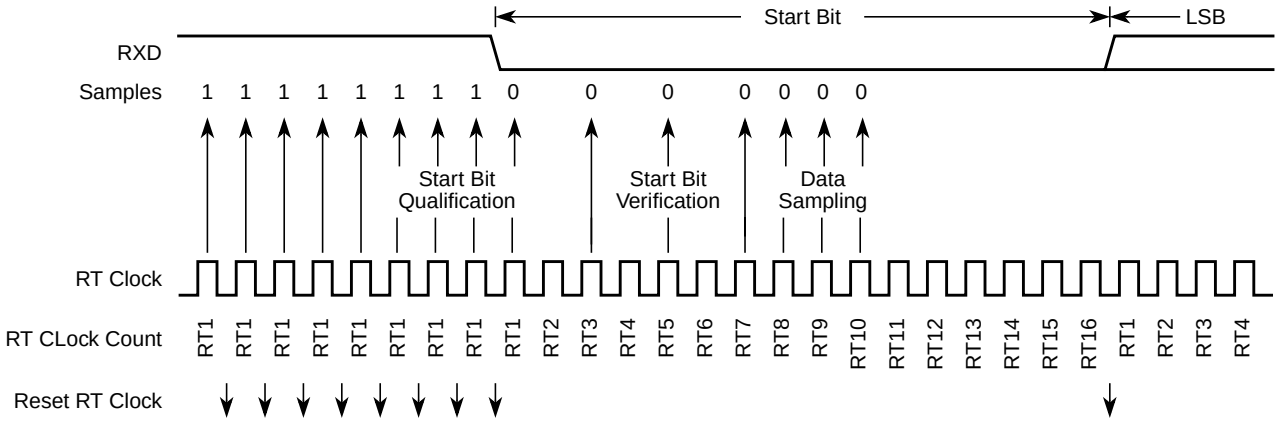


Figure 23-21. Receiver Data Sampling

To verify the start bit and to detect noise, data recovery logic takes samples at RT3, RT5, and RT7. Figure 23-17 summarizes the results of the start bit verification samples.

Table 23-17. Start Bit Verification

RT3, RT5, and RT7 Samples	Start Bit Verification	Noise Flag
000	Yes	0
001	Yes	1
010	Yes	1
011	No	0
100	Yes	1
101	No	0
110	No	0
111	No	0

If start bit verification is not successful, the RT clock is reset and a new search for a start bit begins.

To determine the value of a data bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. Table 23-18 summarizes the results of the data bit samples.

Table 23-18. Data Bit Recovery

RT8, RT9, and RT10 Samples	Data Bit Determination	Noise Flag
000	0	0
001	0	1
010	0	1
011	1	1
100	0	1
101	1	1
110	1	1
111	1	0

NOTE

The RT8, RT9, and RT10 samples do not affect start bit verification. If any or all of the RT8, RT9, and RT10 start bit samples are logic 1s following a successful start bit verification, the noise flag (NF) is set and the receiver assumes that the bit is a start bit (logic 0).

To verify a stop bit and to detect noise, recovery logic takes samples at RT8, RT9, and RT10. Table 23-19 summarizes the results of the stop bit samples.

Table 23-19. Stop Bit Recovery

RT8, RT9, and RT10 Samples	Framing Error Flag	Noise Flag
000	1	0
001	1	1
010	1	1
011	0	1
100	1	1
101	0	1
110	0	1
111	0	0

In Figure 23-22 the verification samples RT3 and RT5 determine that the first low detected was noise and not the beginning of a start bit. The RT clock is reset and the start bit search begins again. The noise flag is not set because the noise occurred before the start bit was found.

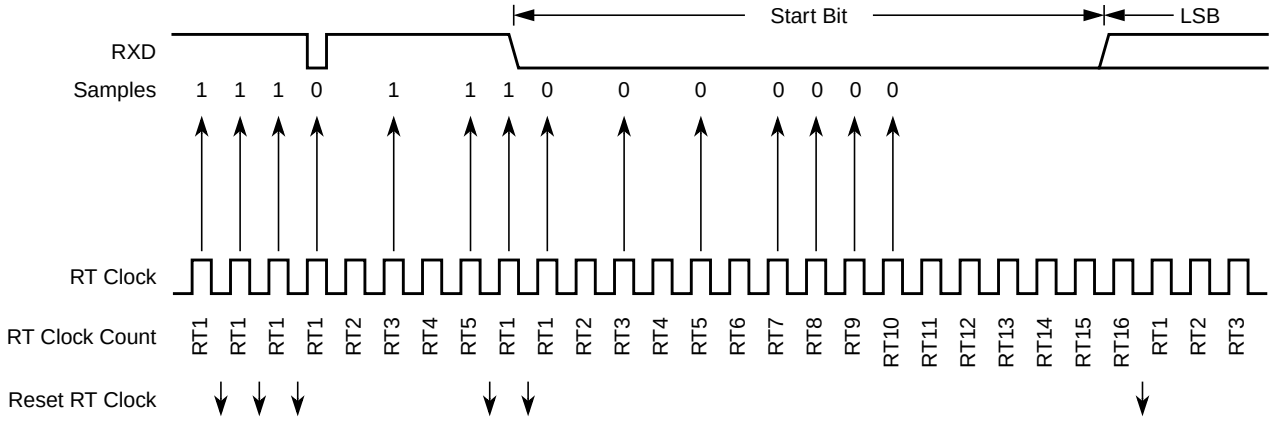


Figure 23-22. Start Bit Search Example 1

In Figure 23-23, verification sample at RT3 is high. The RT3 sample sets the noise flag. Although the perceived bit time is misaligned, the data samples RT8, RT9, and RT10 are within the bit time and data recovery is successful.

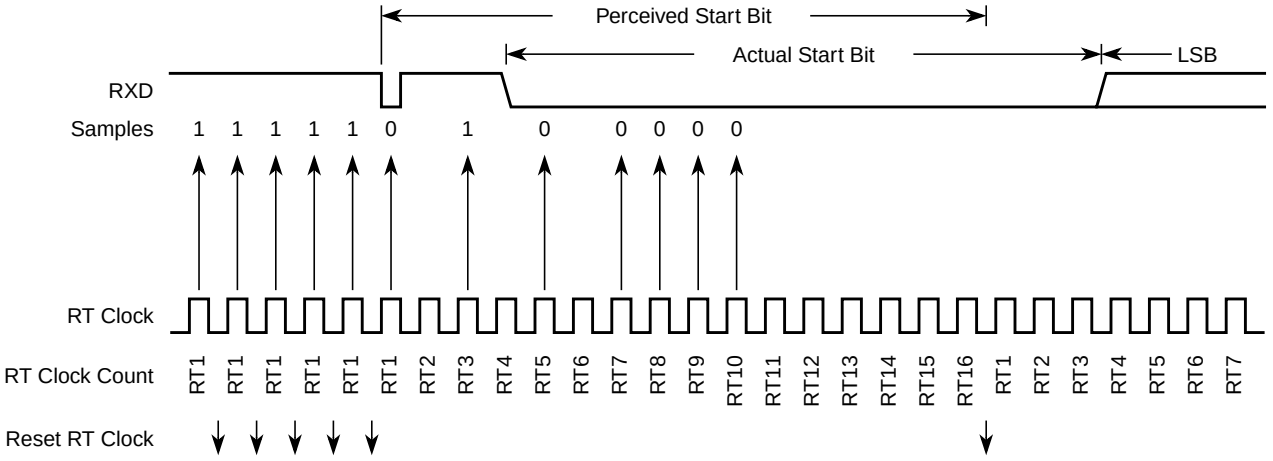


Figure 23-23. Start Bit Search Example 2

In Figure 23-24, a large burst of noise is perceived as the beginning of a start bit, although the test sample at RT5 is high. The RT5 sample sets the noise flag. Although this is a worst-case misalignment of perceived bit time, the data samples RT8, RT9, and RT10 are within the bit time and data recovery is successful.

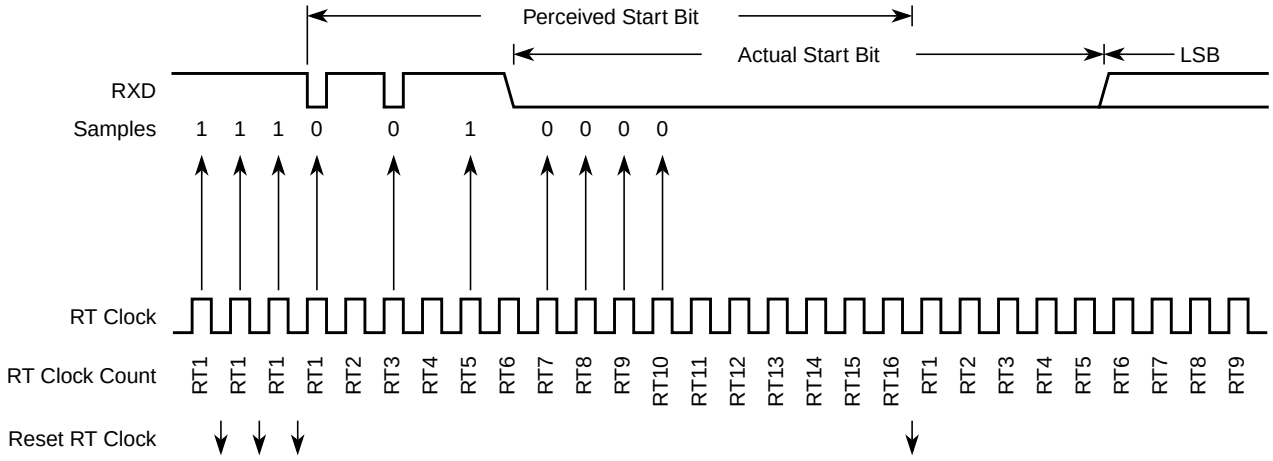


Figure 23-24. Start Bit Search Example 3

Figure 23-25 shows the effect of noise early in the start bit time. Although this noise does not affect proper synchronization with the start bit time, it does set the noise flag.

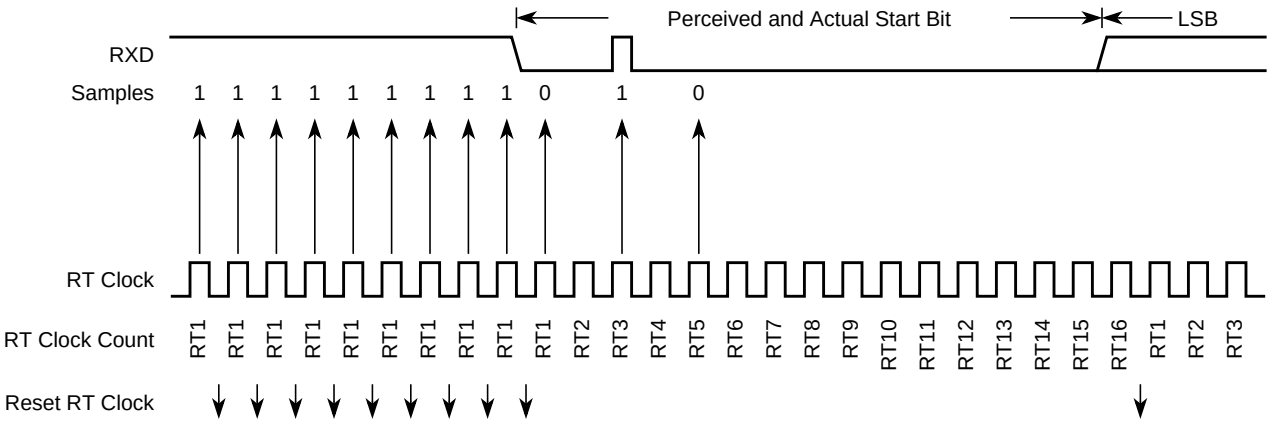


Figure 23-25. Start Bit Search Example 4

Figure 23-26 shows a burst of noise near the beginning of the start bit that resets the RT clock. The sample after the reset is low but is not preceded by three high samples that would qualify as a falling edge. Depending on the timing of the start bit search and on the data, the frame may be missed entirely or it may set the framing error flag.

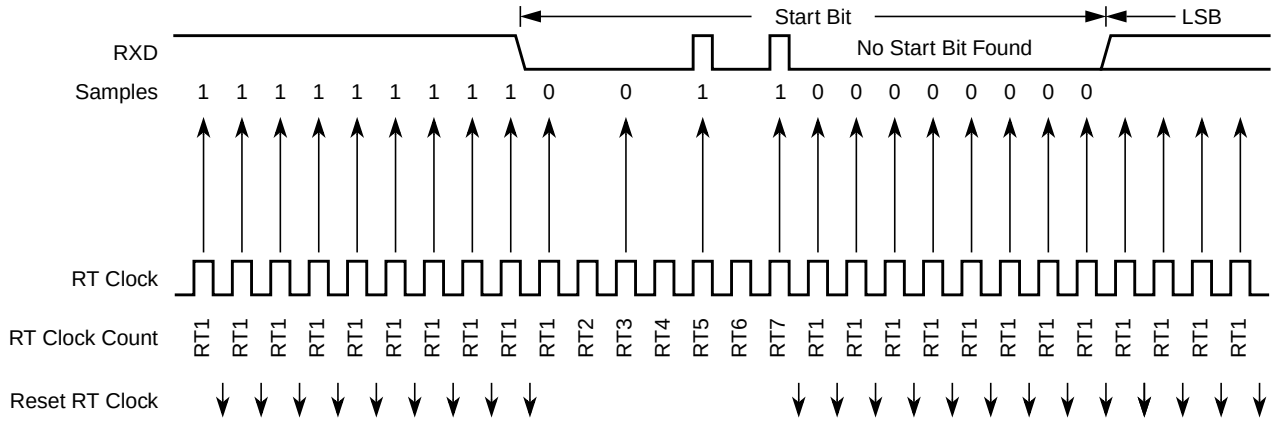


Figure 23-26. Start Bit Search Example 5

In Figure 23-27, a noise burst makes the majority of data samples RT8, RT9, and RT10 high. This sets the noise flag but does not reset the RT clock. In start bits only, the RT8, RT9, and RT10 data samples are ignored.

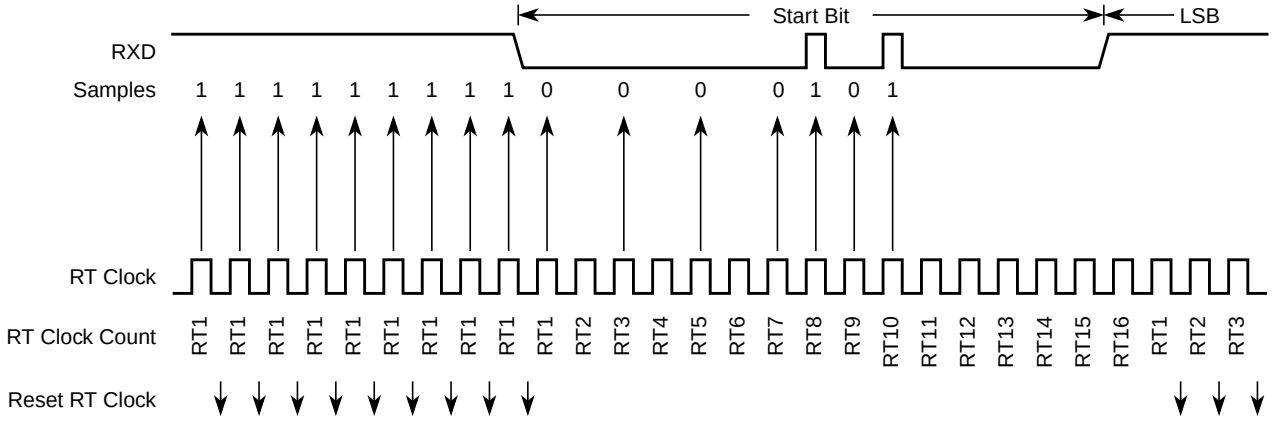


Figure 23-27. Start Bit Search Example 6

23.4.6.4 Framing Errors

If the data recovery logic does not detect a logic 1 where the stop bit should be in an incoming frame, it sets the framing error flag, FE, in SCI status register 1 (SCISR1). A break character also sets the FE flag because a break character has no stop bit. The FE flag is set at the same time that the RDRF flag is set.

23.4.6.5 Baud Rate Tolerance

A transmitting device may be operating at a baud rate below or above the receiver baud rate. Accumulated bit time misalignment can cause one of the three stop bit data samples (RT8, RT9, and RT10) to fall outside the actual stop bit. A noise error will occur if the RT8, RT9, and RT10 samples are not all the same logical values. A framing error will occur if the receiver clock is misaligned in such a way that the majority of the RT8, RT9, and RT10 stop bit samples are a logic zero.

As the receiver samples an incoming frame, it re-synchronizes the RT clock on any valid falling edge within the frame. Re synchronization within frames will correct a misalignment between transmitter bit times and receiver bit times.

23.4.6.5.1 Slow Data Tolerance

Figure 23-28 shows how much a slow received frame can be misaligned without causing a noise error or a framing error. The slow stop bit begins at RT8 instead of RT1 but arrives in time for the stop bit data samples at RT8, RT9, and RT10.

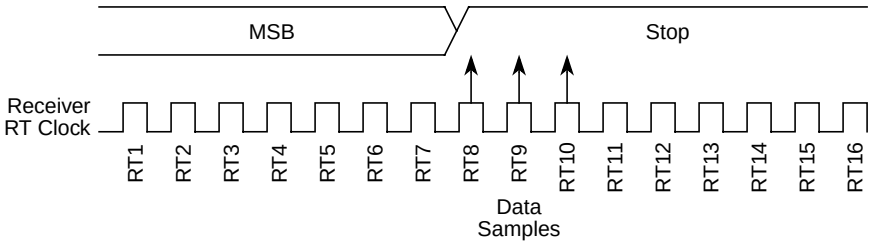


Figure 23-28. Slow Data

Let’s take RTr as receiver RT clock and RTt as transmitter RT clock.

For an 8-bit data character, it takes the receiver 9 bit times x 16 RTr cycles +7 RTr cycles = 151 RTr cycles to start data sampling of the stop bit.

With the misaligned character shown in Figure 23-28, the receiver counts 151 RTr cycles at the point when the count of the transmitting device is 9 bit times x 16 RTt cycles = 144 RTt cycles.

The maximum percent difference between the receiver count and the transmitter count of a slow 8-bit data character with no errors is:

((151 – 144) / 151) x 100 = 4.63%

For a 9-bit data character, it takes the receiver 10 bit times x 16 RTr cycles + 7 RTr cycles = 167 RTr cycles to start data sampling of the stop bit.

With the misaligned character shown in Figure 23-28, the receiver counts 167 RTr cycles at the point when the count of the transmitting device is 10 bit times x 16 RTt cycles = 160 RTt cycles.

The maximum percent difference between the receiver count and the transmitter count of a slow 9-bit character with no errors is:

((167 – 160) / 167) X 100 = 4.19%

23.4.6.5.2 Fast Data Tolerance

Figure 23-29 shows how much a fast received frame can be misaligned. The fast stop bit ends at RT10 instead of RT16 but is still sampled at RT8, RT9, and RT10.

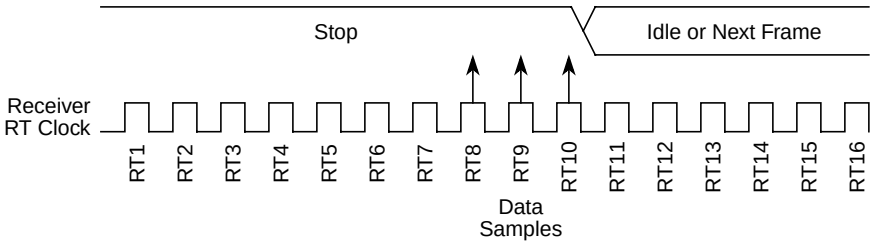


Figure 23-29. Fast Data

For an 8-bit data character, it takes the receiver 9 bit times x 16 RTr cycles + 10 RTr cycles = 154 RTr cycles to finish data sampling of the stop bit.

With the misaligned character shown in Figure 23-29, the receiver counts 154 RTr cycles at the point when the count of the transmitting device is 10 bit times x 16 RTt cycles = 160 RTt cycles.

The maximum percent difference between the receiver count and the transmitter count of a fast 8-bit character with no errors is:

((160 – 154) / 160) x 100 = 3.75%

For a 9-bit data character, it takes the receiver 10 bit times x 16 RTr cycles + 10 RTr cycles = 170 RTr cycles to finish data sampling of the stop bit.

With the misaligned character shown in Figure 23-29, the receiver counts 170 RTr cycles at the point when the count of the transmitting device is 11 bit times x 16 RTt cycles = 176 RTt cycles.

The maximum percent difference between the receiver count and the transmitter count of a fast 9-bit character with no errors is:

((176 – 170) /176) x 100 = 3.40%

23.4.6.6 Receiver Wakeup

To enable the SCI to ignore transmissions intended only for other receivers in multiple-receiver systems, the receiver can be put into a standby state. Setting the receiver wakeup bit, RWU, in SCI control register 2 (SCICR2) puts the receiver into standby state during which receiver interrupts are disabled.The SCI will still load the receive data into the SCIDRH/L registers, but it will not set the RDRF flag.

The transmitting device can address messages to selected receivers by including addressing information in the initial frame or frames of each message.

The WAKE bit in SCI control register 1 (SCICR1) determines how the SCI is brought out of the standby state to process an incoming message. The WAKE bit enables either idle line wakeup or address mark wakeup.

23.4.6.6.1 Idle Input line Wakeup (WAKE = 0)

In this wakeup method, an idle condition on the RXD pin clears the RWU bit and wakes up the SCI. The initial frame or frames of every message contain addressing information. All receivers evaluate the addressing information, and receivers for which the message is addressed process the frames that follow. Any receiver for which a message is not addressed can set its RWU bit and return to the standby state. The RWU bit remains set and the receiver remains on standby until another idle character appears on the RXD pin.

Idle line wakeup requires that messages be separated by at least one idle character and that no message contains idle characters.

The idle character that wakes a receiver does not set the receiver idle bit, IDLE, or the receive data register full flag, RDRF.

The idle line type bit, ILT, determines whether the receiver begins counting logic 1s as idle character bits after the start bit or after the stop bit. ILT is in SCI control register 1 (SCICR1).

23.4.6.6.2 Address Mark Wakeup (WAKE = 1)

In this wakeup method, a logic 1 in the most significant bit (MSB) position of a frame clears the RWU bit and wakes up the SCI. The logic 1 in the MSB position marks a frame as an address frame that contains addressing information. All receivers evaluate the addressing information, and the receivers for which the message is addressed process the frames that follow.Any receiver for which a message is not addressed can set its RWU bit and return to the standby state. The RWU bit remains set and the receiver remains on standby until another address frame appears on the RXD pin.

The logic 1 MSB of an address frame clears the receiver's RWU bit before the stop bit is received and sets the RDRF flag.

Address mark wakeup allows messages to contain idle characters but requires that the MSB be reserved for use in address frames.

NOTE

With the WAKE bit clear, setting the RWU bit after the RXD pin has been idle can cause the receiver to wake up immediately.

23.4.7 Single-Wire Operation

Normally, the SCI uses two pins for transmitting and receiving. In single-wire operation, the RXD pin is disconnected from the SCI. The SCI uses the TXD pin for both receiving and transmitting.

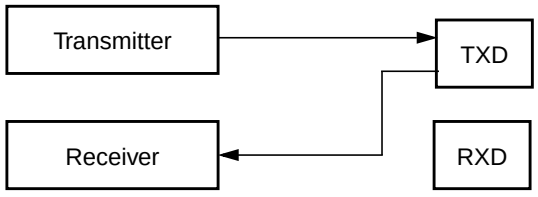


Figure 23-30. Single-Wire Operation (LOOPS = 1, RSRC = 1)

Enable single-wire operation by setting the LOOPS bit and the receiver source bit, RSRC, in SCI control register 1 (SCICR1). Setting the LOOPS bit disables the path from the RXD pin to the receiver. Setting the RSRC bit connects the TXD pin to the receiver. Both the transmitter and receiver must be enabled (TE = 1 and RE = 1).The TXDIR bit (SCISR2[1]) determines whether the TXD pin is going to be used as an input (TXDIR = 0) or an output (TXDIR = 1) in this mode of operation.

NOTE

In single-wire operation data from the TXD pin is inverted if RXPOL is set.

23.4.8 Loop Operation

In loop operation the transmitter output goes to the receiver input. The RXD pin is disconnected from the SCI.

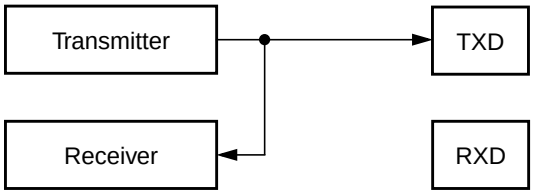


Figure 23-31. Loop Operation (LOOPS = 1, RSRC = 0)

Enable loop operation by setting the LOOPS bit and clearing the RSRC bit in SCI control register 1 (SCICR1). Setting the LOOPS bit disables the path from the RXD pin to the receiver. Clearing the RSRC bit connects the transmitter output to the receiver input. Both the transmitter and receiver must be enabled (TE = 1 and RE = 1).

NOTE

In loop operation data from the transmitter is not recognized by the receiver if RXPOL and TXPOL are not the same.

23.5 Initialization/Application Information

23.5.1 Reset Initialization

See Section 23.3.2, “Register Descriptions”.

23.5.2 Modes of Operation

23.5.2.1 Run Mode

Normal mode of operation.

To initialize a SCI transmission, see Section 23.4.5.2, “Character Transmission”.

23.5.2.2 Wait Mode

SCI operation in wait mode depends on the state of the SCISWAI bit in the SCI control register 1 (SCICR1).

- If SCISWAI is clear, the SCI operates normally when the CPU is in wait mode.
- If SCISWAI is set, SCI clock generation ceases and the SCI module enters a power-conservation state when the CPU is in wait mode. Setting SCISWAI does not affect the state of the receiver enable bit, RE, or the transmitter enable bit, TE.

  If SCISWAI is set, any transmission or reception in progress stops at wait mode entry. The transmission or reception resumes when either an internal or external interrupt brings the CPU out of wait mode. Exiting wait mode by reset aborts any transmission or reception in progress and resets the SCI.

23.5.2.3 Stop Mode

The SCI is inactive during stop mode for reduced power consumption. The STOP instruction does not affect the SCI register states, but the SCI bus clock will be disabled. The SCI operation resumes from where it left off after an external interrupt brings the CPU out of stop mode. Exiting stop mode by reset aborts any transmission or reception in progress and resets the SCI.

The receive input active edge detect circuit is still active in stop mode. An active edge on the receive input can be used to bring the CPU out of stop mode.

23.5.3 Interrupt Operation

This section describes the interrupt originated by the SCI block.The MCU must service the interrupt requests. Table 23-20 lists the eight interrupt sources of the SCI.

Table 23-20. SCI Interrupt Sources

Interrupt	Source	Local Enable	Description
TDRE	SCISR1[7]	TIE	Active high level. Indicates that a byte was transferred from SCIDRH/L to the transmit shift register.
TC	SCISR1[6]	TCIE	Active high level. Indicates that a transmit is complete.
RDRF	SCISR1[5]	RIE	Active high level. The RDRF interrupt indicates that received data is available in the SCI data register.
OR	SCISR1[3]		Active high level. This interrupt indicates that an overrun condition has occurred.
IDLE	SCISR1[4]	ILIE	Active high level. Indicates that receiver input has become idle.
RXEDGIF	SCIASR1[7]	RXEDGIE	Active high level. Indicates that an active edge (falling for RXPOL = 0, rising for RXPOL = 1) was detected.
BERRIF	SCIASR1[1]	BERRIE	Active high level. Indicates that a mismatch between transmitted and received data in a single wire application has happened.
BKDIF	SCIASR1[0]	BRKDIE	Active high level. Indicates that a break character has been received.

23.5.3.1 Description of Interrupt Operation

The SCI only originates interrupt requests. The following is a description of how the SCI makes a request and how the MCU should acknowledge that request. The interrupt vector offset and interrupt number are chip dependent. The SCI only has a single interrupt line (SCI Interrupt Signal, active high operation) and all the following interrupts, when generated, are ORed together and issued through that port.

23.5.3.1.1 TDRE Description

The TDRE interrupt is set high by the SCI when the transmit shift register receives a byte from the SCI data register. A TDRE interrupt indicates that the transmit data register (SCIDRH/L) is empty and that a new byte can be written to the SCIDRH/L for transmission.Clear TDRE by reading SCI status register 1 with TDRE set and then writing to SCI data register low (SCIDRL).

23.5.3.1.2 TC Description

The TC interrupt is set by the SCI when a transmission has been completed. Transmission is completed when all bits including the stop bit (if transmitted) have been shifted out and no data is queued to be transmitted. No stop bit is transmitted when sending a break character and the TC flag is set (providing there is no more data queued for transmission) when the break character has been shifted out. A TC interrupt indicates that there is no transmission in progress. TC is set high when the TDRE flag is set and no data, preamble, or break character is being transmitted. When TC is set, the TXD pin becomes idle (logic 1). Clear TC by reading SCI status register 1 (SCISR1) with TC set and then writing to SCI data register low (SCIDRL).TC is cleared automatically when data, preamble, or break is queued and ready to be sent.

23.5.3.1.3 RDRF Description

The RDRF interrupt is set when the data in the receive shift register transfers to the SCI data register. A RDRF interrupt indicates that the received data has been transferred to the SCI data register and that the byte can now be read by the MCU. The RDRF interrupt is cleared by reading the SCI status register one (SCISR1) and then reading SCI data register low (SCIDRL).

23.5.3.1.4 OR Description

The OR interrupt is set when software fails to read the SCI data register before the receive shift register receives the next frame. The newly acquired data in the shift register will be lost in this case, but the data already in the SCI data registers is not affected. The OR interrupt is cleared by reading the SCI status register one (SCISR1) and then reading SCI data register low (SCIDRL).

23.5.3.1.5 IDLE Description

The IDLE interrupt is set when 10 consecutive logic 1s (if M = 0) or 11 consecutive logic 1s (if M = 1) appear on the receiver input. Once the IDLE is cleared, a valid frame must again set the RDRF flag before an idle condition can set the IDLE flag. Clear IDLE by reading SCI status register 1 (SCISR1) with IDLE set and then reading SCI data register low (SCIDRL).

23.5.3.1.6 RXEDGIF Description

The RXEDGIF interrupt is set when an active edge (falling if RXPOL = 0, rising if RXPOL = 1) on the RXD pin is detected. Clear RXEDGIF by writing a “1” to the SCIASR1 SCI alternative status register 1.

23.5.3.1.7 BERRIF Description

The BERRIF interrupt is set when a mismatch between the transmitted and the received data in a single wire application like LIN was detected. Clear BERRIF by writing a “1” to the SCIASR1 SCI alternative status register 1. This flag is also cleared if the bit error detect feature is disabled.

23.5.3.1.8 BKDIF Description

The BKDIF interrupt is set when a break signal was received. Clear BKDIF by writing a “1” to the SCIASR1 SCI alternative status register 1. This flag is also cleared if break detect feature is disabled.

23.5.4 Recovery from Wait Mode

The SCI interrupt request can be used to bring the CPU out of wait mode.

23.5.5 Recovery from Stop Mode

An active edge on the receive input can be used to bring the CPU out of stop mode.

Chapter 24
Analog-to-Digital Converter (ADC12B16C) Block Description

Revision History

Version Number	Revision Date	Effective Date	Author	Description of Changes
V01.00	13 Oct. 2005	13 Oct. 2005		Initial version
V01.01	4 Mar. 2008	4 Mar. 2008		correchted reference that DJM bit is in ATDCTL3

24.1 Introduction

The ADC12B16C is a 16-channel, 12-bit, multiplexed input successive approximation analog-to-digital converter. Refer to device electrical specifications for ATD accuracy.

24.1.1 Features

- 8-, 10-, or 12-bit resolution.
- Conversion in Stop Mode using internally generated clock
- Automatic return to low power after conversion sequence
- Automatic compare with interrupt for higher than or less/equal than programmable value
- Programmable sample time.
- Left/right justified result data.
- External trigger control.
- Sequence complete interrupt.
- Analog input multiplexer for 16 analog input channels.
- Special conversions for V_{RH}, V_{RL}, $(V_{RL}+V_{RH})/2$.
- 1-to-16 conversion sequence lengths.
- Continuous conversion mode.
- Multiple channel scans.

- Configurable external trigger functionality on any AD channel or any of four additional trigger inputs. The four additional trigger inputs can be chip external or internal. Refer to device specification for availability and connectivity.
- Configurable location for channel wrap around (when converting multiple channels in a sequence).

24.1.2 Modes of Operation

24.1.2.1 Conversion Modes

There is software programmable selection between performing **single** or **continuous conversion** on a **single channel** or **multiple channels**.

24.1.2.2 MCU Operating Modes

- **Stop Mode**
  - **ICLKSTP=0 (in ATDCTL2 register)**
    Entering Stop Mode aborts any conversion sequence in progress and if a sequence was aborted restarts it after exiting stop mode. This has the same effect/consequences as starting a conversion sequence with write to ATDCTL5. So after exiting from stop mode with a previously aborted sequence all flags are cleared etc.
  - **ICLKSTP=1 (in ATDCTL2 register)**
    A/D conversion sequence seamless continues in Stop Mode based on the internally generated clock ICLK as ATD clock. For conversions during transition from Run to Stop Mode or vice versa the result is not written to the results register, no CCF flag is set and no compare is done. When converting in Stop Mode (ICLKSTP=1) an ATD Stop Recovery time $t_{ATDSTPRCV}$ is required to switch back to bus clock based ATDCLK when leaving Stop Mode. Do not access ATD registers during this time.
- **Wait Mode**
  ADC12B16C behaves same in Run and Wait Mode. For reduced power consumption continuos conversions should be aborted before entering Wait mode.
- **Freeze Mode**
  In Freeze Mode the ADC12B16C will either continue or finish or stop converting according to the FRZ1 and FRZ0 bits. This is useful for debugging and emulation.

24.1.3 Block Diagram

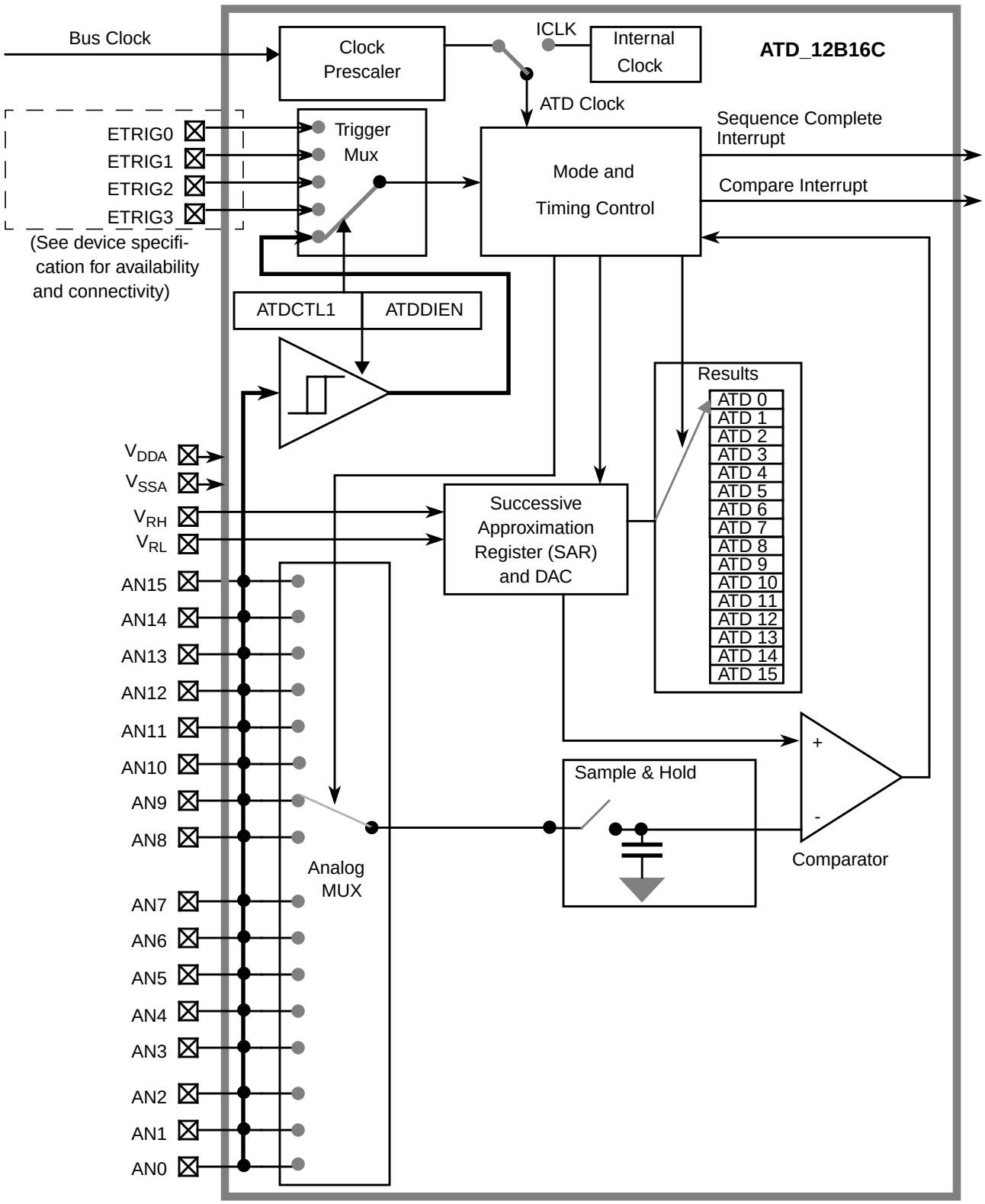


Figure 24-1. ADC12B16C Block Diagram

24.2 Signal Description

This section lists all inputs to the ADC12B16C block.

24.2.1 Detailed Signal Descriptions

24.2.1.1 AN*x* (*x* = 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0)

This pin serves as the analog input Channel *x*. It can also be configured as digital port or external trigger for the ATD conversion.

24.2.1.2 ETRIG3, ETRIG2, ETRIG1, ETRIG0

These inputs can be configured to serve as an external trigger for the ATD conversion.

Refer to device specification for availability and connection of these inputs!

24.2.1.3 V_{RH}, V_{RL}

V_{RH} is the high reference voltage, V_{RL} is the low reference voltage for ATD conversion.

24.2.1.4 V_{DDA}, V_{SSA}

These pins are the power supplies for the analog circuitry of the ADC12B16C block.

24.3 Memory Map and Register Definition

This section provides a detailed description of all registers accessible in the ADC12B16C.

24.3.1 Module Memory Map

Figure 24-2 gives an overview on all ADC12B16C registers.

NOTE

Register Address = Base Address + Address Offset, where the Base Address is defined at the MCU level and the Address Offset is defined at the module level.

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000	ATDCTL0	R W	Reserved	0	0	0	WRAP3	WRAP2	WRAP1	WRAP0
0x0001	ATDCTL1	R W	ETRIGSEL	SRES1	SRES0	SMP_DIS	ETRIGCH3	ETRIGCH2	ETRIGCH1	ETRIGCH0
0x0002	ATDCTL2	R W	0	AFFC	ICLKSTP	ETRIGLE	ETRIGP	ETRIGE	ASCIE	ACMPIE

= Unimplemented or Reserved

Figure 24-2. ADC12B16C Register Summary (Sheet 1 of 3)

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0003	ATDCTL3	R	DJM	S8C	S4C	S2C	S1C	FIFO	FRZ1	FRZ0
		W								
0x0004	ATDCTL4	R	SMP2	SMP1	SMP0	PRS[4:0]				
		W								
0x0005	ATDCTL5	R	0	SC	SCAN	MULT	CD	CC	CB	CA
		W								
0x0006	ATDSTAT0	R	SCF	0	ETORF	FIFOR	CC3	CC2	CC1	CC0
		W								
0x0007	Unimple-mented	R	0	0	0	0	0	0	0	0
		W								
0x0008	ATDCMPEH	R	CMPE[15:8]							
		W								
0x0009	ATDCMPEL	R	CMPE[7:0]							
		W								
0x000A	ATDSTAT2H	R	CCF[15:8]							
		W								
0x000B	ATDSTAT2L	R	CCF[7:0]							
		W								
0x000C	ATDDIENH	R	IEN[15:8]							
		W								
0x000D	ATDDIENL	R	IEN[7:0]							
		W								
0x000E	ATDCMPHTH	R	CMPHT[15:8]							
		W								
0x000F	ATDCMPHTL	R	CMPHT[7:0]							
		W								
0x0010	ATDDR0	R	See Section 24.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 24.3.2.12.2, “Right Justified Result Data (DJM=1)”							
		W								
0x0012	ATDDR1	R	See Section 24.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 24.3.2.12.2, “Right Justified Result Data (DJM=1)”							
		W								
0x0014	ATDDR2	R	See Section 24.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 24.3.2.12.2, “Right Justified Result Data (DJM=1)”							
		W								
0x0016	ATDDR3	R	See Section 24.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 24.3.2.12.2, “Right Justified Result Data (DJM=1)”							
		W								
0x0018	ATDDR4	R	See Section 24.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 24.3.2.12.2, “Right Justified Result Data (DJM=1)”							
		W								
0x001A	ATDDR5	R	See Section 24.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 24.3.2.12.2, “Right Justified Result Data (DJM=1)”							
		W								
0x001C	ATDDR6	R	See Section 24.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 24.3.2.12.2, “Right Justified Result Data (DJM=1)”							
		W								
0x001E	ATDDR7	R	See Section 24.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 24.3.2.12.2, “Right Justified Result Data (DJM=1)”							
		W								
0x0020	ATDDR8	R	See Section 24.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 24.3.2.12.2, “Right Justified Result Data (DJM=1)”							
		W								
0x0022	ATDDR9	R	See Section 24.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 24.3.2.12.2, “Right Justified Result Data (DJM=1)”							
		W								

= Unimplemented or Reserved

Figure 24-2. ADC12B16C Register Summary (Sheet 2 of 3)

Address	Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0024	ATDDR10	R W	See Section 24.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 24.3.2.12.2, “Right Justified Result Data (DJM=1)”							
0x0026	ATDDR11	R W	See Section 24.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 24.3.2.12.2, “Right Justified Result Data (DJM=1)”							
0x0028	ATDDR12	R W	See Section 24.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 24.3.2.12.2, “Right Justified Result Data (DJM=1)”							
0x002A	ATDDR13	R W	See Section 24.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 24.3.2.12.2, “Right Justified Result Data (DJM=1)”							
0x002C	ATDDR14	R W	See Section 24.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 24.3.2.12.2, “Right Justified Result Data (DJM=1)”							
0x002E	ATDDR15	R W	See Section 24.3.2.12.1, “Left Justified Result Data (DJM=0)” and Section 24.3.2.12.2, “Right Justified Result Data (DJM=1)”							

= Unimplemented or Reserved

Figure 24-2. ADC12B16C Register Summary (Sheet 3 of 3)

24.3.2 Register Descriptions

This section describes in address order all the ADC12B16C registers and their individual bits.

24.3.2.1 ATD Control Register 0 (ATDCTL0)

Writes to this register will abort current conversion sequence.

Module Base + 0x0000

	7	6	5	4	3	2	1	0
R	Reserved	0	0	0	WRAP3	WRAP2	WRAP1	WRAP0
W								
Reset	0	0	0	0	1	1	1	1

= Unimplemented or Reserved

Figure 24-3. ATD Control Register 0 (ATDCTL0)

Read: Anytime

Write: Anytime, in special modes always write 0 to Reserved Bit 7.

Table 24-1. ATDCTL0 Field Descriptions

Field	Description
3-0 WRAP[3-0]	**Wrap Around Channel Select Bits** — These bits determine the channel for wrap around when doing multi-channel conversions. The coding is summarized in Table 24-2.

Table 24-2. Multi-Channel Wrap Around Coding

WRAP3	WRAP2	WRAP1	WRAP0	Multiple Channel Conversions (MULT = 1) Wraparound to AN0 after Converting
0	0	0	0	Reserved(1)
0	0	0	1	AN1
0	0	1	0	AN2
0	0	1	1	AN3
0	1	0	0	AN4
0	1	0	1	AN5
0	1	1	0	AN6
0	1	1	1	AN7
1	0	0	0	AN8
1	0	0	1	AN9
1	0	1	0	AN10
1	0	1	1	AN11
1	1	0	0	AN12
1	1	0	1	AN13
1	1	1	0	AN14
1	1	1	1	AN15

1. If only AN0 should be converted use MULT=0.

24.3.2.2 ATD Control Register 1 (ATDCTL1)

Writes to this register will abort current conversion sequence.

Module Base + 0x0001

	7	6	5	4	3	2	1	0
R W	ETRIGSEL	SRES1	SRES0	SMP_DIS	ETRIGCH3	ETRIGCH2	ETRIGCH1	ETRIGCH0
Reset	0	0	1	0	1	1	1	1

Figure 24-4. ATD Control Register 1 (ATDCTL1)

Read: Anytime

Write: Anytime

Table 24-3. ATDCTL1 Field Descriptions

Field	Description
7 ETRIGSEL	**External Trigger Source Select** — This bit selects the external trigger source to be either one of the AD channels or one of the ETRIG3-0 inputs. See device specification for availability and connectivity of ETRIG3-0 inputs. If a particular ETRIG3-0 input option is not available, writing a 1 to ETRISEL only sets the bit but has not effect, this means that one of the AD channels (selected by ETRIGCH3-0) is configured as the source for external trigger. The coding is summarized in Table 24-5.
6–5 SRES[1:0]	**A/D Resolution Select** — These bits select the resolution of A/D conversion results. See Table 24-4 for coding.
4 SMP_DIS	**Discharge Before Sampling Bit** 0 No discharge before sampling. 1 The internal sample capacitor is discharged before sampling the channel. This adds 2 ATD clock cycles to the sampling time. This can help to detect an open circuit instead of measuring the previous sampled channel.
3–0 ETRIGCH[3:0]	**External Trigger Channel Select** — These bits select one of the AD channels or one of the ETRIG3-0 inputs as source for the external trigger. The coding is summarized in Table 24-5.

Table 24-4. A/D Resolution Coding

SRES1	SRES0	A/D Resolution
0	0	8-bit data
0	1	10-bit data
1	0	12-bit data
1	1	Reserved

Table 24-5. External Trigger Channel Select Coding

ETRIGSEL	ETRIGCH3	ETRIGCH2	ETRIGCH1	ETRIGCH0	External trigger source is
0	0	0	0	0	AN0
0	0	0	0	1	AN1
0	0	0	1	0	AN2
0	0	0	1	1	AN3
0	0	1	0	0	AN4
0	0	1	0	1	AN5
0	0	1	1	0	AN6
0	0	1	1	1	AN7
0	1	0	0	0	AN8
0	1	0	0	1	AN9
0	1	0	1	0	AN10
0	1	0	1	1	AN11
0	1	1	0	0	AN12
0	1	1	0	1	AN13
0	1	1	1	0	AN14
0	1	1	1	1	AN15
1	0	0	0	0	ETRIG0[(1)]
1	0	0	0	1	ETRIG1[1]
1	0	0	1	0	ETRIG2[1]
1	0	0	1	1	ETRIG3[1]
1	0	1	X	X	Reserved
1	1	X	X	X	Reserved

1. Only if ETRIG3-0 input option is available (see device specification), else ETRISEL is ignored, that means external trigger source is still on one of the AD channels selected by ETRIGCH3-0

24.3.2.3 ATD Control Register 2 (ATDCTL2)

Writes to this register will abort current conversion sequence.

Module Base + 0x0002

	7	6	5	4	3	2	1	0
R	0	AFFC	ICLKSTP	ETRIGLE	ETRIGP	ETRIGE	ASCIE	ACMPIE
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 24-5. ATD Control Register 2 (ATDCTL2)

Read: Anytime

Write: Anytime

Table 24-6. ATDCTL2 Field Descriptions

Field	Description
6 AFFC	**ATD Fast Flag Clear All** 0 ATD flag clearing done by write 1 to respective CCF[*n*] flag. 1 Changes all ATD conversion complete flags to a fast clear sequence. For compare disabled (CMPE[*n*]=0) a read access to the result register will cause the associated CCF[*n*] flag to clear automatically. For compare enabled (CMPE[*n*]=1) a write access to the result register will cause the associated CCF[*n*] flag to clear automatically.
5 ICLKSTP	**Internal Clock in Stop Mode Bit** — This bit enables A/D conversions in stop mode. When going into stop mode and ICLKSTP=1 the ATD conversion clock is automatically switched to the internally generated clock ICLK. Current conversion sequence will seamless continue. Conversion speed will change from prescaled bus frequency to the ICLK frequency (see ATD Electrical Characteristics in device description). The prescaler bits PRS4-0 in ATDCTL4 have no effect on the ICLK frequency. For conversions during stop mode the automatic compare interrupt or the sequence complete interrupt can be used to inform software handler about changing A/D values. External trigger will not work while converting in stop mode. For conversions during transition from Run to Stop Mode or vice versa the result is not written to the results register, no CCF flag is set and no compare is done. When converting in Stop Mode (ICLKSTP=1) an ATD Stop Recovery time $t_{ATDSTPRCV}$ is required to switch back to bus clock based ATDCLK when leaving Stop Mode. Do not access ATD registers during this time. 0 If A/D conversion sequence is ongoing when going into stop mode, the actual conversion sequence will be aborted and automatically restarted when exiting stop mode. 1 A/D continues to convert in stop mode using internally generated clock (ICLK)
4 ETRIGLE	**External Trigger Level/Edge Control** — This bit controls the sensitivity of the external trigger signal. See Table 24-7 for details.
3 ETRIGP	**External Trigger Polarity** — This bit controls the polarity of the external trigger signal. See Table 24-7 for details.
2 ETRIGE	**External Trigger Mode Enable** — This bit enables the external trigger on one of the AD channels or one of the ETRIG3-0 inputs as described in Table 24-5. If external trigger source is one of the AD channels, the digital input buffer of this channel is enabled. The external trigger allows to synchronize the start of conversion with external events. External trigger will not work while converting in stop mode. 0 Disable external trigger 1 Enable external trigger
1 ASCIE	**ATD Sequence Complete Interrupt Enable** 0 ATD Sequence Complete interrupt requests are disabled. 1 ATD Sequence Complete interrupt will be requested whenever SCF=1 is set.
0 ACMPIE	**ATD Compare Interrupt Enable** — If automatic compare is enabled for conversion *n* (CMPE[*n*]=1 in ATDCMPE register) this bit enables the compare interrupt. If the CCF[*n*] flag is set (showing a successful compare for conversion *n*), the compare interrupt is triggered. 0 ATD Compare interrupt requests are disabled. 1 For the conversions in a sequence for which automatic compare is enabled (CMPE[*n*]=1), ATD Compare Interrupt will be requested whenever any of the respective CCF flags is set.

Table 24-7. External Trigger Configurations

ETRIGLE	ETRIGP	External Trigger Sensitivity
0	0	Falling edge
0	1	Rising edge
1	0	Low level
1	1	High level

24.3.2.4 ATD Control Register 3 (ATDCTL3)

Writes to this register will abort current conversion sequence.

Module Base + 0x0003

	7	6	5	4	3	2	1	0
R W	DJM	S8C	S4C	S2C	S1C	FIFO	FRZ1	FRZ0
Reset	0	0	1	0	0	0	0	0

= Unimplemented or Reserved

Figure 24-6. ATD Control Register 3 (ATDCTL3)

Read: Anytime

Write: Anytime

Table 24-8. ATDCTL3 Field Descriptions

Field	Description
7 DJM	**Result Register Data Justification** — Result data format is always unsigned. This bit controls justification of conversion data in the result registers. 0 Left justified data in the result registers. 1 Right justified data in the result registers. Table 24-9 gives examples ATD results for an input signal range between 0 and 5.12 Volts.
6–3 S8C, S4C, S2C, S1C	**Conversion Sequence Length** — These bits control the number of conversions per sequence. Table 24-10 shows all combinations. At reset, S4C is set to 1 (sequence length is 4). This is to maintain software continuity to HC12 family.
2 FIFO	**Result Register FIFO Mode** — If this bit is zero (non-FIFO mode), the A/D conversion results map into the result registers based on the conversion sequence; the result of the first conversion appears in the first result register (ATDDR0), the second result in the second result register (ATDDR1), and so on. If this bit is one (FIFO mode) the conversion counter is not reset at the beginning or ending of a conversion sequence; sequential conversion results are placed in consecutive result registers. In a continuously scanning conversion sequence, the result register counter will wrap around when it reaches the end of the result register file. The conversion counter value (CC3-0 in ATDSTAT0) can be used to determine where in the result register file, the current conversion result will be placed. Aborting a conversion or starting a new conversion clears the conversion counter even if FIFO=1. So the first result of a new conversion sequence, started by writing to ATDCTL5, will always be place in the first result register (ATDDDR0). Intended usage of FIFO mode is continuos conversion (SCAN=1) or triggered conversion (ETRIG=1). Which result registers hold valid data can be tracked using the conversion complete flags. Fast flag clear mode may or may not be useful in a particular application to track valid data. If this bit is one, automatic compare of result registers is always disabled, that is ADC12B16C will behave as if ACMPIE and all CPME[*n*] were zero. 0 Conversion results are placed in the corresponding result register up to the selected sequence length. 1 Conversion results are placed in consecutive result registers (wrap around at end).
1–0 FRZ[1:0]	**Background Debug Freeze Enable** — When debugging an application, it is useful in many cases to have the ATD pause when a breakpoint (Freeze Mode) is encountered. These 2 bits determine how the ATD will respond to a breakpoint as shown in Table 24-11. Leakage onto the storage node and comparator reference capacitors may compromise the accuracy of an immediately frozen conversion depending on the length of the freeze period.

Table 24-9. Examples of ideal decimal ATD Results

Input Signal V_{RL} = 0 Volts V_{RH} = 5.12 Volts	8-Bit Codes (resolution=20mV)	10-Bit Codes (resolution=5mV)	12-Bit Codes (transfer curve has 1.25mV offset) (resolution=1.25mV)
5.120 Volts	255	1023	4095
...	...	...	...
0.022	1	4	17
0.020	1	4	16
0.018	1	4	14
0.016	1	3	12
0.014	1	3	11
0.012	1	2	9
0.010	1	2	8
0.008	0	2	6
0.006	0	1	4
0.004	0	1	3
0.003	0	0	2
0.002	0	0	1
0.000	0	0	0

Table 24-10. Conversion Sequence Length Coding

S8C	S4C	S2C	S1C	Number of Conversions per Sequence
0	0	0	0	16
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	10
1	0	1	1	11
1	1	0	0	12
1	1	0	1	13
1	1	1	0	14
1	1	1	1	15

Table 24-11. ATD Behavior in Freeze Mode (Breakpoint)

FRZ1	FRZ0	Behavior in Freeze Mode
0	0	Continue conversion
0	1	Reserved
1	0	Finish current conversion, then freeze
1	1	Freeze Immediately

24.3.2.5 ATD Control Register 4 (ATDCTL4)

Writes to this register will abort current conversion sequence.

Module Base + 0x0004

	7	6	5	4	3	2	1	0
R W	SMP2	SMP1	SMP0	PRS[4:0]				
Reset	0	0	0	0	0	1	0	1

Figure 24-7. ATD Control Register 4 (ATDCTL4)

Read: Anytime

Write: Anytime

Table 24-12. ATDCTL4 Field Descriptions

Field	Description
7–5 SMP[2:0]	**Sample Time Select** — These three bits select the length of the sample time in units of ATD conversion clock cycles. Note that the ATD conversion clock period is itself a function of the prescaler value (bits PRS4-0). Table 24-13 lists the available sample time lengths.
4–0 PRS[4:0]	**ATD Clock Prescaler** — These 5 bits are the binary prescaler value PRS. The ATD conversion clock frequency is calculated as follows: $f_{ATDCLK} = \frac{f_{BUS}}{2 \times (PRS + 1)}$ Refer to Device Specification for allowed frequency range of f_{ATDCLK}.

Table 24-13. Sample Time Select

SMP2	SMP1	SMP0	Sample Time in Number of ATD Clock Cycles
0	0	0	4
0	0	1	6
0	1	0	8
0	1	1	10
1	0	0	12
1	0	1	16
1	1	0	20

Table 24-13. Sample Time Select

SMP2	SMP1	SMP0	Sample Time in Number of ATD Clock Cycles
1	1	1	24

24.3.2.6 ATD Control Register 5 (ATDCTL5)

Writes to this register will abort current conversion sequence and start a new conversion sequence. If external trigger is enabled (ETRIGE=1) an initial write to ATDCTL5 is required to allow starting of a conversion sequence which will then occur on each trigger event. Start of conversion means the beginning of the sampling phase.

Module Base + 0x0005

	7	6	5	4	3	2	1	0
R	0	SC	SCAN	MULT	CD	CC	CB	CA
W								
Reset	0	0	0	0	0	0	0	0

Figure 24-8. ATD Control Register 5 (ATDCTL5)

Read: Anytime

Write: Anytime

Table 24-14. ATDCTL5 Field Descriptions

Field	Description
6 SC	**Special Channel Conversion Bit** — If this bit is set, then special channel conversion can be selected using CD, CC, CB and CA of ATDCTL5. Table 24-15 lists the coding. 0 Special channel conversions disabled 1 Special channel conversions enabled
5 SCAN	**Continuous Conversion Sequence Mode** — This bit selects whether conversion sequences are performed continuously or only once. If external trigger is enabled (ETRIGE=1) setting this bit has no effect, that means external trigger always starts a single conversion sequence. 0 Single conversion sequence 1 Continuous conversion sequences (scan mode)

Table 24-14. ATDCTL5 Field Descriptions (continued)

Field	Description
4 MULT	**Multi-Channel Sample Mode** — When MULT is 0, the ATD sequence controller samples only from the specified analog input channel for an entire conversion sequence. The analog channel is selected by channel selection code (control bits CD/CC/CB/CA located in ATDCTL5). When MULT is 1, the ATD sequence controller samples across channels. The number of channels sampled is determined by the sequence length value (S8C, S4C, S2C, S1C). The first analog channel examined is determined by channel selection code (CD, CC, CB, CA control bits); subsequent channels sampled in the sequence are determined by incrementing the channel selection code or wrapping around to AN0 (channel 0). 0 Sample only one channel 1 Sample across several channels
3–0 CD, CC, CB, CA	**Analog Input Channel Select Code** — These bits select the analog input channel(s) whose signals are sampled and converted to digital codes. Table 24-15 lists the coding used to select the various analog input channels. In the case of single channel conversions (MULT=0), this selection code specifies the channel to be examined. In the case of multiple channel conversions (MULT=1), this selection code specifies the first channel to be examined in the conversion sequence. Subsequent channels are determined by incrementing the channel selection code or wrapping around to AN0 (after converting the channel defined by the Wrap Around Channel Select Bits WRAP3-0 in ATDCTL0). In case of starting with a channel number higher than the one defined by WRAP3-0 the first wrap around will be AN15 to AN0.

Table 24-15. Analog Input Channel Select Coding

SC	CD	CC	CB	CA	Analog Input Channel
0	0	0	0	0	AN0
	0	0	0	1	AN1
	0	0	1	0	AN2
	0	0	1	1	AN3
	0	1	0	0	AN4
	0	1	0	1	AN5
	0	1	1	0	AN6
	0	1	1	1	AN7
	1	0	0	0	AN8
	1	0	0	1	AN9
	1	0	1	0	AN10
	1	0	1	1	AN11
	1	1	0	0	AN12
	1	1	0	1	AN13
	1	1	1	0	AN14
	1	1	1	1	AN15

Table 24-15. Analog Input Channel Select Coding

SC	CD	CC	CB	CA	Analog Input Channel
1	0	0	0	0	Reserved
	0	0	0	1	Reserved
	0	0	1	X	Reserved
	0	1	0	0	V_{RH}
	0	1	0	1	V_{RL}
	0	1	1	0	$(V_{RH}+V_{RL}) / 2$
	0	1	1	1	Reserved
	1	X	X	X	Reserved

24.3.2.7 ATD Status Register 0 (ATDSTAT0)

This register contains the Sequence Complete Flag, overrun flags for external trigger and FIFO mode, and the conversion counter.

Module Base + 0x0006

	7	6	5	4	3	2	1	0
R	SCF	0	ETORF	FIFOR	CC3	CC2	CC1	CC0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 24-9. ATD Status Register 0 (ATDSTAT0)

Read: Anytime

Write: Anytime (No effect on (CC3, CC2, CC1, CC0))

Table 24-16. ATDSTAT0 Field Descriptions

Field	Description
7 SCF	**Sequence Complete Flag** — This flag is set upon completion of a conversion sequence. If conversion sequences are continuously performed (SCAN=1), the flag is set after each one is completed. This flag is cleared when one of the following occurs: A) Write “1” to SCF B) Write to ATDCTL5 (a new conversion sequence is started) C) If AFFC=1 and read of a result register 0 Conversion sequence not completed 1 Conversion sequence has completed
5 ETORF	**External Trigger Overrun Flag** — While in edge trigger mode (ETRIGLE=0), if additional active edges are detected while a conversion sequence is in process the overrun flag is set. This flag is cleared when one of the following occurs: A) Write “1” to ETORF B) Write to ATDCTL0,1,2,3,4, ATDCMPE or ATDCMPHT (a conversion sequence is aborted) C) Write to ATDCTL5 (a new conversion sequence is started) 0 No External trigger over run error has occurred 1 External trigger over run error has occurred
4 FIFOR	**Result Register Over Run Flag** — This bit indicates that a result register has been written to before its associated conversion complete flag (CCF) has been cleared. This flag is most useful when using the FIFO mode because the flag potentially indicates that result registers are out of sync with the input channels. However, it is also practical for non-FIFO modes, and indicates that a result register has been over written before it has been read (i.e. the old data has been lost). This flag is cleared when one of the following occurs: A) Write “1” to FIFOR B) Write to ATDCTL0,1,2,3,4, ATDCMPE or ATDCMPHT (a conversion sequence is aborted) C) Write to ATDCTL5 (a new conversion sequence is started) 0 No over run has occurred 1 Overrun condition exists (result register has been written while associated CCFx flag was still set)

Table 24-16. ATDSTAT0 Field Descriptions (continued)

Field	Description
3–0 CC[3:0]	**Conversion Counter** — These 4 read-only bits are the binary value of the conversion counter. The conversion counter points to the result register that will receive the result of the current conversion. E.g. CC3=0, CC2=1, CC1=1, CC0=0 indicates that the result of the current conversion will be in ATD Result Register 6. If in non-FIFO mode (FIFO=0) the conversion counter is initialized to zero at the begin and end of the conversion sequence. If in FIFO mode (FIFO=1) the register counter is not initialized. The conversion counters wraps around when its maximum value is reached. Aborting a conversion or starting a new conversion clears the conversion counter even if FIFO=1.

24.3.2.8 ATD Compare Enable Register (ATDCMPE)

Writes to this register will abort current conversion sequence.

Read: Anytime

Write: Anytime

Module Base + 0x0008

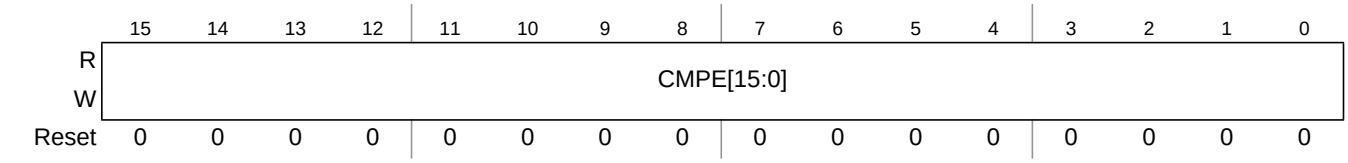


Figure 24-10. ATD Compare Enable Register (ATDCMPE)

Table 24-17. ATDCMPE Field Descriptions

Field	Description
15–0 CMPE[15:0]	**Compare Enable for Conversion Number *n* (*n*= 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0) of a Sequence** — These bits enable automatic compare of conversion results individually for conversions of a sequence. The sense of each comparison is determined by the CMPHT[*n*] bit in the ATDCMPHT register. For each conversion number with CMPE[*n*]=1 do the following: 1) Write compare value to ATDDR*n* result register 2) Write compare operator with CMPHT[*n*] in ATDCPMHT register CCF[*n*] in ATDSTAT2 register will flag individual success of any comparison. 0 No automatic compare 1 Automatic compare of results for conversion *n* of a sequence is enabled.

24.3.2.9 ATD Status Register 2 (ATDSTAT2)

This read-only register contains the Conversion Complete Flags CCF[15:0].

Module Base + 0x000A

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CCF[15:0]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 24-11. ATD Status Register 2 (ATDSTAT2)

Read: Anytime

Write: Anytime, no effect

Table 24-18. ATDSTAT2 Field Descriptions

Field	Description
15–0 CCF[15:0]	**Conversion Complete Flag *n* (*n*= 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0)** — A conversion complete flag is set at the end of each conversion in a sequence. The flags are associated with the conversion position in a sequence (and also the result register number). Therefore in non-fifo mode, CCF[8] is set when the ninth conversion in a sequence is complete and the result is available in result register ATDDR8; CCF[9] is set when the tenth conversion in a sequence is complete and the result is available in ATDDR9, and so forth. If automatic compare of conversion results is enabled (CMPE[*n*]=1 in ATDCMPE), the conversion complete flag is only set if comparison with ATDDR*n* is true and if ACMPIE=1 a compare interrupt will be requested. In this case, as the ATDDR*n* result register is used to hold the compare value, the result will not be stored there at the end of the conversion but is lost. A flag CCF[*n*] is cleared when one of the following occurs: A) Write to ATDCTL5 (a new conversion sequence is started) B) If AFFC=0, write “1” to CCF[*n*] C) If AFFC=1 and CMPE[*n*]=0, read of result register ATDDR*n* D) If AFFC=1 and CMPE[*n*]=1, write to result register ATDDR*n* In case of a concurrent set and clear on CCF[*n*]: The clearing by method A) will overwrite the set. The clearing by methods B) or C) or D) will be overwritten by the set. 0 Conversion number *n* not completed or successfully compared 1 If (CMPE[*n*]=0): Conversion number *n* has completed. Result is ready in ATDDR*n*. If (CMPE[*n*]=1): Compare for conversion result number *n* with compare value in ATDDR*n*, using compare operator CMPGT[*n*] is true. (No result available in ATDDR*n*)

24.3.2.10 ATD Input Enable Register (ATDDIEN)

Module Base + 0x000C

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	IEN[15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 24-12. ATD Input Enable Register (ATDDIEN)

Read: Anytime

Write: Anytime

Table 24-19. ATDDIEN Field Descriptions

Field	Description
15–0 IEN[15:0]	**ATD Digital Input Enable on channel *x* (*x*= 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0)** — This bit controls the digital input buffer from the analog input pin (AN*x*) to the digital data register. 0 Disable digital input buffer to AN*x* pin 1 Enable digital input buffer on AN*x* pin. **Note:** Setting this bit will enable the corresponding digital input buffer continuously. If this bit is set while simultaneously using it as an analog port, there is potentially increased power consumption because the digital input buffer maybe in the linear region.

24.3.2.11 ATD Compare Higher Than Register (ATDCMPHT)

Writes to this register will abort current conversion sequence.

Read: Anytime

Write: Anytime

Module Base + 0x000E

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R W	CMPHT[15:0]															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 24-13. ATD Compare Higher Than Register (ATDCMPHT)

Table 24-20. ATDCMPHT Field Descriptions

Field	Description
15–0 CMPHT[15:0]	**Compare Operation Higher Than Enable for conversion number *n* (*n*= 15, 14, 13, 12, 11, 10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0) of a Sequence** — This bit selects the operator for comparison of conversion results. 0 If result of conversion *n* is **lower or same than** compare value in ATDDR*n*, this is flagged in ATDSTAT2 1 If result of conversion *n* is **higher than** compare value in ATDDR*n*, this is flagged in ATDSTAT2

24.3.2.12 ATD Conversion Result Registers (ATDDR*n*)

The A/D conversion results are stored in 16 result registers. Results are always in unsigned data representation. Left and right justification is selected using the DJM control bit in ATDCTL3.

If automatic compare of conversions results is enabled (CMPE[*n*]=1 in ATDCMPE), these registers must be written with the compare values in left or right justified format depending on the actual value of the DJM bit. In this case, as the ATDDR*n* register is used to hold the compare value, the result will not be stored there at the end of the conversion but is lost.

Read: Anytime

Write: Anytime

NOTE

For conversions not using automatic compare, results are stored in the result registers after each conversion. In this case avoid writing to ATDDRn except for initial values, because an A/D result might be overwritten.

24.3.2.12.1 Left Justified Result Data (DJM=0)

Module Base +
0x0010 = ATDDR0, 0x0012 = ATDDR1, 0x0014 = ATDDR2, 0x0016 = ATDDR3
0x0018 = ATDDR4, 0x001A = ATDDR5, 0x001C = ATDDR6, 0x001E = ATDDR7
0x0020 = ATDDR8, 0x0022 = ATDDR9, 0x0024 = ATDDR10, 0x0026 = ATDDR11
0x0028 = ATDDR12, 0x002A = ATDDR13, 0x002C = ATDDR14, 0x002E = ATDDR15

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	0	0	0	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 24-14. Left justified ATD conversion result register (ATDDR*n*)

24.3.2.12.2 Right Justified Result Data (DJM=1)

Module Base +
0x0010 = ATDDR0, 0x0012 = ATDDR1, 0x0014 = ATDDR2, 0x0016 = ATDDR3
0x0018 = ATDDR4, 0x001A = ATDDR5, 0x001C = ATDDR6, 0x001E = ATDDR7
0x0020 = ATDDR8, 0x0022 = ATDDR9, 0x0024 = ATDDR10, 0x0026 = ATDDR11
0x0028 = ATDDR12, 0x002A = ATDDR13, 0x002C = ATDDR14, 0x002E = ATDDR15

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bi1 1	Bit 0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 24-15. Right justified ATD conversion result register (ATDDR*n*)

Table 24-15 shows how depending on the A/D resolution the conversion result is transferred to the ATD result registers. Compare is always done using all 12 bits of both the conversion result and the compare value in ATDDRn.

Table 24-21. Conversion result mapping to ATDDRn

A/D resolution	DJM	conversion result mapping to ATDDR*n*
8-bit data	0	Bit[11:4] = result, Bit[3:0]=0000
8-bit data	1	Bit[7:0] = result, Bit[11:8]=0000
10-bit data	0	Bit[11:2] = result, Bit[1:0]=00
10-bit data	1	Bit[9:0] = result, Bit[11:10]=00
12-bit data	X	Bit[11:0] = result

24.4 Functional Description

The ADC12B16C is structured into an analog sub-block and a digital sub-block.

24.4.1 Analog Sub-Block

The analog sub-block contains all analog electronics required to perform a single conversion. Separate power supplies V_{DDA} and V_{SSA} allow to isolate noise of other MCU circuitry from the analog sub-block.

24.4.1.1 Sample and Hold Machine

The Sample and Hold (S/H) Machine accepts analog signals from the external world and stores them as capacitor charge on a storage node.

During the sample process the analog input connects directly to the storage node.

The input analog signals are unipolar and must fall within the potential range of V_{SSA} to V_{DDA}.

During the hold process the analog input is disconnected from the storage node.

24.4.1.2 Analog Input Multiplexer

The analog input multiplexer connects one of the 16 external analog input channels to the sample and hold machine.

24.4.1.3 Analog-to-Digital (A/D) Machine

The A/D Machine performs analog to digital conversions. The resolution is program selectable at either 8 or 10 or 12 bits. The A/D machine uses a successive approximation architecture. It functions by comparing the stored analog sample potential with a series of digitally generated analog potentials. By following a binary search algorithm, the A/D machine locates the approximating potential that is nearest to the sampled potential.

When not converting the A/D machine is automatically powered down.

Only analog input signals within the potential range of V_{RL} to V_{RH} (A/D reference potentials) will result in a non-railed digital output code.

24.4.2 Digital Sub-Block

This subsection explains some of the digital features in more detail. See Section 24.3.2, “Register Descriptions” for all details.

24.4.2.1 External Trigger Input

The external trigger feature allows the user to synchronize ATD conversions to the external environment events rather than relying on software to signal the ATD module when ATD conversions are to take place. The external trigger signal (out of reset ATD channel 15, configurable in ATDCTL1) is programmable to

be edge or level sensitive with polarity control. Table 24-22 gives a brief description of the different combinations of control bits and their effect on the external trigger function.

Table 24-22. External Trigger Control Bits

ETRIGLE	ETRIGP	ETRIGE	SCAN	Description
X	X	0	0	Ignores external trigger. Performs one conversion sequence and stops.
X	X	0	1	Ignores external trigger. Performs continuous conversion sequences.
0	0	1	X	Falling edge triggered. Performs one conversion sequence per trigger.
0	1	1	X	Rising edge triggered. Performs one conversion sequence per trigger.
1	0	1	X	Trigger active low. Performs continuous conversions while trigger is active.
1	1	1	X	Trigger active high. Performs continuous conversions while trigger is active.

During a conversion, if additional active edges are detected the overrun error flag ETORF is set.

In either level or edge triggered modes, the first conversion begins when the trigger is received.

Once ETRIGE is enabled, conversions cannot be started by a write to ATDCTL5, but rather must be triggered externally.

If the level mode is active and the external trigger both de-asserts and re-asserts itself during a conversion sequence, this does not constitute an overrun. Therefore, the flag is not set. If the trigger is left asserted in level mode while a sequence is completing, another sequence will be triggered immediately.

24.4.2.2 General-Purpose Digital Port Operation

The input channel pins can be multiplexed between analog and digital data. As analog inputs, they are multiplexed and sampled as analog channels to the A/D converter. The analog/digital multiplex operation is performed in the input pads. The input pad is always connected to the analog input channels of the ADC12B16C. The input pad signal is buffered to the digital port registers. This buffer can be turned on or off with the ATDDIEN register. This is important so that the buffer does not draw excess current when analog potentials are presented at its input.

24.5 Resets

At reset the ADC12B16C is in a power down state. The reset state of each individual bit is listed within the Register Description section (see Section 24.3.2, “Register Descriptions”) which details the registers and their bit-field.

24.6 Interrupts

The interrupts requested by the ADC12B16C are listed in Table 24-23. Refer to MCU specification for related vector address and priority.

Table 24-23. ATD Interrupt Vectors

Interrupt Source	CCR Mask	Local Enable
Sequence Complete Interrupt	I bit	ASCIE in ATDCTL2
Compare Interrupt	I bit	ACMPIE in ATDCTL2

See Section 24.3.2, “Register Descriptions” for further details.

Chapter 25
Serial Peripheral Interface (S12SPIV5)

Revision History

Revision Number	Date	Author	Summary of Changes
05.00	24 MAR 2005		Added 16-bit transfer width feature.

25.1 Introduction

The SPI module allows a duplex, synchronous, serial communication between the MCU and peripheral devices. Software can poll the SPI status flags or the SPI operation can be interrupt driven.

25.1.1 Glossary of Terms

SPI	Serial Peripheral Interface
$\overline{SS}$	Slave Select
SCK	Serial Clock
MOSI	Master Output, Slave Input
MISO	Master Input, Slave Output
MOMI	Master Output, Master Input
SISO	Slave Input, Slave Output

25.1.2 Features

The SPI includes these distinctive features:

- Master mode and slave mode
- Selectable 8 or 16-bit transfer width
- Bidirectional mode
- Slave select output
- Mode fault error flag with CPU interrupt capability

- Double-buffered data register
- Serial clock with programmable polarity and phase
- Control of SPI operation during wait mode

25.1.3 Modes of Operation

The SPI functions in three modes: run, wait, and stop.

- Run mode

  This is the basic mode of operation.
- Wait mode

  SPI operation in wait mode is a configurable low power mode, controlled by the SPISWAI bit located in the SPICR2 register. In wait mode, if the SPISWAI bit is clear, the SPI operates like in run mode. If the SPISWAI bit is set, the SPI goes into a power conservative state, with the SPI clock generation turned off. If the SPI is configured as a master, any transmission in progress stops, but is resumed after CPU goes into run mode. If the SPI is configured as a slave, reception and transmission of data continues, so that the slave stays synchronized to the master.
- Stop mode

  The SPI is inactive in stop mode for reduced power consumption. If the SPI is configured as a master, any transmission in progress stops, but is resumed after CPU goes into run mode. If the SPI is configured as a slave, reception and transmission of data continues, so that the slave stays synchronized to the master.

For a detailed description of operating modes, please refer to Section 25.4.7, “Low Power Mode Options”.

25.1.4 Block Diagram

Figure 25-1 gives an overview on the SPI architecture. The main parts of the SPI are status, control and data registers, shifter logic, baud rate generator, master/slave control logic, and port control logic.

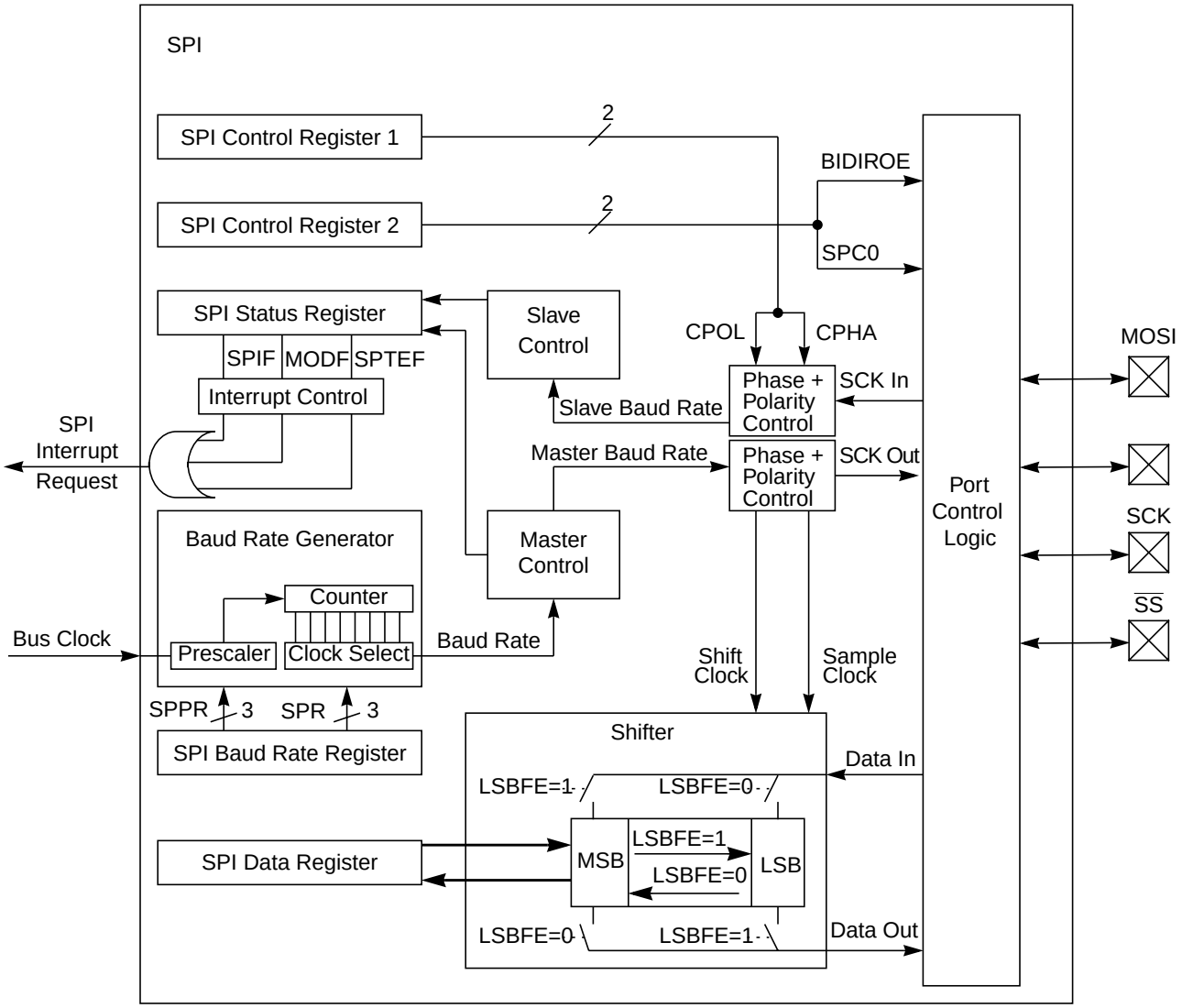


Figure 25-1. SPI Block Diagram

25.2 External Signal Description

This section lists the name and description of all ports including inputs and outputs that do, or may, connect off chip. The SPI module has a total of four external pins.

25.2.1 MOSI — Master Out/Slave In Pin

This pin is used to transmit data out of the SPI module when it is configured as a master and receive data when it is configured as slave.

25.2.2 MISO — Master In/Slave Out Pin

This pin is used to transmit data out of the SPI module when it is configured as a slave and receive data when it is configured as master.

25.2.3 $\overline{SS}$ — Slave Select Pin

This pin is used to output the select signal from the SPI module to another peripheral with which a data transfer is to take place when it is configured as a master and it is used as an input to receive the slave select signal when the SPI is configured as slave.

25.2.4 SCK — Serial Clock Pin

In master mode, this is the synchronous output clock. In slave mode, this is the synchronous input clock.

25.3 Memory Map and Register Definition

This section provides a detailed description of address space and registers used by the SPI.

25.3.1 Module Memory Map

The memory map for the SPI is given in Figure 25-2. The address listed for each register is the sum of a base address and an address offset. The base address is defined at the SoC level and the address offset is defined at the module level. Reads from the reserved bits return zeros and writes to the reserved bits have no effect.

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000 SPICR1	R	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
	W								
0x0001 SPICR2	R	0	XFRW	0	MODFEN	BIDIROE	0	SPISWAI	SPC0
	W								
0x0002 SPIBR	R	0	SPPR2	SPPR1	SPPR0	0	SPR2	SPR1	SPR0
	W								
0x0003 SPISR	R	SPIF	0	SPTEF	MODF	0	0	0	0
	W								
0x0004 SPIDRH	R	R15	R14	R13	R12	R11	R10	R9	R8
	W	T15	T14	T13	T12	T11	T10	T9	T8
0x0005 SPIDRL	R	R7	R6	R5	R4	R3	R2	R1	R0
	W	T7	T6	T5	T4	T3	T2	T1	T0
0x0006 Reserved	R								
	W								
0x0007 Reserved	R								
	W								

= Unimplemented or Reserved

Figure 25-2. SPI Register Summary

25.3.2 Register Descriptions

This section consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number. Details of register bit and field function follow the register diagrams, in bit order.

25.3.2.1 SPI Control Register 1 (SPICR1)

Module Base +0x0000

	7	6	5	4	3	2	1	0
R W	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
Reset	0	0	0	0	0	1	0	0

Figure 25-3. SPI Control Register 1 (SPICR1)

Read: Anytime

Write: Anytime

Table 25-1. SPICR1 Field Descriptions

Field	Description
7 SPIE	**SPI Interrupt Enable Bit** — This bit enables SPI interrupt requests, if SPIF or MODF status flag is set. 0 SPI interrupts disabled. 1 SPI interrupts enabled.
6 SPE	**SPI System Enable Bit** — This bit enables the SPI system and dedicates the SPI port pins to SPI system functions. If SPE is cleared, SPI is disabled and forced into idle state, status bits in SPISR register are reset. 0 SPI disabled (lower power consumption). 1 SPI enabled, port pins are dedicated to SPI functions.
5 SPTIE	**SPI Transmit Interrupt Enable** — This bit enables SPI interrupt requests, if SPTEF flag is set. 0 SPTEF interrupt disabled. 1 SPTEF interrupt enabled.
4 MSTR	**SPI Master/Slave Mode Select Bit** — This bit selects whether the SPI operates in master or slave mode. Switching the SPI from master to slave or vice versa forces the SPI system into idle state. 0 SPI is in slave mode. 1 SPI is in master mode.
3 CPOL	**SPI Clock Polarity Bit** — This bit selects an inverted or non-inverted SPI clock. To transmit data between SPI modules, the SPI modules must have identical CPOL values. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state. 0 Active-high clocks selected. In idle state SCK is low. 1 Active-low clocks selected. In idle state SCK is high.
2 CPHA	**SPI Clock Phase Bit** — This bit is used to select the SPI clock format. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state. 0 Sampling of data occurs at odd edges (1,3,5,...) of the SCK clock. 1 Sampling of data occurs at even edges (2,4,6,...) of the SCK clock.
1 SSOE	**Slave Select Output Enable** — The $\overline{SS}$ output feature is enabled only in master mode, if MODFEN is set, by asserting the SSOE as shown in Table 25-2. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state.
0 LSBFE	**LSB-First Enable** — This bit does not affect the position of the MSB and LSB in the data register. Reads and writes of the data register always have the MSB in the highest bit position. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state. 0 Data is transferred most significant bit first. 1 Data is transferred least significant bit first.

Table 25-2. $\overline{SS}$ Input / Output Selection

MODFEN	SSOE	Master Mode	Slave Mode
0	0	$\overline{SS}$ not used by SPI	$\overline{SS}$ input
0	1	$\overline{SS}$ not used by SPI	$\overline{SS}$ input
1	0	$\overline{SS}$ input with MODF feature	$\overline{SS}$ input
1	1	$\overline{SS}$ is slave select output	$\overline{SS}$ input

25.3.2.2 SPI Control Register 2 (SPICR2)

Module Base +0x0001

	7	6	5	4	3	2	1	0
R	0	XFRW	0	MODFEN	BIDIROE	0	SPISWAI	SPC0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 25-4. SPI Control Register 2 (SPICR2)

Read: Anytime

Write: Anytime; writes to the reserved bits have no effect

Table 25-3. SPICR2 Field Descriptions

Field	Description
6 XFRW	**Transfer Width —** This bit is used for selecting the data transfer width. If 8-bit transfer width is selected, SPIDRL becomes the dedicated data register and SPIDRH is unused. If 16-bit transfer width is selected, SPIDRH and SPIDRL form a 16-bit data register. Please refer to Section 25.3.2.4, “SPI Status Register (SPISR) for information about transmit/receive data handling and the interrupt flag clearing mechanism. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state. 0 8-bit Transfer Width (n = 8)(1) 1 16-bit Transfer Width (n = 16)1
4 MODFEN	**Mode Fault Enable Bit** — This bit allows the MODF failure to be detected. If the SPI is in master mode and MODFEN is cleared, then the $\overline{SS}$ port pin is not used by the SPI. In slave mode, the $\overline{SS}$ is available only as an input regardless of the value of MODFEN. For an overview on the impact of the MODFEN bit on the $\overline{SS}$ port pin configuration, refer to Table 25-2. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state. 0 $\overline{SS}$ port pin is not used by the SPI. 1 $\overline{SS}$ port pin with MODF feature.
3 BIDIROE	**Output Enable in the Bidirectional Mode of Operation** — This bit controls the MOSI and MISO output buffer of the SPI, when in bidirectional mode of operation (SPC0 is set). In master mode, this bit controls the output buffer of the MOSI port, in slave mode it controls the output buffer of the MISO port. In master mode, with SPC0 set, a change of this bit will abort a transmission in progress and force the SPI into idle state. 0 Output buffer disabled. 1 Output buffer enabled.

Table 25-3. SPICR2 Field Descriptions (continued)

Field	Description
6 XFRW	**Transfer Width —** This bit is used for selecting the data transfer width. If 8-bit transfer width is selected, SPIDRL becomes the dedicated data register and SPIDRH is unused. If 16-bit transfer width is selected, SPIDRH and SPIDRL form a 16-bit data register. Please refer to Section 25.3.2.4, “SPI Status Register (SPISR) for information about transmit/receive data handling and the interrupt flag clearing mechanism. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state. 0 8-bit Transfer Width (n = 8)(1) 1 16-bit Transfer Width (n = 16)1
1 SPISWAI	**SPI Stop in Wait Mode Bit** — This bit is used for power conservation while in wait mode. 0 SPI clock operates normally in wait mode. 1 Stop SPI clock generation when in wait mode.
0 SPC0	**Serial Pin Control Bit 0** — This bit enables bidirectional pin configurations as shown in Table 25-4. In master mode, a change of this bit will abort a transmission in progress and force the SPI system into idle state.

1. n is used later in this document as a placeholder for the selected transfer width.

Table 25-4. Bidirectional Pin Configurations

Pin Mode	SPC0	BIDIROE	MISO	MOSI
Master Mode of Operation				
Normal	0	X	Master In	Master Out
Bidirectional	1	0	MISO not used by SPI	Master In
		1		Master I/O
Slave Mode of Operation				
Normal	0	X	Slave Out	Slave In
Bidirectional	1	0	Slave In	MOSI not used by SPI
		1	Slave I/O	

25.3.2.3 SPI Baud Rate Register (SPIBR)

Module Base +0x0002

	7	6	5	4	3	2	1	0
R	0	SPPR2	SPPR1	SPPR0	0	SPR2	SPR1	SPR0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 25-5. SPI Baud Rate Register (SPIBR)

Read: Anytime

Write: Anytime; writes to the reserved bits have no effect

Table 25-5. SPIBR Field Descriptions

Field	Description
6–4 SPPR[2:0]	**SPI Baud Rate Preselection Bits** — These bits specify the SPI baud rates as shown in Table 25-6. In master mode, a change of these bits will abort a transmission in progress and force the SPI system into idle state.
2–0 SPR[2:0]	**SPI Baud Rate Selection Bits** — These bits specify the SPI baud rates as shown in Table 25-6. In master mode, a change of these bits will abort a transmission in progress and force the SPI system into idle state.

The baud rate divisor equation is as follows:

$$\text{BaudRateDivisor} = (\text{SPPR} + 1) \bullet 2^{(\text{SPR} + 1)} \qquad \textit{Eqn. 25-1}$$

The baud rate can be calculated with the following equation:

$$\text{Baud Rate} = \text{BusClock} / \text{BaudRateDivisor} \qquad \textit{Eqn. 25-2}$$

NOTE

For maximum allowed baud rates, please refer to the SPI Electrical Specification in the Electricals chapter of this data sheet.

Table 25-6. Example SPI Baud Rate Selection (25 MHz Bus Clock)

SPPR2	SPPR1	SPPR0	SPR2	SPR1	SPR0	Baud Rate Divisor	Baud Rate
0	0	0	0	0	0	2	12.5 Mbit/s
0	0	0	0	0	1	4	6.25 Mbit/s
0	0	0	0	1	0	8	3.125 Mbit/s
0	0	0	0	1	1	16	1.5625 Mbit/s
0	0	0	1	0	0	32	781.25 kbit/s
0	0	0	1	0	1	64	390.63 kbit/s
0	0	0	1	1	0	128	195.31 kbit/s
0	0	0	1	1	1	256	97.66 kbit/s
0	0	1	0	0	0	4	6.25 Mbit/s
0	0	1	0	0	1	8	3.125 Mbit/s
0	0	1	0	1	0	16	1.5625 Mbit/s
0	0	1	0	1	1	32	781.25 kbit/s
0	0	1	1	0	0	64	390.63 kbit/s
0	0	1	1	0	1	128	195.31 kbit/s
0	0	1	1	1	0	256	97.66 kbit/s
0	0	1	1	1	1	512	48.83 kbit/s
0	1	0	0	0	0	6	4.16667 Mbit/s
0	1	0	0	0	1	12	2.08333 Mbit/s
0	1	0	0	1	0	24	1.04167 Mbit/s
0	1	0	0	1	1	48	520.83 kbit/s
0	1	0	1	0	0	96	260.42 kbit/s
0	1	0	1	0	1	192	130.21 kbit/s
0	1	0	1	1	0	384	65.10 kbit/s
0	1	0	1	1	1	768	32.55 kbit/s
0	1	1	0	0	0	8	3.125 Mbit/s
0	1	1	0	0	1	16	1.5625 Mbit/s
0	1	1	0	1	0	32	781.25 kbit/s
0	1	1	0	1	1	64	390.63 kbit/s
0	1	1	1	0	0	128	195.31 kbit/s
0	1	1	1	0	1	256	97.66 kbit/s
0	1	1	1	1	0	512	48.83 kbit/s
0	1	1	1	1	1	1024	24.41 kbit/s
1	0	0	0	0	0	10	2.5 Mbit/s
1	0	0	0	0	1	20	1.25 Mbit/s
1	0	0	0	1	0	40	625 kbit/s
1	0	0	0	1	1	80	312.5 kbit/s
1	0	0	1	0	0	160	156.25 kbit/s
1	0	0	1	0	1	320	78.13 kbit/s
1	0	0	1	1	0	640	39.06 kbit/s

Table 25-6. Example SPI Baud Rate Selection (25 MHz Bus Clock) (continued)

SPPR2	SPPR1	SPPR0	SPR2	SPR1	SPR0	Baud Rate Divisor	Baud Rate
1	0	0	1	1	1	1280	19.53 kbit/s
1	0	1	0	0	0	12	2.08333 Mbit/s
1	0	1	0	0	1	24	1.04167 Mbit/s
1	0	1	0	1	0	48	520.83 kbit/s
1	0	1	0	1	1	96	260.42 kbit/s
1	0	1	1	0	0	192	130.21 kbit/s
1	0	1	1	0	1	384	65.10 kbit/s
1	0	1	1	1	0	768	32.55 kbit/s
1	0	1	1	1	1	1536	16.28 kbit/s
1	1	0	0	0	0	14	1.78571 Mbit/s
1	1	0	0	0	1	28	892.86 kbit/s
1	1	0	0	1	0	56	446.43 kbit/s
1	1	0	0	1	1	112	223.21 kbit/s
1	1	0	1	0	0	224	111.61 kbit/s
1	1	0	1	0	1	448	55.80 kbit/s
1	1	0	1	1	0	896	27.90 kbit/s
1	1	0	1	1	1	1792	13.95 kbit/s
1	1	1	0	0	0	16	1.5625 Mbit/s
1	1	1	0	0	1	32	781.25 kbit/s
1	1	1	0	1	0	64	390.63 kbit/s
1	1	1	0	1	1	128	195.31 kbit/s
1	1	1	1	0	0	256	97.66 kbit/s
1	1	1	1	0	1	512	48.83 kbit/s
1	1	1	1	1	0	1024	24.41 kbit/s
1	1	1	1	1	1	2048	12.21 kbit/s

25.3.2.4 SPI Status Register (SPISR)

Module Base +0x0003

	7	6	5	4	3	2	1	0
R	SPIF	0	SPTEF	MODF	0	0	0	0
W								
Reset	0	0	1	0	0	0	0	0

= Unimplemented or Reserved

Figure 25-6. SPI Status Register (SPISR)

Read: Anytime

Write: Has no effect

Table 25-7. SPISR Field Descriptions

Field	Description
7 SPIF	**SPIF Interrupt Flag** — This bit is set after received data has been transferred into the SPI data register. For information about clearing SPIF Flag, please refer to Table 25-8. 0 Transfer not yet complete. 1 New data copied to SPIDR.
5 SPTEF	**SPI Transmit Empty Interrupt Flag** — If set, this bit indicates that the transmit data register is empty. For information about clearing this bit and placing data into the transmit data register, please refer to Table 25-9. 0 SPI data register not empty. 1 SPI data register empty.
4 MODF	**Mode Fault Flag** — This bit is set if the $\overline{SS}$ input becomes low while the SPI is configured as a master and mode fault detection is enabled, MODFEN bit of SPICR2 register is set. Refer to MODFEN bit description in Section 25.3.2.2, “SPI Control Register 2 (SPICR2)”. The flag is cleared automatically by a read of the SPI status register (with MODF set) followed by a write to the SPI control register 1. 0 Mode fault has not occurred. 1 Mode fault has occurred.

Table 25-8. SPIF Interrupt Flag Clearing Sequence

XFRW Bit	SPIF Interrupt Flag Clearing Sequence			
0	Read SPISR with SPIF == 1	**then**	Read SPIDRL	
1	Read SPISR with SPIF == 1	**then**	Byte Read SPIDRL (1)	
			or	
			Byte Read SPIDRH (2)	Byte Read SPIDRL
			or	
			Word Read (SPIDRH:SPIDRL)	

1. Data in SPIDRH is lost in this case.
2. SPIDRH can be read repeatedly without any effect on SPIF. SPIF Flag is cleared only by the read of SPIDRL after reading SPISR with SPIF == 1.

Table 25-9. SPTEF Interrupt Flag Clearing Sequence

XFRW Bit	SPTEF Interrupt Flag Clearing Sequence			
0	Read SPISR with SPTEF == 1	**then**	Write to SPIDRL (1)	
1	Read SPISR with SPTEF == 1	**then**	Byte Write to SPIDRL 1(2)	
			or	
			Byte Write to SPIDRH 1(3)	Byte Write to SPIDRL 1
			or	
			Word Write to (SPIDRH:SPIDRL) 1	

1. Any write to SPIDRH or SPIDRL with SPTEF == 0 is effectively ignored.
2. Data in SPIDRH is undefined in this case.
3. SPIDRH can be written repeatedly without any effect on SPTEF. SPTEF Flag is cleared only by writing to SPIDRL after reading SPISR with SPTEF == 1.

25.3.2.5 SPI Data Register (SPIDR = SPIDRH:SPIDRL)

Module Base +0x0004

	7	6	5	4	3	2	1	0
R	R15	R14	R13	R12	R11	R10	R9	R8
W	T15	T14	T13	T12	T11	T10	T9	T8
Reset	0	0	0	0	0	0	0	0

Figure 25-7. SPI Data Register High (SPIDRH)

Module Base +0x0005

	7	6	5	4	3	2	1	0
R	R7	R6	R5	R4	R3	R2	R1	R0
W	T7	T6	T5	T4	T3	T2	T1	T0
Reset	0	0	0	0	0	0	0	0

Figure 25-8. SPI Data Register Low (SPIDRL)

Read: Anytime; read data only valid when SPIF is set

Write: Anytime

The SPI data register is both the input and output register for SPI data. A write to this register allows data to be queued and transmitted. For an SPI configured as a master, queued data is transmitted immediately after the previous transmission has completed. The SPI transmitter empty flag SPTEF in the SPISR register indicates when the SPI data register is ready to accept new data.

Received data in the SPIDR is valid when SPIF is set.

If SPIF is cleared and data has been received, the received data is transferred from the receive shift register to the SPIDR and SPIF is set.

If SPIF is set and not serviced, and a second data value has been received, the second received data is kept as valid data in the receive shift register until the start of another transmission. The data in the SPIDR does not change.

If SPIF is set and valid data is in the receive shift register, and SPIF is serviced before the start of a third transmission, the data in the receive shift register is transferred into the SPIDR and SPIF remains set (see Figure 25-9).

If SPIF is set and valid data is in the receive shift register, and SPIF is serviced after the start of a third transmission, the data in the receive shift register has become invalid and is not transferred into the SPIDR (see Figure 25-10).

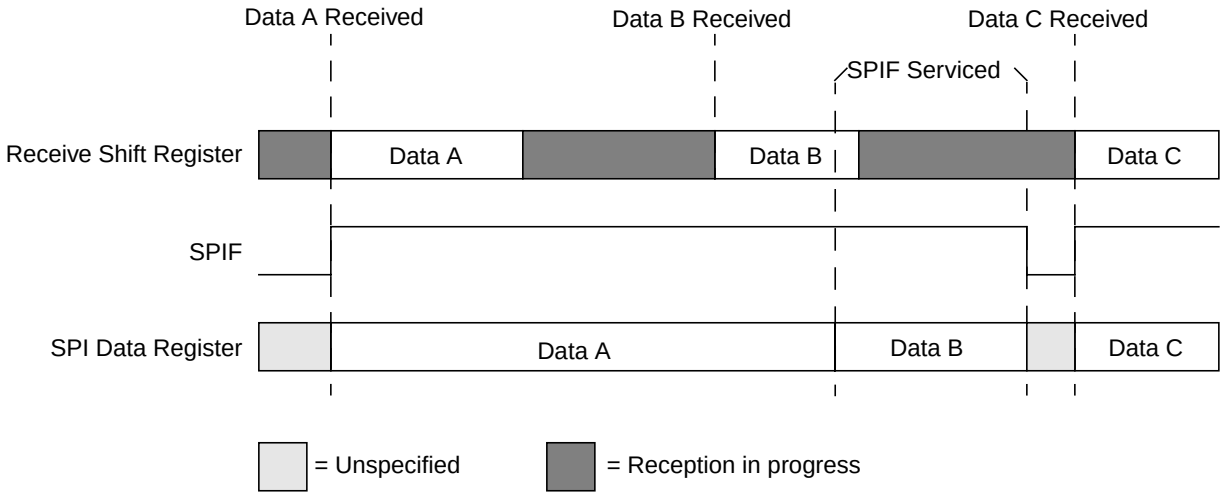


Figure 25-9. Reception with SPIF serviced in Time

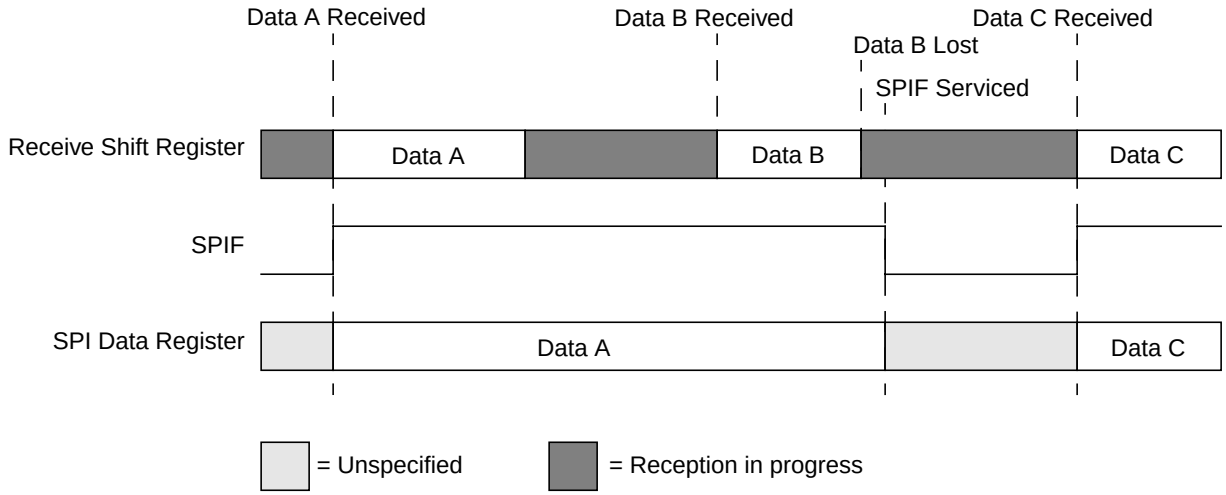


Figure 25-10. Reception with SPIF serviced too late

25.4 Functional Description

The SPI module allows a duplex, synchronous, serial communication between the MCU and peripheral devices. Software can poll the SPI status flags or SPI operation can be interrupt driven.

The SPI system is enabled by setting the SPI enable (SPE) bit in SPI control register 1. While SPE is set, the four associated SPI port pins are dedicated to the SPI function as:

- Slave select ($\overline{SS}$)
- Serial clock (SCK)
- Master out/slave in (MOSI)
- Master in/slave out (MISO)

The main element of the SPI system is the SPI data register. The n-bit[1] data register in the master and the n-bit[1] data register in the slave are linked by the MOSI and MISO pins to form a distributed 2n-bit[1] register. When a data transfer operation is performed, this 2n-bit[1] register is serially shifted n[1] bit positions by the S-clock from the master, so data is exchanged between the master and the slave. Data written to the master SPI data register becomes the output data for the slave, and data read from the master SPI data register after a transfer operation is the input data from the slave.

A read of SPISR with SPTEF = 1 followed by a write to SPIDR puts data into the transmit data register. When a transfer is complete and SPIF is cleared, received data is moved into the receive data register. This data register acts as the SPI receive data register for reads and as the SPI transmit data register for writes. A common SPI data register address is shared for reading data from the read data buffer and for writing data to the transmit data register.

The clock phase control bit (CPHA) and a clock polarity control bit (CPOL) in the SPI control register 1 (SPICR1) select one of four possible clock formats to be used by the SPI system. The CPOL bit simply selects a non-inverted or inverted clock. The CPHA bit is used to accommodate two fundamentally different protocols by sampling data on odd numbered SCK edges or on even numbered SCK edges (see Section 25.4.3, “Transmission Formats”).

The SPI can be configured to operate as a master or as a slave. When the MSTR bit in SPI control register1 is set, master mode is selected, when the MSTR bit is clear, slave mode is selected.

NOTE

A change of CPOL or MSTR bit while there is a received byte pending in the receive shift register will destroy the received byte and must be avoided.

1. n depends on the selected transfer width, please refer to Section 25.3.2.2, “SPI Control Register 2 (SPICR2)

25.4.1 Master Mode

The SPI operates in master mode when the MSTR bit is set. Only a master SPI module can initiate transmissions. A transmission begins by writing to the master SPI data register. If the shift register is empty, data immediately transfers to the shift register. Data begins shifting out on the MOSI pin under the control of the serial clock.

- Serial clock

  The SPR2, SPR1, and SPR0 baud rate selection bits, in conjunction with the SPPR2, SPPR1, and SPPR0 baud rate preselection bits in the SPI baud rate register, control the baud rate generator and determine the speed of the transmission. The SCK pin is the SPI clock output. Through the SCK pin, the baud rate generator of the master controls the shift register of the slave peripheral.
- MOSI, MISO pin

  In master mode, the function of the serial data output pin (MOSI) and the serial data input pin (MISO) is determined by the SPC0 and BIDIROE control bits.
- $\overline{SS}$ pin

  If MODFEN and SSOE are set, the $\overline{SS}$ pin is configured as slave select output. The $\overline{SS}$ output becomes low during each transmission and is high when the SPI is in idle state.

  If MODFEN is set and SSOE is cleared, the $\overline{SS}$ pin is configured as input for detecting mode fault error. If the $\overline{SS}$ input becomes low this indicates a mode fault error where another master tries to drive the MOSI and SCK lines. In this case, the SPI immediately switches to slave mode, by clearing the MSTR bit and also disables the slave output buffer MISO (or SISO in bidirectional mode). So the result is that all outputs are disabled and SCK, MOSI, and MISO are inputs. If a transmission is in progress when the mode fault occurs, the transmission is aborted and the SPI is forced into idle state.

  This mode fault error also sets the mode fault (MODF) flag in the SPI status register (SPISR). If the SPI interrupt enable bit (SPIE) is set when the MODF flag becomes set, then an SPI interrupt sequence is also requested.

  When a write to the SPI data register in the master occurs, there is a half SCK-cycle delay. After the delay, SCK is started within the master. The rest of the transfer operation differs slightly, depending on the clock format specified by the SPI clock phase bit, CPHA, in SPI control register 1 (see Section 25.4.3, “Transmission Formats”).

NOTE

A change of the bits CPOL, CPHA, SSOE, LSBFE, XFRW, MODFEN, SPC0, or BIDIROE with SPC0 set, SPPR2-SPPR0 and SPR2-SPR0 in master mode will abort a transmission in progress and force the SPI into idle state. The remote slave cannot detect this, therefore the master must ensure that the remote slave is returned to idle state.

25.4.2 Slave Mode

The SPI operates in slave mode when the MSTR bit in SPI control register 1 is clear.

- Serial clock

  In slave mode, SCK is the SPI clock input from the master.
- MISO, MOSI pin

  In slave mode, the function of the serial data output pin (MISO) and serial data input pin (MOSI) is determined by the SPC0 bit and BIDIROE bit in SPI control register 2.
- $\overline{SS}$ pin

  The $\overline{SS}$ pin is the slave select input. Before a data transmission occurs, the $\overline{SS}$ pin of the slave SPI must be low. $\overline{SS}$ must remain low until the transmission is complete. If $\overline{SS}$ goes high, the SPI is forced into idle state.

  The $\overline{SS}$ input also controls the serial data output pin, if $\overline{SS}$ is high (not selected), the serial data output pin is high impedance, and, if $\overline{SS}$ is low, the first bit in the SPI data register is driven out of the serial data output pin. Also, if the slave is not selected ($\overline{SS}$ is high), then the SCK input is ignored and no internal shifting of the SPI shift register occurs.

  Although the SPI is capable of duplex operation, some SPI peripherals are capable of only receiving SPI data in a slave mode. For these simpler devices, there is no serial data out pin.

NOTE

When peripherals with duplex capability are used, take care not to simultaneously enable two receivers whose serial outputs drive the same system slave's serial data output line.

As long as no more than one slave device drives the system slave's serial data output line, it is possible for several slaves to receive the same transmission from a master, although the master would not receive return information from all of the receiving slaves.

If the CPHA bit in SPI control register 1 is clear, odd numbered edges on the SCK input cause the data at the serial data input pin to be latched. Even numbered edges cause the value previously latched from the serial data input pin to shift into the LSB or MSB of the SPI shift register, depending on the LSBFE bit.

If the CPHA bit is set, even numbered edges on the SCK input cause the data at the serial data input pin to be latched. Odd numbered edges cause the value previously latched from the serial data input pin to shift into the LSB or MSB of the SPI shift register, depending on the LSBFE bit.

When CPHA is set, the first edge is used to get the first data bit onto the serial data output pin. When CPHA is clear and the $\overline{SS}$ input is low (slave selected), the first bit of the SPI data is driven out of the serial data output pin. After the nth[1] shift, the transfer is considered complete and the received data is transferred into the SPI data register. To indicate transfer is complete, the SPIF flag in the SPI status register is set.

NOTE

A change of the bits CPOL, CPHA, SSOE, LSBFE, MODFEN, SPC0, or BIDIROE with SPC0 set in slave mode will corrupt a transmission in progress and must be avoided.

1. n depends on the selected transfer width, please refer to Section 25.3.2.2, "SPI Control Register 2 (SPICR2)

25.4.3 Transmission Formats

During an SPI transmission, data is transmitted (shifted out serially) and received (shifted in serially) simultaneously. The serial clock (SCK) synchronizes shifting and sampling of the information on the two serial data lines. A slave select line allows selection of an individual slave SPI device; slave devices that are not selected do not interfere with SPI bus activities. Optionally, on a master SPI device, the slave select line can be used to indicate multiple-master bus contention.

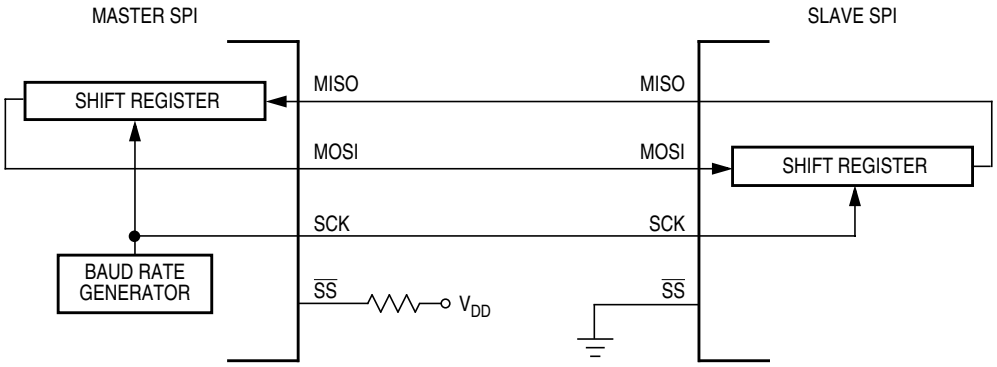


Figure 25-11. Master/Slave Transfer Block Diagram

25.4.3.1 Clock Phase and Polarity Controls

Using two bits in the SPI control register 1, software selects one of four combinations of serial clock phase and polarity.

The CPOL clock polarity control bit specifies an active high or low clock and has no significant effect on the transmission format.

The CPHA clock phase control bit selects one of two fundamentally different transmission formats.

Clock phase and polarity should be identical for the master SPI device and the communicating slave device. In some cases, the phase and polarity are changed between transmissions to allow a master device to communicate with peripheral slaves having different requirements.

25.4.3.2 CPHA = 0 Transfer Format

The first edge on the SCK line is used to clock the first data bit of the slave into the master and the first data bit of the master into the slave. In some peripherals, the first bit of the slave's data is available at the slave's data out pin as soon as the slave is selected. In this format, the first SCK edge is issued a half cycle after $\overline{SS}$ has become low.

A half SCK cycle later, the second edge appears on the SCK line. When this second edge occurs, the value previously latched from the serial data input pin is shifted into the LSB or MSB of the shift register, depending on LSBFE bit.

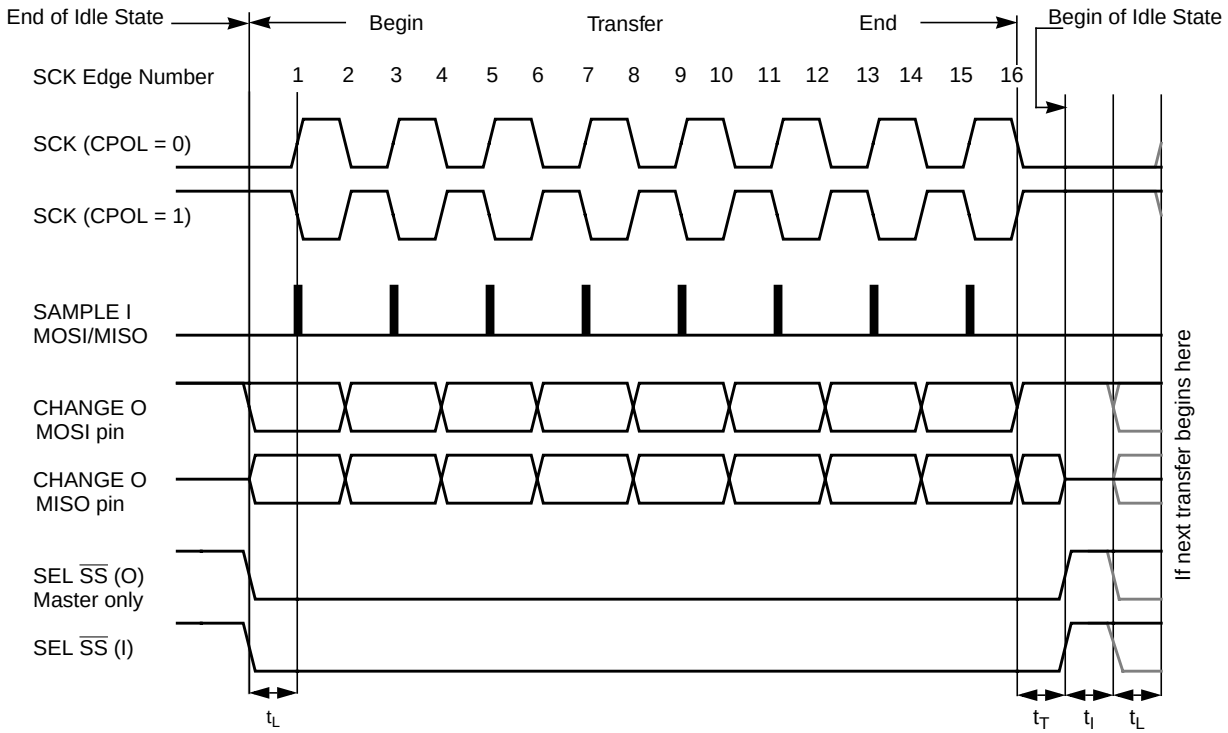
After this second edge, the next bit of the SPI master data is transmitted out of the serial data output pin of the master to the serial input pin on the slave. This process continues for a total of 16 edges on the SCK line, with data being latched on odd numbered edges and shifted on even numbered edges.

Data reception is double buffered. Data is shifted serially into the SPI shift register during the transfer and is transferred to the parallel SPI data register after the last bit is shifted in.

After 2n[1] (last) SCK edges:

- Data that was previously in the master SPI data register should now be in the slave data register and the data that was in the slave data register should be in the master.
- The SPIF flag in the SPI status register is set, indicating that the transfer is complete.

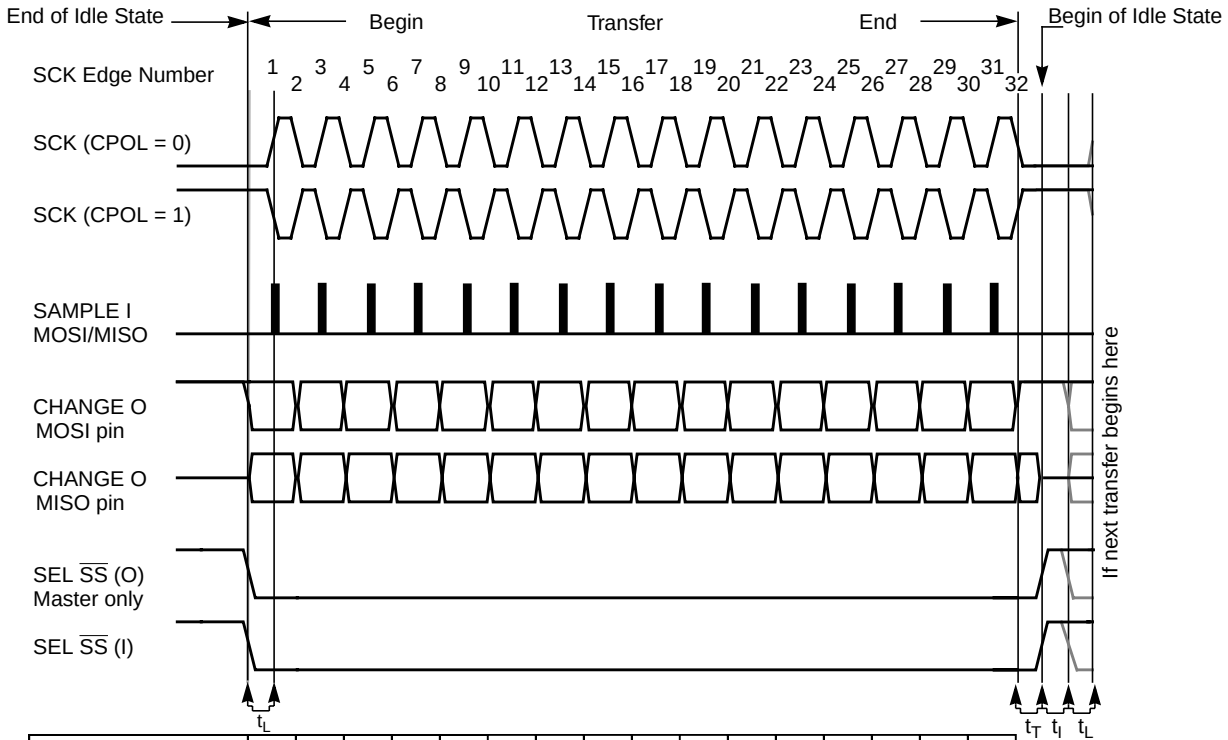
Figure 25-12 is a timing diagram of an SPI transfer where CPHA = 0. SCK waveforms are shown for CPOL = 0 and CPOL = 1. The diagram may be interpreted as a master or slave timing diagram because the SCK, MISO, and MOSI pins are connected directly between the master and the slave. The MISO signal is the output from the slave and the MOSI signal is the output from the master. The $\overline{SS}$ pin of the master must be either high or reconfigured as a general-purpose output not affecting the SPI.



t_L = Minimum leading time before the first SCK edge
t_T = Minimum trailing time after the last SCK edge
t_I = Minimum idling time between transfers (minimum $\overline{SS}$ high time)
t_L, t_T, and t_I are guaranteed for the master mode and required for the slave mode.

Figure 25-12. SPI Clock Format 0 (CPHA = 0), with 8-bit Transfer Width selected (XFRW = 0)

1. n depends on the selected transfer width, please refer to Section 25.3.2.2, “SPI Control Register 2 (SPICR2)



t_L = Minimum leading time before the first SCK edge

t_T = Minimum trailing time after the last SCK edge

t_I = Minimum idling time between transfers (minimum $\overline{SS}$ high time)

t_L, t_T, and t_I are guaranteed for the master mode and required for the slave mode.

Figure 25-13. SPI Clock Format 0 (CPHA = 0), with 16-Bit Transfer Width selected (XFRW = 1)

In slave mode, if the $\overline{SS}$ line is not deasserted between the successive transmissions then the content of the SPI data register is not transmitted; instead the last received data is transmitted. If the $\overline{SS}$ line is deasserted for at least minimum idle time (half SCK cycle) between successive transmissions, then the content of the SPI data register is transmitted.

In master mode, with slave select output enabled the $\overline{SS}$ line is always deasserted and reasserted between successive transfers for at least minimum idle time.

25.4.3.3 CPHA = 1 Transfer Format

Some peripherals require the first SCK edge before the first data bit becomes available at the data out pin, the second edge clocks data into the system. In this format, the first SCK edge is issued by setting the CPHA bit at the beginning of the n[1]-cycle transfer operation.

The first edge of SCK occurs immediately after the half SCK clock cycle synchronization delay. This first edge commands the slave to transfer its first data bit to the serial data input pin of the master.

A half SCK cycle later, the second edge appears on the SCK pin. This is the latching edge for both the master and slave.

When the third edge occurs, the value previously latched from the serial data input pin is shifted into the LSB or MSB of the SPI shift register, depending on LSBFE bit. After this edge, the next bit of the master data is coupled out of the serial data output pin of the master to the serial input pin on the slave.

This process continues for a total of n[1] edges on the SCK line with data being latched on even numbered edges and shifting taking place on odd numbered edges.

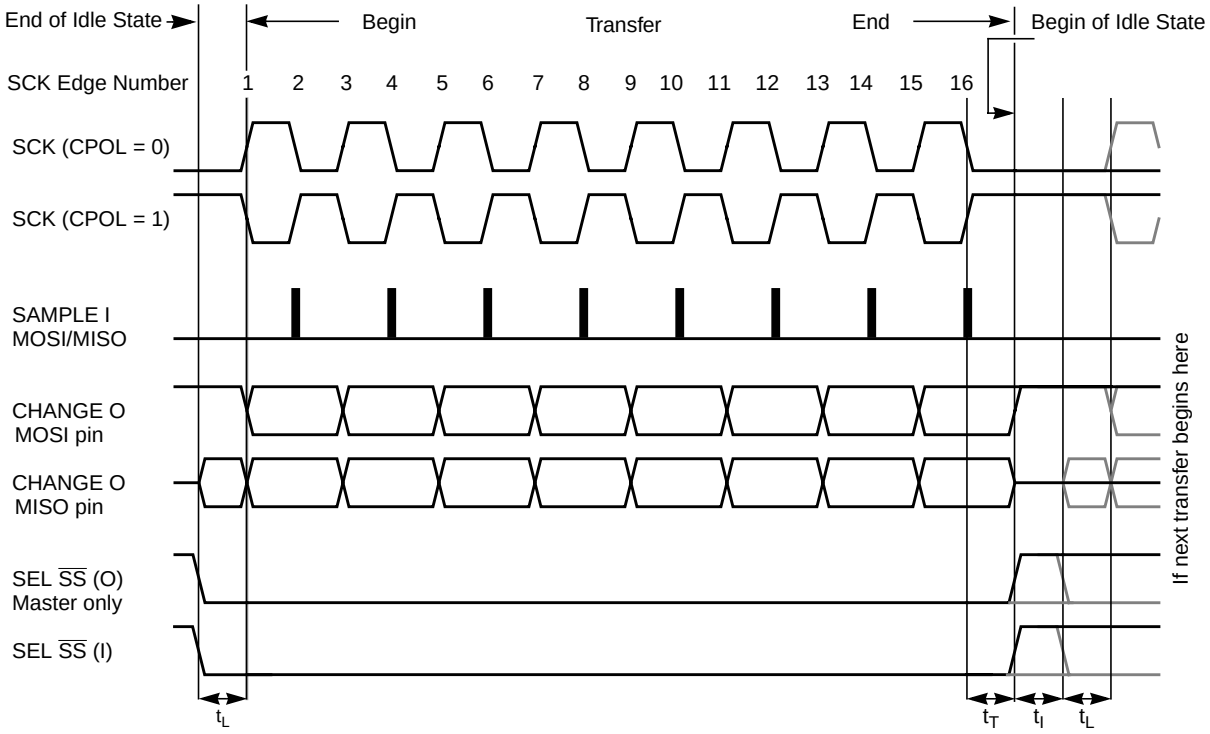
Data reception is double buffered, data is serially shifted into the SPI shift register during the transfer and is transferred to the parallel SPI data register after the last bit is shifted in.

After 2n[1] SCK edges:

- Data that was previously in the SPI data register of the master is now in the data register of the slave, and data that was in the data register of the slave is in the master.
- The SPIF flag bit in SPISR is set indicating that the transfer is complete.

Figure 25-14 shows two clocking variations for CPHA = 1. The diagram may be interpreted as a master or slave timing diagram because the SCK, MISO, and MOSI pins are connected directly between the master and the slave. The MISO signal is the output from the slave, and the MOSI signal is the output from the master. The $\overline{SS}$ line is the slave select input to the slave. The $\overline{SS}$ pin of the master must be either high or reconfigured as a general-purpose output not affecting the SPI.

1. n depends on the selected transfer width, please refer to Section 25.3.2.2, “SPI Control Register 2 (SPICR2)

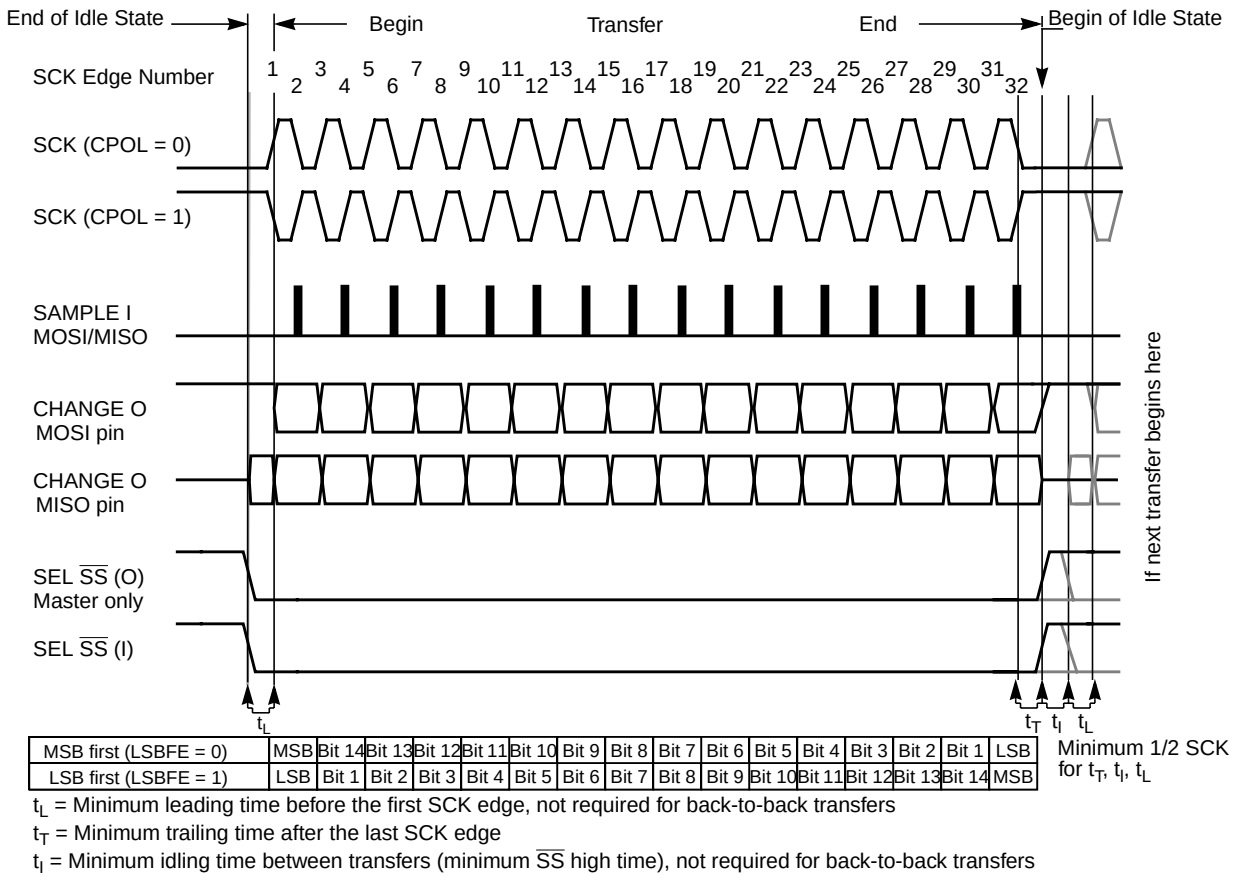


t_L = Minimum leading time before the first SCK edge, not required for back-to-back transfers

t_T = Minimum trailing time after the last SCK edge

t_I = Minimum idling time between transfers (minimum $\overline{SS}$ high time), not required for back-to-back transfers

Figure 25-14. SPI Clock Format 1 (CPHA = 1), with 8-Bit Transfer Width selected (XFRW = 0)



t_L = Minimum leading time before the first SCK edge, not required for back-to-back transfers

t_T = Minimum trailing time after the last SCK edge

t_I = Minimum idling time between transfers (minimum $\overline{SS}$ high time), not required for back-to-back transfers

Figure 25-15. SPI Clock Format 1 (CPHA = 1), with 16-Bit Transfer Width selected (XFRW = 1)

The $\overline{SS}$ line can remain active low between successive transfers (can be tied low at all times). This format is sometimes preferred in systems having a single fixed master and a single slave that drive the MISO data line.

- Back-to-back transfers in master mode

  In master mode, if a transmission has completed and new data is available in the SPI data register, this data is sent out immediately without a trailing and minimum idle time.

The SPI interrupt request flag (SPIF) is common to both the master and slave modes. SPIF gets set one half SCK cycle after the last SCK edge.

25.4.4 SPI Baud Rate Generation

Baud rate generation consists of a series of divider stages. Six bits in the SPI baud rate register (SPPR2, SPPR1, SPPR0, SPR2, SPR1, and SPR0) determine the divisor to the SPI module clock which results in the SPI baud rate.

The SPI clock rate is determined by the product of the value in the baud rate preselection bits (SPPR2–SPPR0) and the value in the baud rate selection bits (SPR2–SPR0). The module clock divisor equation is shown in Equation 25-3.

$$\text{BaudRateDivisor} = (\text{SPPR} + 1) \bullet 2^{(\text{SPR} + 1)} \qquad \textit{Eqn. 25-3}$$

When all bits are clear (the default condition), the SPI module clock is divided by 2. When the selection bits (SPR2–SPR0) are 001 and the preselection bits (SPPR2–SPPR0) are 000, the module clock divisor becomes 4. When the selection bits are 010, the module clock divisor becomes 8, etc.

When the preselection bits are 001, the divisor determined by the selection bits is multiplied by 2. When the preselection bits are 010, the divisor is multiplied by 3, etc. See Table 25-6 for baud rate calculations for all bit conditions, based on a 25 MHz bus clock. The two sets of selects allows the clock to be divided by a non-power of two to achieve other baud rates such as divide by 6, divide by 10, etc.

The baud rate generator is activated only when the SPI is in master mode and a serial transfer is taking place. In the other cases, the divider is disabled to decrease I_{DD} current.

NOTE

For maximum allowed baud rates, please refer to the SPI Electrical Specification in the Electricals chapter of this data sheet.

25.4.5 Special Features

25.4.5.1 $\overline{SS}$ Output

The $\overline{SS}$ output feature automatically drives the $\overline{SS}$ pin low during transmission to select external devices and drives it high during idle to deselect external devices. When $\overline{SS}$ output is selected, the $\overline{SS}$ output pin is connected to the $\overline{SS}$ input pin of the external device.

The $\overline{SS}$ output is available only in master mode during normal SPI operation by asserting SSOE and MODFEN bit as shown in Table 25-2.

The mode fault feature is disabled while $\overline{SS}$ output is enabled.

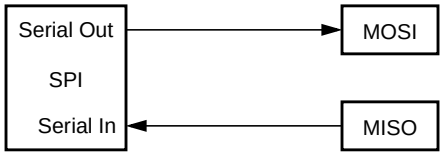
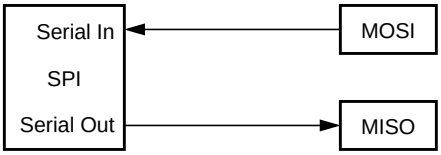
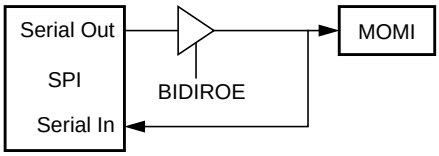
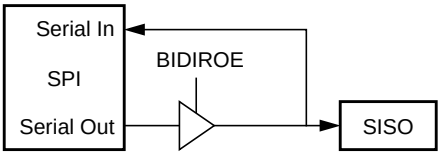
NOTE

Care must be taken when using the $\overline{SS}$ output feature in a multimaster system because the mode fault feature is not available for detecting system errors between masters.

25.4.5.2 Bidirectional Mode (MOMI or SISO)

The bidirectional mode is selected when the SPC0 bit is set in SPI control register 2 (see Table 25-10). In this mode, the SPI uses only one serial data pin for the interface with external device(s). The MSTR bit decides which pin to use. The MOSI pin becomes the serial data I/O (MOMI) pin for the master mode, and the MISO pin becomes serial data I/O (SISO) pin for the slave mode. The MISO pin in master mode and MOSI pin in slave mode are not used by the SPI.

Table 25-10. Normal Mode and Bidirectional Mode

When SPE = 1	Master Mode MSTR = 1	Slave Mode MSTR = 0
Normal Mode SPC0 = 0		
Bidirectional Mode SPC0 = 1		

The direction of each serial I/O pin depends on the BIDIROE bit. If the pin is configured as an output, serial data from the shift register is driven out on the pin. The same pin is also the serial input to the shift register.

- The SCK is output for the master mode and input for the slave mode.
- The $\overline{SS}$ is the input or output for the master mode, and it is always the input for the slave mode.
- The bidirectional mode does not affect SCK and $\overline{SS}$ functions.

NOTE

In bidirectional master mode, with mode fault enabled, both data pins MISO and MOSI can be occupied by the SPI, though MOSI is normally used for transmissions in bidirectional mode and MISO is not used by the SPI. If a mode fault occurs, the SPI is automatically switched to slave mode. In this case MISO becomes occupied by the SPI and MOSI is not used. This must be considered, if the MISO pin is used for another purpose.

25.4.6 Error Conditions

The SPI has one error condition:

- Mode fault error

25.4.6.1 Mode Fault Error

If the $\overline{SS}$ input becomes low while the SPI is configured as a master, it indicates a system error where more than one master may be trying to drive the MOSI and SCK lines simultaneously. This condition is not permitted in normal operation, the MODF bit in the SPI status register is set automatically, provided the MODFEN bit is set.

In the special case where the SPI is in master mode and MODFEN bit is cleared, the $\overline{SS}$ pin is not used by the SPI. In this special case, the mode fault error function is inhibited and MODF remains cleared. In case the SPI system is configured as a slave, the $\overline{SS}$ pin is a dedicated input pin. Mode fault error doesn't occur in slave mode.

If a mode fault error occurs, the SPI is switched to slave mode, with the exception that the slave output buffer is disabled. So SCK, MISO, and MOSI pins are forced to be high impedance inputs to avoid any possibility of conflict with another output driver. A transmission in progress is aborted and the SPI is forced into idle state.

If the mode fault error occurs in the bidirectional mode for a SPI system configured in master mode, output enable of the MOMI (MOSI in bidirectional mode) is cleared if it was set. No mode fault error occurs in the bidirectional mode for SPI system configured in slave mode.

The mode fault flag is cleared automatically by a read of the SPI status register (with MODF set) followed by a write to SPI control register 1. If the mode fault flag is cleared, the SPI becomes a normal master or slave again.

NOTE

If a mode fault error occurs and a received data byte is pending in the receive shift register, this data byte will be lost.

25.4.7 Low Power Mode Options

25.4.7.1 SPI in Run Mode

In run mode with the SPI system enable (SPE) bit in the SPI control register clear, the SPI system is in a low-power, disabled state. SPI registers remain accessible, but clocks to the core of this module are disabled.

25.4.7.2 SPI in Wait Mode

SPI operation in wait mode depends upon the state of the SPISWAI bit in SPI control register 2.

- If SPISWAI is clear, the SPI operates normally when the CPU is in wait mode
- If SPISWAI is set, SPI clock generation ceases and the SPI module enters a power conservation state when the CPU is in wait mode.
  - If SPISWAI is set and the SPI is configured for master, any transmission and reception in progress stops at wait mode entry. The transmission and reception resumes when the SPI exits wait mode.
  - If SPISWAI is set and the SPI is configured as a slave, any transmission and reception in progress continues if the SCK continues to be driven from the master. This keeps the slave synchronized to the master and the SCK.

    If the master transmits several bytes while the slave is in wait mode, the slave will continue to send out bytes consistent with the operation mode at the start of wait mode (i.e., if the slave is currently sending its SPIDR to the master, it will continue to send the same byte. Else if the slave is currently sending the last received byte from the master, it will continue to send each previous master byte).

NOTE

Care must be taken when expecting data from a master while the slave is in wait or stop mode. Even though the shift register will continue to operate, the rest of the SPI is shut down (i.e., a SPIF interrupt will **not** be generated until exiting stop or wait mode). Also, the byte from the shift register will not be copied into the SPIDR register until after the slave SPI has exited wait or stop mode. In slave mode, a received byte pending in the receive shift register will be lost when entering wait or stop mode. An SPIF flag and SPIDR copy is generated only if wait mode is entered or exited during a tranmission. If the slave enters wait mode in idle mode and exits wait mode in idle mode, neither a SPIF nor a SPIDR copy will occur.

25.4.7.3 SPI in Stop Mode

Stop mode is dependent on the system. The SPI enters stop mode when the module clock is disabled (held high or low). If the SPI is in master mode and exchanging data when the CPU enters stop mode, the transmission is frozen until the CPU exits stop mode. After stop, data to and from the external SPI is exchanged correctly. In slave mode, the SPI will stay synchronized with the master.

The stop mode is not dependent on the SPISWAI bit.

25.4.7.4 Reset

The reset values of registers and signals are described in Section 25.3, “Memory Map and Register Definition”, which details the registers and their bit fields.

- If a data transmission occurs in slave mode after reset without a write to SPIDR, it will transmit garbage, or the data last received from the master before the reset.
- Reading from the SPIDR after reset will always read zeros.

25.4.7.5 Interrupts

The SPI only originates interrupt requests when SPI is enabled (SPE bit in SPICR1 set). The following is a description of how the SPI makes a request and how the MCU should acknowledge that request. The interrupt vector offset and interrupt priority are chip dependent.

The interrupt flags MODF, SPIF, and SPTEF are logically ORed to generate an interrupt request.

25.4.7.5.1 MODF

MODF occurs when the master detects an error on the $\overline{SS}$ pin. The master SPI must be configured for the MODF feature (see Table 25-2). After MODF is set, the current transfer is aborted and the following bit is changed:

- MSTR = 0, The master bit in SPICR1 resets.

The MODF interrupt is reflected in the status register MODF flag. Clearing the flag will also clear the interrupt. This interrupt will stay active while the MODF flag is set. MODF has an automatic clearing process which is described in Section 25.3.2.4, “SPI Status Register (SPISR)”.

25.4.7.5.2 SPIF

SPIF occurs when new data has been received and copied to the SPI data register. After SPIF is set, it does not clear until it is serviced. SPIF has an automatic clearing process, which is described in Section 25.3.2.4, “SPI Status Register (SPISR)”.

25.4.7.5.3 SPTEF

SPTEF occurs when the SPI data register is ready to accept new data. After SPTEF is set, it does not clear until it is serviced. SPTEF has an automatic clearing process, which is described in Section 25.3.2.4, “SPI Status Register (SPISR)”.

Chapter 26
Enhanced Capture Timer (ECT16B8CV3)

Table 26-1. Revision History

Revision Number	Revision Date	Sections Affected	Description of Changes
V03.05	09 Sep 2008	26.4.1.2/26-1175	Fixed a typo and refined a statement pertaining to switch between I/O and timer functionlality.
V03.06	05 Aug 2009	26.3.2.15/26-1149 26.3.2.16/26-1151 26.3.2.24/26-1157 26.3.2.29/26-1162 26.4.1.1.2/26-1173	update register PACTL bit4 PEDGE PT7 to IC7 update register PAFLG bit0 PAIF PT7 to IC7,update bit1 PAOVF PT3 to IC3 update register ICSYS bit3 TFMOD PTx to ICx update register PBFLG bit1 PBOVF PT1 to IC1 update IC Queue Mode description.
V03.07	26 Aug 2009	26.3.2.2/26-1136 26.3.2.3/26-1136 26.3.2.4/26-1137	- Add description, ?a counter overflow when TTOV[7] is set?, to be the condition of channel 7 override event. - Phrase the description of OC7M to make it more explicit
V03.08	04 May 2010	26.3.2.8/26-1140 26.3.2.11/26-1143	- Add Table 26-11 - TCRE description part,add Note and Figure 26-17

26.1 Introduction

The HCS12 enhanced capture timer module has the features of the HCS12 standard timer module enhanced by additional features in order to enlarge the field of applications, in particular for automotive ABS applications.

This design specification describes the standard timer as well as the additional features.

The basic timer consists of a 16-bit, software-programmable counter driven by a prescaler. This timer can be used for many purposes, including input waveform measurements while simultaneously generating an output waveform. Pulse widths can vary from microseconds to many seconds.

A full access for the counter registers or the input capture/output compare registers will take place in one clock cycle. Accessing high byte and low byte separately for all of these registers will not yield the same result as accessing them in one word.

26.1.1 Features

- 16-bit buffer register for four input capture (IC) channels.
- Four 8-bit pulse accumulators with 8-bit buffer registers associated with the four buffered IC channels. Configurable also as two 16-bit pulse accumulators.
- 16-bit modulus down-counter with 8-bit prescaler.
- Four user-selectable delay counters for input noise immunity increase.

26.1.2 Modes of Operation

- Stop — Timer and modulus counter are off since clocks are stopped.
- Freeze — Timer and modulus counter keep on running, unless the TSFRZ bit in the TSCR1 register is set to one.
- Wait — Counters keep on running, unless the TSWAI bit in the TSCR1 register is set to one.
- Normal — Timer and modulus counter keep on running, unless the TEN bit in the TSCR1 register or the MCEN bit in the MCCTL register are cleared.

26.1.3 Block Diagram

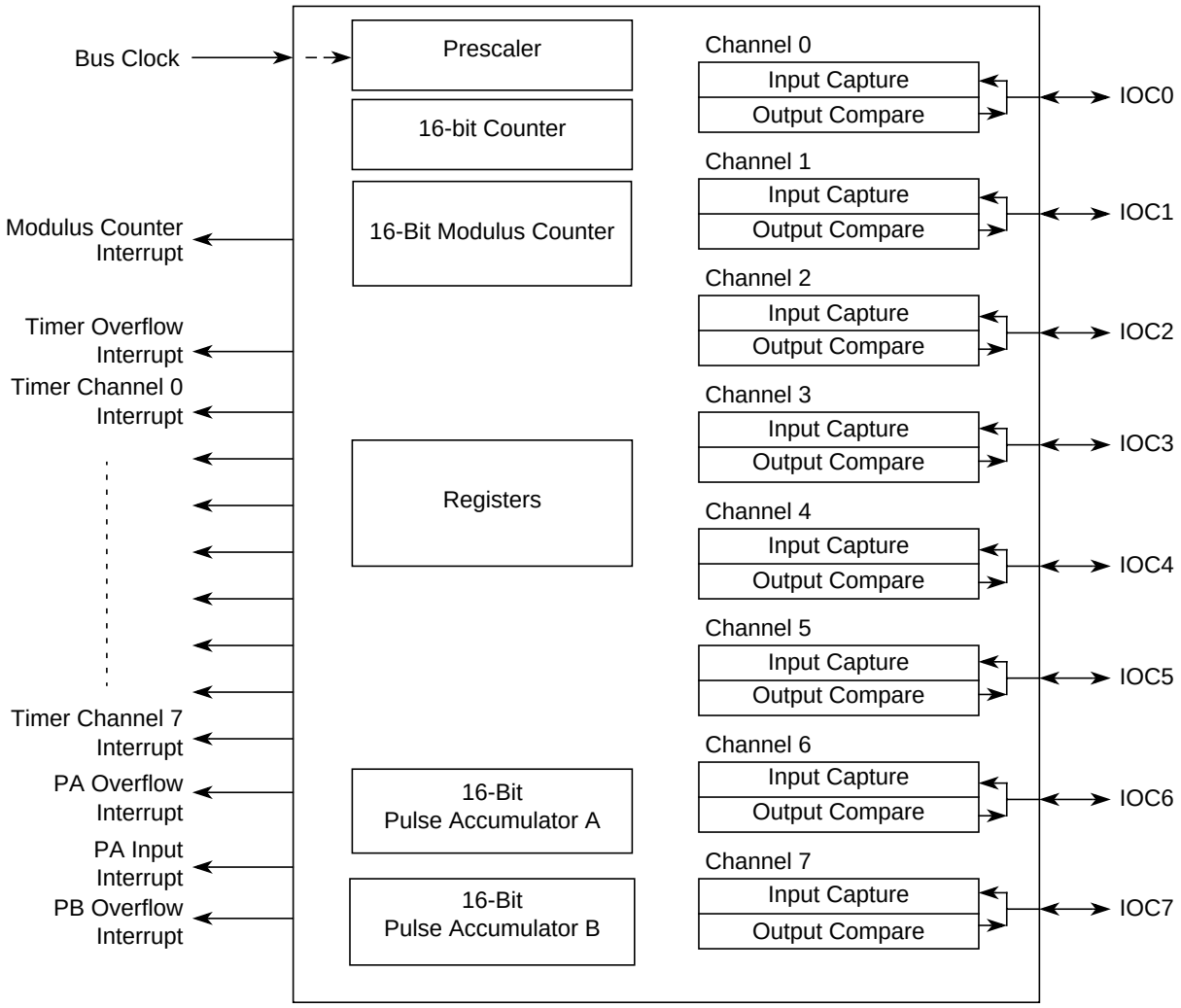


Figure 26-1. ECT Block Diagram

26.2 External Signal Description

The ECT module has a total of eight external pins.

26.2.1 IOC7 — Input Capture and Output Compare Channel 7

This pin serves as input capture or output compare for channel 7.

26.2.2 IOC6 — Input Capture and Output Compare Channel 6

This pin serves as input capture or output compare for channel 6.

26.2.3 IOC5 — Input Capture and Output Compare Channel 5

This pin serves as input capture or output compare for channel 5.

26.2.4 IOC4 — Input Capture and Output Compare Channel 4

This pin serves as input capture or output compare for channel 4.

26.2.5 IOC3 — Input Capture and Output Compare Channel 3

This pin serves as input capture or output compare for channel 3.

26.2.6 IOC2 — Input Capture and Output Compare Channel 2

This pin serves as input capture or output compare for channel 2.

26.2.7 IOC1 — Input Capture and Output Compare Channel 1

This pin serves as input capture or output compare for channel 1.

26.2.8 IOC0 — Input Capture and Output Compare Channel 0

This pin serves as input capture or output compare for channel 0.

NOTE

For the description of interrupts see Section 26.4.3, “Interrupts”.

26.3 Memory Map and Register Definition

This section provides a detailed description of all memory and registers.

26.3.1 Module Memory Map

The memory map for the ECT module is given below in the Table 26-2. The address listed for each register is the address offset. The total address for each register is the sum of the base address for the ECT module and the address offset for each register.

26.3.2 Register Descriptions

This section consists of register descriptions in address order. Each description includes a standard register diagram with an associated figure number. Details of register bit and field function follow the register diagrams, in bit order.

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x0000 TIOS	R	IOS7	IOS6	IOS5	IOS4	IOS3	IOS2	IOS1	IOS0
	W								
0x0001 CFORC	R	0	0	0	0	0	0	0	0
	W	FOC7	FOC6	FOC5	FOC4	FOC3	FOC2	FOC1	FOC0
0x0002 OC7M	R	OC7M7	OC7M6	OC7M5	OC7M4	OC7M3	OC7M2	OC7M1	OC7M0
	W								
0x0003 OC7D	R	OC7D7	OC7D6	OC7D5	OC7D4	OC7D3	OC7D2	OC7D1	OC7D0
	W								
0x0004 TCNT (High)	R	TCNT15	TCNT14	TCNT13	TCNT12	TCNT11	TCNT10	TCNT9	TCNT8
	W								
0x0005 TCNT (Low)	R	TCNT7	TCNT6	TCNT5	TCNT4	TCNT3	TCNT2	TCNT1	TCNT0
	W								
0x0006 TSCR1	R	TEN	TSWAI	TSFRZ	TFFCA	PRNT	0	0	0
	W								
0x0007 TTOF	R	TOV7	TOV6	TOV5	TOV4	TOV3	TOV2	TOV1	TOV0
	W								
0x0008 TCTL1	R	OM7	OL7	OM6	OL6	OM5	OL5	OM4	OL4
	W								
0x0009 TCTL2	R	OM3	OL3	OM2	OL2	OM1	OL1	OM0	OL0
	W								
0x000A TCTL3	R	EDG7B	EDG7A	EDG6B	EDG6A	EDG5B	EDG5A	EDG4B	EDG4A
	W								
0x000B TCTL4	R	EDG3B	EDG3A	EDG2B	EDG2A	EDG1B	EDG1A	EDG0B	EDG0A
	W								
0x000C TIE	R	C7I	C6I	C5I	C4I	C3I	C2I	C1I	C0I
	W								

= Unimplemented or Reserved

Figure 26-2. ECT Register Summary (Sheet 1 of 5)

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x000D TSCR2	R	TOI	0	0	0	TCRE	PR2	PR1	PR0
	W								
0x000E TFLG1	R	C7F	C6F	C5F	C4F	C3F	C2F	C1F	C0F
	W								
0x000F TFLG2	R	TOF	0	0	0	0	0	0	0
	W								
0x0010 TC0 (High)	R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
	W								
0x0011 TC0 (Low)	R	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	W								
0x0012 TC1 (High)	R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
	W								
0x0013 TC1 (Low)	R	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	W								
0x0014 TC2 (High)	R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
	W								
0x0015 TC2 (Low)	R	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	W								
0x0016 TC3 (High)	R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
	W								
0x0017 TC3 (Low)	R	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	W								
0x0018 TC4 (High)	R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
	W								
0x0019 TC4 (Low)	R	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	W								
0x001A TC5 (High)	R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
	W								
0x001B TC5 (Low)	R	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	W								

= Unimplemented or Reserved

Figure 26-2. ECT Register Summary (Sheet 2 of 5)

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x001C TC6 (High)	R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
	W								
0x001D TC6 (Low)	R	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	W								
0x001E TC7 (High)	R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
	W								
0x001F TC7 (Low)	R	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	W								
0x0020 PACTL	R	0	PAEN	PAMOD	PEDGE	CLK1	CLK0	PA0VI	PAI
	W								
0x0021 PAFLG	R	0	0	0	0	0	0	PA0VF	PAIF
	W								
0x0022 PACN3	R	PACNT7(15)	PACNT6(14)	PACNT5(13)	PACNT4(12)	PACNT3(11)	PACNT2(10)	PACNT1(9)	PACNT0(8)
	W								
0x0023 PACN2	R	PACNT7	PACNT6	PACNT5	PACNT4	PACNT3	PACNT2	PACNT1	PACNT0
	W								
0x0024 PACN1	R	PACNT7(15)	PACNT6(14)	PACNT5(13)	PACNT4(12)	PACNT3(11)	PACNT2(10)	PACNT1(9)	PACNT0(8)
	W								
0x0025 PACN0	R	PACNT7	PACNT6	PACNT5	PACNT4	PACNT3	PACNT2	PACNT1	PACNT0
	W								
0x0026 MCCTL	R	MCZI	MODMC	RDMCL	0	0	MCEN	MCPR1	MCPR0
	W				ICLAT	FLMC			
0x0027 MCFLG	R	MCZF	0	0	0	POLF3	POLF2	POLF1	POLF0
	W								
0x0028 ICPAR	R	0	0	0	0	PA3EN	PA2EN	PA1EN	PA0EN
	W								
0x0029 DLYCT	R	DLY7	DLY6	DLY5	DLY4	DLY3	DLY2	DLY1	DLY0
	W								
0x002A ICOVW	R	NOVW7	NOVW6	NOVW5	NOVW4	NOVW3	NOVW2	NOVW1	NOVW0
	W								

= Unimplemented or Reserved

Figure 26-2. ECT Register Summary (Sheet 3 of 5)

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x002B ICSYS	R	SH37	SH26	SH15	SH04	TFMOD	PACMX	BUFEN	LATQ
	W								
0x002C OCPD	R	OCPD7	OCPD6	OCPD5	OCPD4	OCPD3	OCPD2	OCPD1	OCPD0
	W								
0x002D TIMTST	R	Timer Test Register							
	W								
0x002E PTPSR	R	PTPS7	PTPS6	PTPS5	PTPS4	PTPS3	PTPS2	PTPS1	PTPS0
	W								
0x002F PTMCPSR	R	PTMPS7	PTMPS6	PTMPS5	PTMPS4	PTMPS3	PTMPS2	PTMPS1	PTMPS0
	W								
0x0030 PBCTL	R	0	PBEN	0	0	0	0	PBOVI	0
	W								
0x0031 PBFLG	R	0	0	0	0	0	0	PBOVF	0
	W								
0x0032 PA3H	R	PA3H7	PA3H6	PA3H5	PA3H4	PA3H3	PA3H2	PA3H1	PA3H0
	W								
0x0033 PA2H	R	PA2H7	PA2H6	PA2H5	PA2H4	PA2H3	PA2H2	PA2H1	PA2H0
	W								
0x0034 PA1H	R	PA1H7	PA1H6	PA1H5	PA1H4	PA1H3	PA1H2	PA1H1	PA1H0
	W								
0x0035 PA0H	R	PA0H7	PA0H6	PA0H5	PA0H4	PA0H3	PA0H2	PA0H1	PA0H0
	W								
0x0036 MCCNT (High)	R	MCCNT15	MCCNT14	MCCNT13	MCCNT12	MCCNT11	MCCNT10	MCCNT9	MCCNT8
	W								
0x0037 MCCNT (Low)	R	MCCNT7	MCCNT6	MCCNT5	MCCNT4	MCCNT3	MCCNT2	MCCNT1	MCCNT0
	W								
0x0038 TC0H (High)	R	TC15	TC14	TC13	TC12	TC11	TC10	TC9	TC8
	W								
0x0039 TC0H (Low)	R	TC7	TC6	TC5	TC4	TC3	TC2	TC1	TC0
		= Unimplemented or Reserved							

Figure 26-2. ECT Register Summary (Sheet 4 of 5)

Register Name		Bit 7	6	5	4	3	2	1	Bit 0
0x003A TC1H (High)	R	TC15	TC14	TC13	TC12	TC11	TC10	TC9	TC8
	W								
0x003B TC1H (Low)	R	TC7	TC6	TC5	TC4	TC3	TC2	TC1	TC0
	W								
0x003C TC2H (High)	R	TC15	TC14	TC13	TC12	TC11	TC10	TC9	TC8
	W								
0x003D TC2H (Low)	R	TC7	TC6	TC5	TC4	TC3	TC2	TC1	TC0
	W								
0x003E TC3H (High)	R	TC15	TC14	TC13	TC12	TC11	TC10	TC9	TC8
	W								
0x003F TC3H (Low)	R	TC7	TC6	TC5	TC4	TC3	TC2	TC1	TC0
	W								

= Unimplemented or Reserved

Figure 26-2. ECT Register Summary (Sheet 5 of 5)

26.3.2.1 Timer Input Capture/Output Compare Select Register (TIOS)

Module Base + 0x0000

	7	6	5	4	3	2	1	0
R W	IOS7	IOS6	IOS5	IOS4	IOS3	IOS2	IOS1	IOS0
Reset	0	0	0	0	0	0	0	0

Figure 26-3. Timer Input Capture/Output Compare Register (TIOS)

Read or write: Anytime

All bits reset to zero.

Table 26-2. TIOS Field Descriptions

Field	Description
7:0 IOS[7:0]	**Input Capture or Output Compare Channel Configuration** 0 The corresponding channel acts as an input capture. 1 The corresponding channel acts as an output compare.

26.3.2.2 Timer Compare Force Register (CFORC)

Module Base + 0x0001

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0
W	FOC7	FOC6	FOC5	FOC4	FOC3	FOC2	FOC1	FOC0
Reset	0	0	0	0	0	0	0	0

Figure 26-4. Timer Compare Force Register (CFORC)

Read or write: Anytime but reads will always return 0x0000 (1 state is transient).

All bits reset to zero.

Table 26-3. CFORC Field Descriptions

Field	Description
7:0 FOC[7:0]	**Force Output Compare Action for Channel 7:0** — A write to this register with the corresponding data bit(s) set causes the action which is programmed for output compare “x” to occur immediately. The action taken is the same as if a successful comparison had just taken place with the TCx register except the interrupt flag does not get set. **Note:** A channel 7 event, which can be a counter overflow when TTOV[7] is set or A successful channel 7 output compare overrides any channel 6:0 compares. If a forced output compare on any channel occurs at the same time as the successful output compare, then the forced output compare action will take precedence and the interrupt flag will not get set.

26.3.2.3 Output Compare 7 Mask Register (OC7M)

Module Base + 0x0002

	7	6	5	4	3	2	1	0
R W	OC7M7	OC7M6	OC7M5	OC7M4	OC7M3	OC7M2	OC7M1	OC7M0
Reset	0	0	0	0	0	0	0	0

Figure 26-5. Output Compare 7 Mask Register (OC7M)

Read or write: Anytime

All bits reset to zero.

Table 26-4. OC7M Field Descriptions

Field	Description
7:0 OC7M[7:0]	**Output Compare Mask Action for Channel 7:0** **A channel 7 event, which can be a counter overflow when TTOV[7] is set or a successful output compare on channel 7, overrides any channel 6:0 compares. For each OC7M bit that is set,the output compare action reflects the corresponding OC7D bit.** 0 The corresponding OC7Dx bit in the output compare 7 data register will not be transferred to the timer port on a channel 7 event, even if the corresponding pin is setup for output compare. 1 The corresponding OC7Dx bit in the output compare 7 data register will be transferred to the timer port on a channel 7 event. **Note:** The corresponding channel must also be setup for output compare (IOSx = 1 andOCPDx = 0) for data to be transferred from the output compare 7 data register to the timer port.

26.3.2.4 Output Compare 7 Data Register (OC7D)

Module Base + 0x0003

	7	6	5	4	3	2	1	0
R W	OC7D7	OC7D6	OC7D5	OC7D4	OC7D3	OC7D2	OC7D1	OC7D0
Reset	0	0	0	0	0	0	0	0

Figure 26-6. Output Compare 7 Data Register (OC7D)

Read or write: Anytime

All bits reset to zero.

Table 26-5. OC7D Field Descriptions

Field	Description
7:0 OC7D[7:0]	**Output Compare 7 Data Bits** — A channel 7 event, which can be a counter overflow when TTOV[7] is set or A channel 7 output compare can cause bits in the output compare 7 data register to transfer to the timer port data register depending on the output compare 7 mask register.

26.3.2.5 Timer Count Register (TCNT)

Module Base + 0x0004

	15	14	13	12	11	10	9	8
R W	TCNT15	TCNT14	TCNT13	TCNT12	TCNT11	TCNT10	TCNT9	TCNT8
Reset	0	0	0	0	0	0	0	0

Figure 26-7. Timer Count Register High (TCNT)

Module Base + 0x0005

	7	6	5	4	3	2	1	0
R W	TCNT7	TCNT6	TCNT5	TCNT4	TCNT3	TCNT2	TCNT1	TCNT0
Reset	0	0	0	0	0	0	0	0

Figure 26-8. Timer Count Register Low (TCNT)

Read: Anytime

Write: Writable in special modes.

All bits reset to zero.

Table 26-6. TCNT Field Descriptions

Field	Description
15:0 TCNT[15:0]	**Timer Counter Bits** — The 16-bit main timer is an up counter. A read to this register will return the current value of the counter. Access to the counter register will take place in one clock cycle. **Note:** A separate read/write for high byte and low byte in test mode will give a different result than accessing them as a word. The period of the first count after a write to the TCNT registers may be a different size because the write is not synchronized with the prescaler clock.

26.3.2.6 Timer System Control Register 1 (TSCR1)

Module Base + 0x0006

	7	6	5	4	3	2	1	0
R	TEN	TSWAI	TSFRZ	TFFCA	PRNT	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 26-9. Timer System Control Register 1 (TSCR1)

Read or write: Anytime except PRNT bit is write once

All bits reset to zero.

Table 26-7. TSCR1 Field Descriptions

Field	Description
7 TEN	**Timer Enable** 0 Disables the main timer, including the counter. Can be used for reducing power consumption. 1 Allows the timer to function normally. **Note:** If for any reason the timer is not active, there is no ÷64 clock for the pulse accumulator since the ÷64 is generated by the timer prescaler.
6 TSWAI	**Timer Module Stops While in Wait** 0 Allows the timer module to continue running during wait. 1 Disables the timer counter, pulse accumulators and modulus down counter when the MCU is in wait mode. Timer interrupts cannot be used to get the MCU out of wait.

Table 26-7. TSCR1 Field Descriptions (continued)

Field	Description
5 TSFRZ	**Timer and Modulus Counter Stop While in Freeze Mode** 0 Allows the timer and modulus counter to continue running while in freeze mode. 1 Disables the timer and modulus counter whenever the MCU is in freeze mode. This is useful for emulation. The pulse accumulators do not stop in freeze mode.
4 TFFCA	**Timer Fast Flag Clear All** 0 Allows the timer flag clearing to function normally. 1 A read from an input capture or a write to the output compare channel registers causes the corresponding channel flag, CxF, to be cleared in the TFLG1 register. Any access to the TCNT register clears the TOF flag in the TFLG2 register. Any access to the PACN3 and PACN2 registers clears the PAOVF and PAIF flags in the PAFLG register. Any access to the PACN1 and PACN0 registers clears the PBOVF flag in the PBFLG register. Any access to the MCCNT register clears the MCZF flag in the MCFLG register. This has the advantage of eliminating software overhead in a separate clear sequence. Extra care is required to avoid accidental flag clearing due to unintended accesses. **Note:** The flags cannot be cleared via the normal flag clearing mechanism (writing a one to the flag) when TFFCA = 1.
3 PRNT	**Precision Timer** 0 Enables legacy timer. Only bits DLY0 and DLY1 of the DLYCT register are used for the delay selection of the delay counter. PR0, PR1, and PR2 bits of the TSCR2 register are used for timer counter prescaler selection. MCPR0 and MCPR1 bits of the MCCTL register are used for modulus down counter prescaler selection. 1 Enables precision timer. All bits in the DLYCT register are used for the delay selection, all bits of the PTPSR register are used for Precision Timer Prescaler Selection, and all bits of PTMCPSR register are used for the prescaler Precision Timer Modulus Counter Prescaler selection.

26.3.2.7 Timer Toggle On Overflow Register 1 (TTOV)

Module Base + 0x0007

	7	6	5	4	3	2	1	0
R W	TOV7	TOV6	TOV5	TOV4	TOV3	TOV2	TOV1	TOV0
Reset	0	0	0	0	0	0	0	0

Figure 26-10. Timer Toggle On Overflow Register 1 (TTOV)

Read or write: Anytime

All bits reset to zero.

Table 26-8. TTOV Field Descriptions

Field	Description
7:0 TOV[7:0]	**Toggle On Overflow Bits** — TOV97:0] toggles output compare pin on timer counter overflow. This feature only takes effect when in output compare mode. When set, it takes precedence over forced output compare but not channel 7 override events. 0 Toggle output compare pin on overflow feature disabled. 1 Toggle output compare pin on overflow feature enabled.

26.3.2.8 Timer Control Register 1/Timer Control Register 2 (TCTL1/TCTL2)

Module Base + 0x0008

	7	6	5	4	3	2	1	0
R W	OM7	OL7	OM6	OL6	OM5	OL5	OM4	OL4
Reset	0	0	0	0	0	0	0	0

Figure 26-11. Timer Control Register 1 (TCTL1)

Module Base + 0x0009

	7	6	5	4	3	2	1	0
R W	OM3	OL3	OM2	OL2	OM1	OL1	OM0	OL0
Reset	0	0	0	0	0	0	0	0

Figure 26-12. Timer Control Register 2 (TCTL2)

Read or write: Anytime

All bits reset to zero.

Table 26-9. TCTL1/TCTL2 Field Descriptions

Field	Description
OM[7:0] 7, 5, 3, 1	OMx — Output Mode OLx — Output Level These eight pairs of control bits are encoded to specify the output action to be taken as a result of a successful OCx compare. When either OMx or OLx is one, the pin associated with OCx becomes an output tied to OCx. See Table 26-10.
OL[7:0] 6, 4, 2, 0	

Table 26-10. Compare Result Output Action

OMx	OLx	Action
0	0	No output compare action on the timer output signal
0	1	Toggle OCx output line
1	0	Clear OCx output line to zero
1	1	Set OCx output line to one

NOTE

To enable output action by OMx and OLx bits on timer port, the corresponding bit in OC7M should be cleared. The settings for these bits can be seen in Table 26-11

Table 26-11. The OC7 and OCx event priority

OC7M7=0				OC7M7=1			
OC7Mx=1		OC7Mx=0		OC7Mx=1		OC7Mx=0	
TC7=TCx	TC7>TCx	TC7=TCx	TC7>TCx	TC7=TCx	TC7>TCx	TC7=TCx	TC7>TCx
IOCx=OC7Dx IOC7=OM7/OL7	IOCx=OC7Dx +OMx/OLx IOC7=OM7/OL7	IOCx=OMx/OLx IOC7=OM7/OL7		IOCx=OC7Dx IOC7=OC7D7	IOCx=OC7Dx +OMx/OLx IOC7=OC7D7	IOCx=OMx/OLx IOC7=OC7D7	

Note: in Table 26-11,the IOS7 and IOSx should be set to 1

IOSx is the register TIOS bit x,

OC7Mx is the register OC7M bit x,

TCx is timer Input Capture/Output Compare register,

IOCx is channel x,

OMx/OLx is the register TCTL1/TCTL2,

OC7Dx is the register OC7D bit x.

IOCx = OC7Dx+ OMx/OLx, means that both OC7 event and OCx event will change channel x value.

26.3.2.9 Timer Control Register 3/Timer Control Register 4 (TCTL3/TCTL4)

Module Base + 0x000A

	7	6	5	4	3	2	1	0
R W	EDG7B	EDG7A	EDG6B	EDG6A	EDG5B	EDG5A	EDG4B	EDG4A
Reset	0	0	0	0	0	0	0	0

Figure 26-13. Timer Control Register 3 (TCTL3)

Module Base + 0x000B

	7	6	5	4	3	2	1	0
R W	EDG3B	EDG3A	EDG2B	EDG2A	EDG1B	EDG1A	EDG0B	EDG0A
Reset	0	0	0	0	0	0	0	0

Figure 26-14. Timer Control Register 4 (TCTL4)

Read or write: Anytime

All bits reset to zero.

Table 26-12. TCTL3/TCTL4 Field Descriptions

Field	Description
EDG[7:0]B 7, 5, 3, 1	**Input Capture Edge Control** — These eight pairs of control bits configure the input capture edge detector circuits for each input capture channel. The four pairs of control bits in TCTL4 also configure the input capture edge control for the four 8-bit pulse accumulators PAC0–PAC3.EDG0B and EDG0A in TCTL4 also determine the active edge for the 16-bit pulse accumulator PACB. See Table 26-13.
EDG[7:0]A 6, 4, 2, 0	

Table 26-13. Edge Detector Circuit Configuration

EDGxB	EDGxA	Configuration
0	0	Capture disabled
0	1	Capture on rising edges only
1	0	Capture on falling edges only
1	1	Capture on any edge (rising or falling)

26.3.2.10 Timer Interrupt Enable Register (TIE)

Module Base + 0x000C

	7	6	5	4	3	2	1	0
R W	C7I	C6I	C5I	C4I	C3I	C2I	C1I	C0I
Reset	0	0	0	0	0	0	0	0

Figure 26-15. Timer Interrupt Enable Register (TIE)

Read or write: Anytime

All bits reset to zero.

The bits C7I–C0I correspond bit-for-bit with the flags in the TFLG1 status register.

Table 26-14. TIE Field Descriptions

Field	Description
7:0 C[7:0]I	**Input Capture/Output Compare “x” Interrupt Enable** 0 The corresponding flag is disabled from causing a hardware interrupt. 1 The corresponding flag is enabled to cause an interrupt.

26.3.2.11 Timer System Control Register 2 (TSCR2)

Module Base + 0x000D

	7	6	5	4	3	2	1	0
R	TOI	0	0	0	TCRE	PR2	PR1	PR0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 26-16. Timer System Control Register 2 (TSCR2)

Read or write: Anytime

All bits reset to zero.

Table 26-15. TSCR2 Field Descriptions

Field	Description
7 TOI	**Timer Overflow Interrupt Enable** 0 Timer overflow interrupt disabled. 1 Hardware interrupt requested when TOF flag set.
3 TCRE	**Timer Counter Reset Enable** — This bit allows the timer counter to be reset by a successful channel 7 output compare. This mode of operation is similar to an up-counting modulus counter. 0 Counter reset disabled and counter free runs. 1 Counter reset by a successful output compare on channel 7. **Note:** If register TC7 = 0x0000 and TCRE = 1, then the TCNT register will stay at 0x0000 continuously. If register TC7 = 0xFFFF and TCRE = 1, the TOF flag will never be set when TCNT is reset from 0xFFFF to 0x0000. **Note:** TCRE=1 and TC7!=0, the TCNT cycle period will be TC7 x "prescaler counter width" + "1 Bus Clock". When TCRE is set and TC7 is not equal to 0, TCNT will cycle from 0 to TC7. When TCNT reaches TC7 value, it will last only one bus cycle then reset to 0. for a more detail explanation please refer to Figure 26-17. **Note:** in Figure 26-17,if PR[2:0] is equal to 0, one prescaler counter equal to one bus clock
2:0 PR[2:0]	**Timer Prescaler Select** — These three bits specify the division rate of the main Timer prescaler when the PRNT bit of register TSCR1 is set to 0. The newly selected prescale factor will not take effect until the next synchronized edge where all prescale counter stages equal zero. See Table 26-16.

Figure 26-17. The TCNT cycle diagram under TCRE=1 condition

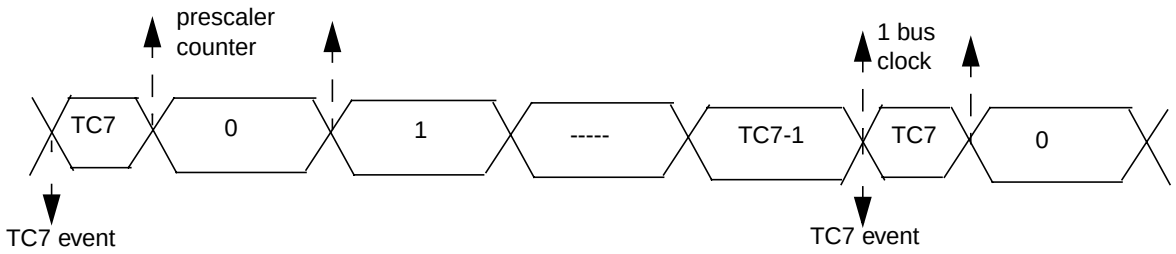


Table 26-16. Prescaler Selection

PR2	PR1	PR0	Prescale Factor
0	0	0	1
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

26.3.2.12 Main Timer Interrupt Flag 1 (TFLG1)

Module Base + 0x000E

	7	6	5	4	3	2	1	0
R W	C7F	C6F	C5F	C4F	C3F	C2F	C1F	C0F
Reset	0	0	0	0	0	0	0	0

Figure 26-18. Main Timer Interrupt Flag 1 (TFLG1)

Read: Anytime

Write used in the flag clearing mechanism. Writing a one to the flag clears the flag. Writing a zero will not affect the current status of the bit.

NOTE

When TFFCA = 1, the flags cannot be cleared via the normal flag clearing mechanism (writing a one to the flag). Reference Section 26.3.2.6, “Timer System Control Register 1 (TSCR1)”.

All bits reset to zero.

TFLG1 indicates when interrupt conditions have occurred. The flags can be cleared via the normal flag clearing mechanism (writing a one to the flag) or via the fast flag clearing mechanism (reference TFFCA bit in Section 26.3.2.6, “Timer System Control Register 1 (TSCR1)”).

Use of the TFMOD bit in the ICSYS register in conjunction with the use of the ICOVW register allows a timer interrupt to be generated after capturing two values in the capture and holding registers, instead of generating an interrupt for every capture.

Table 26-17. TFLG1 Field Descriptions

Field	Description
7:0 C[7:0]F	**Input Capture/Output Compare Channel “x” Flag** — A CxF flag is set when a corresponding input capture or output compare is detected. C0F can also be set by 16-bit Pulse Accumulator B (PACB). C3F–C0F can also be set by 8-bit pulse accumulators PAC3–PAC0. If the delay counter is enabled, the CxF flag will not be set until after the delay.

26.3.2.13 Main Timer Interrupt Flag 2 (TFLG2)

Module Base + 0x000F

	7	6	5	4	3	2	1	0
R	TOF	0	0	0	0	0	0	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 26-19. Main Timer Interrupt Flag 2 (TFLG2)

Read: Anytime

Write used in the flag clearing mechanism. Writing a one to the flag clears the flag. Writing a zero will not affect the current status of the bit.

NOTE

When TFFCA = 1, the flag cannot be cleared via the normal flag clearing mechanism (writing a one to the flag). Reference Section 26.3.2.6, “Timer System Control Register 1 (TSCR1)”.

All bits reset to zero.

TFLG2 indicates when interrupt conditions have occurred. The flag can be cleared via the normal flag clearing mechanism (writing a one to the flag) or via the fast flag clearing mechanism (Reference TFFCA bit in Section 26.3.2.6, “Timer System Control Register 1 (TSCR1)”).

Table 26-18. TFLG2 Field Descriptions

Field	Description
7 TOF	**Timer Overflow Flag** — Set when 16-bit free-running timer overflows from 0xFFFF to 0x0000.

26.3.2.14 Timer Input Capture/Output Compare Registers 0–7

Module Base + 0x0010

	15	14	13	12	11	10	9	8
R W	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Reset	0	0	0	0	0	0	0	0

Figure 26-20. Timer Input Capture/Output Compare Register 0 High (TC0)

Module Base + 0x0011

	7	6	5	4	3	2	1	0
R W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset	0	0	0	0	0	0	0	0

Figure 26-21. Timer Input Capture/Output Compare Register 0 Low (TC0)

Module Base + 0x0012

	15	14	13	12	11	10	9	8
R W	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Reset	0	0	0	0	0	0	0	0

Figure 26-22. Timer Input Capture/Output Compare Register 1 High (TC1)

Module Base + 0x0013

	7	6	5	4	3	2	1	0
R W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset	0	0	0	0	0	0	0	0

Figure 26-23. Timer Input Capture/Output Compare Register 1 Low (TC1)

Module Base + 0x0014

	15	14	13	12	11	10	9	8
R W	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Reset	0	0	0	0	0	0	0	0

Figure 26-24. Timer Input Capture/Output Compare Register 2 High (TC2)

Module Base + 0x0015

	7	6	5	4	3	2	1	0
R W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset	0	0	0	0	0	0	0	0

Figure 26-25. Timer Input Capture/Output Compare Register 2 Low (TC2)

Module Base + 0x0016

	15	14	13	12	11	10	9	8
R W	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Reset	0	0	0	0	0	0	0	0

Figure 26-26. Timer Input Capture/Output Compare Register 3 High (TC3)

Module Base + 0x0017

	7	6	5	4	3	2	1	0
R W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset	0	0	0	0	0	0	0	0

Figure 26-27. Timer Input Capture/Output Compare Register 3 Low (TC3)

Module Base + 0x0018

	15	14	13	12	11	10	9	8
R W	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Reset	0	0	0	0	0	0	0	0

Figure 26-28. Timer Input Capture/Output Compare Register 4 High (TC4)

Module Base + 0x0019

	7	6	5	4	3	2	1	0
R W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset	0	0	0	0	0	0	0	0

Figure 26-29. Timer Input Capture/Output Compare Register 4 Low (TC4)

Module Base + 0x001A

	15	14	13	12	11	10	9	8
R W	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Reset	0	0	0	0	0	0	0	0

Figure 26-30. Timer Input Capture/Output Compare Register 5 High (TC5)

Module Base + 0x001B

	7	6	5	4	3	2	1	0
R W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset	0	0	0	0	0	0	0	0

Figure 26-31. Timer Input Capture/Output Compare Register 5 Low (TC5)

Module Base + 0x001C

	15	14	13	12	11	10	9	8
R W	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Reset	0	0	0	0	0	0	0	0

Figure 26-32. Timer Input Capture/Output Compare Register 6 High (TC6)

Module Base + 0x001D

	7	6	5	4	3	2	1	0
R W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset	0	0	0	0	0	0	0	0

Figure 26-33. Timer Input Capture/Output Compare Register 6 Low (TC6)

Module Base + 0x001E

	15	14	13	12	11	10	9	8
R W	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
Reset	0	0	0	0	0	0	0	0

Figure 26-34. Timer Input Capture/Output Compare Register 7 High (TC7)

Module Base + 0x001F

	7	6	5	4	3	2	1	0
R W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Reset	0	0	0	0	0	0	0	0

Figure 26-35. Timer Input Capture/Output Compare Register 7 Low (TC7)

Read: Anytime

Write anytime for output compare function. Writes to these registers have no meaning or effect during input capture.

All bits reset to zero.

Depending on the TIOS bit for the corresponding channel, these registers are used to latch the value of the free-running counter when a defined transition is sensed by the corresponding input capture edge detector or to trigger an output action for output compare.

26.3.2.15 16-Bit Pulse Accumulator A Control Register (PACTL)

Module Base + 0x0020

	7	6	5	4	3	2	1	0
R	0	PAEN	PAMOD	PEDGE	CLK1	CLK0	PAOVI	PAI
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 26-36. 16-Bit Pulse Accumulator Control Register (PACTL)

Read: Anytime

Write: Anytime

All bits reset to zero.

Table 26-19. PACTL Field Descriptions

Field	Description
6 PAEN	**Pulse Accumulator A System Enable** — PAEN is independent from TEN. With timer disabled, the pulse accumulator can still function unless pulse accumulator is disabled. 0 16-Bit Pulse Accumulator A system disabled. 8-bit PAC3 and PAC2 can be enabled when their related enable bits in ICPAR are set. Pulse Accumulator Input Edge Flag (PAIF) function is disabled. 1 16-Bit Pulse Accumulator A system enabled. The two 8-bit pulse accumulators PAC3 and PAC2 are cascaded to form the PACA 16-bit pulse accumulator. When PACA in enabled, the PACN3 and PACN2 registers contents are respectively the high and low byte of the PACA. PA3EN and PA2EN control bits in ICPAR have no effect. Pulse Accumulator Input Edge Flag (PAIF) function is enabled. The PACA shares the input pin with IC7.
5 PAMOD	**Pulse Accumulator Mode** — This bit is active only when the Pulse Accumulator A is enabled (PAEN = 1). 0 Event counter mode 1 Gated time accumulation mode

Table 26-19. PACTL Field Descriptions (continued)

Field	Description
4 PEDGE	**Pulse Accumulator Edge Control** — This bit is active only when the Pulse Accumulator A is enabled (PAEN = 1). Refer to Table 26-20. For PAMOD bit = 0 (event counter mode). 0 Falling edges on IC7 pin cause the count to be incremented 1 Rising edges on IC7 pin cause the count to be incremented For PAMOD bit = 1 (gated time accumulation mode). 0 IC7 input pin high enables bus clock divided by 64 to Pulse Accumulator and the trailing falling edge on IC7 sets the PAIF flag. 1 IC7 input pin low enables bus clock divided by 64 to Pulse Accumulator and the trailing rising edge on IC7 sets the PAIF flag. If the timer is not active (TEN = 0 in TSCR1), there is no divide-by-64 since the ÷64 clock is generated by the timer prescaler.
3:2 CLK[1:0]	**Clock Select Bits** — For the description of PACLK please refer to Figure 26-72. If the pulse accumulator is disabled (PAEN = 0), the prescaler clock from the timer is always used as an input clock to the timer counter. The change from one selected clock to the other happens immediately after these bits are written. Refer to Table 26-21.
2 PAOVI	**Pulse Accumulator A Overflow Interrupt Enable** 0 Interrupt inhibited 1 Interrupt requested if PAOVF is set
0 PAI	**Pulse Accumulator Input Interrupt Enable** 0 Interrupt inhibited 1 Interrupt requested if PAIF is set

Table 26-20. Pin Action

PAMOD	PEDGE	Pin Action
0	0	Falling edge
0	1	Rising edge
1	0	Divide by 64 clock enabled with pin high level
1	1	Divide by 64 clock enabled with pin low level

Table 26-21. Clock Selection

CLK1	CLK0	Clock Source
0	0	Use timer prescaler clock as timer counter clock
0	1	Use PACLK as input to timer counter clock
1	0	Use PACLK/256 as timer counter clock frequency
1	1	Use PACLK/65536 as timer counter clock frequency

26.3.2.16 Pulse Accumulator A Flag Register (PAFLG)

Module Base + 0x0021

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	PAOVF	PAIF
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 26-37. Pulse Accumulator A Flag Register (PAFLG)

Read: Anytime

Write used in the flag clearing mechanism. Writing a one to the flag clears the flag. Writing a zero will not affect the current status of the bit.

NOTE

When TFFCA = 1, the flags cannot be cleared via the normal flag clearing mechanism (writing a one to the flag). Reference Section 26.3.2.6, “Timer System Control Register 1 (TSCR1)”.

All bits reset to zero.

PAFLG indicates when interrupt conditions have occurred. The flags can be cleared via the normal flag clearing mechanism (writing a one to the flag) or via the fast flag clearing mechanism (Reference TFFCA bit in Section 26.3.2.6, “Timer System Control Register 1 (TSCR1)”).

Table 26-22. PAFLG Field Descriptions

Field	Description
1 PAOVF	**Pulse Accumulator A Overflow Flag** — Set when the 16-bit pulse accumulator A overflows from 0xFFFF to 0x0000, or when 8-bit pulse accumulator 3 (PAC3) overflows from 0x00FF to 0x0000. When PACMX = 1, PAOVF bit can also be set if 8-bit pulse accumulator 3 (PAC3) reaches 0x00FF followed by an active edge on IC3.
0 PAIF	**Pulse Accumulator Input edge Flag** — Set when the selected edge is detected at the IC7 input pin. In event mode the event edge triggers PAIF and in gated time accumulation mode the trailing edge of the gate signal at the IC7 input pin triggers PAIF.

26.3.2.17 Pulse Accumulators Count Registers (PACN3 and PACN2)

Module Base + 0x0022

	7	6	5	4	3	2	1	0
R W	PACNT7(15)	PACNT6(14)	PACNT5(13)	PACNT4(12)	PACNT3(11)	PACNT2(10)	PACNT1(9)	PACNT0(8)
Reset	0	0	0	0	0	0	0	0

Figure 26-38. Pulse Accumulators Count Register 3 (PACN3)

Module Base + 0x0023

	7	6	5	4	3	2	1	0
R W	PACNT7	PACNT6	PACNT5	PACNT4	PACNT3	PACNT2	PACNT1	PACNT0
Reset	0	0	0	0	0	0	0	0

Figure 26-39. Pulse Accumulators Count Register 2 (PACN2)

Read: Anytime

Write: Anytime

All bits reset to zero.

The two 8-bit pulse accumulators PAC3 and PAC2 are cascaded to form the PACA 16-bit pulse accumulator. When PACA in enabled (PAEN = 1 in PACTL), the PACN3 and PACN2 registers contents are respectively the high and low byte of the PACA.

When PACN3 overflows from 0x00FF to 0x0000, the interrupt flag PAOVF in PAFLG is set.

Full count register access will take place in one clock cycle.

NOTE

A separate read/write for high byte and low byte will give a different result than accessing them as a word.

When clocking pulse and write to the registers occurs simultaneously, write takes priority and the register is not incremented.

26.3.2.18 Pulse Accumulators Count Registers (PACN1 and PACN0)

Module Base + 0x0024

	7	6	5	4	3	2	1	0
R W	PACNT7(15)	PACNT6(14)	PACNT5(13)	PACNT4(12)	PACNT3(11)	PACNT2(10)	PACNT1(9)	PACNT0(8)
Reset	0	0	0	0	0	0	0	0

Figure 26-40. Pulse Accumulators Count Register 1 (PACN1)

Module Base + 0x0025

	7	6	5	4	3	2	1	0
R W	PACNT7	PACNT6	PACNT5	PACNT4	PACNT3	PACNT2	PACNT1	PACNT0
Reset	0	0	0	0	0	0	0	0

Figure 26-41. Pulse Accumulators Count Register 0 (PACN0)

Read: Anytime

Write: Anytime

All bits reset to zero.

The two 8-bit pulse accumulators PAC1 and PAC0 are cascaded to form the PACB 16-bit pulse accumulator. When PACB in enabled, (PBEN = 1 in PBCTL) the PACN1 and PACN0 registers contents are respectively the high and low byte of the PACB.

When PACN1 overflows from 0x00FF to 0x0000, the interrupt flag PBOVF in PBFLG is set.

Full count register access will take place in one clock cycle.

NOTE

A separate read/write for high byte and low byte will give a different result than accessing them as a word.

When clocking pulse and write to the registers occurs simultaneously, write takes priority and the register is not incremented.

26.3.2.19 16-Bit Modulus Down-Counter Control Register (MCCTL)

Module Base + 0x0026

	7	6	5	4	3	2	1	0
R	MCZI	MODMC	RDMCL	0	0	MCEN	MCPR1	MCPR0
W				ICLAT	FLMC			
Reset	0	0	0	0	0	0	0	0

Figure 26-42. 16-Bit Modulus Down-Counter Control Register (MCCTL)

Read: Anytime

Write: Anytime

All bits reset to zero.

Table 26-23. MCCTL Field Descriptions

Field	Description
7 MCZI	**Modulus Counter Underflow Interrupt Enable** 0 Modulus counter interrupt is disabled. 1 Modulus counter interrupt is enabled.
6 MODMC	**Modulus Mode Enable** 0 The modulus counter counts down from the value written to it and will stop at 0x0000. 1 Modulus mode is enabled. When the modulus counter reaches 0x0000, the counter is loaded with the latest value written to the modulus count register. **Note:** For proper operation, the MCEN bit should be cleared before modifying the MODMC bit in order to reset the modulus counter to 0xFFFF.
5 RDMCL	**Read Modulus Down-Counter Load** 0 Reads of the modulus count register (MCCNT) will return the present value of the count register. 1 Reads of the modulus count register (MCCNT) will return the contents of the load register.

Table 26-23. MCCTL Field Descriptions (continued)

Field	Description
4 ICLAT	**Input Capture Force Latch Action** — When input capture latch mode is enabled (LATQ and BUFEN bit in ICSYS are set), a write one to this bit immediately forces the contents of the input capture registers TC0 to TC3 and their corresponding 8-bit pulse accumulators to be latched into the associated holding registers. The pulse accumulators will be automatically cleared when the latch action occurs. Writing zero to this bit has no effect. Read of this bit will always return zero.
3 FLMC	**Force Load Register into the Modulus Counter Count Register** — This bit is active only when the modulus down-counter is enabled (MCEN = 1). A write one into this bit loads the load register into the modulus counter count register (MCCNT). This also resets the modulus counter prescaler. Write zero to this bit has no effect. Read of this bit will return always zero.
2 MCEN	**Modulus Down-Counter Enable** 0 Modulus counter disabled. The modulus counter (MCCNT) is preset to 0xFFFF. This will prevent an early interrupt flag when the modulus down-counter is enabled. 1 Modulus counter is enabled.
1:0 MCPR[1:0]	**Modulus Counter Prescaler Select** — These two bits specify the division rate of the modulus counter prescaler when PRNT of TSCR1 is set to 0. The newly selected prescaler division rate will not be effective until a load of the load register into the modulus counter count register occurs.

Table 26-24. Modulus Counter Prescaler Select

MCPR1	MCPR0	Prescaler Division
0	0	1
0	1	4
1	0	8
1	1	16

26.3.2.20 16-Bit Modulus Down-Counter FLAG Register (MCFLG)

Module Base + 0x0027

	7	6	5	4	3	2	1	0
R	MCZF	0	0	0	POLF3	POLF2	POLF1	POLF0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 26-43. 16-Bit Modulus Down-Counter FLAG Register (MCFLG)

Read: Anytime

Write only used in the flag clearing mechanism for bit 7. Writing a one to bit 7 clears the flag. Writing a zero will not affect the current status of the bit.

NOTE

When TFFCA = 1, the flag cannot be cleared via the normal flag clearing mechanism (writing a one to the flag). Reference Section 26.3.2.6, “Timer System Control Register 1 (TSCR1)”.

All bits reset to zero.

Table 26-25. MCFLG Field Descriptions

Field	Description
7 MCZF	**Modulus Counter Underflow Flag** — The flag is set when the modulus down-counter reaches 0x0000. The flag indicates when interrupt conditions have occurred. The flag can be cleared via the normal flag clearing mechanism (writing a one to the flag) or via the fast flag clearing mechanism (Reference TFFCA bit in Section 26.3.2.6, “Timer System Control Register 1 (TSCR1)”).
3:0 POLF[3:0]	**First Input Capture Polarity Status** — These are read only bits. Writes to these bits have no effect. Each status bit gives the polarity of the first edge which has caused an input capture to occur after capture latch has been read. Each POLFx corresponds to a timer PORTx input. 0 The first input capture has been caused by a falling edge. 1 The first input capture has been caused by a rising edge.

26.3.2.21 ICPAR — Input Control Pulse Accumulators Register (ICPAR)

Module Base + 0x0028

	7	6	5	4	3	2	1	0
R	0	0	0	0	PA3EN	PA2EN	PA1EN	PA0EN
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 26-44. Input Control Pulse Accumulators Register (ICPAR)

Read: Anytime

Write: Anytime.

All bits reset to zero.

The 8-bit pulse accumulators PAC3 and PAC2 can be enabled only if PAEN in PACTL is cleared. If PAEN is set, PA3EN and PA2EN have no effect.

The 8-bit pulse accumulators PAC1 and PAC0 can be enabled only if PBEN in PBCTL is cleared. If PBEN is set, PA1EN and PA0EN have no effect.

Table 26-26. ICPAR Field Descriptions

Field	Description
3:0 PA[3:0]EN	**8-Bit Pulse Accumulator ‘x’ Enable** 0 8-Bit Pulse Accumulator is disabled. 1 8-Bit Pulse Accumulator is enabled.

26.3.2.22 Delay Counter Control Register (DLYCT)

Module Base + 0x0029

	7	6	5	4	3	2	1	0
R W	DLY7	DLY6	DLY5	DLY4	DLY3	DLY2	DLY1	DLY0
Reset	0	0	0	0	0	0	0	0

Figure 26-45. Delay Counter Control Register (DLYCT)

Read: Anytime

Write: Anytime

All bits reset to zero.

Table 26-27. DLYCT Field Descriptions

Field	Description
7:0 DLY[7:0]	**Delay Counter Select** — When the PRNT bit of TSCR1 register is set to 0, only bits DLY0, DLY1 are used to calculate the delay.Table 26-28 shows the delay settings in this case. When the PRNT bit of TSCR1 register is set to 1, all bits are used to set a more precise delay. Table 26-29 shows the delay settings in this case. After detection of a valid edge on an input capture pin, the delay counter counts the pre-selected number of [(dly_cnt + 1)*4]bus clock cycles, then it will generate a pulse on its output if the level of input signal, after the preset delay, is the opposite of the level before the transition.This will avoid reaction to narrow input pulses. Delay between two active edges of the input signal period should be longer than the selected counter delay. **Note:** It is recommended to not write to this register while the timer is enabled, that is when TEN is set in register TSCR1.

Table 26-28. Delay Counter Select when PRNT = 0

DLY1	DLY0	Delay
0	0	Disabled
0	1	256 bus clock cycles
1	0	512 bus clock cycles
1	1	1024 bus clock cycles

Table 26-29. Delay Counter Select Examples when PRNT = 1

DLY7	DLY6	DLY5	DLY4	DLY3	DLY2	DLY1	DLY0	Delay
0	0	0	0	0	0	0	0	Disabled (bypassed)
0	0	0	0	0	0	0	1	8 bus clock cycles
0	0	0	0	0	0	1	0	12 bus clock cycles
0	0	0	0	0	0	1	1	16 bus clock cycles
0	0	0	0	0	1	0	0	20 bus clock cycles
0	0	0	0	0	1	0	1	24 bus clock cycles
0	0	0	0	0	1	1	0	28 bus clock cycles

Table 26-29. Delay Counter Select Examples when PRNT = 1

DLY7	DLY6	DLY5	DLY4	DLY3	DLY2	DLY1	DLY0	Delay
0	0	0	0	0	1	1	1	32 bus clock cycles
0	0	0	0	1	1	1	1	64 bus clock cycles
0	0	0	1	1	1	1	1	128 bus clock cycles
0	0	1	1	1	1	1	1	256 bus clock cycles
0	1	1	1	1	1	1	1	512 bus clock cycles
1	1	1	1	1	1	1	1	1024 bus clock cycles

26.3.2.23 Input Control Overwrite Register (ICOVW)

Module Base + 0x002A

	7	6	5	4	3	2	1	0
R W	NOVW7	NOVW6	NOVW5	NOVW4	NOVW3	NOVW2	NOVW1	NOVW0
Reset	0	0	0	0	0	0	0	0

Figure 26-46. Input Control Overwrite Register (ICOVW)

Read: Anytime

Write: Anytime

All bits reset to zero.

Table 26-30. ICOVW Field Descriptions

Field	Description
7:0 NOVW[7:0]	**No Input Capture Overwrite** 0 The contents of the related capture register or holding register can be overwritten when a new input capture or latch occurs. 1 The related capture register or holding register cannot be written by an event unless they are empty (see Section 26.4.1.1, “IC Channels”). This will prevent the captured value being overwritten until it is read or latched in the holding register.

26.3.2.24 Input Control System Control Register (ICSYS)

Module Base + 0x002B

	7	6	5	4	3	2	1	0
R W	SH37	SH26	SH15	SH04	TFMOD	PACMX	BUFEN	LATQ
Reset	0	0	0	0	0	0	0	0

Figure 26-47. Input Control System Register (ICSYS)

Read: Anytime

Write: Once in normal modes

All bits reset to zero.

Table 26-31. ICSYS Field Descriptions

Field	Description
7:4 SHxy	**Share Input action of Input Capture Channels x and y** 0 Normal operation 1 The channel input ‘x’ causes the same action on the channel ‘y’. The port pin ‘x’ and the corresponding edge detector is used to be active on the channel ‘y’.
3 TFMOD	**Timer Flag Setting Mode** — Use of the TFMOD bit in conjunction with the use of the ICOVW register allows a timer interrupt to be generated after capturing two values in the capture and holding registers instead of generating an interrupt for every capture. By setting TFMOD in queue mode, when NOVWx bit is set and the corresponding capture and holding registers are emptied, an input capture event will first update the related input capture register with the main timer contents. At the next event, the TCx data is transferred to the TCxH register, the TCx is updated and the CxF interrupt flag is set. In all other input capture cases the interrupt flag is set by a valid external event on ICx. 0 The timer flags C3F–C0F in TFLG1 are set when a valid input capture transition on the corresponding port pin occurs. 1 If in queue mode (BUFEN = 1 and LATQ = 0), the timer flags C3F–C0F in TFLG1 are set only when a latch on the corresponding holding register occurs. If the queue mode is not engaged, the timer flags C3F–C0F are set the same way as for TFMOD = 0.
2 PACMX	**8-Bit Pulse Accumulators Maximum Count** 0 Normal operation. When the 8-bit pulse accumulator has reached the value 0x00FF, with the next active edge, it will be incremented to 0x0000. 1 When the 8-bit pulse accumulator has reached the value 0x00FF, it will not be incremented further. The value 0x00FF indicates a count of 255 or more.
1 BUFFEN	**IC Buffer Enable** 0 Input capture and pulse accumulator holding registers are disabled. 1 Input capture and pulse accumulator holding registers are enabled. The latching mode is defined by LATQ control bit.
0 LATQ	**Input Control Latch or Queue Mode Enable** — The BUFEN control bit should be set in order to enable the IC and pulse accumulators holding registers. Otherwise LATQ latching modes are disabled. Write one into ICLAT bit in MCCTL, when LATQ and BUFEN are set will produce latching of input capture and pulse accumulators registers into their holding registers. 0 Queue mode of Input Capture is enabled. The main timer value is memorized in the IC register by a valid input pin transition. With a new occurrence of a capture, the value of the IC register will be transferred to its holding register and the IC register memorizes the new timer value. 1 Latch mode is enabled. Latching function occurs when modulus down-counter reaches zero or a zero is written into the count register MCCNT (see Section 26.4.1.1.2, “Buffered IC Channels”). With a latching event the contents of IC registers and 8-bit pulse accumulators are transferred to their holding registers. 8-bit pulse accumulators are cleared.

26.3.2.25 Output Compare Pin Disconnect Register (OCPD)

Module Base + 0x002C

	7	6	5	4	3	2	1	0
R W	OCPD7	OCPD6	OCPD5	OCPD4	OCPD3	OCPD2	OCPD1	OCPD0
Reset	0	0	0	0	0	0	0	0

Figure 26-48. Output Compare Pin Disconnect Register (OCPD)

Read: Anytime

Write: Anytime

All bits reset to zero.

Table 26-32. OCPD Field Descriptions

Field	Description
7:0 OCPD[7:0]	**Output Compare Pin Disconnect Bits** 0 Enables the timer channel IO port. Output Compare actions will occur on the channel pin. These bits do not affect the input capture or pulse accumulator functions. 1 Disables the timer channel IO port. Output Compare actions will not affect on the channel pin; the output compare flag will still be set on an Output Compare event.

26.3.2.26 Precision Timer Prescaler Select Register (PTPSR)

Module Base + 0x002E

	7	6	5	4	3	2	1	0
R W	PTPS7	PTPS6	PTPS5	PTPS4	PTPS3	PTPS2	PTPS1	PTPS0
Reset	0	0	0	0	0	0	0	0

Figure 26-49. Precision Timer Prescaler Select Register (PTPSR)

Read: Anytime

Write: Anytime

All bits reset to zero.

Table 26-33. PTPSR Field Descriptions

Field	Description
7:0 PTPS[7:0]	**Precision Timer Prescaler Select Bits** — These eight bits specify the division rate of the main Timer prescaler. These are effective only when the PRNT bit of TSCR1 is set to 1. Table 26-34 shows some selection examples in this case. The newly selected prescale factor will not take effect until the next synchronized edge where all prescale counter stages equal zero.

Table 26-34. Precision Timer Prescaler Selection Examples when PRNT = 1

PTPS7	PTPS6	PTPS5	PTPS4	PTPS3	PTPS2	PTPS1	PTPS0	Prescale Factor
0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	1	2
0	0	0	0	0	0	1	0	3
0	0	0	0	0	0	1	1	4
0	0	0	0	0	1	0	0	5
0	0	0	0	0	1	0	1	6
0	0	0	0	0	1	1	0	7
0	0	0	0	0	1	1	1	8
0	0	0	0	1	1	1	1	16
0	0	0	1	1	1	1	1	32
0	0	1	1	1	1	1	1	64
0	1	1	1	1	1	1	1	128
1	1	1	1	1	1	1	1	256

26.3.2.27 Precision Timer Modulus Counter Prescaler Select Register (PTMCPSR)

Module Base + 0x002F

	7	6	5	4	3	2	1	0
R W	PTMPS7	PTMPS6	PTMPS5	PTMPS4	PTMPS3	PTMPS2	PTMPS1	PTMPS0
Reset	0	0	0	0	0	0	0	0

Figure 26-50. Precision Timer Modulus Counter Prescaler Select Register (PTMCPSR)

Read: Anytime

Write: Anytime

All bits reset to zero.

Table 26-35. PTMCPSR Field Descriptions

Field	Description
7:0 PTMPS[7:0]	**Precision Timer Modulus Counter Prescaler Select Bits** — These eight bits specify the division rate of the modulus counter prescaler. These are effective only when the PRNT bit of TSCR1 is set to 1. Table 26-36 shows some possible division rates. The newly selected prescaler division rate will not be effective until a load of the load register into the modulus counter count register occurs.

Table 26-36. Precision Timer Modulus Counter Prescaler Select Examples when PRNT = 1

PTMPS7	PTMPS6	PTMPS5	PTMPS4	PTMPS3	PTMPS2	PTMPS1	PTMPS0	Prescaler Division Rate
0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	1	2
0	0	0	0	0	0	1	0	3
0	0	0	0	0	0	1	1	4
0	0	0	0	0	1	0	0	5
0	0	0	0	0	1	0	1	6
0	0	0	0	0	1	1	0	7
0	0	0	0	0	1	1	1	8
0	0	0	0	1	1	1	1	16
0	0	0	1	1	1	1	1	32
0	0	1	1	1	1	1	1	64
0	1	1	1	1	1	1	1	128
1	1	1	1	1	1	1	1	256

26.3.2.28 16-Bit Pulse Accumulator B Control Register (PBCTL)

Module Base + 0x0030

	7	6	5	4	3	2	1	0
R	0	PBEN	0	0	0	0	PBOVI	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 26-51. 16-Bit Pulse Accumulator B Control Register (PBCTL)

Read: Anytime

Write: Anytime

All bits reset to zero.

Table 26-37. PBCTL Field Descriptions

Field	Description
6 PBEN	**Pulse Accumulator B System Enable** — PBEN is independent from TEN. With timer disabled, the pulse accumulator can still function unless pulse accumulator is disabled. 0 16-bit pulse accumulator system disabled. 8-bit PAC1 and PAC0 can be enabled when their related enable bits in ICPAR are set. 1 Pulse accumulator B system enabled. The two 8-bit pulse accumulators PAC1 and PAC0 are cascaded to form the PACB 16-bit pulse accumulator B. When PACB is enabled, the PACN1 and PACN0 registers contents are respectively the high and low byte of the PACB. PA1EN and PA0EN control bits in ICPAR have no effect. The PACB shares the input pin with IC0.
1 PBOVI	**Pulse Accumulator B Overflow Interrupt Enable** 0 Interrupt inhibited 1 Interrupt requested if PBOVF is set

26.3.2.29 Pulse Accumulator B Flag Register (PBFLG)

Module Base + 0x0031

	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	PBOVF	0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 26-52. Pulse Accumulator B Flag Register (PBFLG)

Read: Anytime

Write used in the flag clearing mechanism. Writing a one to the flag clears the flag. Writing a zero will not affect the current status of the bit.

NOTE

When TFFCA = 1, the flag cannot be cleared via the normal flag clearing mechanism (writing a one to the flag). Reference Section 26.3.2.6, “Timer System Control Register 1 (TSCR1)”.

All bits reset to zero.

PBFLG indicates when interrupt conditions have occurred. The flag can be cleared via the normal flag clearing mechanism (writing a one to the flag) or via the fast flag clearing mechanism (Reference TFFCA bit in Section 26.3.2.6, “Timer System Control Register 1 (TSCR1)”).

Table 26-38. PBFLG Field Descriptions

Field	Description
1 PBOVF	**Pulse Accumulator B Overflow Flag** — This bit is set when the 16-bit pulse accumulator B overflows from 0xFFFF to 0x0000, or when 8-bit pulse accumulator 1 (PAC1) overflows from 0x00FF to 0x0000. When PACMX = 1, PBOVF bit can also be set if 8-bit pulse accumulator 1 (PAC1) reaches 0x00FF and an active edge follows on IC1.

26.3.2.30 8-Bit Pulse Accumulators Holding Registers (PA3H–PA0H)

Module Base + 0x0032

	7	6	5	4	3	2	1	0
R	PA3H7	PA3H6	PA3H5	PA3H4	PA3H3	PA3H2	PA3H1	PA3H0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 26-53. 8-Bit Pulse Accumulators Holding Register 3 (PA3H)

Module Base + 0x0033

	7	6	5	4	3	2	1	0
R	PA2H7	PA2H6	PA2H5	PA2H4	PA2H3	PA2H2	PA2H1	PA2H0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 26-54. 8-Bit Pulse Accumulators Holding Register 2 (PA2H)

Module Base + 0x0034

	7	6	5	4	3	2	1	0
R	PA1H7	PA1H6	PA1H5	PA1H4	PA1H3	PA1H2	PA1H1	PA1H0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 26-55. 8-Bit Pulse Accumulators Holding Register 1 (PA1H)

Module Base + 0x0035

	7	6	5	4	3	2	1	0
R	PA0H7	PA0H6	PA0H5	PA0H4	PA0H3	PA0H2	PA0H1	PA0H0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 26-56. 8-Bit Pulse Accumulators Holding Register 0 (PA0H)

Read: Anytime.

Write: Has no effect.

All bits reset to zero.

These registers are used to latch the value of the corresponding pulse accumulator when the related bits in register ICPAR are enabled (see Section 26.4.1.3, “Pulse Accumulators”).

26.3.2.31 Modulus Down-Counter Count Register (MCCNT)

Module Base + 0x0036

	15	14	13	12	11	10	9	8
R W	MCCNT15	MCCNT14	MCCNT13	MCCNT12	MCCNT11	MCCNT10	MCCNT9	MCCNT8
Reset	1	1	1	1	1	1	1	1

Figure 26-57. Modulus Down-Counter Count Register High (MCCNT)

Module Base + 0x0037

	7	6	5	4	3	2	1	0
R W	MCCNT7	MCCNT6	MCCNT5	MCCNT4	MCCNT3	MCCNT2	MCCNT1	MCCNT0
Reset	1	1	1	1	1	1	1	1

Figure 26-58. Modulus Down-Counter Count Register Low (MCCNT)

Read: Anytime

Write: Anytime.

All bits reset to one.

A full access for the counter register will take place in one clock cycle.

NOTE

A separate read/write for high byte and low byte will give different results than accessing them as a word.

If the RDMCL bit in MCCTL register is cleared, reads of the MCCNT register will return the present value of the count register. If the RDMCL bit is set, reads of the MCCNT will return the contents of the load register.

If a 0x0000 is written into MCCNT when LATQ and BUFEN in ICSYS register are set, the input capture and pulse accumulator registers will be latched.

With a 0x0000 write to the MCCNT, the modulus counter will stay at zero and does not set the MCZF flag in MCFLG register.

If the modulus down counter is enabled (MCEN = 1) and modulus mode is enabled (MODMC = 1), a write to MCCNT will update the load register with the value written to it. The count register will not be updated with the new value until the next counter underflow.

If modulus mode is not enabled (MODMC = 0), a write to MCCNT will clear the modulus prescaler and will immediately update the counter register with the value written to it and down-counts to 0x0000 and stops.

The FLMC bit in MCCTL can be used to immediately update the count register with the new value if an immediate load is desired.

26.3.2.32 Timer Input Capture Holding Registers 0–3 (TCxH)

Module Base + 0x0038

	15	14	13	12	11	10	9	8
R	TC15	TC14	TC13	TC12	TC11	TC10	TC9	TC8
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 26-59. Timer Input Capture Holding Register 0 High (TC0H)

Module Base + 0x0039

	7	6	5	4	3	2	1	0
R	TC7	TC6	TC5	TC4	TC3	TC2	TC1	TC0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 26-60. Timer Input Capture Holding Register 0 Low (TC0H)

Module Base + 0x003A

	15	14	13	12	11	10	9	8
R	TC15	TC14	TC13	TC12	TC11	TC10	TC9	TC8
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 26-61. Timer Input Capture Holding Register 1 High (TC1H)

Module Base + 0x003B

	7	6	5	4	3	2	1	0
R	TC7	TC6	TC5	TC4	TC3	TC2	TC1	TC0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 26-62. Timer Input Capture Holding Register 1 Low (TC1H)

Module Base + 0x003C

	15	14	13	12	11	10	9	8
R	TC15	TC14	TC13	TC12	TC11	TC10	TC9	TC8
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 26-63. Timer Input Capture Holding Register 2 High (TC2H)

Module Base + 0x003D

	7	6	5	4	3	2	1	0
R	TC7	TC6	TC5	TC4	TC3	TC2	TC1	TC0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 26-64. Timer Input Capture Holding Register 2 Low (TC2H)

Module Base + 0x003E

	15	14	13	12	11	10	9	8
R	TC15	TC14	TC13	TC12	TC11	TC10	TC9	TC8
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 26-65. Timer Input Capture Holding Register 3 High (TC3H)

Module Base + 0x003F

	7	6	5	4	3	2	1	0
R	TC7	TC6	TC5	TC4	TC3	TC2	TC1	TC0
W								
Reset	0	0	0	0	0	0	0	0

= Unimplemented or Reserved

Figure 26-66. Timer Input Capture Holding Register 3 Low (TC3H)

Read: Anytime

Write: Has no effect.

All bits reset to zero.

These registers are used to latch the value of the input capture registers TC0–TC3. The corresponding IOSx bits in TIOS should be cleared (see Section 26.4.1.1, “IC Channels”).

26.4 Functional Description

This section provides a complete functional description of the ECT block, detailing the operation of the design from the end user perspective in a number of subsections.

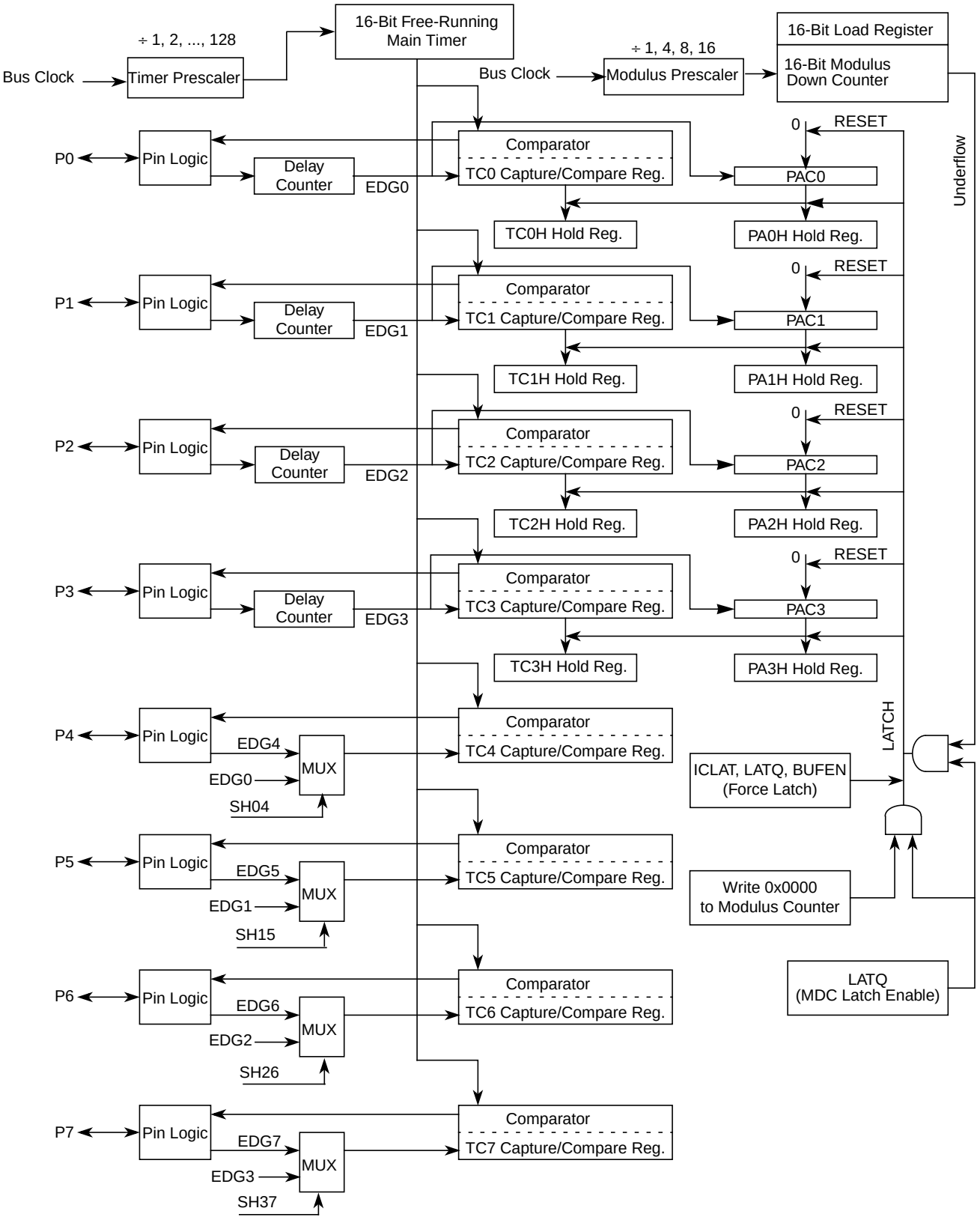


Figure 26-67. Detailed Timer Block Diagram in Latch Mode when PRNT = 0

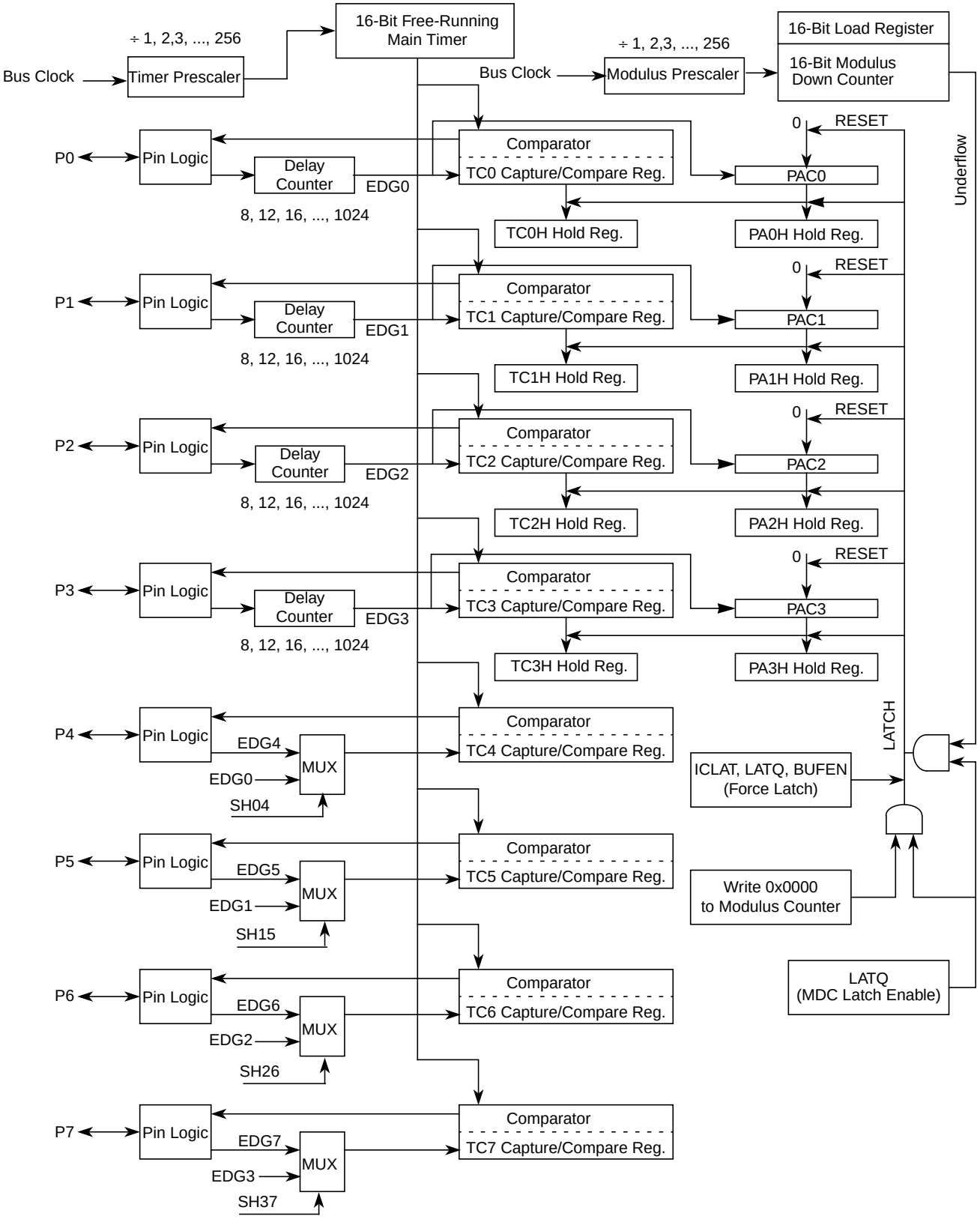


Figure 26-68. Detailed Timer Block Diagram in Latch Mode when PRNT = 1

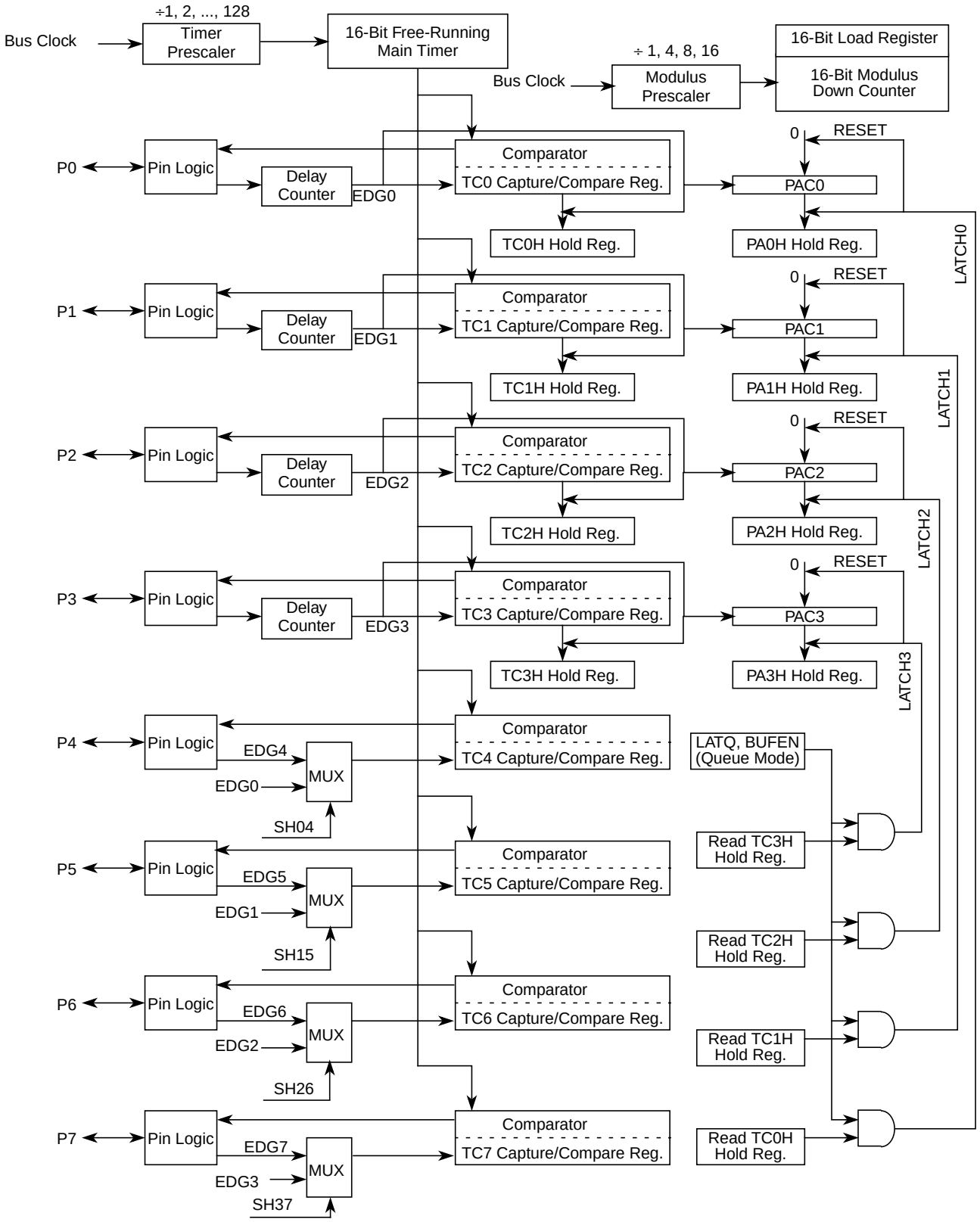


Figure 26-69. Detailed Timer Block Diagram in Queue Mode when PRNT = 0

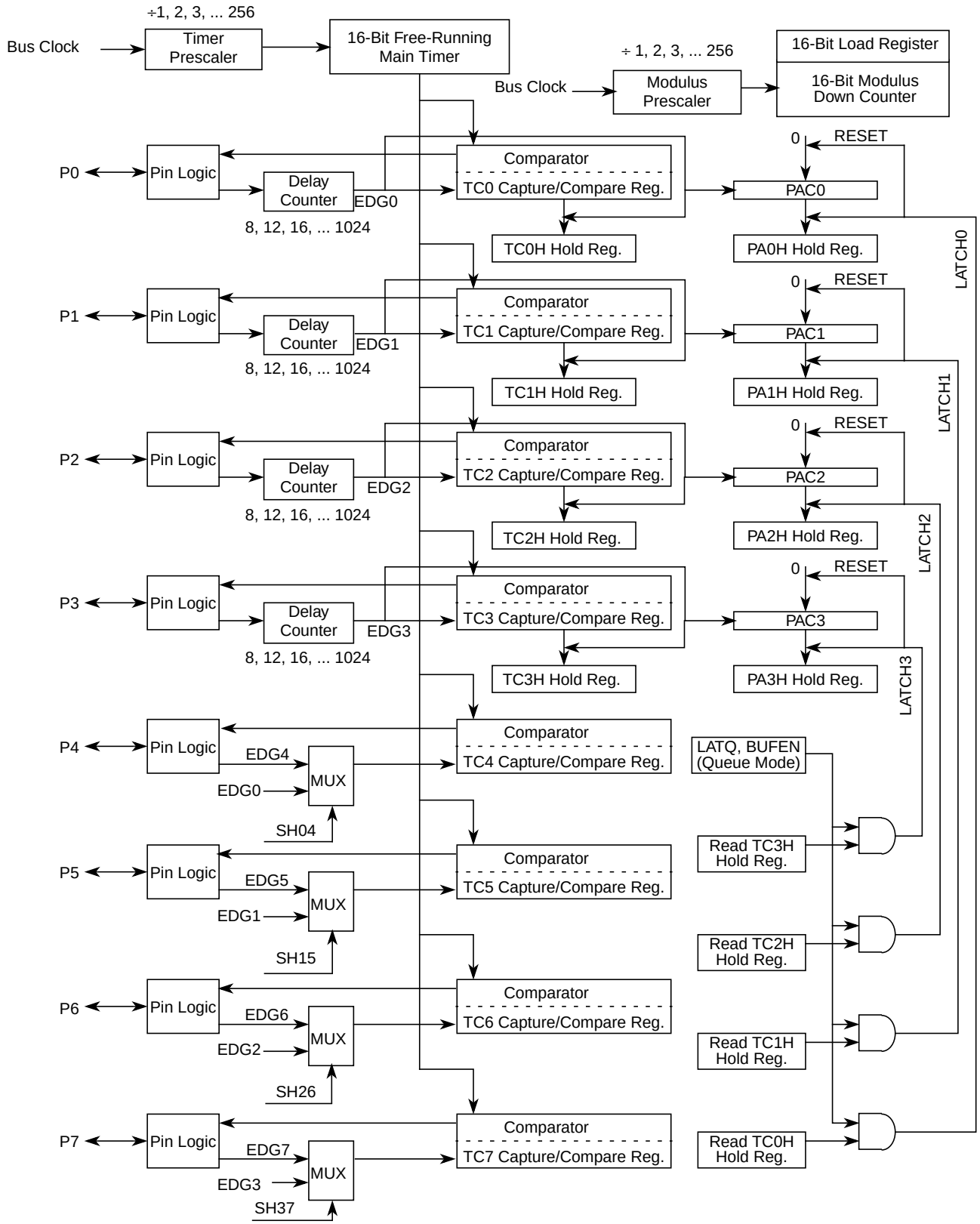


Figure 26-70. Detailed Timer Block Diagram in Queue Mode when PRNT = 1

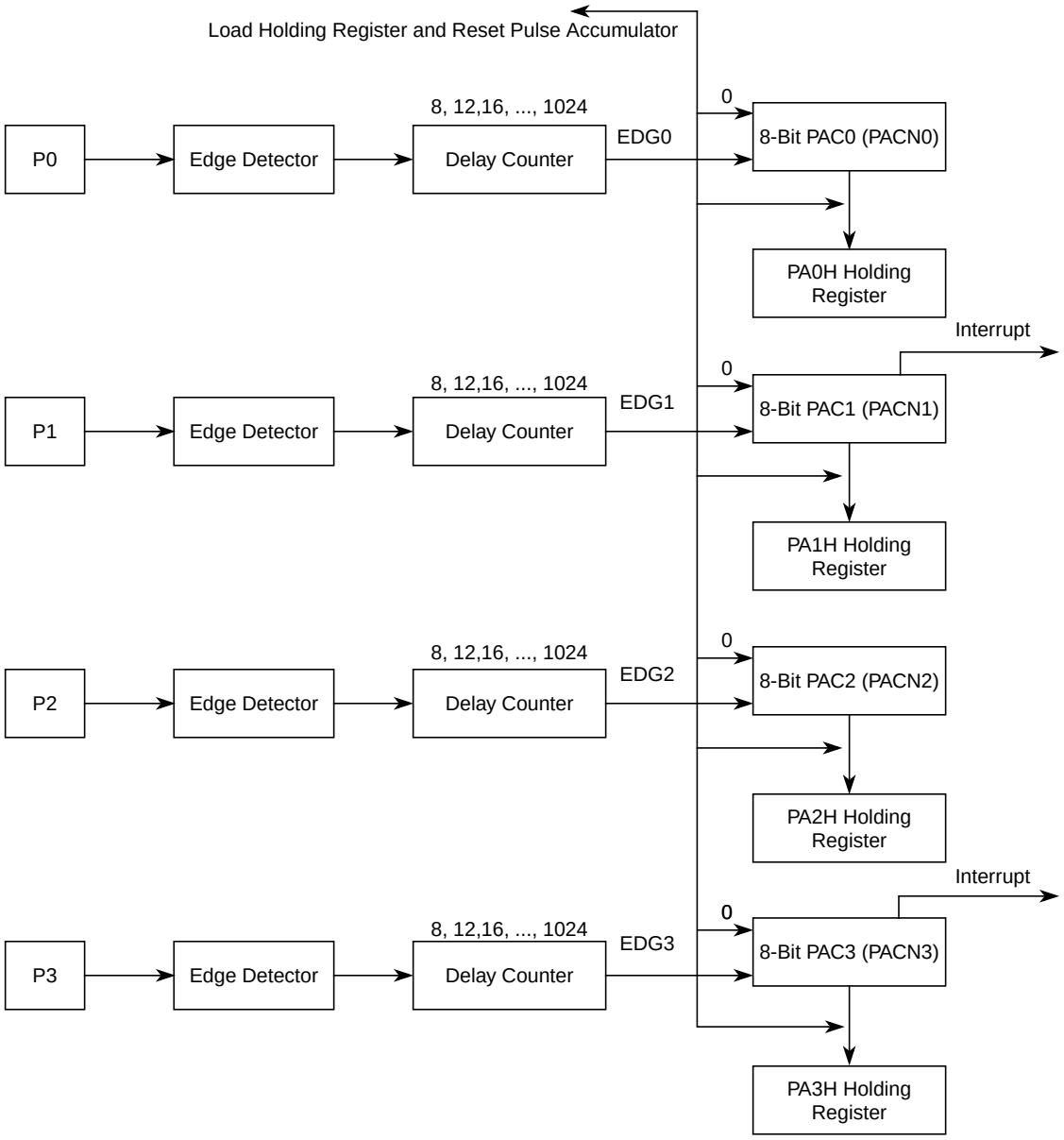


Figure 26-71. 8-Bit Pulse Accumulators Block Diagram

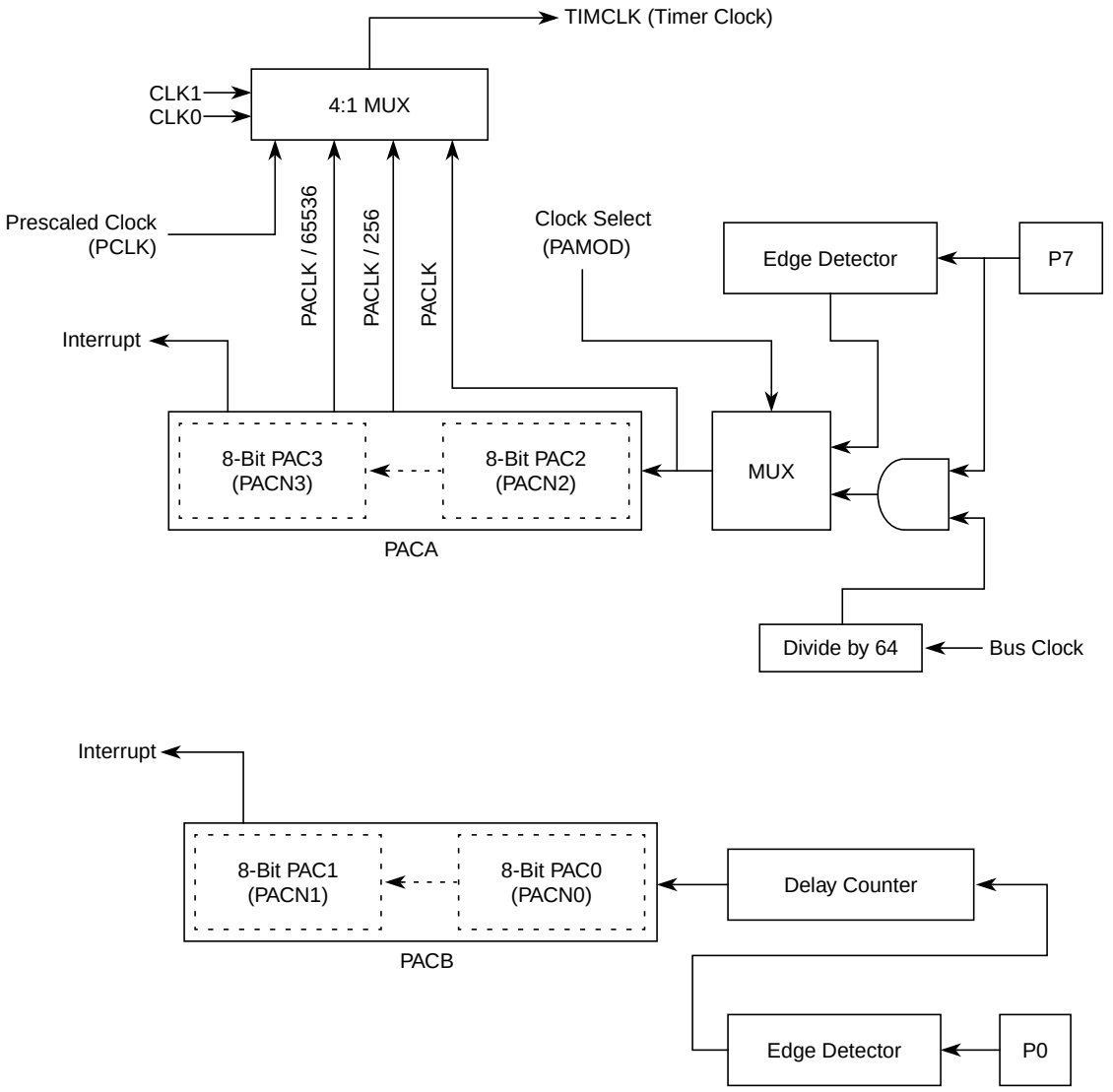


Figure 26-72. 16-Bit Pulse Accumulators Block Diagram

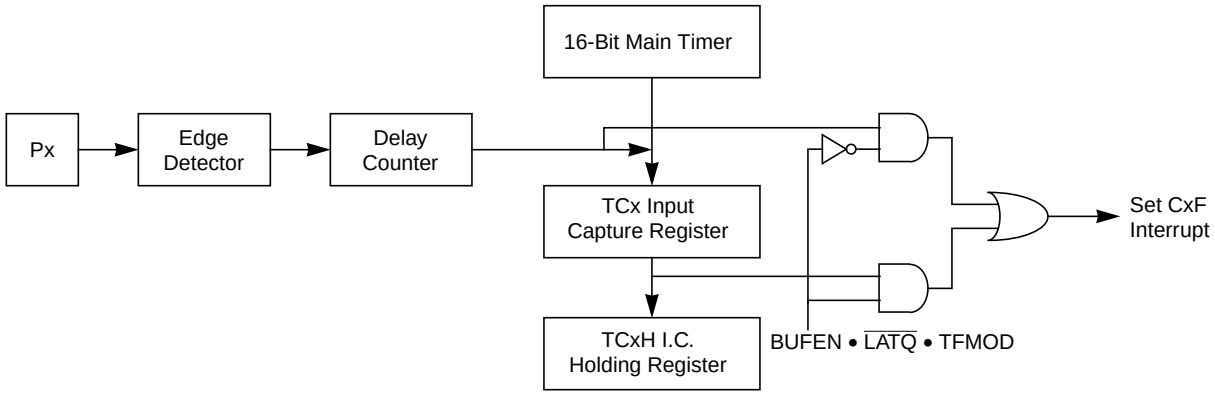


Figure 26-73. Block Diagram for Port 7 with Output Compare/Pulse Accumulator A

26.4.1 Enhanced Capture Timer Modes of Operation

The enhanced capture timer has 8 input capture, output compare (IC/OC) channels, same as on the HC12 standard timer (timer channels TC0 to TC7). When channels are selected as input capture by selecting the IOSx bit in TIOS register, they are called input capture (IC) channels.

Four IC channels (channels 7–4) are the same as on the standard timer with one capture register each that memorizes the timer value captured by an action on the associated input pin.

Four other IC channels (channels 3–0), in addition to the capture register, also have one buffer each called a holding register. This allows two different timer values to be saved without generating any interrupts.

Four 8-bit pulse accumulators are associated with the four buffered IC channels (channels 3–0). Each pulse accumulator has a holding register to memorize their value by an action on its external input. Each pair of pulse accumulators can be used as a 16-bit pulse accumulator.

The 16-bit modulus down-counter can control the transfer of the IC registers and the pulse accumulators contents to the respective holding registers for a given period, every time the count reaches zero.

The modulus down-counter can also be used as a stand-alone time base with periodic interrupt capability.

26.4.1.1 IC Channels

The IC channels are composed of four standard IC registers and four buffered IC channels.

- An IC register is empty when it has been read or latched into the holding register.
- A holding register is empty when it has been read.

26.4.1.1.1 Non-Buffered IC Channels

The main timer value is memorized in the IC register by a valid input pin transition. If the corresponding NOVWx bit of the ICOVW register is cleared, with a new occurrence of a capture, the contents of IC register are overwritten by the new value. If the corresponding NOVWx bit of the ICOVW register is set, the capture register cannot be written unless it is empty. This will prevent the captured value from being overwritten until it is read.

26.4.1.1.2 Buffered IC Channels

There are two modes of operations for the buffered IC channels:

1. IC latch mode (LATQ = 1)

   The main timer value is memorized in the IC register by a valid input pin transition (see Figure 26-67 and Figure 26-68).

   The value of the buffered IC register is latched to its holding register by the modulus counter for a given period when the count reaches zero, by a write 0x0000 to the modulus counter or by a write to ICLAT in the MCCTL register.

   If the corresponding NOVWx bit of the ICOVW register is cleared, with a new occurrence of a capture, the contents of IC register are overwritten by the new value. In case of latching, the contents of its holding register are overwritten.

If the corresponding NOVWx bit of the ICOVW register is set, the capture register or its holding register cannot be written by an event unless they are empty (see Section 26.4.1.1, “IC Channels”). This will prevent the captured value from being overwritten until it is read or latched in the holding register.

2. IC Queue Mode (LATQ = 0)

   The main timer value is memorized in the IC register by a valid input pin transition (see Figure 26-69 and Figure 26-70).

   If the corresponding NOVWx bit of the ICOVW register is cleared, with a new occurrence of a capture, the value of the IC register will be transferred to its holding register and the IC register memorizes the new timer value.

   If the corresponding NOVWx bit of the ICOVW register is set, the capture register or its holding register cannot be written by an event unless they are empty (see Section 26.4.1.1, “IC Channels”).

   if the TFMOD bit of the ICSYS register is set,the timer flags C3F--C0F in TFLG register are set only when a latch on the corresponding holding register occurs,after C3F--C0F are set,user should clear flag C3F--C0F,then read TCx and TCxH to make TCx and TCxH be empty.

   In queue mode, reads of the holding register will latch the corresponding pulse accumulator value to its holding register.

26.4.1.1.3 Delayed IC Channels

There are four delay counters in this module associated with IC channels 0–3. The use of this feature is explained in the diagram and notes below.

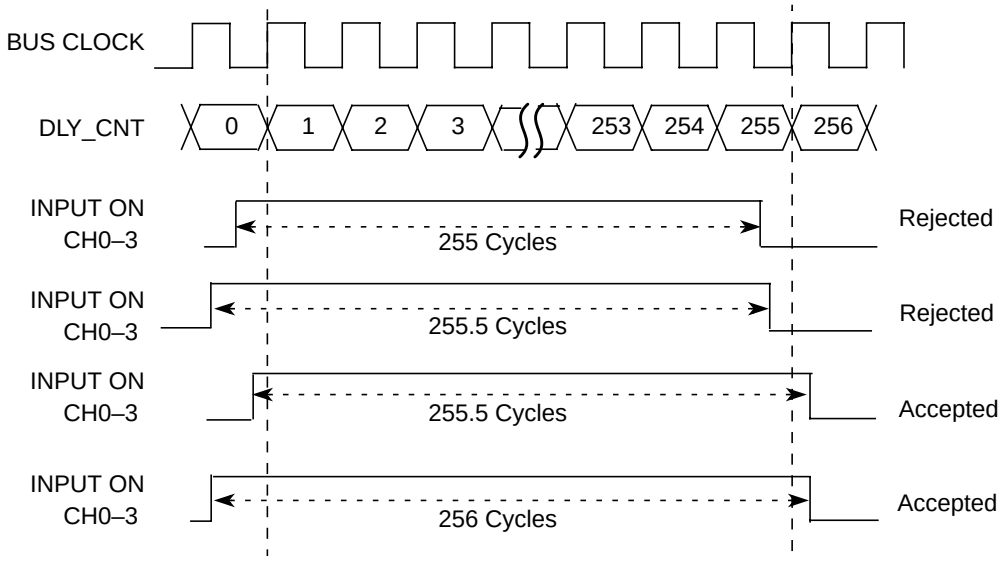


Figure 26-74. Channel Input Validity with Delay Counter Feature

In Figure 26-74 a delay counter value of 256 bus cycles is considered.

1. Input pulses with a duration of (DLY_CNT – 1) cycles or shorter are rejected.
2. Input pulses with a duration between (DLY_CNT – 1) and DLY_CNT cycles may be rejected or accepted, depending on their relative alignment with the sample points.

3. Input pulses with a duration between (DLY_CNT – 1) and DLY_CNT cycles may be rejected or accepted, depending on their relative alignment with the sample points.
4. Input pulses with a duration of DLY_CNT or longer are accepted.

26.4.1.2 OC Channel Initialization

An internal compare channel whose output drives OCx may be programmed before the timer drives the output compare state (OCx). The required output of the compare logic can be disconnected from the pin, leaving it driven by the GP IO port, by setting the appropriate OCPDx bit before enabling the output compare channel (by default the OCPD bits are cleared which would enable the output compare logic to drive the pin as soon as the timer output compare channel is enabled). The desired initial state can then be configured in the internal output compare logic by forcing a compare action with the logic disconnected from the IO (by writing a one to CFORCx bit with TIOSx, OCPDx and TEN bits set to one). Clearing the output compare disconnect bit (OCPDx) will then allow the internal compare logic to drive the programmed state to OCx. This allows a glitch free switching between general purpose I/O and timer output functionality.

26.4.1.3 Pulse Accumulators

There are four 8-bit pulse accumulators with four 8-bit holding registers associated with the four IC buffered channels 3–0. A pulse accumulator counts the number of active edges at the input of its channel.

The minimum pulse width for the PAI input is greater than two bus clocks.The maximum input frequency on the pulse accumulator channel is one half the bus frequency or Eclk.

The user can prevent the 8-bit pulse accumulators from counting further than 0x00FF by utilizing the PACMX control bit in the ICSYS register. In this case, a value of 0x00FF means that 255 counts or more have occurred.

Each pair of pulse accumulators can be used as a 16-bit pulse accumulator (see Figure 26-72).

Pulse accumulator B operates only as an event counter, it does not feature gated time accumulation mode. The edge control for pulse accumulator B as a 16-bit pulse accumulator is defined by TCTL4[1:0].

To operate the 16-bit pulse accumulators A and B (PACA and PACB) independently of input capture or output compare 7 and 0 respectively, the user must set the corresponding bits: IOSx = 1, OMx = 0, and OLx = 0. OC7M7 or OC7M0 in the OC7M register must also be cleared.

There are two modes of operation for the pulse accumulators:

- Pulse accumulator latch mode

  The value of the pulse accumulator is transferred to its holding register when the modulus down-counter reaches zero, a write 0x0000 to the modulus counter or when the force latch control bit ICLAT is written.

  At the same time the pulse accumulator is cleared.
- Pulse accumulator queue mode

  When queue mode is enabled, reads of an input capture holding register will transfer the contents of the associated pulse accumulator to its holding register.

At the same time the pulse accumulator is cleared.

26.4.1.4 Modulus Down-Counter

The modulus down-counter can be used as a time base to generate a periodic interrupt. It can also be used to latch the values of the IC registers and the pulse accumulators to their holding registers.

The action of latching can be programmed to be periodic or only once.

26.4.1.5 Precision Timer

By enabling the PRNT bit of the TSCR1 register, the performance of the timer can be enhanced. In this case, it is possible to set additional prescaler settings for the main timer counter and modulus down counter and enhance delay counter settings compared to the settings in the present ECT timer.

26.4.1.6 Flag Clearing Mechanisms

The flags in the ECT can be cleared one of two ways:

1. Normal flag clearing mechanism (TFFCA = 0)

   Any of the ECT flags can be cleared by writing a one to the flag.
2. Fast flag clearing mechanism (TFFCA = 1)

   With the timer fast flag clear all (TFFCA) enabled, the ECT flags can only be cleared by accessing the various registers associated with the ECT modes of operation as described below. The flags cannot be cleared via the normal flag clearing mechanism. This fast flag clearing mechanism has the advantage of eliminating the software overhead required by a separate clear sequence. Extra care must be taken to avoid accidental flag clearing due to unintended accesses.

   — Input capture

   A read from an input capture channel register causes the corresponding channel flag, CxF, to be cleared in the TFLG1 register.

   — Output compare

   A write to the output compare channel register causes the corresponding channel flag, CxF, to be cleared in the TFLG1 register.

   — Timer counter

   Any access to the TCNT register clears the TOF flag in the TFLG2 register.

   — Pulse accumulator A

   Any access to the PACN3 and PACN2 registers clears the PAOVF and PAIF flags in the PAFLG register.

   — Pulse accumulator B

   Any access to the PACN1 and PACN0 registers clears the PBOVF flag in the PBFLG register.

   — Modulus down counter

   Any access to the MCCNT register clears the MCZF flag in the MCFLG register.

26.4.2 Reset

The reset state of each individual bit is listed within the register description section (Section 26.3, “Memory Map and Register Definition”) which details the registers and their bit-fields.

26.4.3 Interrupts

This section describes interrupts originated by the ECT block. The MCU must service the interrupt requests. Table 26-39 lists the interrupts generated by the ECT to communicate with the MCU.

Table 26-39. ECT Interrupts

Interrupt Source	Description
Timer channel 7–0	Active high timer channel interrupts 7–0
Modulus counter underflow	Active high modulus counter interrupt
Pulse accumulator B overflow	Active high pulse accumulator B interrupt
Pulse accumulator A input	Active high pulse accumulator A input interrupt
Pulse accumulator A overflow	Pulse accumulator overflow interrupt
Timer overflow	Timer 0verflow interrupt

The ECT only originates interrupt requests. The following is a description of how the module makes a request and how the MCU should acknowledge that request. The interrupt vector offset and interrupt number are chip dependent.

26.4.3.1 Channel [7:0] Interrupt

This active high output will be asserted by the module to request a timer channel 7–0 interrupt to be serviced by the system controller.

26.4.3.2 Modulus Counter Interrupt

This active high output will be asserted by the module to request a modulus counter underflow interrupt to be serviced by the system controller.

26.4.3.3 Pulse Accumulator B Overflow Interrupt

This active high output will be asserted by the module to request a timer pulse accumulator B overflow interrupt to be serviced by the system controller.

26.4.3.4 Pulse Accumulator A Input Interrupt

This active high output will be asserted by the module to request a timer pulse accumulator A input interrupt to be serviced by the system controller.

26.4.3.5 Pulse Accumulator A Overflow Interrupt

This active high output will be asserted by the module to request a timer pulse accumulator A overflow interrupt to be serviced by the system controller.

26.4.3.6 Timer Overflow Interrupt

This active high output will be asserted by the module to request a timer overflow interrupt to be serviced by the system controller.

Appendix A
Electrical Characteristics

A.1 General

This supplement contains the most accurate electrical information for the MC9S12XF-Family microcontroller available at the time of publication.

This introduction is intended to give an overview on several common topics like power supply, current injection etc.

A.1.1 Parameter Classification

The electrical parameters shown in this supplement are guaranteed by various methods. To give the customer a better understanding the following classification is used and the parameters are tagged accordingly in the tables where appropriate.

NOTE

This classification is shown in the column labeled “C” in the parameter tables where appropriate.

P: Those parameters are guaranteed during production testing on each individual device.

C: Those parameters are achieved by the design characterization by measuring a statistically relevant sample size across process variations.

T: Those parameters are achieved by design characterization on a small sample size from typical devices under typical conditions unless otherwise noted. All values shown in the typical column are within this category.

D: Those parameters are derived mainly from simulations.

A.1.2 Power Supply

The MC9S12XF-Family utilizes several pins to supply power to the I/O ports, A/D converter, oscillator, and PLL as well as the digital core.

The VDDA, VSSA pin pairs supply the A/D converter and parts of the internal voltage regulator.

The VDDX, VSSX pin pairs [4:1] supply the I/O pins.

VDDR supplies the internal voltage regulator.

NOTE

Connecting VDDR to VSS disables the internal voltage regulator.

The VDDF, VSS1 pin pair supplies the internal NVM logic.

The VDD, VSS2 are the supply pins for the internal digital logic.

VDDPLL, VSSPLL pin pair supply the oscillator and the PLL.

VSS1, VSS2 and VSS3 are internally connected by metal.

All VDDX pins are internally connected by metal.

All VSSX pins are internally connected by metal.

VDDA is connected to all VDDX pins by diodes for ESD protection such that VDDX must not exceed VDDA by more than a diode voltage drop. VDDA can exceed VDDX by more than a diode drop in order to support applications with a 5V A/D converter range and 3.3V I/O pin range.
VSSA and VSSX are connected by anti-parallel diodes for ESD protection.

NOTE

In the following context V_{DD35} is used for either VDDA, VDDR, and VDDX; V_{SS35} is used for either VSSA and VSSX unless otherwise noted.

I_{DD35} denotes the sum of the currents flowing into the VDDA and VDDR pins.

V_{DD} is used for VDD, V_{SS} is used for VSS1, VSS2 and VSS3.

V_{DDPLL} is used for VDDPLL, V_{SSPLL} is used for VSSPLL

I_{DD} is used for the sum of the currents flowing into VDD, VDDF and VDDPLL.

A.1.3 Pins

There are four groups of functional pins.

A.1.3.1 I/O Pins

Those I/O pins have a level in the range of 3.13V to 5.5 V. This class of pins is comprised of all port I/O pins, the analog inputs, BKGD and the $\overline{\text{RESET}}$ pins.The internal structure of all those pins is identical; however, some of the functionality may be disabled. For example the BKGD pin pull up is always enabled.

A.1.3.2 Analog Reference

This group is made up by the V_{RH} and V_{RL} pins.

A.1.3.3 Oscillator

The pins EXTAL, XTAL dedicated to the oscillator have a nominal 1.8 V level. They are supplied by VDDPLL.

A.1.3.4 TEST

This pin is used for production testing only.

A.1.4 Current Injection

Power supply must maintain regulation within operating V_{DD35} or V_{DD} range during instantaneous and operating maximum current conditions. If positive injection current ($V_{in} > V_{DD35}$) is greater than I_{DD35}, the injection current may flow out of V_{DD35} and could result in external power supply going out of regulation. Ensure external V_{DD35} load will shunt current greater than maximum injection current. This will be the greatest risk when the MCU is not consuming power; e.g., if no system clock is present, or if clock rate is very low which would reduce overall power consumption.

A.1.5 Absolute Maximum Ratings

Absolute maximum ratings are stress ratings only. A functional operation under or outside those maxima is not guaranteed. Stress beyond those limits may affect the reliability or cause permanent damage of the device.

This device contains circuitry protecting against damage due to high static voltage or electrical fields; however, it is advised that normal precautions be taken to avoid application of any voltages higher than maximum-rated voltages to this high-impedance circuit. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level (e.g., either V_{SS35} or V_{DD35}).

Table A-1. Absolute Maximum Ratings(1)

Num	Rating	Symbol	Min	Max	Unit
1	I/O, regulator and analog supply voltage	V_{DD35}	–0.3	6.0	V
2	Digital logic supply voltage(2)	V_{DD}	–0.3	2.16	V
3	PLL supply voltage2	V_{DDPLL}	–0.3	2.16	V
4	NVM supply voltage2	V_{DDF}	–0.3	3.6	V
5	Voltage difference V_{DDX} to V_{DDA}	Δ_{VDDX}	–6.0	0.3	V
6	Voltage difference V_{SSX} to V_{SSA}	Δ_{VSSX}	–0.3	0.3	V
7	Digital I/O input voltage	V_{IN}	–0.3	6.0	V
8	Analog reference	V_{RH}, V_{RL}	–0.3	6.0	V
9	EXTAL, XTAL	V_{ILV}	–0.3	2.16	V
10	TEST input	V_{TEST}	–0.3	10.0	V
11	Instantaneous maximum current Single pin limit for all digital I/O pins(3)	I_D	–25	+25	mA
12	Instantaneous maximum current Single pin limit for EXTAL, XTAL(4)	I_{DL}	–25	+25	mA
13	Instantaneous maximum current Single pin limit for TEST (5)	I_{DT}	–0.25	0	mA
14	Maximum current Single pin limit for power supply pins	I_{DV}	–100	+100	mA
15	Storage temperature range	T_{stg}	–65	155	°C

1. Beyond absolute maximum ratings device might be damaged.
2. The device contains an internal voltage regulator to generate the logic and PLL supply out of the I/O supply. The absolute maximum ratings apply when the device is powered from an external source.
3. All digital I/O pins are internally clamped to V_{SSX} and V_{DDX}, or V_{SSA} and V_{DDA}.
4. Those pins are internally clamped to V_{SSPLL} and V_{DDPLL}.
5. This pin is clamped low to V_{SSPLL}, but not clamped high. This pin must be tied low in applications.

A.1.6 ESD Protection and Latch-up Immunity

All ESD testing is in conformity with CDF-AEC-Q100 stress test qualification for automotive grade integrated circuits. During the device qualification ESD stresses were performed for the Human Body Model (HBM) and the Charge Device Model.

A device will be defined as a failure if after exposure to ESD pulses the device no longer meets the device specification. Complete DC parametric and functional testing is performed per the applicable device specification at room temperature followed by hot temperature, unless specified otherwise in the device specification.

Table A-2. ESD and Latch-up Test Conditions

Model	Description	Symbol	Value	Unit
Human Body	Series resistance	R1	1500	Ohm
	Storage capacitance	C	100	pF
	Number of pulse per pin Positive Negative	 — —	 1 1	
Charged Device	Number of pulse per pin Positive Negative	 — —	 3 3	
Latch-up	Minimum input voltage limit		–2.5	V
	Maximum input voltage limit		7.5	V

Table A-3. ESD and Latch-Up Protection Characteristics

Num	C	Rating	Symbol	Min	Max	Unit
1	C	Human Body Model (HBM)	V_{HBM}	2000	—	V
2	C	Charge Device Model (CDM) corner pins Charge Device Model (CDM) edge pins	V_{CDM}	750 500	— —	V
3	C	Latch-up current at T_A = 125°C Positive Negative	I_{LAT}	 +100 –100	 — —	mA
4	C	Latch-up current at T_A = 27°C Positive Negative	I_{LAT}	 +200 –200	 — —	mA

A.1.7 Operating Conditions

This section describes the operating conditions of the device. Unless otherwise noted those conditions apply to all the following data.

NOTE

Please refer to the temperature rating of the device (C, V, M) with regards to the ambient temperature T_A and the junction temperature T_J. For power dissipation calculations refer to Section A.1.8, "Power Dissipation and Thermal Characteristics".

Table A-4. Operating Conditions

Rating	Symbol	Min	Typ	Max	Unit
I/O, regulator and analog supply voltage	V_{DD35}	3.13	5	5.5	V
NVM logic supply voltage(1)	V_{DDF}	2.7	2.8	2.9	V
Voltage difference V_{DDX} to V_{DDA}	Δ_{VDDX}	refer to Table A-12			
Voltage difference V_{DDR} to V_{DDX}	Δ_{VDDR}	–0.1	0	0.1	V
Voltage difference V_{SSX} to V_{SSA}	Δ_{VSSX}	refer to Table A-12			
Voltage difference V_{SS1} , V_{SS2} , V_{SS3} , V_{SSPLL} to V_{SSX}	Δ_{VSS}	–0.1	0	0.1	V
Digital logic supply voltage[1]	V_{DD}	1.72	1.8	1.98	V
PLL supply voltage	V_{DDPLL}	1.72	1.8	1.98	V
Oscillator(2) (Loop Controlled Pierce) (Full Swing Pierce)	f_{osc}	4 2	— —	16 40	MHz
Bus frequency(3)	f_{bus}	0.5	—	50	MHz
MC9S12XF512**C** Operating junction temperature range Operating ambient temperature range(4)	 T_J T_A	 –40 –40	 — 27	 110 85	°C
MC9S12XF512**V** Operating junction temperature range Operating ambient temperature range[2]	 T_J T_A	 –40 –40	 — 27	 130 105	°C
MC9S12XF512**M** Operating junction temperature range Operating ambient temperature range[2]	 T_J T_A	 –40 –40	 — 27	 150 125	°C

1. The device contains an internal voltage regulator to generate the logic, NVM and PLL supply out of the I/O supply.
2. This refers to the oscillator base frequency. Typical crystal & resonator tolerances are supported.
3. Please refer to Table A-22 for maximum bus frequency limits with frequency modulation enabled.
4. Please refer to Section A.1.8, "Power Dissipation and Thermal Characteristics" for more details about the relation between ambient temperature T_A and device junction temperature T_J.

NOTE

Using the internal voltage regulator, operation is guaranteed in a power down until a low voltage reset assertion.

A.1.8 Power Dissipation and Thermal Characteristics

Power dissipation and thermal characteristics are closely related. The user must assure that the maximum operating junction temperature is not exceeded. The average chip-junction temperature (T_J) in °C can be obtained from:

$$T_J = T_A + (P_D \bullet \Theta_{JA})$$

T_J = Junction Temperature, [°C]

T_A = Ambient Temperature, [°C]

P_D = Total Chip Power Dissipation, [W]

Θ_{JA} = Package Thermal Resistance, [°C/W]

The total power dissipation can be calculated from:

$$P_D = P_{INT} + P_{IO}$$

P_{INT} = Chip Internal Power Dissipation, [W]

$$P_{IO} = \sum_i R_{DSON} \cdot I_{IO_i}^2$$

P_{IO} is the sum of all output currents on I/O ports associated with V_{DDX}, whereby

$$R_{DSON} = \frac{V_{OL}}{I_{OL}}; \text{for outputs driven low}$$

$$R_{DSON} = \frac{V_{DD35} - V_{OH}}{I_{OH}}; \text{for outputs driven high}$$

Two cases with internal voltage regulator enabled and disabled must be considered:

1. Internal voltage regulator disabled

$$P_{INT} = I_{DD} \cdot V_{DD} + I_{DDPLL} \cdot V_{DDPLL} + I_{DDA} \cdot V_{DDA}$$

2. Internal voltage regulator enabled

$$P_{INT} = I_{DDR} \cdot V_{DDR} + I_{DDA} \cdot V_{DDA}$$

Table A-5. Thermal Package Characteristics (MC9S12XF512) (1)

Num	C	Rating	Symbol	Min	Typ	Max	Unit
LQFP144							
1	D	Thermal resistance LQFP144, single sided PCB[2]	θ_{JA}	—	—	48	°C/W
2	D	Thermal resistance LQFP144, double sided PCB with 2 internal planes[3]	θ_{JA}	—	—	40	°C/W
3	D	Junction to Board LQFP 144	θ_{JB}	—	—	28	°C/W
4	D	Junction to Case LQFP 144[4]	θ_{JC}	—	—	9	°C/W
5	D	Junction to Package Top LQFP144[5]	Ψ_{JT}	—	—	2	°C/W
LQFP112							
6	D	Thermal resistance LQFP112, single sided PCB(2)	θ_{JA}	—	—	49	°C/W
7	D	Thermal resistance LQFP112, double sided PCB with 2 internal planes(3)	θ_{JA}	—	—	40	°C/W
8	D	Junction to Board LQFP112	θ_{JB}	—	—	28	°C/W
9	D	Junction to Case LQFP112[4]	θ_{JC}	—	—	9	°C/W
10	D	Junction to Package Top LQFP112[5]	Ψ_{JT}	—	—	2	°C/W
LQFP64							
11	D	Thermal resistance LQFP 64, single sided PCB[2]	θ_{JA}	—	—	62	°C/W
12	D	Thermal resistance LQFP 64, double sided PCB with 2 internal planes[3]	θ_{JA}	—	—	44	°C/W
13	D	Junction to Board LQFP 64	θ_{JB}	—	—	27	°C/W
14	D	Junction to Case LQFP 64(4)	θ_{JC}	—	—	10	°C/W
15	D	Junction to Package Top LQFP 64(5)	Ψ_{JT}	—	—	2	°C/W

1. The values for thermal resistance are achieved by package simulations
2. Junction to ambient thermal resistance, θ_{JA} was simulated to be equivalent to the JEDEC specification JESD51-2 in a horizontal configuration in natural convection.
3. Junction to ambient thermal resistance, θ_{JA} was simulated to be equivalent to the JEDEC specification JESD51-7 in a horizontal configuration in natural convection.
4. Junction to case thermal resistance was simulated to be equivalent to the measured values using the cold plate technique with the cold plate temperature used as the “case” temperature. This basic cold plate measurement technique is described by MIL-STD 883D, Method 1012.1. This is the correct thermal metric to use to calculate thermal performance when the package is being used with a heat sink.
5. Thermal characterization parameter Ψ_{JT} is the “resistance” from junction to reference point thermocouple on top center of the case as defined in JESD51-2. Ψ_{JT} is a useful value to use to estimate junction temperature in a steady state customer enviroment.

A.1.9 I/O Characteristics

This section describes the characteristics of all I/O pins except EXTAL, XTAL, TEST and supply pins.

Table A-6. 3.3-V I/O Characteristics

Conditions are 3.13 V < V_{DD35} < 3.6 V temperature from –40°C to +150°C, unless otherwise noted I/O Characteristics for all I/O pins except EXTAL, XTAL,TEST and supply pins.							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	P	Input high voltage	V_{IH}	0.65*V_{DD35}	—	—	V
	T	Input high voltage	V_{IH}	—	—	V_{DD35} + 0.3	V
2	P	Input low voltage	V_{IL}	—	—	0.35*V_{DD35}	V
	T	Input low voltage	V_{IL}	V_{SS35} – 0.3	—	—	V
3	T	Input hysteresis	V_{HYS}	—	250	—	mV
4a	C	Input leakage current (pins in high impedance input mode)[(1)] V_{in} = V_{DD35} or V_{SS35} **M** Temperature range -40°C to 150°C **V** Temperature range -40°C to 130°C **C** Temperature range -40°C to 110°C	I_{in}	 –1 –0.75 –0.5	 — — —	 1 0.75 0.5	µA
4b	C	Input leakage current (pins in high impedance input mode) V_{in} = V_{DD35} or V_{SS35} -40°C 27°C 70°C 85°C 100°C 105°C 110°C 120°C 125°C 130°C 150°C	I_{in}	—	— — ±1 ±1 ±8 ±14 ±26 ±32 ±40 ±60 ±74 ±92 ±240	—	nA
5	C	Output high voltage (pins in output mode) Partial drive I_{OH} = –0.75 mA	V_{OH}	V_{DD35} – 0.4	—	—	V
6	P	Output high voltage (pins in output mode) Full drive I_{OH} = –4 mA	V_{OH}	V_{DD35} – 0.4	—	—	V
7	C	Output low voltage (pins in output mode) Partial Drive I_{OL} = +0.9 mA	V_{OL}	—	—	0.4	V
8	P	Output low voltage (pins in output mode) Full Drive I_{OL} = +4.75 mA	V_{OL}	—	—	0.4	V
9	P	Internal pull up resistance V_{IH} min > input voltage > V_{IL} max	R_{PUL}	25	—	50	KΩ
10	P	Internal pull down resistance V_{IH} min > input voltage > V_{IL} max	R_{PDH}	25	—	50	KΩ
11	D	Input capacitance	C_{in}	—	6	—	pF

Table A-6. 3.3-V I/O Characteristics

Conditions are 3.13 V < V_{DD35} < 3.6 V temperature from –40°C to +150°C, unless otherwise noted I/O Characteristics for all I/O pins except EXTAL, XTAL,TEST and supply pins.							
12	T	Injection current[2] Single pin limit Total device limit, sum of all injected currents	 I_{ICS} I_{ICP}	 –2.5 –25	—	 2.5 25	mA
13	D	Port H, J, P interrupt input pulse filtered (STOP)[3]	t_{PULSE}	—	—	3	µs
14	D	Port H, J, P interrupt input pulse passed(STOP)[3]	t_{PULSE}	10	—	—	µs
15	D	Port H, J, P interrupt input pulse filtered ($\overline{STOP}$)	t_{PULSE}	—	—	3	tcyc
16	D	Port H, J, P interrupt input pulse passed($\overline{STOP}$)	t_{PULSE}	4	—	—	tcyc
17	D	$\overline{IRQ}$ pulse width, edge-sensitive mode ($\overline{STOP}$)	PW_{IRQ}	1	—	—	tcyc
18	D	$\overline{XIRQ}$ pulse width with X-bit set (STOP)	PW_{XIRQ}	4	—	—	tosc

1. Maximum leakage current occurs at maximum operating temperature.
2. Refer to Section A.1.4, “Current Injection” for more details
3. Parameter only applies in stop or pseudo stop mode.

Table A-7. 5-V I/O Characteristics

Conditions are 4.5 V < V_{DD35} < 5.5 V temperature from –40°C to +150°C, unless otherwise noted
I/O Characteristics for all I/O pins except EXTAL, XTAL,TEST and supply pins.

Num	C	Rating	Symbol	Min	Typ	Max	Unit
1	P	Input high voltage	V_{IH}	0.65*V_{DD35}	—	—	V
	T	Input high voltage	V_{IH}	—	—	V_{DD35} + 0.3	V
2	P	Input low voltage	V_{IL}	—	—	0.35*V_{DD35}	V
	T	Input low voltage	V_{IL}	V_{SS35} – 0.3	—	—	V
3	T	Input hysteresis	V_{HYS}	—	250	—	mV
4a	P	Input leakage current (pins in high impedance input mode)[1] V_{in} = V_{DD35} or V_{SS35} **M** Temperature range -40°C to 150°C **V** Temperature range -40°C to 130°C **C** Temperature range -40°C to 110°C	I_{in}	 –1 –0.75 –0.5	 — — —	 1 0.75 0.5	µA
4b	C	Input leakage current (pins in high impedance input mode) V_{in} = V_{DD35} or V_{SS35} -40°C 27°C 70°C 85°C 100°C 105°C 110°C 120°C 125°C 130°C 150°C	I_{in}	—	— — ±1 ±1 ±8 ±14 ±26 ±32 40 ±60 ±74 ±92 ±240	—	nA
5	C	Output high voltage (pins in output mode) Partial drive I_{OH} = –2 mA	V_{OH}	V_{DD35} – 0.8	—	—	V
6	P	Output high voltage (pins in output mode) Full drive I_{OH} = –10 mA	V_{OH}	V_{DD35} – 0.8	—	—	V
7	C	Output low voltage (pins in output mode) Partial drive I_{OL} = +2 mA	V_{OL}	—	—	0.8	V
8	P	Output low voltage (pins in output mode) Full drive I_{OL} = +10 mA	V_{OL}	—	—	0.8	V
9	P	Internal pull up resistance V_{IH} min > input voltage > V_{IL} max	R_{PUL}	25	—	50	KΩ
10	P	Internal pull down resistance V_{IH} min > input voltage > V_{IL} max	R_{PDH}	25	—	50	KΩ
11	D	Input capacitance	C_{in}	—	6	—	pF
12	T	Injection current[2] Single pin limit Total device Limit, sum of all injected currents	 I_{ICS} I_{ICP}	 –2.5 –25	—	 2.5 25	mA
13	P	Port H, J, P interrupt input pulse filtered(STOP)[3]	t_{PULSE}	—	—	3	µs
14	P	Port H, J, P interrupt input pulse passed(STOP)[3]	t_{PULSE}	10	—	—	µs
15	D	Port H, J, P interrupt input pulse filtered ($\overline{STOP}$)	t_{PULSE}	—	—	3	tcyc

Table A-7. 5-V I/O Characteristics

Conditions are 4.5 V < V_{DD35} < 5.5 V temperature from –40°C to +150°C, unless otherwise noted I/O Characteristics for all I/O pins except EXTAL, XTAL,TEST and supply pins.							
16	D	Port H, J, P interrupt input pulse passed ($\overline{\text{STOP}}$)	t_{PULSE}	4	—	—	tcyc
17	D	$\overline{\text{IRQ}}$ pulse width, edge-sensitive mode ($\overline{\text{STOP}}$)	PW_{IRQ}	1	—	—	tcyc
18	D	$\overline{\text{XIRQ}}$ pulse width with X-bit set (STOP)	PW_{XIRQ}	4	—	—	tosc

1. Maximum leakage current occurs at maximum operating temperature.
2. Refer to Section A.1.4, “Current Injection” for more details
3. Parameter only applies in stop or pseudo stop mode.

Table A-8. Characteristics of Expantion Bus Inputs Port C, D, PE5, PE6, and PE7 for Reduced Input Voltage Thresholds

Conditions are 4.5 V < V_{DD35} < 5.5 V Temperature from –40°C to +150°C, unless otherwise noted							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	D	Input high voltage	V_{IH}	1.75	—	—	V
2	D	Input low voltage	V_{IL}	—	—	0.75	V
3	T	Input hysteresis	V_{HYS}	—	100	—	mV

A.1.10 Supply Currents

This section describes the current consumption characteristics of the device as well as the conditions for the measurements.

A.1.10.1 Typical Current Measurement Conditions

Since the current consumption of the output drivers is load dependent, all measurements are without output loads and with minimum I/O activity. Unless otherwise noted the currents are measured in single chip mode, S12X CPU code is executed from Flash and XGATE code is executed from RAM. V_{DD35}=5V, internal voltage regulator is enabled and the bus frequency is 50MHz using a 4-MHz oscillator in loop controlled Pierce mode.

Furthermore in expanded modes the currents flowing in the system are highly dependent on the load at the address, data, and control signals as well as on the duty cycle of those signals. No generally applicable numbers can be given. A very good estimate is to take the single chip currents and add the currents due to the external loads.

Since the DBG and BDM modules are typically not used in the end application, the supply current values for these modules is not specified.

An overhead of current consumption exisits independent of the listed modules, due to voltage regulation and clock logic that is not dedicated to a specific module. This is listed in a separate table row named “overhead”.

A.1.10.2 Maximum Current Measurement Conditions

Currents are measured in single chip mode, S12XCPU and XGATE code is executed from RAM with V_{DD35}=5.5V, internal voltage regulator enabled and a 50MHz bus frequency from a 4-MHz input. Characterized parameters are derived using a 4MHz loop controlled Pierce oscillator. Production test parameters are tested with a 4MHz square wave oscillator.

Table A-9. Module Configurations for Maximum Run Supply(VDDR+VDDA) Current V_{DD35}=5.5V

Peripheral	Configuration
S12XCPU	420 cycle loop: 384 DBNE cycles plus subroutine entry to stimulate stacking (RAM access)
XGATE	XGATE fetches code from RAM, XGATE runs in an infinite loop, reading the Status and Flag registers of CAN's, SPI's, SCI's in sequence and doing some bit manipulation on the data
MSCAN	Configured to loop-back mode using a bit rate of 1Mbit/s
SPI	Configured to master mode, continuously transmit data (0x55 or 0xAA) at 4Mbit/s
SCI	Configured into loop mode, continuously transmit data (0x55) at speed of 57600 baud
PMF	Configured to toggle its pins at the rate of 24MHz
ECT	The peripheral shall be configured in output compare mode. Pulse accumulator and modulus counter enabled.
ATD	The peripheral is configured to operate at its maximum specified frequency and to continuously convert voltages on all input channels in sequence.
RTI	Enabled with RTI Control Register (RTICTL) set to $FF
API	The module is configured to run from the RC oscillator clock source.
EPIT	PIT is enabled, Micro-timer register 0 and 1 loaded with $0F and timer registers 0 to 3 are loaded with $03/07/0F/1F.
FLEXRAY	Configure FlexRay and perform Startup procedure.
Overhead	VREG supplying 1.8V from a 5V input voltage, PLL on

Table A-10. Run and Wait Current Characteristics

Conditions are shown in Table A-4 unless otherwise noted							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
Run supply current (Peripheral Configuration see Table A-9.)							
1	P	Peripheral Set(1) f_{osc}=4MHz, f_{bus}=50MHz	I_{DD35}	—	—	110	mA
Wait supply current							
2	P	Peripheral Set1 ,PLL on XGATE executing code from RAM	I_{DDW}	—	—	95	mA
3	P	All modules disabled, RTI enabled, PLL off		—	—	10	

1. The following peripherals are on: ATD/ECT/PMF/SPI0-SPI1/SCI0-SCI1/CAN/XGATE/FLEXRAY

Table A-11. Pseudo Stop and Full Stop Current - Subject to Change

Conditions are shown in Table A-4 unless otherwise noted							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
Pseudo stop current (API, RTI, and COP disabled) PLL off							
10	C	–40°C	I_{DDPS}	—	45	—	µA
	P	27°C		—	55	255	
	C	70°C		—	100	—	
	C	85°C		—	145	—	
	P	110°C		—	295	2155	
	P	130°C		—	555	3655	
	P	150°C		—	1035	7000	
Pseudo stop current (API, RTI, and COP enabled) PLL off							
11	C	–40°C	I_{DDPS}	—	65	—	µA
	C	27°C		—	75	—	
	C	70°C		—	120	—	
	C	85°C		—	160	—	
	C	110°C		—	315	—	
	C	130°C		—	570	—	
	C	150°C		—	1045	—	
Stop Current							
12	C	–40°C	I_{DDS}	—	15	—	µA
	P	27°C		—	25	100	
	C	70°C		—	75	—	
	C	85°C		—	115	—	
	P	110°C		—	280	2000	
	P	130°C		—	550	3500	
	P	150°C		—	1050	7500	
Stop Current (API active)							
13	T	-40°C	I_{DDS}	—	20	—	µA
	T	27°C		—	30	—	
	T	85°C		—	120	—	
	T	110°C		—	280	—	
	T	130°C		—	245	—	
Stop Current (ATD active)							
14	T	27°C	I_{DDS}	—	290	—	µA
	T	85°C		—	400	—	
	T	110°C		—	565	—	
	T	130°C		—	835	—	

A.2 ATD Characteristics

This section describes the characteristics of the analog-to-digital converter.

A.2.1 ATD Operating Characteristics

The Table A-12 and Table A-13 show conditions under which the ATD operates.

The following constraints exist to obtain full-scale, full range results:

$$V_{SSA} \le V_{RL} \le V_{IN} \le V_{RH} \le V_{DDA}.$$

This constraint exists since the sample buffer amplifier can not drive beyond the power supply levels that it ties to. If the input level goes outside of this range it will effectively be clipped.

Table A-12. ATD Operating Characteristics

Conditions are shown in Table A-4 unless otherwise noted, supply voltage 3.13 V < V_{DDA} < 5.5 V							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	D	Reference potential Low High	 V_{RL} V_{RH}	 V_{SSA} $V_{DDA}/2$	 — —	 $V_{DDA}/2$ V_{DDA}	 V V
2	D	Voltage difference V_{DDX} to V_{DDA}	Δ_{VDDX}	–2.35	0	0.1	V
3	D	Voltage difference V_{SSX} to V_{SSA}	Δ_{VSSX}	–0.1	0	0.1	V
4	C	Differential reference voltage(1)	V_{RH}-V_{RL}	3.13	5.0	5.5	V
5	C	ATD Clock Frequency (derived from bus clock via the prescaler)	f_{ATDCLk}	0.25	—	8.3	MHz
6	P	ATD Clock Frequency in Stop mode (internal generated temperature and voltage dependent clock, ICLK)		0.6	1	1.7	MHz
7	D	ADC conversion in stop, recovery time(2)	$t_{ATDSTPRCV}$	—	—	1.5	us
8	D	ATD Conversion Period(3) 12 bit resolution: 10 bit resolution: 8 bit resolution:	 N_{CONV12} N_{CONV10} N_{CONV8}	 20 19 17	 — — —	 42 41 39	 ATD clock Cycles

1. Full accuracy is not guaranteed when differential voltage is less than 4.50 V
2. When converting in Stop Mode (ICLKSTP=1) an ATD Stop Recovery time tATDSTPRCV is required to switch back to bus clock based ATDCLK when leaving Stop Mode. Do not access ATD registers during this time.
3. The minimum time assumes a sample time of 4 ATD clock cycles. The maximum time assumes a sample time of 24 ATD clock cycles and the discharge feature (SMP_DIS) enabled, which adds 2 ATD clock cycles.

A.2.2 Factors Influencing Accuracy

Source resistance, source capacitance and current injection have an influence on the accuracy of the ATD. A further factor is that PortAD pins that are configured as output drivers switching.

A.2.2.1 Port AD Output Drivers Switching

PortAD output drivers switching can adversely affect the ATD accuracy whilst converting the analog voltage on other PortAD pins because the output drivers are supplied from the VDDA/VSSA ATD supply pins. Although internal design measures are implemented to minimize the affect of output driver noise, it is recommended to configure PortAD pins as outputs only for low frequency, low load outputs. The impact on ATD accuracy is load dependent and not specified. The values specified are valid under condition that no PortAD output drivers switch during conversion.

A.2.2.2 Source Resistance

Due to the input pin leakage current as specified in Table A-7 in conjunction with the source resistance there will be a voltage drop from the signal source to the ATD input. The maximum source resistance R_S specifies results in an error (10-bit resolution) of less than 1/2 LSB (2.5 mV) at the maximum leakage current. If device or operating conditions are less than worst case or leakage-induced error is acceptable, larger values of source resistance of up to 10Kohm are allowed.

A.2.2.3 Source Capacitance

When sampling an additional internal capacitor is switched to the input. This can cause a voltage drop due to charge sharing with the external and the pin capacitance. For a maximum sampling error of the input voltage ≤ 1LSB (10-bit resilution), then the external filter capacitor, $C_f \geq 1024 * (C_{INS}–C_{INN})$.

A.2.2.4 Current Injection

There are two cases to consider.

1. A current is injected into the channel being converted. The channel being stressed has conversion values of $3FF (in 10-bit mode) for analog inputs greater than V_{RH} and $000 for values less than V_{RL} unless the current is higher than specified as disruptive condition.
2. Current is injected into pins in the neighborhood of the channel being converted. A portion of this current is picked up by the channel (coupling ratio K), This additional current impacts the accuracy of the conversion depending on the source resistance.

The additional input voltage error on the converted channel can be calculated as:

$$V_{ERR} = K * R_S * I_{INJ}$$

with I_{INJ} being the sum of the currents injected into the two pins adjacent to the converted channel.

Table A-13. ATD Electrical Characteristics

Conditions are shown in Table A-4 unless otherwise noted							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	C	Max input source resistance	R_S	—	—	1	KΩ
2	T	Total input capacitance Non sampling Total input capacitance Sampling	C_{INN} C_{INS}	— —	— —	10 16	pF
3	T	Input internal Resistance	R_{INA}	—	5	15	kΩ
4	C	Disruptive analog input current	I_{NA}	–2.5	—	2.5	mA
5	C	Coupling ratio positive current injection	K_p	—	—	1E-4	A/A
6	C	Coupling ratio negative current injection	K_n	—	—	2E-3	A/A

A.2.3 ATD Accuracy

Table A-14 and Table A-15 specify the ATD conversion performance excluding any errors due to current injection, input capacitance and source resistance.

A.2.3.1 ATD Accuracy Definitions

For the following definitions see also Figure A-1.
Differential non-linearity (DNL) is defined as the difference between two adjacent switching steps.

$$DNL(i) = \frac{V_i - V_{i-1}}{1LSB} - 1$$

The integral non-linearity (INL) is defined as the sum of all DNLs:

$$INL(n) = \sum_{i=1}^{n} DNL(i) = \frac{V_n - V_0}{1LSB} - n$$

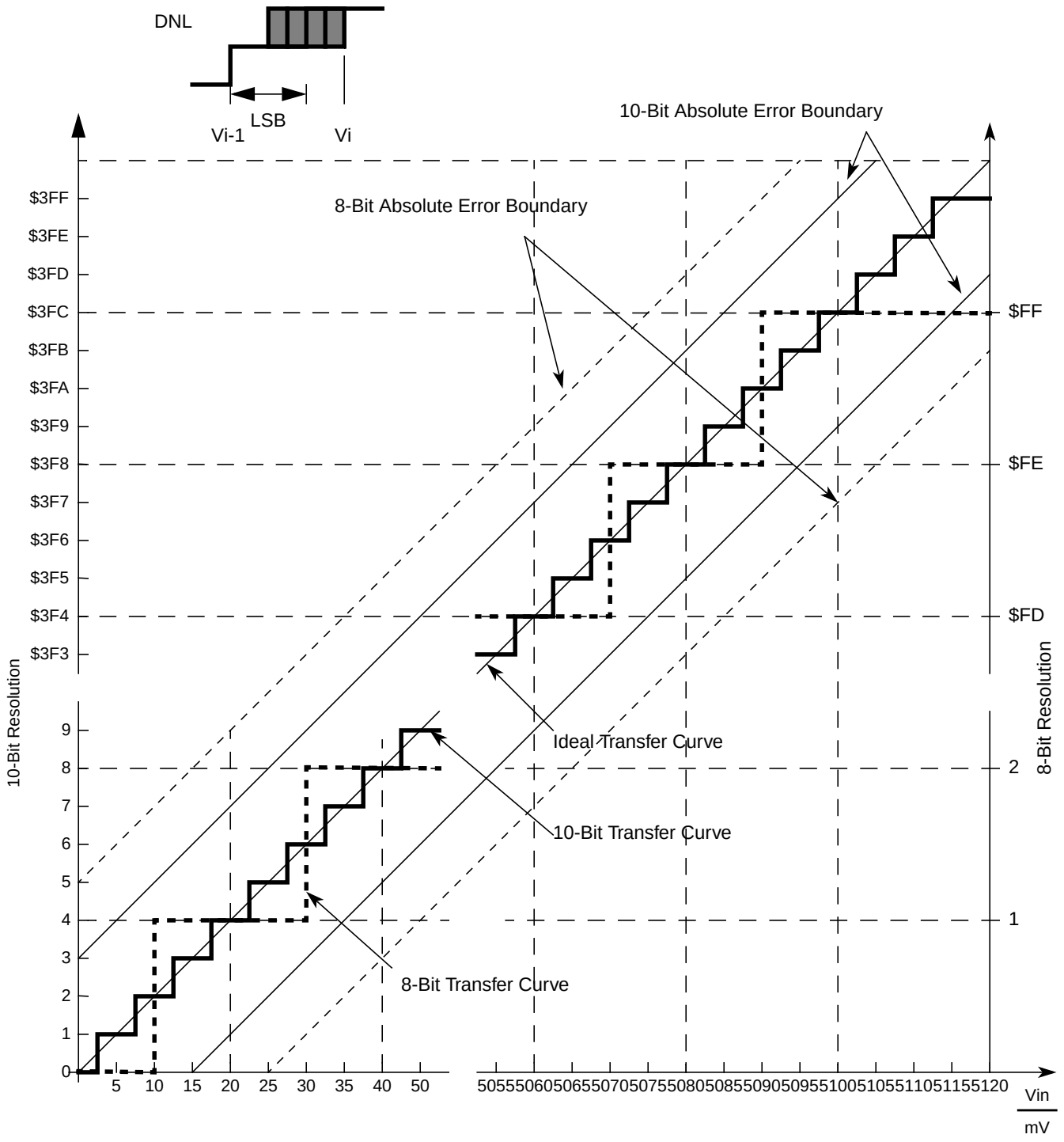


Figure A-1. ATD Accuracy Definitions

NOTE

Figure A-1 shows only definitions, for specification values refer to Table A-14 and Table A-15

Table A-14. ATD Conversion Performance 5V range

Conditions are shown in Table A-4 unless otherwise noted. $V_{REF} = V_{RH} - V_{RL} = 5.12V$. $f_{ATDCLK} = 8.3MHz$
The values are tested to be valid with no PortAD output drivers switching simultaneous with conversions.

Num	C	Rating[1],[2]		Symbol	Min	Typ	Max	Unit
1	P	Resolution	12-Bit	LSB	—	1.25	—	mV
2	P	Differential Nonlinearity	12-Bit	DNL	-4	±2	4	counts
3	P	Integral Nonlinearity	12-Bit	INL	-5	±2.5	5	counts
4	P	Absolute Error[3]	12-Bit	AE	-7	±4	7	counts
5	C	Resolution	10-Bit	LSB	—	5	—	mV
6	C	Differential Nonlinearity	10-Bit	DNL	-1	±0.5	1	counts
7	C	Integral Nonlinearity	10-Bit	INL	-2	±1	2	counts
8	C	Absolute Error[3]	10-Bit	AE	-3	±2	3	counts
9	C	Resolution	8-Bit	LSB	—	20	—	mV
10	C	Differential Nonlinearity	8-Bit	DNL	-0.5	±0.3	0.5	counts
11	C	Integral Nonlinearity	8-Bit	INL	-1	±0.5	1	counts
12	C	Absolute Error[3]	8-Bit	AE	-1.5	±1	1.5	counts

1. The 8-bit and 10-bit mode operation is structurally tested in production test. Absolute values are tested in 12-bit mode.
2. Better performance is possible using specially designed multi-layer PCBs or averaging techniques.
3. These values include the quantization error which is inherently 1/2 count for any A/D converter.

Table A-15. ATD Conversion Performance 3.3V range

Conditions are shown in Table A-4 unless otherwise noted. $V_{REF} = V_{RH} - V_{RL} = 3.3V$. $f_{ATDCLK} = 8.3MHz$
The values are tested to be valid with no PortAD output drivers switching simultaneous with conversions.

Num	C	Rating[1],[2]		Symbol	Min	Typ	Max	Unit
1	P	Resolution	12-Bit	LSB	—	0.80	—	mV
2	P	Differential Nonlinearity	12-Bit	DNL	-6	±3	6	counts
3	P	Integral Nonlinearity	12-Bit	INL	-7	±3	7	counts
4	P	Absolute Error[3]	12-Bit	AE	-8	±4	8	counts
5	C	Resolution	10-Bit	LSB	—	3.22	—	mV
6	C	Differential Nonlinearity	10-Bit	DNL	-1.5	±1	1.5	counts
7	C	Integral Nonlinearity	10-Bit	INL	-2	±1	2	counts
8	C	Absolute Error[3]	10-Bit	AE	-3	±2	3	counts
9	C	Resolution	8-Bit	LSB	—	12.89	—	mV
10	C	Differential Nonlinearity	8-Bit	DNL	-0.5	±0.3	0.5	counts
11	C	Integral Nonlinearity	8-Bit	INL	-1	±0.5	1	counts
12	C	Absolute Error[3]	8-Bit	AE	-1.5	±1	1.5	counts

1. The 8-bit and 10-bit mode operation is structurally tested in production test. Absolute values are tested in 12-bit mode.
2. Better performance is possible using specially designed multi-layer PCBs or averaging techniques.
3. These values include the quantization error which is inherently 1/2 count for any A/D converter.

A.3 NVM, Flash and Emulated EEPROM

A.3.1 Timing Parameters

The time base for all NVM program or erase operations is derived from the oscillator. A minimum oscillator frequency f_{NVMOSC} is required for performing program or erase operations. The NVM modules do not have any means to monitor the frequency and will not prevent program or erase operation at frequencies above or below the specified minimum. When attempting to program or erase the NVM modules at a lower frequency, a full program or erase transition is not assured.

The program and erase operations are timed using a clock derived from the oscillator using the FCLKDIV register. The frequency of this clock must be set within the limits specified as f_{NVMOP}.

The minimum program and erase times shown in Table A-16 are calculated for maximum f_{NVMOP} and maximum f_{NVMBUS} unless otherwise shown. The maximum times are calculated for minimum f_{NVMOP}

A.3.1.1 Erase Verify All Blocks (Blank Check) (FCMD=0x01)

The time it takes to perform a blank check is dependant on the location of the first non-blank word starting at relative address zero. It takes one bus cycle per phrase to verify plus a setup of the command. Assuming that no non blank location is found, then the erase verify all blocks is given by.

$$t_{check} = 33500 \cdot \frac{1}{f_{NVMBUS}}$$

A.3.1.2 Erase Verify Block (Blank Check) (FCMD=0x02)

The time it takes to perform a blank check is dependant on the location of the first non-blank word starting at relative address zero. It takes one bus cycle per phrase to verify plus a setup of the command. Assuming that no non blank location is found, then the erase verify time for a single 256K NVM array is given by

$$t_{check} = 33500 \cdot \frac{1}{f_{NVMBUS}}$$

For a 128K NVM or D-Flash array the erase verify time is given by

$$t_{check} = 17200 \cdot \frac{1}{f_{NVMBUS}}$$

A.3.1.3 Erase Verify P-Flash Section (FCMD=0x03)

The maximum time depends on the number of phrases being verified (N_{VP})

$$t_{check} = (752 + N_{VP}) \cdot \frac{1}{f_{NVMBUS}}$$

A.3.1.4 Read Once (FCMD=0x04)

The maximum read once time is given by

$$t = (400) \cdot \frac{1}{f_{NVMBUS}}$$

A.3.1.5 Load Data Field (FCMD=0x05)

The maximum load data field time is given by

$$t = (450) \cdot \frac{1}{f_{NVMBUS}}$$

A.3.1.6 Program P-Flash (FCMD=0x06)

The programming time for a single phrase of four P-Flash words + associated eight ECC bits is dependant on the bus frequency as a well as on the frequency f_{NVMOP} and can be calculated according to the following formulas, whereby N_{DLOAD} is the number of extra blocks being programmed by the Load Data Field command (DLOAD), i.e. programming 2 blocks using DLOAD, N_{DLOAD} =1 respectively.

The typical phrase programming time can be calculated using the following equation

$$t_{bwpgm} = (128 + (12 \cdot N_{DLOAD})) \cdot \frac{1}{f_{NVMOP}} + (1725 + (510 \cdot N_{DLOAD})) \cdot \frac{1}{f_{NVMBUS}}$$

The maximum phrase programming time can be calculated using the following equation

$$t_{bwpgm} = (130 + (14 \cdot N_{DLOAD})) \cdot \frac{1}{f_{NVMOP}} + (2125 + (510 \cdot N_{DLOAD})) \cdot \frac{1}{f_{NVMBUS}}$$

A.3.1.7 P-Flash Program Once (FCMD=0x07)

The maximum P-Flash Program Once time is given by

$$t_{bwpgm} \approx 162 \cdot \frac{1}{f_{NVMOP}} + 2400 \cdot \frac{1}{f_{NVMBUS}}$$

A.3.1.8 Erase All Blocks (FCMD=0x08)

For S12XF512, S12XF384, S12XF256 and S12XF128 erasing all blocks takes:

$$t_{mass} \approx 100100 \cdot \frac{1}{f_{NVMOP}} + 70000 \cdot \frac{1}{f_{NVMBUS}}$$

A.3.1.9 Erase P-Flash Block (FCMD=0x09)

Erasing a 256K NVM block takes

$$t_{mass} \approx 100100 \cdot \frac{1}{f_{NVMOP}} + 70000 \cdot \frac{1}{f_{NVMBUS}}$$

Erasing a 128K NVM block takes

$$t_{mass} \approx 100100 \cdot \frac{1}{f_{NVMOP}} + 35000 \cdot \frac{1}{f_{NVMBUS}}$$

A.3.1.10 Erase P-Flash Sector (FCMD=0x0A)

The typical time to erase a1024-byte P-Flash sector can be calculated using

$$t_{era} = \left(20020 \cdot \frac{1}{f_{NVMOP}}\right) + \left(700 \cdot \frac{1}{f_{NVMBUS}}\right)$$

The maximum time to erase a1024-byte P-Flash sector can be calculated using

$$t_{era} = \left(20020 \cdot \frac{1}{f_{NVMOP}}\right) + \left(1100 \cdot \frac{1}{f_{NVMBUS}}\right)$$

A.3.1.11 Unsecure Flash (FCMD=0x0B)

The maximum time for unsecuring the flash is given by

$$t_{uns} = \left(100100 \cdot \frac{1}{f_{NVMOP}} + 70000 \cdot \frac{1}{f_{NVMBUS}}\right)$$

A.3.1.12 Verify Backdoor Access Key (FCMD=0x0C)

The maximum verify backdoor access key time is given by

$$t = 400 \cdot \frac{1}{f_{NVMBUS}}$$

A.3.1.13 Set User Margin Level (FCMD=0x0D)

The maximum set user margin level time is given by

$$t = 350 \cdot \frac{1}{f_{NVMBUS}}$$

A.3.1.14 Set Field Margin Level (FCMD=0x0E)

The maximum set field margin level time is given by

$$t = 350 \cdot \frac{1}{f_{NVMBUS}}$$

A.3.1.15 Full Partition D-Flash (FCMD=0x0F)

The maximum time for partitioning the D-flash (ERPART=16, DFPART=0) is given by :

$$t_{part} \approx 21800 \cdot \frac{1}{f_{NVMOP}} + 400000 \cdot \frac{1}{f_{NVMBUS}} + t_{mass}$$

A.3.1.16 Erase Verify D-Flash Section (FCMD=0x10)

Erase Verify D-Flash for a given number of words N_W is given by .

$$t_{check} \approx (840 + N_W) \cdot \frac{1}{f_{NVMBUS}}$$

A.3.1.17 D-Flash Programming (FCMD=0x11)

D-Flash programming time is dependent on the number of words being programmed and their location with respect to a row boundary, because programming across a row boundary requires extra steps. The D-Flash programming time is specified for different cases (1,2,3,4 words and 4 words across a row boundary) at a 50MHz bus frequency. The typical programming time can be calculated using the following equation, whereby N_w denotes the number of words; BC=0 if no boundary is crossed and BC=1 if a boundary is crossed.

$$t_{dpgm} = \left((15 + (54 \cdot N_w) + (16 \cdot BC)) \cdot \frac{1}{f_{NVMOP}}\right) + \left((460 + (640 \cdot N_W) + (500 \cdot BC)) \cdot \frac{1}{f_{NVMBUS}}\right)$$

The maximum programming time can be calculated using the following equation

$$t_{dpgm} = \left((15 + (56 \cdot N_w) + (16 \cdot BC)) \cdot \frac{1}{f_{NVMOP}}\right) + \left((460 + (840 \cdot N_W) + (500 \cdot BC)) \cdot \frac{1}{f_{NVMBUS}}\right)$$

A.3.1.18 Erase D-Flash Sector (FCMD=0x12)

Typical D-Flash sector erase times are those expected on a new device, where no margin verify fails occur. They can be calculated using the following equation.

$$t_{eradf} \approx 5025 \cdot \frac{1}{f_{NVMOP}} + 700 \cdot \frac{1}{f_{NVMBUS}}$$

Maximum D-Fash sector erase times can be calculated using the following equation.

$$t_{eradf} \approx 20100 \cdot \frac{1}{f_{NVMOP}} + 3300 \cdot \frac{1}{f_{NVMBUS}}$$

The D-Flash sector erase time on a new device is ~5ms and can extend to 20ms as the flash is cycled.

A.3.1.19 Enable EEE (FCMD=0x13)

Maximum time to enable EPROM emulation is given by

$$t = \left(\left(((1100 \cdot BWN + (176 \cdot (1 + BWN) + (BWN + N_{SEC}) \cdot 32364))) \cdot \frac{1}{f_{NVMOP}}\right) + \left((3050 \cdot (1 + BWN) + (N_{SEC} + BWN) \cdot 290500) \cdot \frac{1}{f_{NVMBUS}}\right)\right)$$

where N_{SEC} is the number of sectors of constant data. A constant sector is one in which all 63 records contain the latest active data and would need to be copied. The maximum possible is 33 (2048 EEE RAM words /63 =32.5) although this is a highly unlikely scenario. The impact of a worst case brownout recovery scenario is denoted by BWN = 2 for non brownout situations BWN =0.

A.3.1.20 Maximum CCOB Latency

The maximum time a CCOB command has to wait to be actioned due to an EEE clean up is given where BWN = 1 if a brownout has occured otherwise BWN = 0. BWN = 1 only for the first ENEEE after reset.

$$t \approx \left(32364 \cdot \frac{1}{f_{NVMOP}} + 292600 \cdot \frac{1}{f_{NVMBUS}}\right) \cdot (1 + BWN) + BWN \cdot \left(350 \cdot \frac{1}{f_{NVMOP}} + \frac{1100}{f_{NVMBUS}}\right)$$

A.3.1.21 Disable EEE (FCMD=0x14)

Maximum time to disable EPROM emulation is given by

$$t = 300 \cdot \frac{1}{f_{NVMBUS}}$$

A.3.1.22 EEE Query (FCMD=0x15)

Maximum time for the EEE query command is given by

$$t = 300 \cdot \frac{1}{f_{NVMBUS}}$$

A.3.1.23 Partition D-Flash (FCMD=0x20)

The maximum time for partitioning the D-flash (ERPART=16, DFPART=0) is given by

$$t \approx 21800 \cdot \frac{1}{f_{NVMOP}} + 400000 \cdot \frac{1}{f_{NVMBUS}}$$

A.3.1.24 EEE Copy Down

The typical EEE copy down time is given by the following equation

$$t_{dfcd} = (14000 + (316 \cdot ERPART) + (1500 \cdot (124 - DFPART))) \times \frac{1}{f_{NVMBUS}}$$

The maximum EEE copy down time is given by the following equation

$$t_{dfcd} = (34000 + (316 \cdot ERPART) + (1500 \cdot (124 - DFPART))) \times \frac{1}{f_{NVMBUS}}$$

The worst case for Enable EEPROM Emulation allows for all the EEE records to be copied. This low probability scenario can only occur when the EEE is mostly full of unchanging data (the records for which are stored in consecutive D-Flash sectors).

Table A-16. NVM Timing Characteristics

Conditions are as shown in Table A-4, with 50MHz bus and f_{NVMOP}= 1MHz unless otherwise noted.							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	D	External oscillator clock	f_{NVMOSC}	2	—	50(1)	MHz
2	D	Bus frequency for programming or erase operations	f_{NVMBUS}	1	—	50	MHz
3	D	Operating frequency	f_{NVMOP}	800	—	1050	kHz
4	D	P-Flash phrase programming	t_{bwpgm}	—	162	173	µs
5	D	P-Flash phrase program time using D-LOAD on 2 blocks	t_{bwpgm2}	—	185	202	µs
6	P	P-Flash sector erase time	t_{era}	—	20	21	ms
7	P	Erase All Blocks (Mass erase) time	t_{mass}	—	101	102	ms
7a	D	Unsecure Flash	t_{uns}	—	101	102	ms
8	D	P-Flash erase verify (blank check) time(2)	t_{check}	—	—	33500[2]	t_{cyc}
9a	D	D-Flash word programming one word	t_{dpgm}	—	88	95	µs
9b	D	D-Flash word programming two words	t_{dpgm}	—	153	165	µs
9c	D	D-Flash word programming three words	t_{dpgm}	—	212	230	µs
9d	D	D-Flash word programming four words	t_{dpgm}	—	282	316	µs
9e	D	D-Flash word programming four words crossing row boundary	t_{dpgm}	—	298	342	µs
10	D	D-Flash sector erase time	t_{eradf}	—	5.2(3)	21	ms
11	D	D-Flash erase verify (blank check) time	t_{check}	—	—	17500	t_{cyc}
12	D	EEE copy down (2M64J Maskset)	t_{dfrcd}	—	255000	275000(4)	t_{cyc}
12	D	EEE copy down (Other Masksets)	t_{dfrcd}	—	205000	225000(5)	t_{cyc}

1. Restrictions for oscillator in crystal mode apply.
2. Valid for both “Erase verify all” or “Erase verify block” on 256K block without failing locations
3. This is a typical value for a new device
4. Maximum partitioning
5. Maximum partitioning

A.3.2 NVM Reliability Parameters

The reliability of the NVM blocks is guaranteed by stress test during qualification, constant process monitors and burn-in to screen early life failures.

The data retention and program/erase cycling failure rates are specified at the operating conditions noted. The program/erase cycle count on the sector is incremented every time a sector or mass erase event is executed.

The standard shipping condition for both the D-Flash and P-Flash memory is erased with security disabled. However it is recommended that each block or sector is erased before factory programming to ensure that the full data retention capability is achieved. Data retention time is measured from the last erase operation.

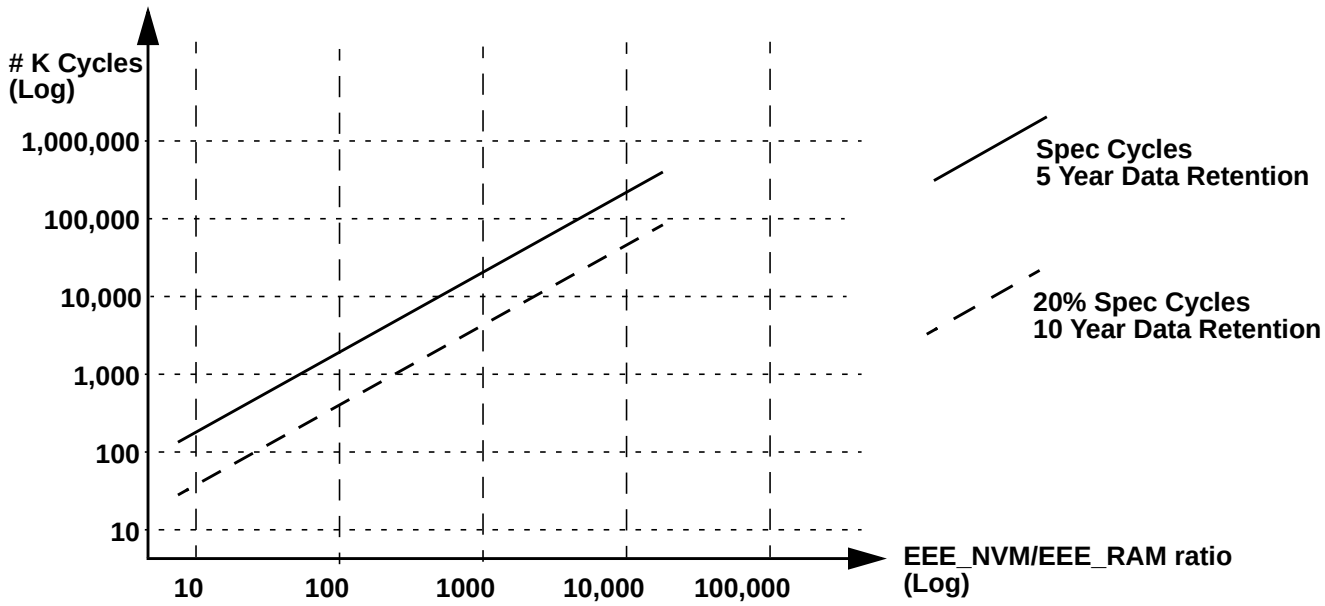
Table A-17. NVM Reliability Characteristics

Conditions are shown in Table A-4 unless otherwise noted							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
		P-Flash Arrays					
1	C	Data retention at an average junction temperature of T_{Javg} = 85°C[1] after up to 10,000 program/erase cycles	$t_{PNVMRET}$	15	100[2]	—	Years
2	C	Data retention at an average junction temperature of T_{Javg} = 85°C[3] after less than 100 program/erase cycles	$t_{PNVMRET}$	20	100[2]	—	Years
3	C	P-Flash number of program/erase cycles (-40°C ≤ tj ≤ 150°C)	n_{PFLPE}	10K	100K[3]	—	Cycles
		D-Flash Array					
4	C	Data retention at an average junction temperature of T_{Javg} = 85°C[3] after up to 50,000 program/erase cycles	$t_{DNVMRET}$	5	100[2]	—	Years
5	C	Data retention at an average junction temperature of T_{Javg} = 85°C[3] after less than 10,000 program/erase cycles	$t_{DNVMRET}$	10	100[2]	—	Years
6	C	Data retention at an average junction temperature of T_{Javg} = 85°C[3] after less than 100 program/erase cycles	$t_{DNVMRET}$	20	100[2]	—	Years
7	C	D-Flash number of program/erase cycles (-40°C ≤ tj ≤ 150°C)	n_{DFLPE}	50K	500K[3]	—	Cycles
		Emulated EEPROM					
8	C	Data retention at an average junction temperature of T_{Javg} = 85°C[1] after spec. program/erase cycles	$t_{EENVMRET}$	5[4]	100[2]	—	Years
9	C	Data retention at an average junction temperature of T_{Javg} = 85°C[3] after less than 20% spec.program/erase cycles. (e.g. after <20,000 cycles / Spec 100,000 cycles)	$t_{EENVMRET}$	10	100[2]	—	Years
10	C	Data retention at an average junction temperature of T_{Javg} = 85°C[3] after less than 0.2% spec. program/erase cycles (e.g. after < 200 cycles / Spec 100,000 cycles)	$t_{EENVMRET}$	20	100[2]	—	Years
11	C	EEPROM number of program/erase cycles with a ratio of EEE_NVM to EEE_RAM = 8 (-40°C ≤ tj ≤ 150°C)	n_{EEPE}	100K[4]	1M[5]	—	Cycles
12	C	EEPROM number of program/erase cycles with a ratio of EEE_NVM to EEE_RAM = 128 (-40°C ≤ tj ≤ 150°C)	n_{EEPE}	3M[4]	30M[5]	—	Cycles
13	C	EEPROM number of program/erase cycles with a ratio of EEE_NVM to EEE_RAM = 16384[6] (-40°C ≤ tj ≤ 150°C)	n_{EEPE}	325M[4]	3.2G[5]	—	Cycles

1. T_{Javg} does not exceed 85°C in a typical temperature profile over the lifetime of a consumer, industrial or automotive application.
2. Typical data retention values are based on intrinsic capability of the technology measured at high temperature and de-rated to 25°C using the Arrhenius equation. For additional information on how Freescale defines Typical Data Retention, please refer to Engineering Bulletin EB618
3. T_{Javg} does not exceed 85°C in a typical temperature profile over the lifetime of a consumer, industrial or automotive application.
4. This represents the number of writes of updated data words to the EEE_RAM partition. Minimum specification (endurance and data retention) of the Emulated EEPROM array is based on the minimum specification of the D-Flash array per item 6.
5. This represents the number of writes of updated data words to the EEE_RAM partition. Typical endurance performance for the Emulated EEPROM array is based on typical endurance performance and the EEE algorithm implemented on this product family. Spec. table quotes typical endurance evaluated at 25°C for this product family.
6. This is equivalent to using a single byte or aligned word in the EEE_RAM with 32K D-Flash allocated for EEEPROM

The number of program/erase cycles for the EEPROM/D-Flash resources depends upon the partitioning of D-Flash used for EEPROM Emulation. Defining RAM size allocated for EEE as RAM-EE and D-Flash partition used for emulation as FLASH-EE, the minimum number of program/erase cycles is specified depending upon the ratio of FLASH-EE/RAM-EE. The minimum ratio FLASH-EE/RAM-EE=8.

Figure A-2. Program/Erase Dependency on D-Flash Partitioning



A.4 Voltage Regulator

Table A-18. Voltage Regulator Electrical Characteristics

Conditions are shown in Table A-4 unless otherwise noted							
Num	**C**	**Characteristic**	**Symbol**	**Min**	**Typical**	**Max**	**Unit**
1	P	Input Voltages	$V_{VDDR,A}$	3.13	—	5.5	V
2	P	Output Voltage Core Full Performance Mode Reduced Power Mode (MCU STOP mode) Shutdown Mode	V_{DD}	1.72 — —	1.84 1.6 —(1)	1.98 — —	V V V
3	P	Output Voltage Flash Full Performance Mode Reduced Power Mode (MCU STOP mode) Shutdown Mode	V_{DDF}	2.6 — —	2.82 2.2 —1	2.9 — —	V V V
4	P	Output Voltage PLL Full Performance Mode Reduced Power Mode (MCU STOP mode) Shutdown Mode	V_{DDPLL}	1.72 — —	1.84 1.6 —1	1.98 — —	V V V
5	P	Low Voltage Interrupt Asser Level (2) Low Voltage InterruptDeassert Level	V_{LVIA} V_{LVID}	4.04 4.19	4.23 4.38	4.40 4.49	V V
6	P	VDDX Low Voltage Reset Deassert (3) (4)	V_{LVRXD}	—	—	3.13	V
7	C	Trimmed API internal clock(5) $\Delta f / f_{nominal}$	df_{API}	- 5%	—	+ 5%	—
8	D	The first period after enabling the counter by APIFE might be reduced by API start up delay	t_{sdel}	—	—	100	us
9	T	Temperature Sensor Slope	dV_{TS}	5.05	5.25	5.45	mV/°C
10	T	High Temperature Interrupt Assert (VREGHTTR=$88)(6) High Temperature Interrupt Deassert (VREGHTTR=$88)	T_{HTIA} T_{HTID}	120 110	132 122	144 134	°C

1. Voltage Regulator Disabled. High Impedance Output
2. Monitors VDDA, active only in Full Performance Mode. Indicates I/O & ADC performance degradation due to low supply voltage.
3. Device functionality is guaranteed on power down to the LVR assert level
4. Monitors VDDX, active only in Full Performance Mode. MCU is monitored by the POR in RPM (see Figure A-3)
5. The API Trimming bits must be set that the minimum period equals to 0.2 ms.
6. A hysteresis is guaranteed by design

A.5 Output Loads

A.5.1 Resistive Loads

The voltage regulator is intended to supply the internal logic and oscillator. It allows no external DC loads.

A.5.2 Capacitive Loads

The capacitive loads are specified in Table A-19. Ceramic capacitors with X7R dielectricum are required.

Table A-19. MC9S12XF512 - Required Capacitive Loads

Num	Characteristic	Symbol	Min	Recommended	Max	Unit
1	VDD/VDDF external capacitive load	C_{DDext}	176	220	264	nF
3	VDDPLL external capacitive load	$C_{DDPLLext}$	80	220	264	nF

A.5.3 Chip Power-up and Voltage Drops

LVI (low voltage interrupt), POR (power-on reset) and LVRs (low voltage reset) handle chip power-up or drops of the supply voltage. Their function is shown in Figure A-1 .

Figure A-3. MC9S12XF-Family - Chip Power-up and Voltage Drops (not scaled)

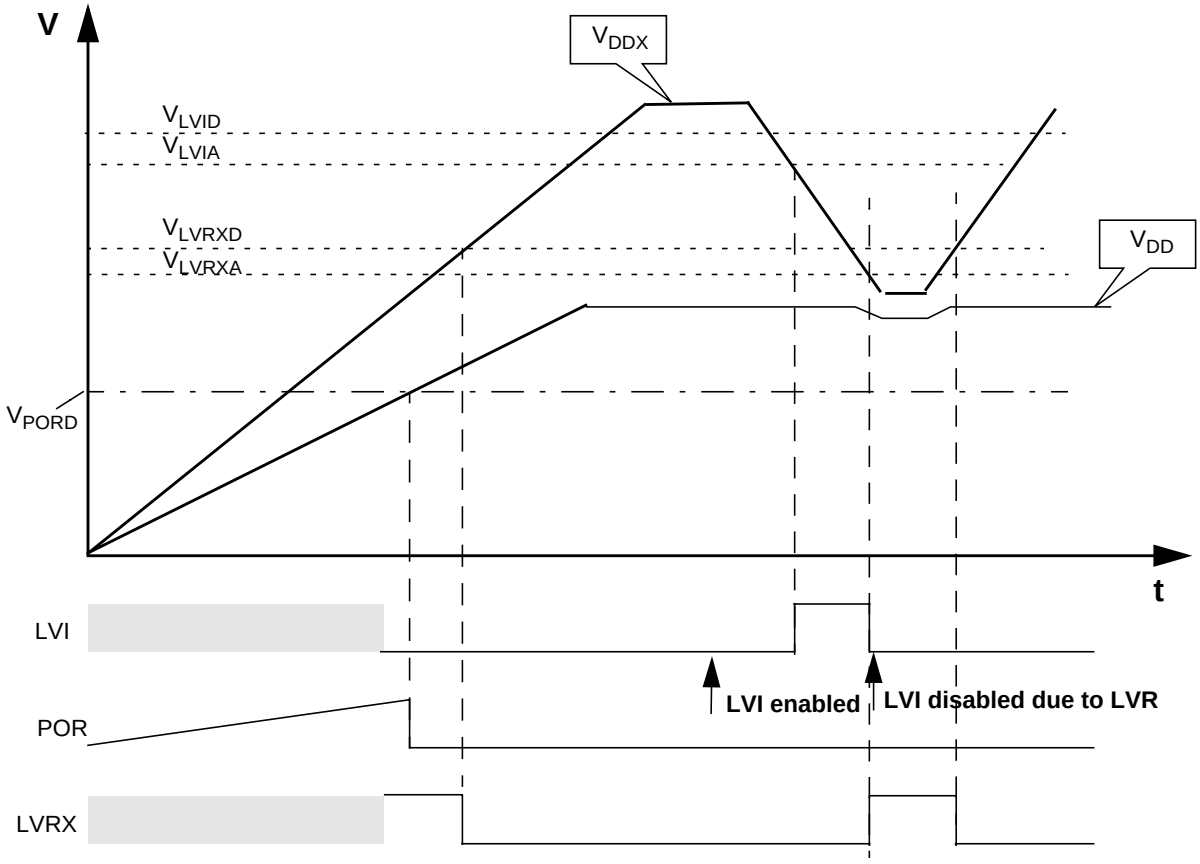
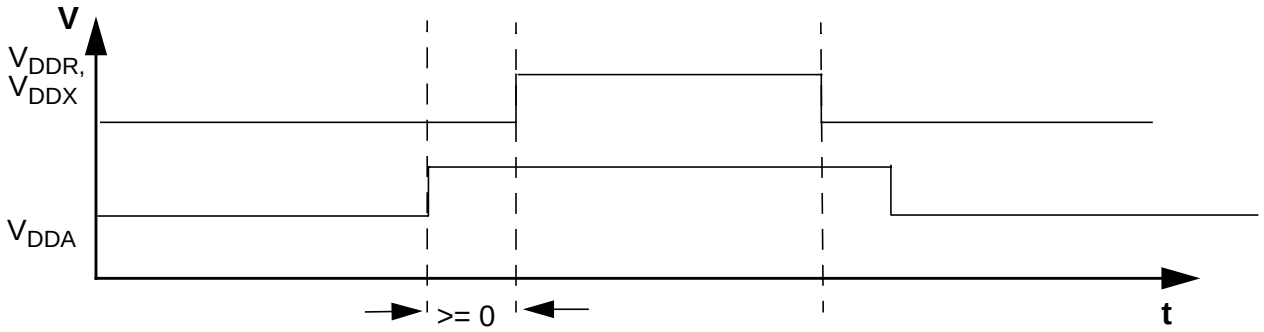


Figure A-4. MC9S12XF-Family Power Sequencing



During power sequencing V_{DDA} can be powered up before V_{DDR}, V_{DDX}.
V_{DDR} and V_{DDX} must be powered up together adhering to the operating conditions differential.
V_{RH} power up must follow V_{DDA} to avoid current injection.

A.6 Reset, Oscillator and PLL

This section summarizes the electrical characteristics of the various startup scenarios for oscillator and phase-locked loop (PLL).

A.6.1 Startup

Table A-20 summarizes several startup characteristics explained in this section. Detailed description of the startup behavior can be found in the Clock and Reset Generator (CRG) block description

Table A-20. Startup Characteristics

Conditions are shown in Table A-4.unless otherwise noted							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	D	Reset input pulse width, minimum input time	PW_{RSTL}	2	—	—	t_{osc}
2	D	Startup from reset	t_{RST}	96	—	4000(1)	n_{bus}
3	D	Wait recovery startup time	t_{WRS}	—	—	14	t_{cyc}
4	D	Fast wakeup from STOP(2)	t_{fws}	—	50	100	µs

1. This is the time between RESET deassertion and start of CPU code execution.
2. Including voltage regulator startup; V_{DD} /V_{DDF} filter capacitors 220 nF, V_{DD35} = 5 V, T= 25°C

A.6.1.1 POR

The release level V_{PORR} and the assert level V_{PORA} are derived from the V_{DD} supply. They are also valid if the device is powered externally. After releasing the POR reset the oscillator and the clock quality check are started. If after a time t_{CQOUT} no valid oscillation is detected, the MCU will start using the internal self clock. The fastest startup time possible is given by n_{uposc}.

A.6.1.2 SRAM Data Retention

Provided an appropriate external reset signal is applied to the MCU, preventing the CPU from executing code when V_{DD35} is out of specification limits, the SRAM contents integrity is guaranteed if after the reset the PORF bit in the CRG flags register has not been set.

A.6.1.3 External Reset

When external reset is asserted for a time greater than PW_{RSTL} the CRG module generates an internal reset, and the CPU starts fetching the reset vector without doing a clock quality check, if there was an oscillation before reset.

A.6.1.4 Stop Recovery

Out of stop the controller can be woken up by an external interrupt. A clock quality check as after POR is performed before releasing the clocks to the system.

If the MCU is woken-up by an interrupt and the fast wake-up feature is enabled (FSTWKP = 1 and SCME = 1), the system will resume operation in self-clock mode after t_{fws}.

A.6.1.5 Pseudo Stop and Wait Recovery

The recovery from pseudo stop and wait is essentially the same since the oscillator is not stopped in both modes. The controller can be woken up by internal or external interrupts. After t_{wrs} the CPU starts fetching the interrupt vector.

A.6.2 Oscillator

Table A-21. Oscillator Characteristics

Conditions are shown in Table A-4. unless otherwise noted							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1a	C	Crystal oscillator range (loop controlled Pierce)	f_{OSC}	4.0	—	16	MHz
1b	C	Crystal oscillator range (full swing Pierce) (1),(2)	f_{OSC}	2.0	—	40	MHz
2	P	Startup Current	i_{OSC}	100	—	—	µA
3a	C	Oscillator start-up time (LCP, 4MHz)(3)	t_{UPOSC}	—	2	10	ms
3b	C	Oscillator start-up time (LCP, 8MHz)3	t_{UPOSC}	—	1.6	8	ms
3c	C	Oscillator start-up time (LCP, 16MHz)3	t_{UPOSC}	—	1	5	ms
4a	C	Oscillator start-up time (full swing Pierce, 2MHz)3	t_{UPOSC}	—	8	40	ms
4b	C	Oscillator start-up time (full swing Pierce, 4MHz)3	t_{UPOSC}	—	4	20	ms
4c	C	Oscillator start-up time (full swing Pierce, 8MHz)3	t_{UPOSC}	—	2	10	ms
4d	C	Oscillator start-up time (full swing Pierce, 16MHz)3	t_{UPOSC}	—	1	5	ms
4e	C	Oscillator start-up time (full swing Pierce, 40MHz)3	t_{UPOSC}	—	0.8	4	ms
5	D	Clock Quality check time-out	t_{CQOUT}	0.45	—	2.5	s
6	P	Clock Monitor Failure Assert Frequency	f_{CMFA}	200	400	1000	KHz
7	P	External square wave input frequency	f_{EXT}	2.0	—	50	MHz
8	D	External square wave pulse width low	t_{EXTL}	9.5	—	—	ns
9	D	External square wave pulse width high	t_{EXTH}	9.5	—	—	ns
10	D	External square wave rise time	t_{EXTR}	—	—	1	ns
11	D	External square wave fall time	t_{EXTF}	—	—	1	ns
12	D	Input Capacitance (EXTAL, XTAL pins)	C_{IN}	—	7	—	pF
13	P	EXTAL Pin Input High Voltage	$V_{IH,EXTAL}$	0.75*V_{DDPLL}	—	—	V
	T	EXTAL Pin Input High Voltage(4)	$V_{IH,EXTAL}$	—	—	V_{DDPLL} + 0.3	V
14	P	EXTAL Pin Input Low Voltage	$V_{IL,EXTAL}$	—	—	0.25*V_{DDPLL}	V
	T	EXTAL Pin Input Low Voltage4	$V_{IL,EXTAL}$	V_{SSPLL} - 0.3	—	—	V
15	C	EXTAL Pin Input Hysteresis	$V_{HYS,EXTAL}$	—	180	—	mV
16	C	EXTAL Pin oscillation amplitude (loop controlled Pierce)	$V_{PP,EXTAL}$	—	0.9	—	V

1. Depending on the crystal a damping series resistor might be necessary
2. Only valid if full swing Pierce oscillator/external clock mode is selected
3. These values apply for carefully designed PCB layouts with capacitors that match the crystal/resonator requirements..
4. Only applies if EXTAL is externally driven

A.6.3 Phase Locked Loop

A.6.3.1 Jitter Information

With each transition of the clock f_{cmp}, the deviation from the reference clock f_{ref} is measured and input voltage to the VCO is adjusted accordingly.The adjustment is done continuously with no abrupt changes in the clock output frequency. Noise, voltage, temperature and other factors cause slight variations in the control loop resulting in a clock jitter. This jitter affects the real minimum and maximum clock periods as illustrated in Figure A-5.

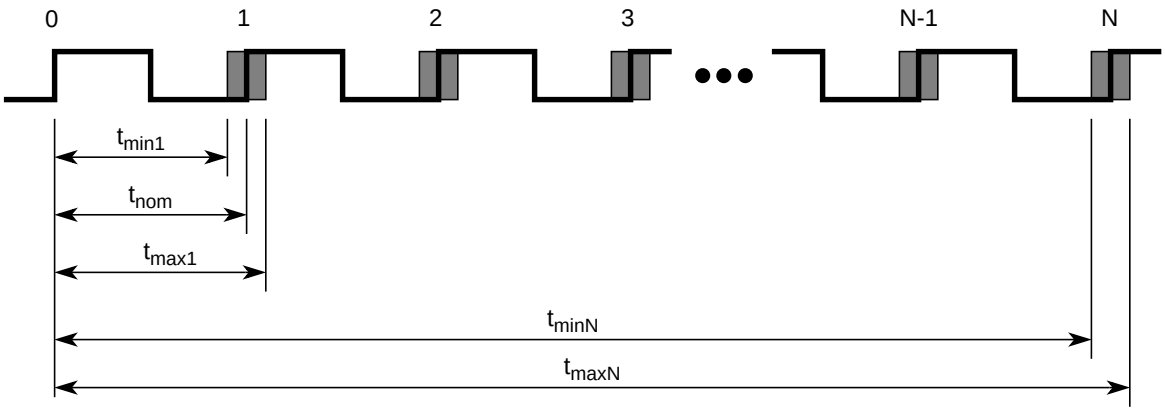


Figure A-5. Jitter Definitions

The relative deviation of t_{nom} is at its maximum for one clock period, and decreases towards zero for larger number of clock periods (N).

Defining the jitter as:

$$J(N) = \max\left(\left|1 - \frac{t_{max}(N)}{N \cdot t_{nom}}\right|, \left|1 - \frac{t_{min}(N)}{N \cdot t_{nom}}\right|\right)$$

For N < 10000, the following equation is a good fit for the maximum jitter:

$$J(N) = \frac{j_1}{\sqrt{N}} + j_2$$

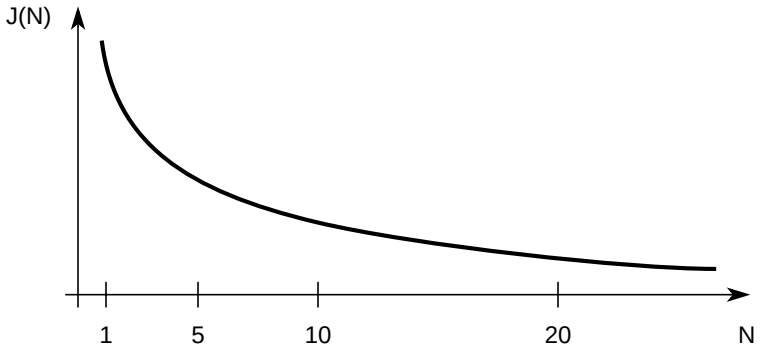


Figure A-6. Maximum bus clock or CGMPLL clock jitter approximation

This is important to note with respect to timers, serial modules where a prescaler will eliminate the effect of the jitter to a large extent.

Table A-22. IPLL Characteristics

Conditions are shown in Table A-4 unless otherwise noted							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	P	Self Clock Mode frequency	f_{SCM}	1		5.5	MHz
2	T	VCO locking range	f_{VCO}	32		120	MHz
3	T	Reference Clock	f_{REF}	1		40	MHz
4	D	System IPLL: Lock Detection	$\|\Delta_{Lock}\|$	0		1.5	%(1)
5	D	System IPLL: Un-Lock Detection	$\|\Delta_{unl}\|$	0.5		2.5	%1
7	C	System IPLL: PLLON Stabilization delay(2)	t_{stab}		0.25		ms
8	C	System IPLL: Jitter fit parameter 1[2]	j_1			1.2	%
9	C	System IPLL: Jitter fit parameter 2[2]	j_2			0	%
10	D	System IPLL: Bus Frequency for FM1=1, FM0=1 (frequency modulation in PLLCTL register of s12xe_crg)	f_{bus}			48	MHz
11	D	System IPLL: Bus Frequency for FM1=1, FM0=0 (frequency modulation in PLLCTL register of s12xe_crg)	f_{bus}			49	MHz
12	D	System IPLL: Bus Frequency for FM1=0, FM0=1 (frequency modulation in PLLCTL register of s12xe_crg)	f_{bus}			49	MHz
13	D	FlexRay IPLL: Lock Detection	$\|\Delta_{Lock}\|$	0		1.5	%1
14	D	FlexRay IPLL: Un-Lock Detection	$\|\Delta_{unl}\|$	0.5		2.5	%1
15	C	FlexRay IPLL: PLLON Stabilization delay(3)	t_{stab}		0.25		ms
16	C	FlexRay IPLL: Jitter fit parameter 1[3]	j_1			1.7	%
17	C	FlexRay IPLL: Jitter fit parameter 2[3]	j_2			0	%

1. System IPLL: % deviation from target frequency
2. System IPLL: f_{OSC} = 8MHz, f_{BUS} = 50MHz equivalent f_{PLL} = 100MHz: REFDIV=$03, REFRQ=00, SYNDIV=$18, VCOFRQ=11, POSTDIV=$ 00. Jitter spec is valid for N=1 to 10000 bus clock cycles
3. FlexRay IPLL: f_{OSC} = 8MHz equivalent f_{CGMPLL} = 80MHz: REFDIV=$00, REFRQ=10, SYNDIV=$04, VCOFRQ=01, POSTDIV=$ 00. Jitter spec is valid for N=1 to 10000 CGMPLL clock cycles

A.7 External Interface Timing

A.7.1 MSCAN

Table A-23. MSCAN Wake-up Pulse Characteristics

Conditions are shown in Table A-4 unless otherwise noted							
Num	**C**	**Rating**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
1	P	MSCAN wakeup dominant pulse filtered	t_{WUP}	—	—	1.5	µs
2	P	MSCAN wakeup dominant pulse pass	t_{WUP}	5	—	—	µs

A.7.2 SPI Timing

This section provides electrical parametrics and ratings for the SPI. In Table A-24 the measurement conditions are listed.

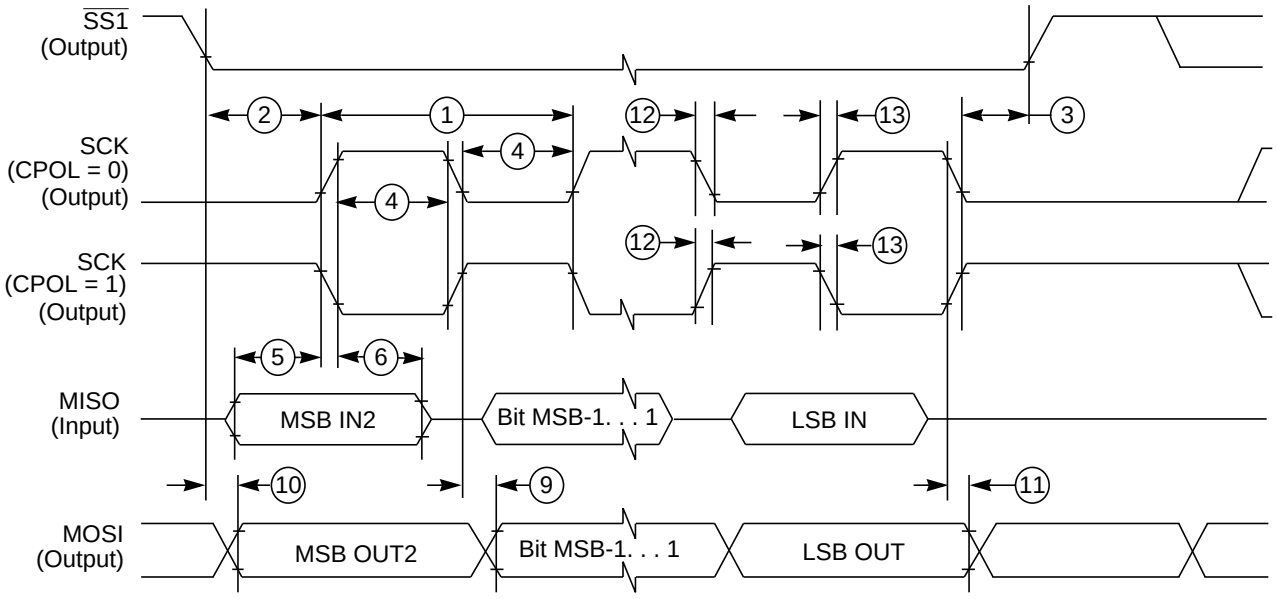
Table A-24. Measurement Conditions

Description	Value	Unit
Drive mode	Full drive mode	—
Load capacitance C_{LOAD}[(1)], on all outputs	50	pF
Thresholds for delay measurement points	(20% / 80%) V_{DDX}	V

1. Timing specified for equal load on all SPI output pins. Avoid asymmetric load.

A.7.2.1 Master Mode

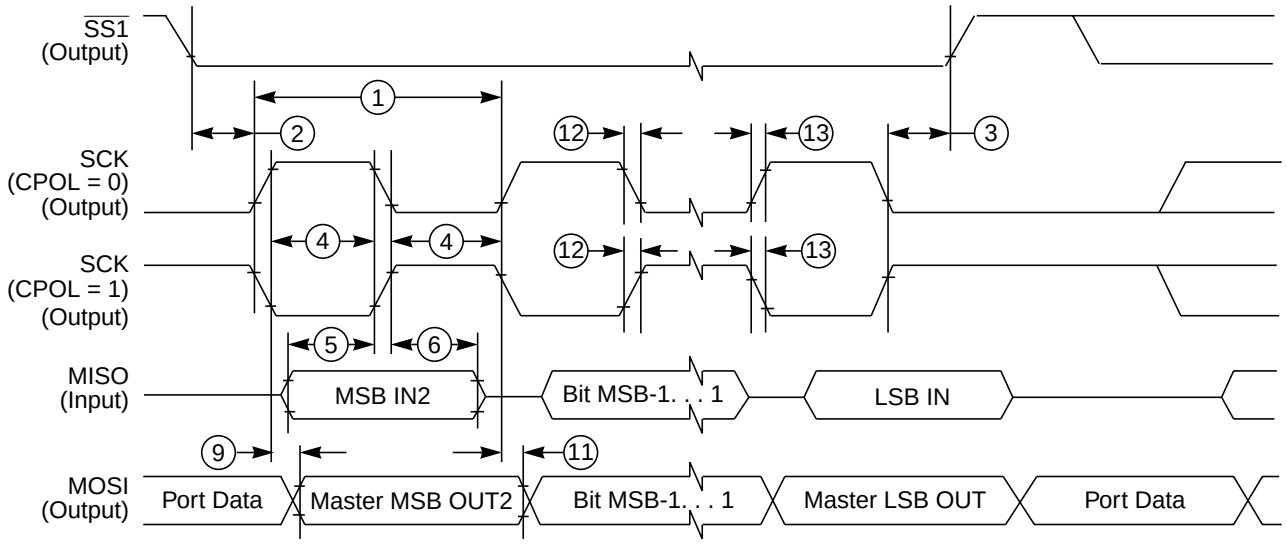
In Figure A-7 the timing diagram for master mode with transmission format CPHA = 0 is depicted.



1. If configured as an output.
2. LSBF = 0. For LSBF = 1, bit order is LSB, bit 1, bit 2... MSB.

Figure A-7. SPI Master Timing (CPHA = 0)

In Figure A-8 the timing diagram for master mode with transmission format CPHA=1 is depicted.



1.If configured as output
2. LSBF = 0. For LSBF = 1, bit order is LSB, bit 1,bit 2... MSB.

Figure A-8. SPI Master Timing (CPHA = 1)

In Table A-25 the timing characteristics for master mode are listed.

Table A-25. SPI Master Mode Timing Characteristics

Num	C	Characteristic	Symbol	Min	Typ	Max	Unit
1	D	SCK frequency	f_{sck}	1/2048	—	1/2[1]	f_{bus}
1	D	SCK period	t_{sck}	2^1	—	2048	t_{bus}
2	D	Enable lead time	t_{lead}	—	1/2	—	t_{sck}
3	D	Enable lag time	t_{lag}	—	1/2	—	t_{sck}
4	D	Clock (SCK) high or low time	t_{wsck}	—	1/2	—	t_{sck}
5	D	Data setup time (inputs)	t_{su}	8	—	—	ns
6	D	Data hold time (inputs)	t_{hi}	8	—	—	ns
9	D	Data valid after SCK edge	t_{vsck}	—	—	15	ns
10	D	Data valid after $\overline{SS}$ fall (CPHA = 0)	t_{vss}	—	—	15	ns
11	D	Data hold time (outputs)	t_{ho}	0	—	—	ns
12	D	Rise and fall time inputs	t_{rfi}	—	—	8	ns
13	D	Rise and fall time outputs	t_{rfo}	—	—	8	ns

1. See **Figure A-9.**

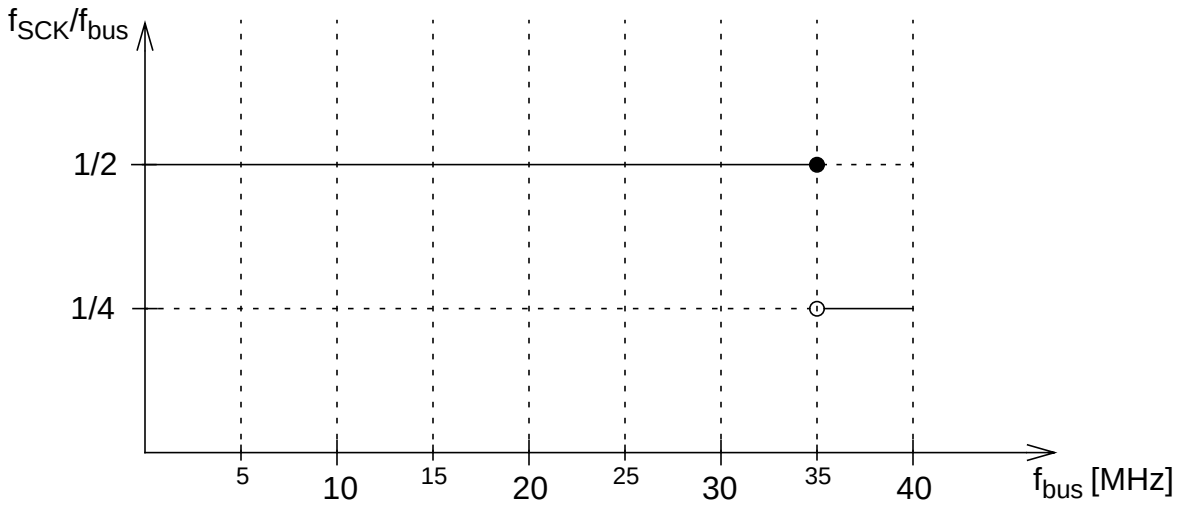
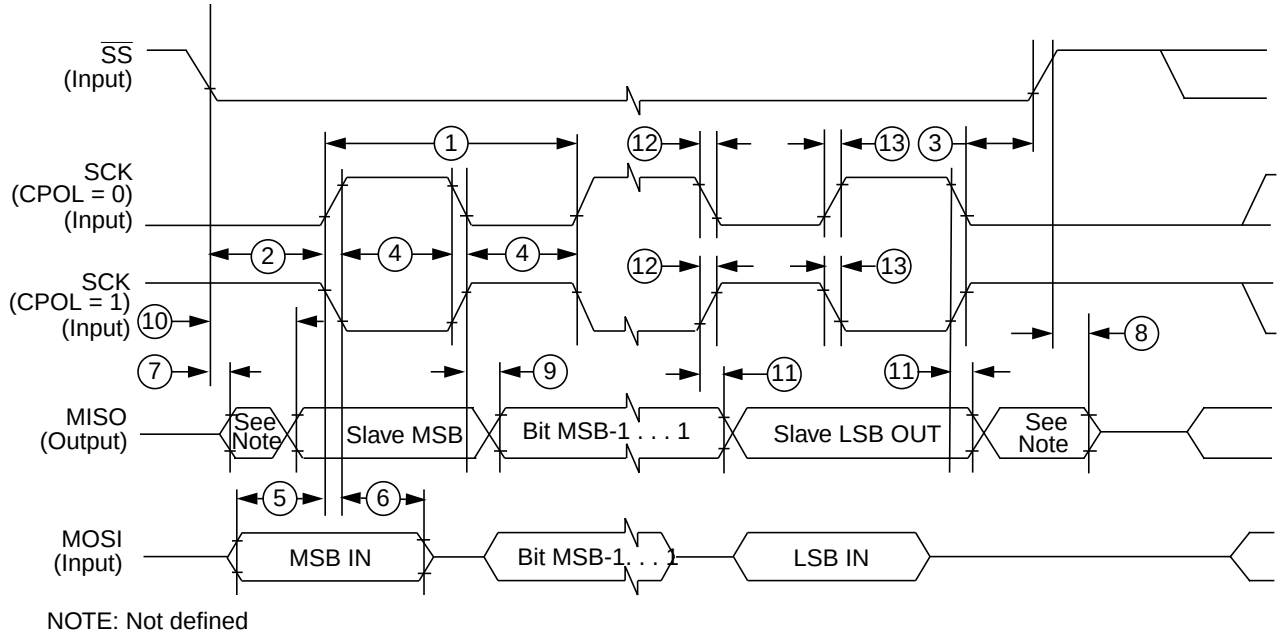


Figure A-9. Derating of maximum f_{SCK} to f_{bus} ratio in Master Mode

A.7.2.2 Slave Mode

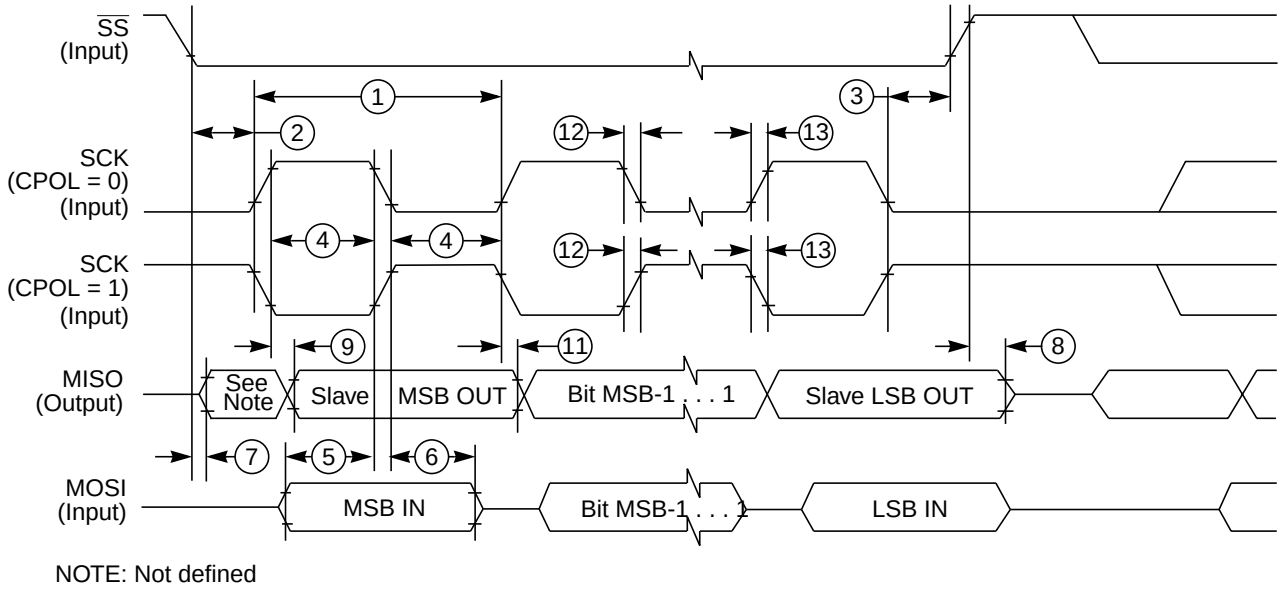
In Figure A-10 the timing diagram for slave mode with transmission format CPHA = 0 is depicted.



NOTE: Not defined

Figure A-10. SPI Slave Timing (CPHA = 0)

In Figure A-11 the timing diagram for slave mode with transmission format CPHA = 1 is depicted.



NOTE: Not defined

Figure A-11. SPI Slave Timing (CPHA = 1)

In Table A-26 the timing characteristics for slave mode are listed.

Table A-26. SPI Slave Mode Timing Characteristics

Num	C	Characteristic	Symbol	Min	Typ	Max	Unit
1	D	SCK frequency	f_{sck}	DC	—	1/4	f_{bus}
1	D	SCK period	t_{sck}	4	—	∞	t_{bus}
2	D	Enable lead time	t_{lead}	4	—	—	t_{bus}
3	D	Enable lag time	t_{lag}	4	—	—	t_{bus}
4	D	Clock (SCK) high or low time	t_{wsck}	4	—	—	t_{bus}
5	D	Data setup time (inputs)	t_{su}	8	—	—	ns
6	D	Data hold time (inputs)	t_{hi}	8	—	—	ns
7	D	Slave access time (time to data active)	t_a	—	—	20	ns
8	D	Slave MISO disable time	t_{dis}	—	—	22	ns
9	D	Data valid after SCK edge	t_{vsck}	—	—	$28 + 0.5 \cdot t_{bus}$ (1)	ns
10	D	Data valid after SS fall	t_{vss}	—	—	$28 + 0.5 \cdot t_{bus}$ 1	ns
11	D	Data hold time (outputs)	t_{ho}	20	—	—	ns
12	D	Rise and fall time inputs	t_{rfi}	—	—	8	ns
13	D	Rise and fall time outputs	t_{rfo}	—	—	8	ns

1. 0.5 t_{bus} added due to internal synchronization delay

A.7.3 External Bus Timing

The following conditions are assumed for all following external bus timing values:

- Crystal input within 45% to 55% duty
- Equal loads of pins
- Pad full drive (reduced drive must be off)

A.7.3.1 Normal Expanded Mode (External Wait Feature Disabled)

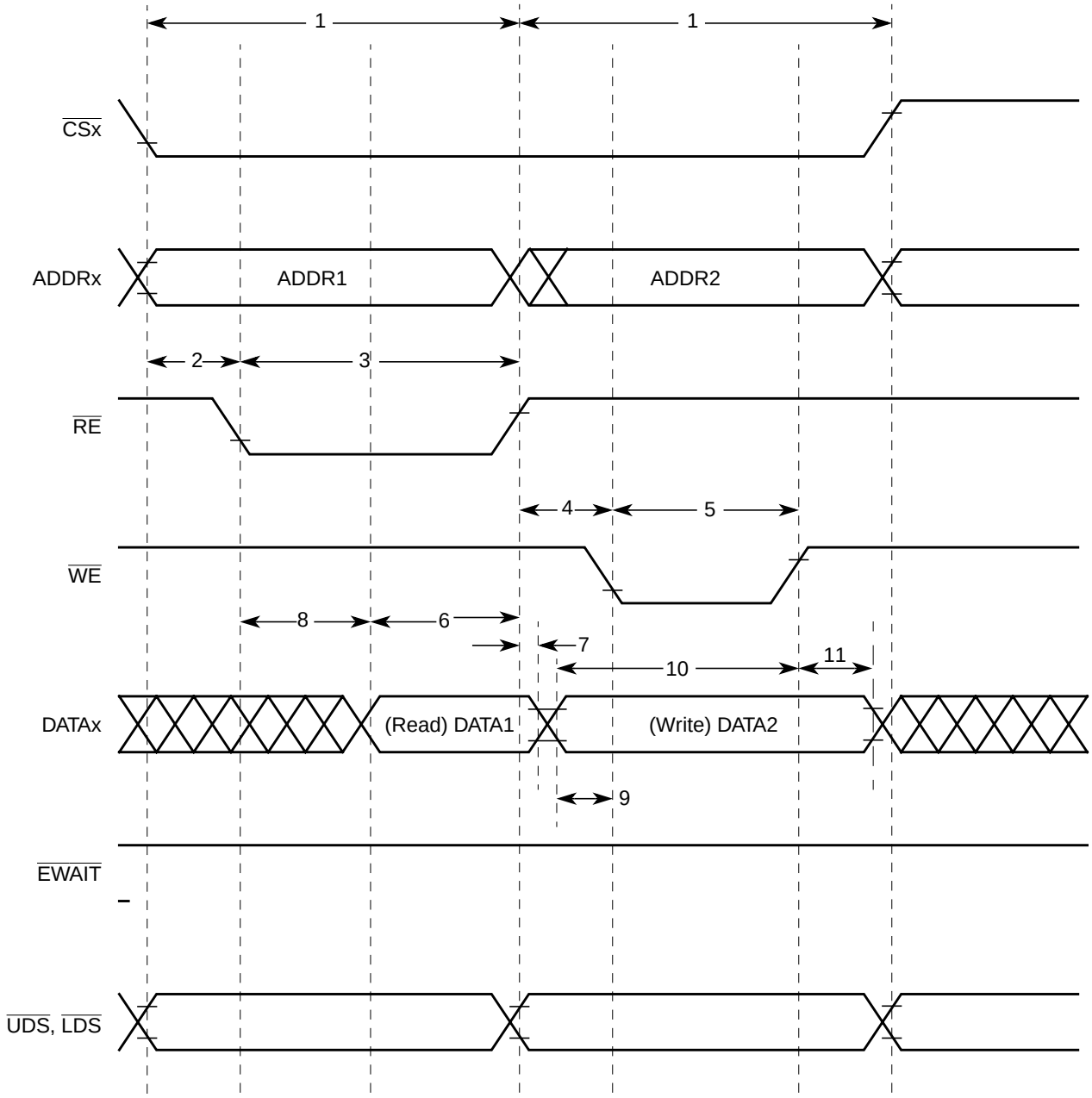


Figure A-12. Example 1a: Normal Expanded Mode — Read Followed by Write

Table A-27. Example 1a: Normal Expanded Mode Timing 50 MHz bus (EWAIT disabled)

No.	Characteristic	Symbol	V_{DD35}=5.0V			Unit
			C	Min	Max	
-	Frequency of internal bus	f_i	-	D.C.	50.0	MHz
-	Internal cycle time	t_{cyc}	-	20	∞	ns
-	Frequency of external bus	f_o	-	D.C.	25.0	MHz
1	External cycle time (selected by EXSTR)	t_{cyce}	-	40	∞	ns
2	Address (1) valid to $\overline{RE}$ fall	t_{ADRE}	D	4	-	ns
3	Pulse width, $\overline{RE}$	PW_{RE}	D	28	-	ns
4	Address valid to $\overline{WE}$ fall	t_{ADWE}	D	4	-	ns
5	Pulse width, $\overline{WE}$	PW_{WE}	D	18	-	ns
6	Read data setup time (if ITHRS = 0)	t_{DSR}	D	19	-	ns
	Read data setup time (if ITHRS = 1)	t_{DSR}	D	23	-	ns
7	Read data hold time	t_{DHR}	D	0	-	ns
8	Read enable access time	t_{ACCR}	D	4	-	ns
9	Write data valid to $\overline{WE}$ fall	t_{WDWE}	D	5	-	ns
10	Write data setup time	t_{DSW}	D	23	-	ns
11	Write data hold time	t_{DHW}	D	6	-	ns

1. Includes the following signals: ADDRx, $\overline{UDS}$, $\overline{LDS}$, and $\overline{CSx}$.

A.7.3.2 Normal Expanded Mode (External Wait Feature Enabled)

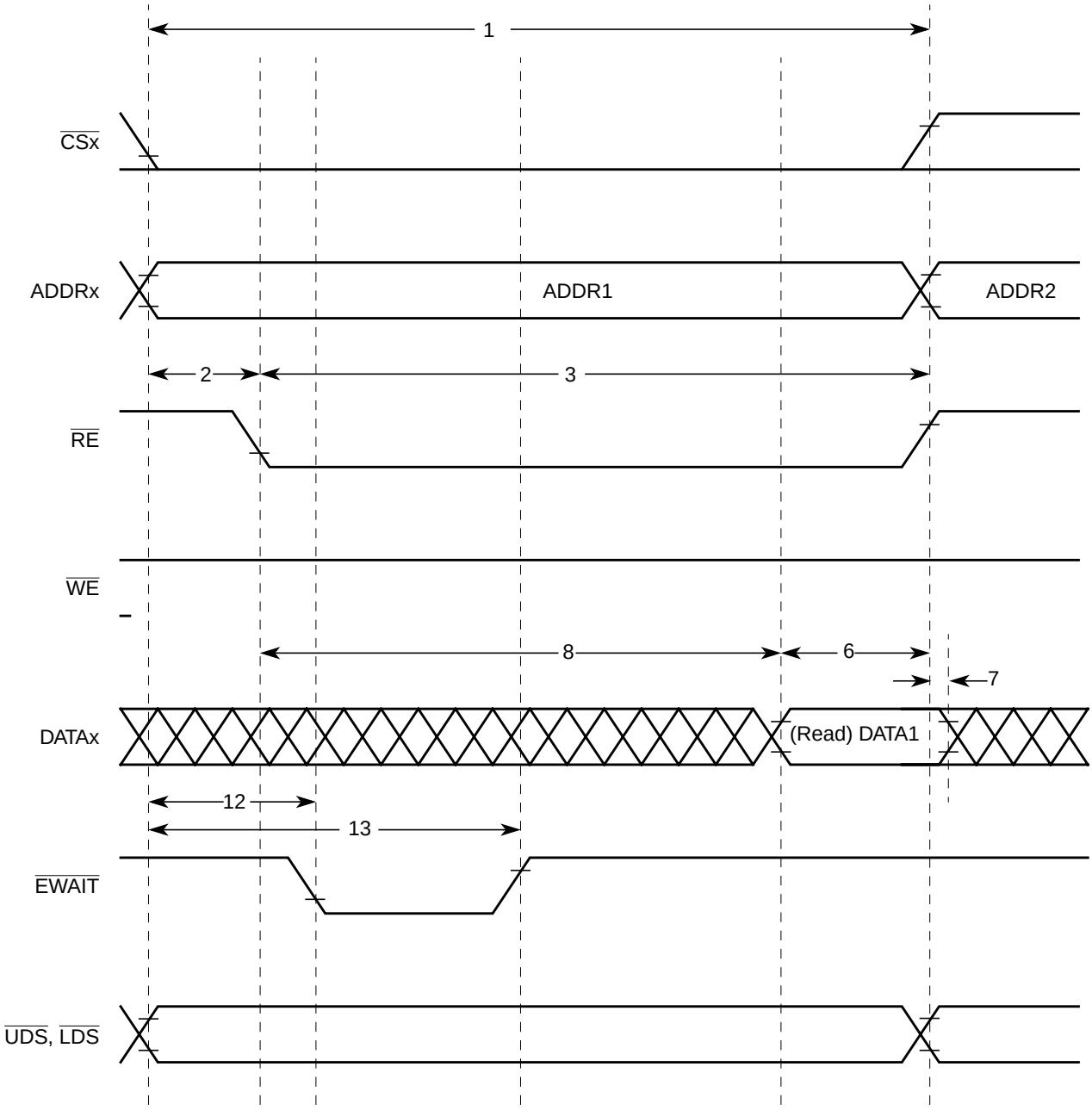


Figure A-13. Example 1b: Normal Expanded Mode — Stretched Read Access

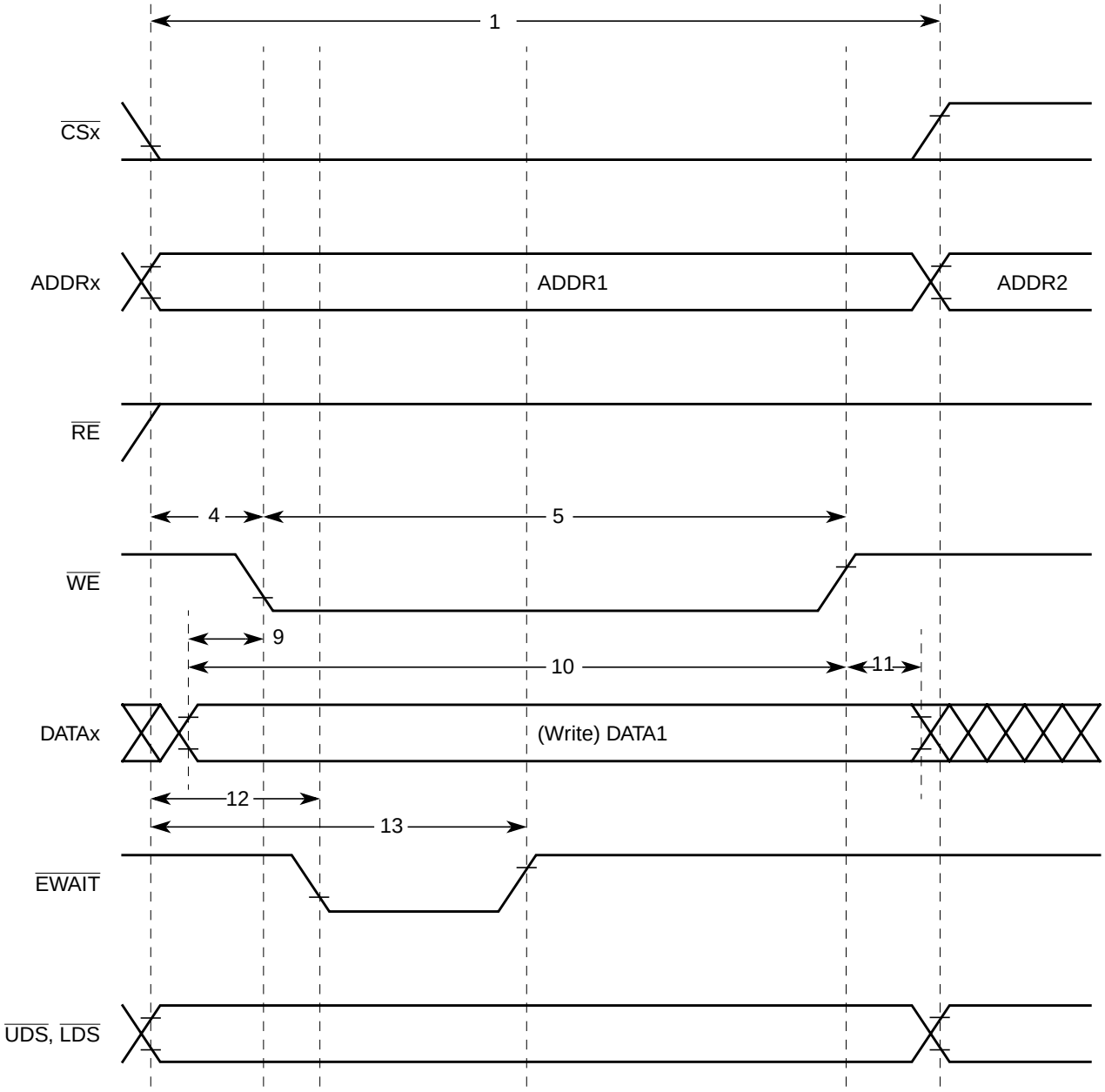


Figure A-14. Example 1b: Normal Expanded Mode — Stretched Write Access

Table A-28. Example 1b: Normal Expanded Mode Timing at 50MHz bus (EWAIT enabled)

No.	Characteristic	Symbol	C	V_{DD5} = 5.0V				Unit
				2 stretch cycles		3 stretch cycles		
				Min	Max	Min	Max	
-	Frequency of internal bus	f_i	-	D.C.	50.0	D.C.	50.0	MHz
-	Internal cycle time	t_{cyc}	-	20	∞	20	∞	ns
-	Frequency of external bus	f_o	-	D.C.	16.7	D.C.	12.5	MHz
-	External cycle time (selected by EXSTR)	t_{cyce}	-	60	∞	80	∞	ns
1	External cycle time (EXSTR+1EWAIT)	t_{cycew}	-	80	∞	100	∞	ns
2	Address (1) valid to $\overline{RE}$ fall	t_{ADRE}	D	4	-	4	-	ns
3	Pulse width, $\overline{RE}$ (2)	PW_{RE}	D	68	-	88	-	ns
4	Address valid to $\overline{WE}$ fall	t_{ADWE}	D	4	-	4	-	ns
5	Pulse width, $\overline{WE}$	PW_{WE}	D	58	-	78	-	ns
6	Read data setup time (if ITHRS = 0)	t_{DSR}	D	19	-	19	-	ns
	Read data setup time (if ITHRS = 1)	t_{DSR}	D	23	-	23	-	ns
7	Read data hold time	t_{DHR}	D	0	-	0	-	ns
8	Read enable access time	t_{ACCR}	D	49	-	69	-	ns
9	Write data valid to $\overline{WE}$ fall	t_{WDWE}	D	5	-	5	-	ns
10	Write data setup time	t_{DSW}	D	63	-	93	-	ns
11	Write data hold time	t_{DHW}	D	6	-	6	-	ns
12	Address to $\overline{EWAIT}$ fall	t_{ADWF}	D	0	16	0	36	ns
13	Address to $\overline{EWAIT}$ rise	t_{ADWR}	D	30	39	50	58	ns

1. Includes the following signals: ADDRx, $\overline{UDS}$, $\overline{LDS}$, and $\overline{CSx}$.
2. Affected by $\overline{EWAIT}$.

A.7.3.3 Emulation Single-Chip Mode (Without Wait States)

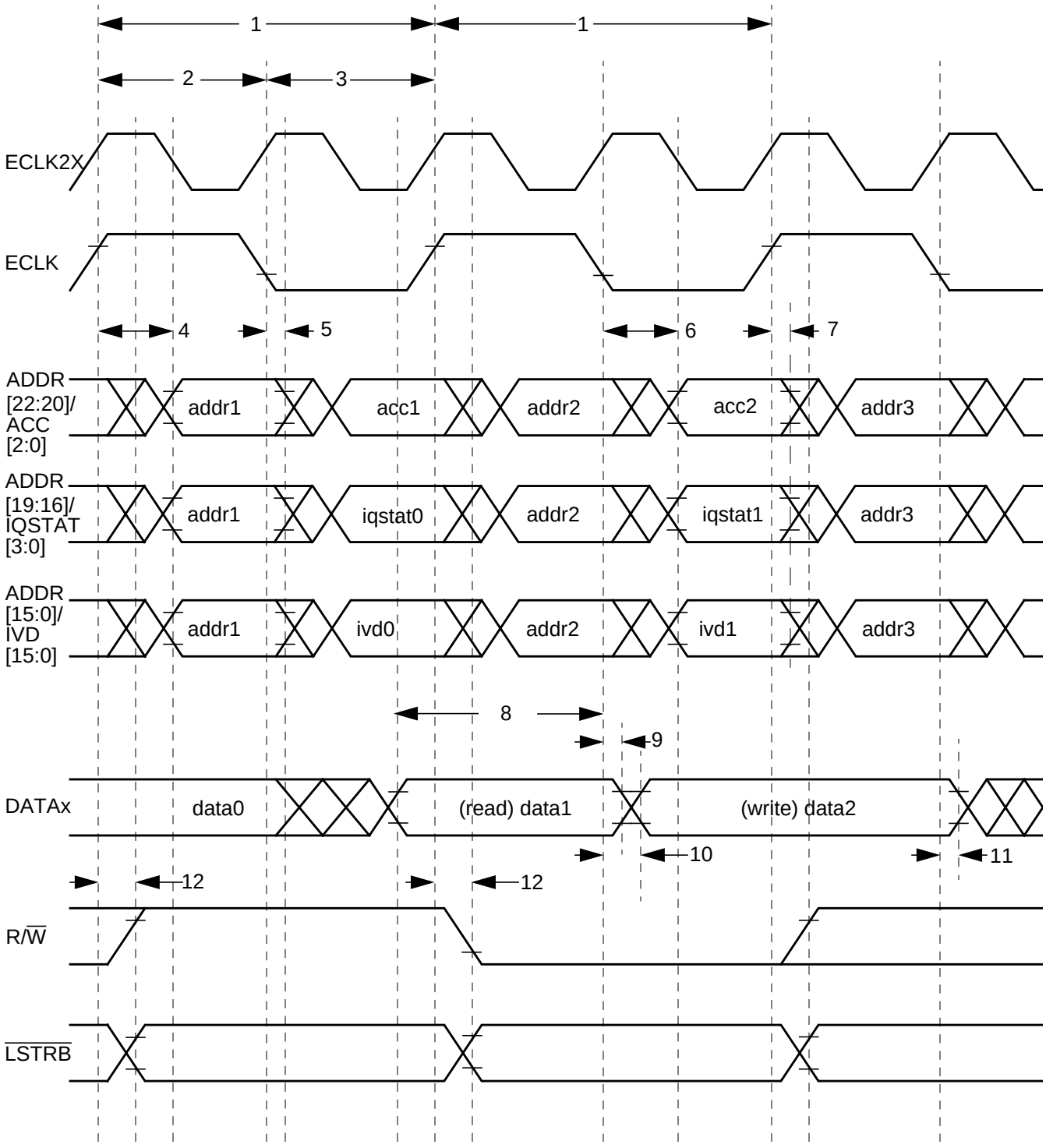


Figure A-15. Example 2a: Emulation Single-Chip Mode — Read Followed by Write

Table A-29. Example 2a: Emulation Single-Chip Mode Timing 50 Mhz bus, V_{DD5}=5.0V (EWAIT disabled)

No.	C	Characteristic [1]	Symbol	Min	Max	Unit
-	-	Frequency of internal bus	f_i	D.C.	50.0	MHz
1	-	Cycle time	t_{cyc}	20	∞	ns
2	D	Pulse width, E high	PW_{EH}	9	-	ns
3	D	Pulse width, E low	PW_{EL}	9	-	ns
4	D	Address delay time	t_{AD}	-	5	ns
5	D	Address hold time	t_{AH}	0	-	ns
6	D	IVDx delay time [2]	t_{IVDD}	-	4.5	ns
7	D	IVDx hold time	t_{IVDH}	0	-	ns
8	D	Read data setup time (ITHRS = 1 only)	t_{DSR}	12	-	ns
9	D	Read data hold time	t_{DHR}	0	-	ns
10	D	Write data delay time	t_{DDW}	-	5	ns
11	D	Write data hold time	t_{DHW}	0	-	ns
12	D	Read/write data delay time [3]	t_{RWD}	-1	5	ns

1. Typical Supply and Silicon, Room Temperature Only
2. Includes also ACCx, IQSTATx
3. Includes LSTRB

A.7.3.4 Emulation Expanded Mode (With Optional Access Stretching)

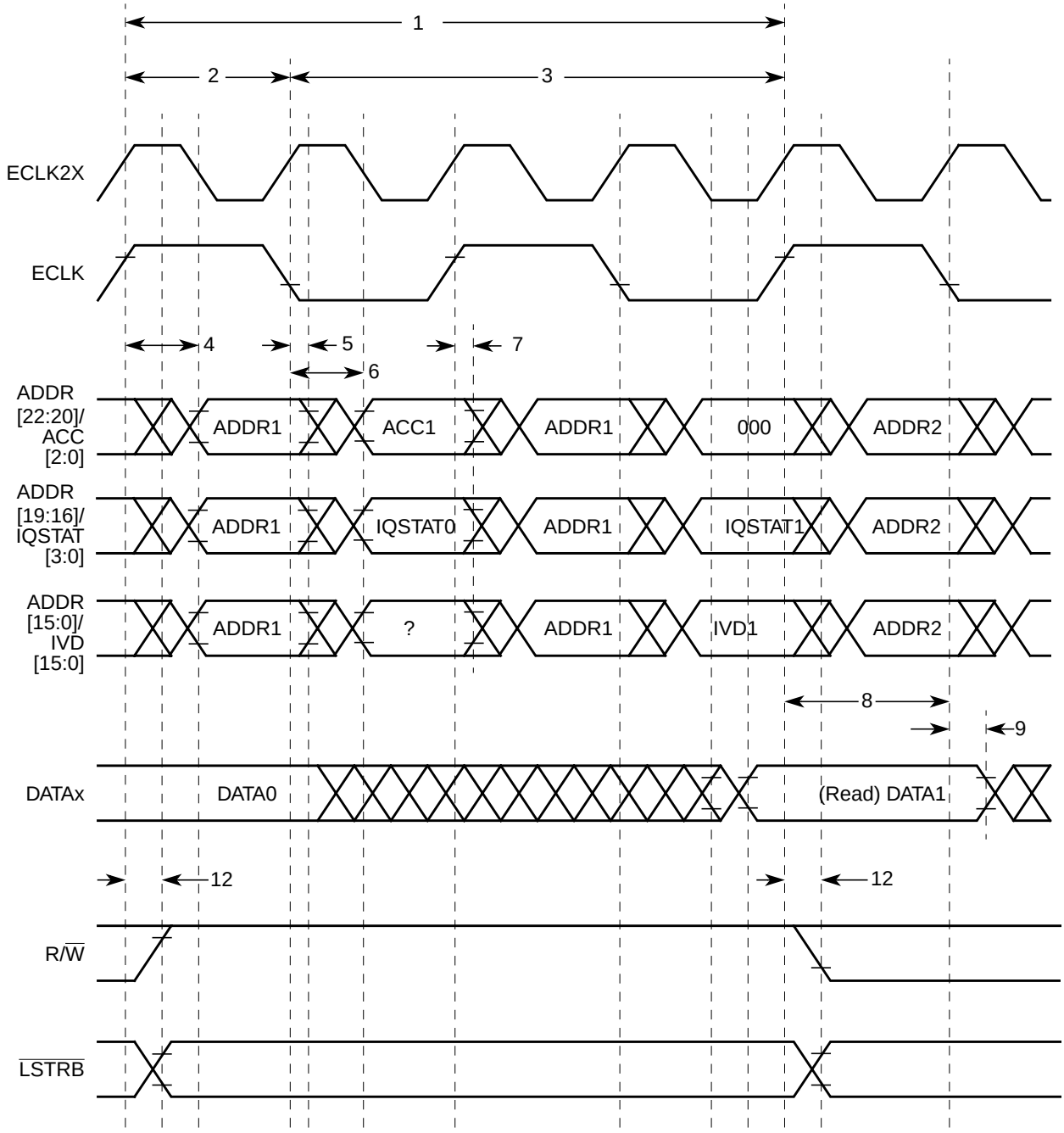


Figure A-16. Example 2b: Emulation Expanded Mode — Read with 1 Stretch Cycle

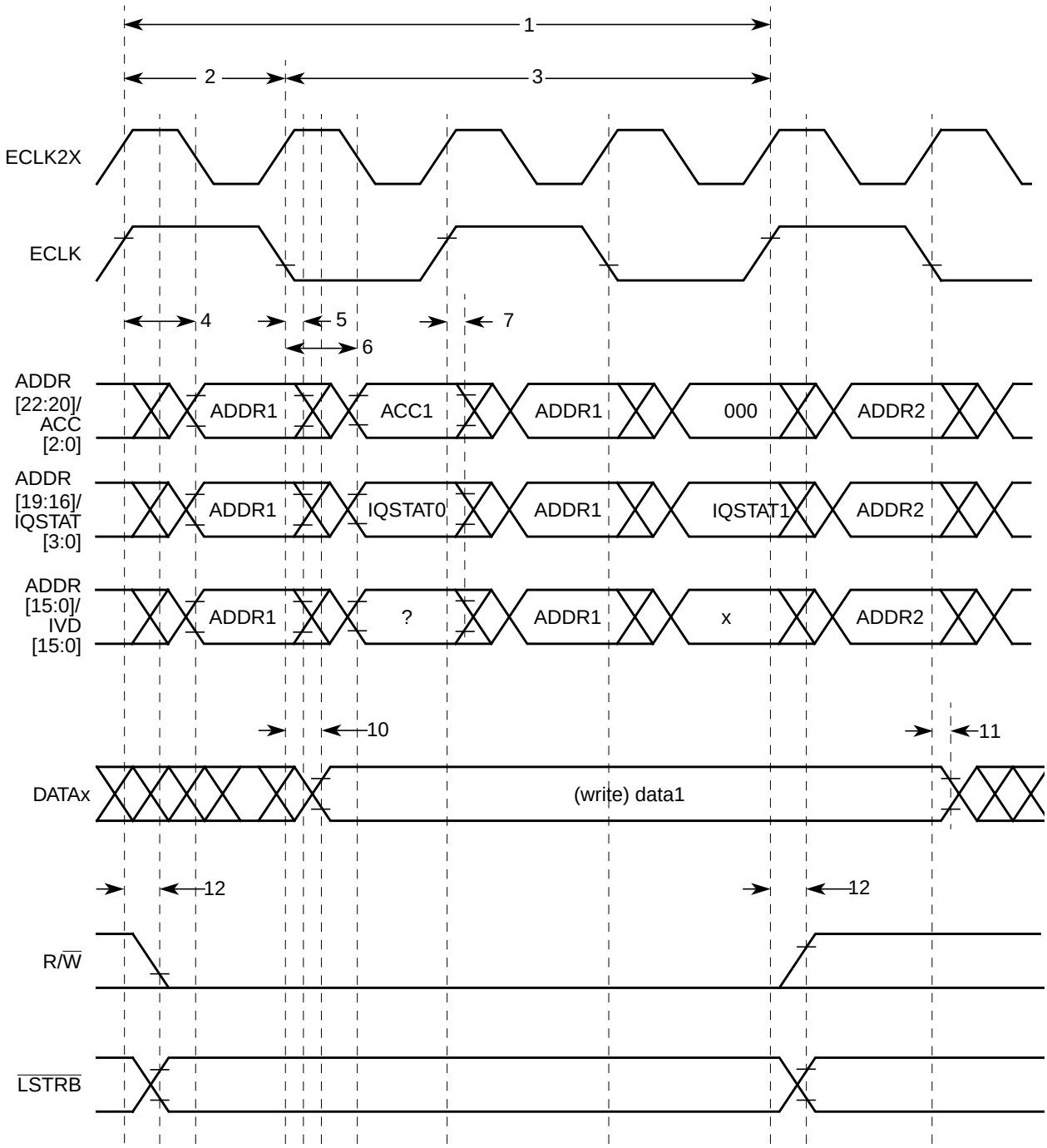


Figure A-17. Example 2b: Emulation Expanded Mode Ò Write with 1 Stretch Cycle

Table A-30. Example 2b: Emulation Expanded Mode Timing 50 MHz bus, V_{DD5}=5.0V (EWAIT disabled)

No.	C	Characteristic [1]	Symbol	1 stretch cycle		2 stretch cycles		3 stretch cycles		Unit
				Min	Max	Min	Max	Min	Max	
-	-	Internal cycle time	t_{cyc}	20	∞	20	∞	20	∞	ns
1	-	Cycle time	t_{cyce}	40	∞	60	∞	80	∞	ns
2	D	Pulse width, E high	PW_{EH}	9	11	9	11	9	11	ns
3	D	E falling to sampling E rising	t_{EFSR}	28	32	48	52	68	72	ns
4	D	Address delay time	t_{AD}	refer to table Table A-29		refer to table Table A-29		refer to table Table A-29		ns
5	D	Address hold time	t_{AH}							ns
6	D	IVD delay time [2]	t_{IVDD}							ns
7	D	IVD hold time	t_{IVDH}							ns
8	D	Read data setup time	t_{DSR}							ns
9	D	Read data hold time	t_{DHR}							ns
10	D	Write data delay time	t_{DDW}							ns
11	D	Write data hold time	t_{DHW}							ns
12	D	Read/write data delay time [3]	t_{RWD}							ns

1. Typical Supply and Silicon, Room Temperature Only
2. Includes also ACCx, IQSTATx
3. Includes $\overline{LSTRB}$

A.7.3.5 External Tag Trigger Timing

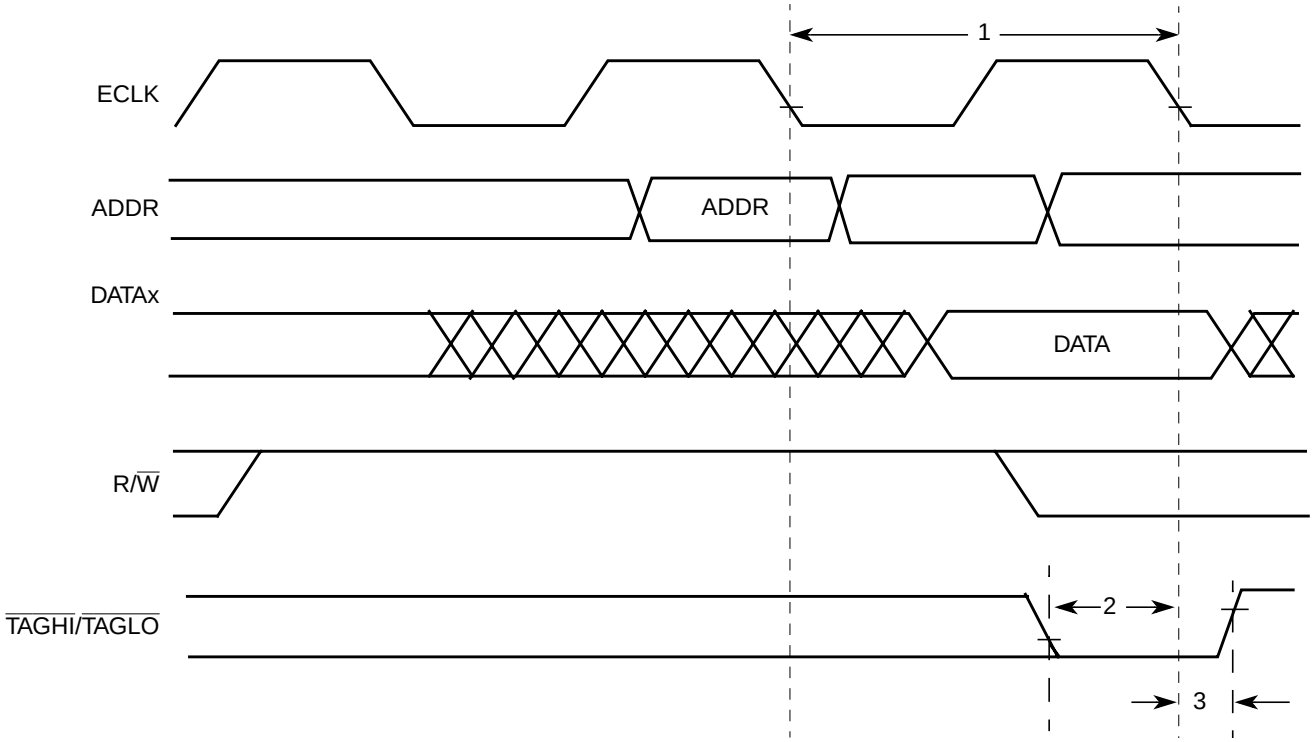


Figure A-18. External Trigger Timing

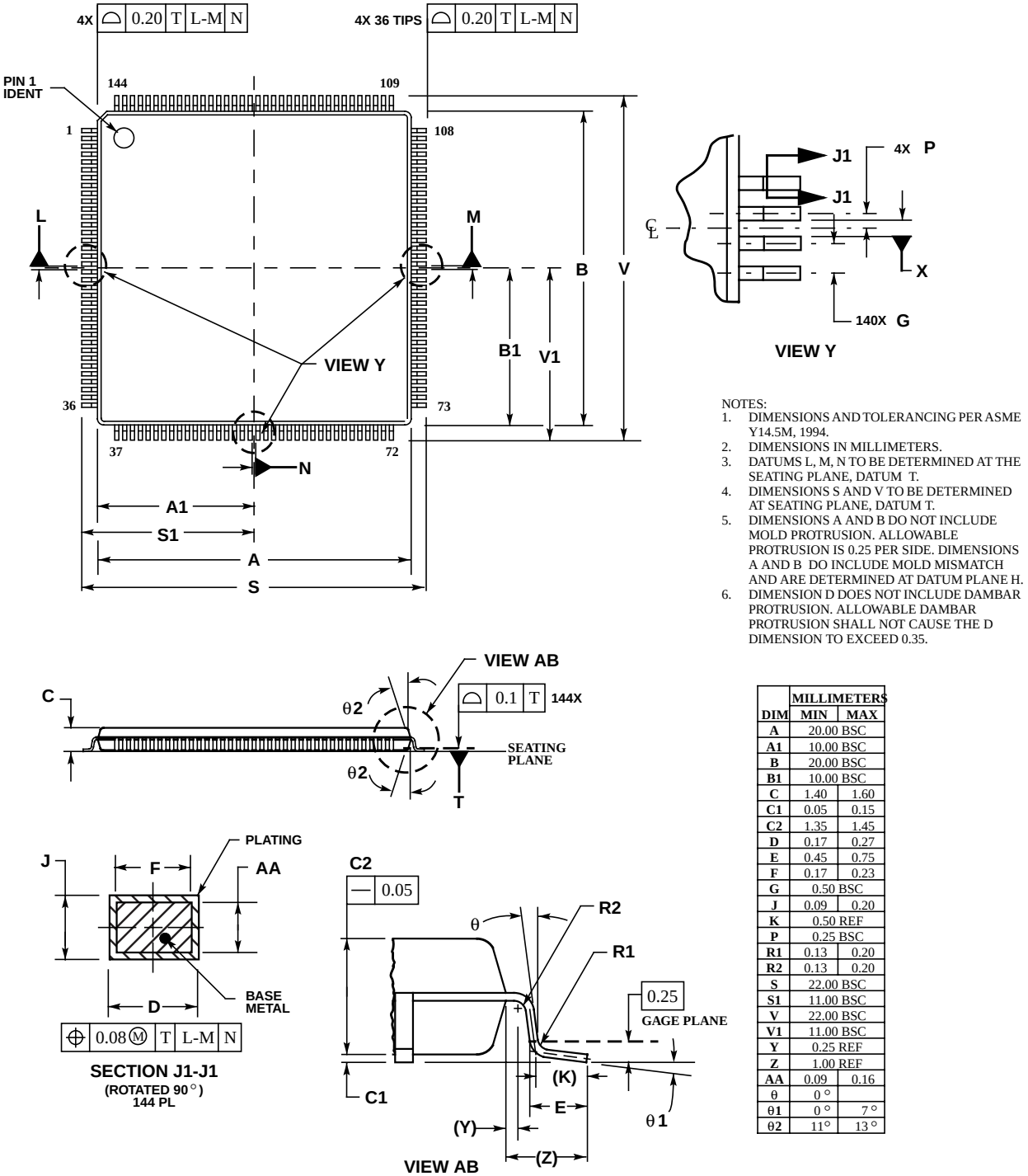
Table A-31. External Tag Trigger Timing V_{DD35} = 5.0 V

No.	C	Characteristic (1)	Symbol	Min	Max	Unit
-	D	Frequency of internal bus	f_i	D.C.	50.0	MHz
1	D	Cycle time	t_{cyc}	20	∞	ns
2	D	$\overline{TAGHI}/\overline{TAGLO}$ setup time	t_{TS}	10	—	ns
3	D	$\overline{TAGHI}/\overline{TAGLO}$ hold time	t_{TH}	0	—	ns

1. Typical supply and silicon, room temperature only

Appendix B
Package Physical Dimension Information.

B.1 144-Pin LQFP Package[1]

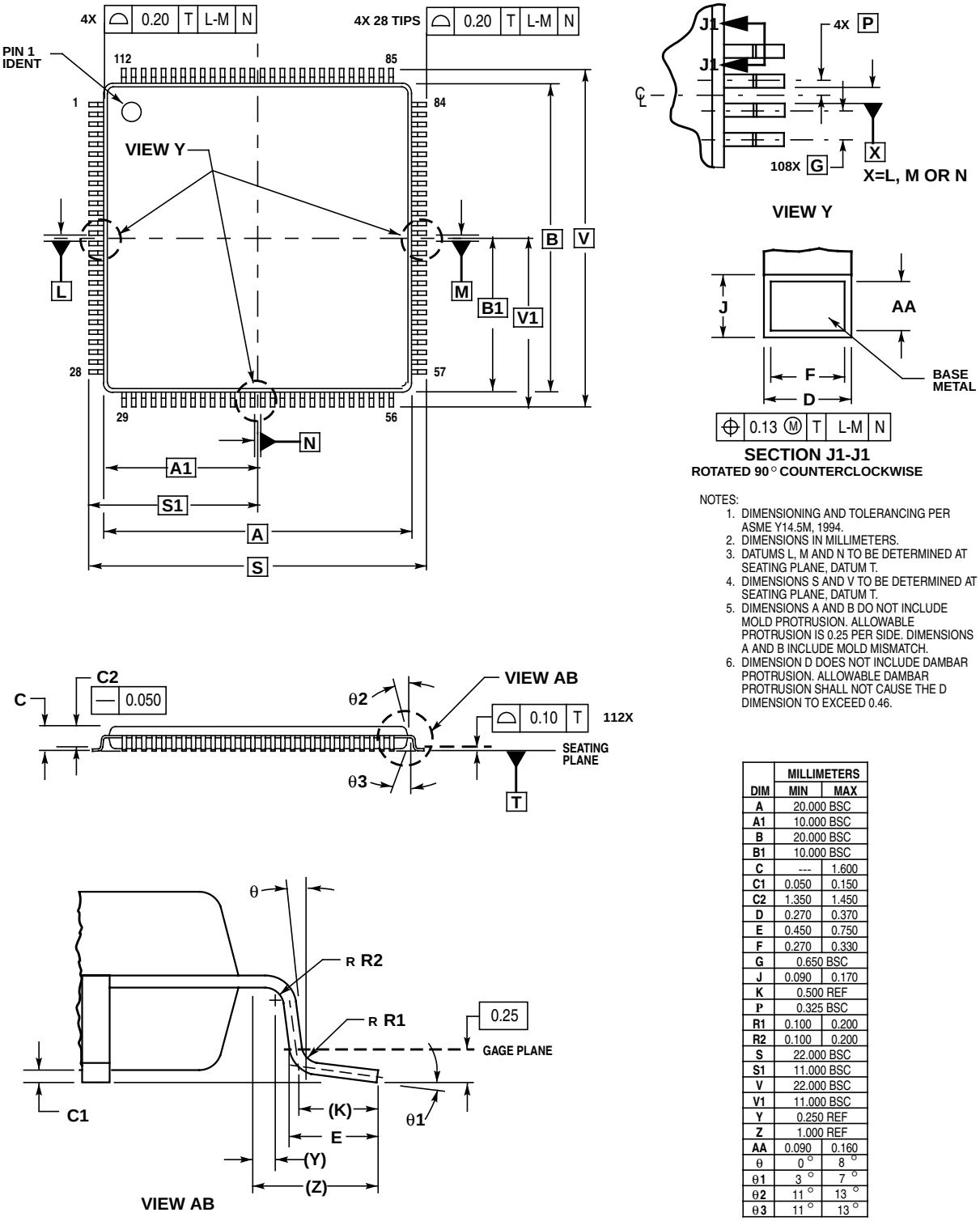


DIM	MILLIMETERS MIN	MAX
A	20.00 BSC	
A1	10.00 BSC	
B	20.00 BSC	
B1	10.00 BSC	
C	1.40	1.60
C1	0.05	0.15
C2	1.35	1.45
D	0.17	0.27
E	0.45	0.75
F	0.17	0.23
G	0.50 BSC	
J	0.09	0.20
K	0.50 REF	
P	0.25 BSC	
R1	0.13	0.20
R2	0.13	0.20
S	22.00 BSC	
S1	11.00 BSC	
V	22.00 BSC	
V1	11.00 BSC	
Y	0.25 REF	
Z	1.00 REF	
AA	0.09	0.16
θ	0 °	
θ1	0 °	7 °
θ2	11°	13 °

1. The 144-Pin LQFP version will not be qualified for production and is intended to be used for emulation (development tools) only.

Figure B-1. 144-Pin LQFP Mechanical Dimensions (Case No. 918-03)

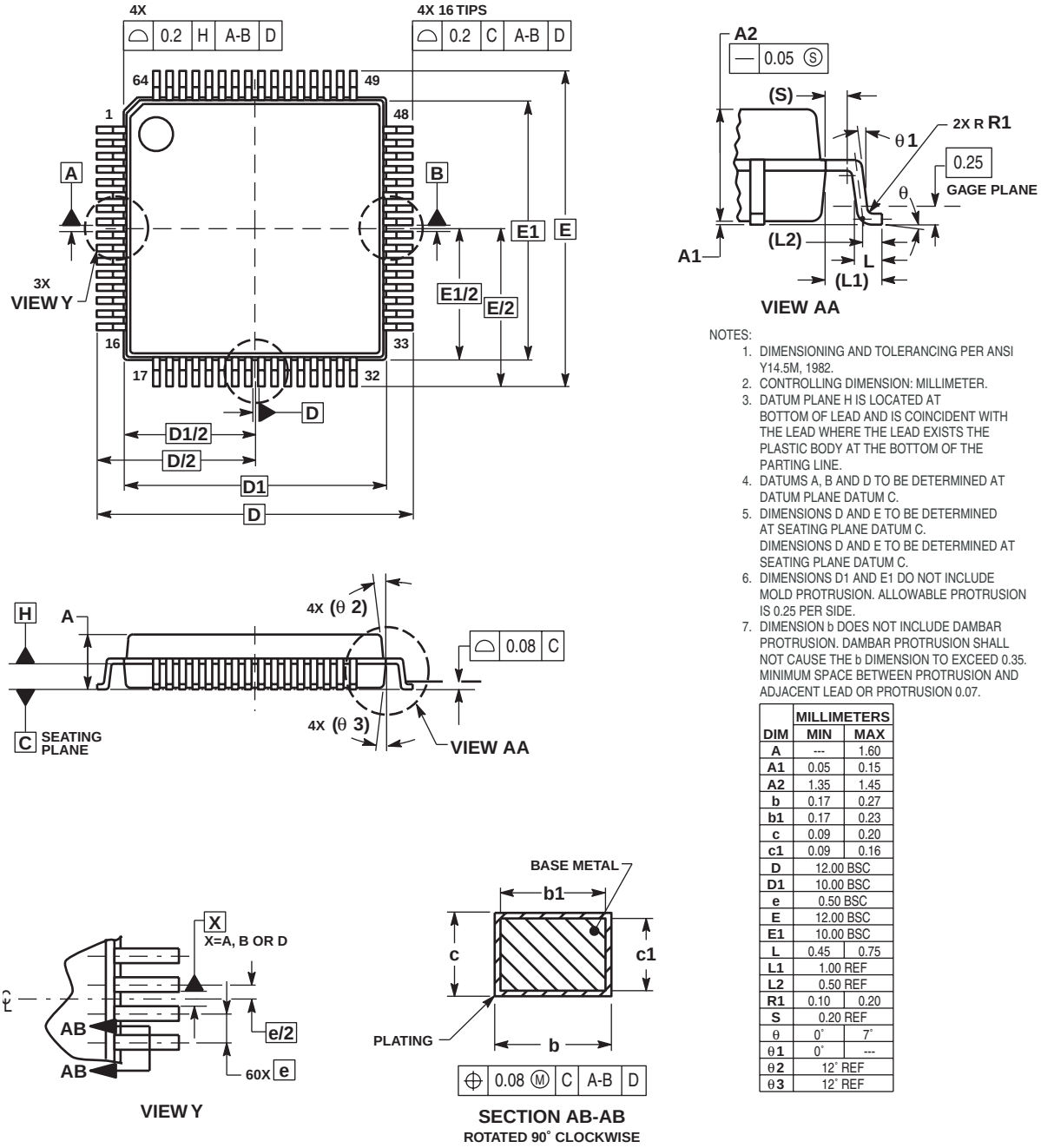
B.2 112-Pin LQFP Package



NOTES:
1. DIMENSIONING AND TOLERANCING PER ASME Y14.5M, 1994.
2. DIMENSIONS IN MILLIMETERS.
3. DATUMS L, M AND N TO BE DETERMINED AT SEATING PLANE, DATUM T.
4. DIMENSIONS S AND V TO BE DETERMINED AT SEATING PLANE, DATUM T.
5. DIMENSIONS A AND B DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE PROTRUSION IS 0.25 PER SIDE. DIMENSIONS A AND B INCLUDE MOLD MISMATCH.
6. DIMENSION D DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL NOT CAUSE THE D DIMENSION TO EXCEED 0.46.

	MILLIMETERS	
DIM	MIN	MAX
A	20.000 BSC	
A1	10.000 BSC	
B	20.000 BSC	
B1	10.000 BSC	
C	---	1.600
C1	0.050	0.150
C2	1.350	1.450
D	0.270	0.370
E	0.450	0.750
F	0.270	0.330
G	0.650 BSC	
J	0.090	0.170
K	0.500 REF	
P	0.325 BSC	
R1	0.100	0.200
R2	0.100	0.200
S	22.000 BSC	
S1	11.000 BSC	
V	22.000 BSC	
V1	11.000 BSC	
Y	0.250 REF	
Z	1.000 REF	
AA	0.090	0.160
θ	0°	8°
θ1	3°	7°
θ2	11°	13°
θ3	11°	13°

Figure B-2. 112-Pin LQFP Mechanical Dimensions (Case No. 987)



NOTES:

1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: MILLIMETER.
3. DATUM PLANE H IS LOCATED AT BOTTOM OF LEAD AND IS COINCIDENT WITH THE LEAD WHERE THE LEAD EXISTS THE PLASTIC BODY AT THE BOTTOM OF THE PARTING LINE.
4. DATUMS A, B AND D TO BE DETERMINED AT DATUM PLANE DATUM C.
5. DIMENSIONS D AND E TO BE DETERMINED AT SEATING PLANE DATUM C. DIMENSIONS D AND E TO BE DETERMINED AT SEATING PLANE DATUM C.
6. DIMENSIONS D1 AND E1 DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE PROTRUSION IS 0.25 PER SIDE.
7. DIMENSION b DOES NOT INCLUDE DAMBAR PROTRUSION. DAMBAR PROTRUSION SHALL NOT CAUSE THE b DIMENSION TO EXCEED 0.35. MINIMUM SPACE BETWEEN PROTRUSION AND ADJACENT LEAD OR PROTRUSION 0.07.

	MILLIMETERS	
DIM	MIN	MAX
A	---	1.60
A1	0.05	0.15
A2	1.35	1.45
b	0.17	0.27
b1	0.17	0.23
c	0.09	0.20
c1	0.09	0.16
D	12.00 BSC	
D1	10.00 BSC	
e	0.50 BSC	
E	12.00 BSC	
E1	10.00 BSC	
L	0.45	0.75
L1	1.00 REF	
L2	0.50 REF	
R1	0.10	0.20
S	0.20 REF	
θ	0°	7°
θ1	0°	---
θ2	12° REF	
θ3	12° REF	

CASE 840F-02
ISSUE B

B.3 64-Pin LQFP Package

Figure B-3. 64-Pin LQFP Mechanical Dimensions (case no. 840F-02)

Appendix C
PCB Layout Guidelines

The PCB must be carefully laid out to ensure proper operation of the voltage regulator as well as of the MCU itself. The following rules must be observed:

- Every supply pair must be decoupled by a ceramic capacitor connected as near as possible to the corresponding pins .
- Central point of the ground star should be the VSS2 pin.
- Use low ohmic low inductance connections between VSS1, VSS2 and VSS3.
- VSSPLL must be directly connected to VSS3.
- Keep traces of VSSPLL, EXTAL, and XTAL as short as possible and occupied board area for C7, C8, and Q1 as small as possible.
- Do not place other signals or supplies underneath area occupied by C7, C8, and Q1 and the connection area to the MCU.
- Central power input should be fed in at the VDDA/VSSA pins.

Example layouts are illustrated on the following pages.

Table C-1. Recommended Decoupling Capacitor Choice

Component	Purpose	Type	Value
C1	V_{DDF} filter capacitor	Ceramic X7R	220 nF
C2	V_{DDX3} filter capacitor (LQFP144 only)	X7R/tantalum	>=100 nF
C3	V_{DDX2} filter capacitor	X7R/tantalum	>=100 nF
C4	V_{DDPLL} filter capacitor	Ceramic X7R	220 nF
C5	OSC load capacitor	From crystal manufacturer	
C6	OSC load capacitor		
C7	V_{DDR} filter capacitor	X7R/tantalum	>=100 nF
C8	V_{DDX4} filter capacitor (LQFP144 only)	X7R/tantalum	>=100 nF
C9	V_{DD} filter capacitor	Ceramic X7R	220 nF
C10	V_{DDA} filter capacitor	Ceramic X7R	>=100 nF
C11	V_{DDX1} filter capacitor	X7R/tantalum	>=100 nF

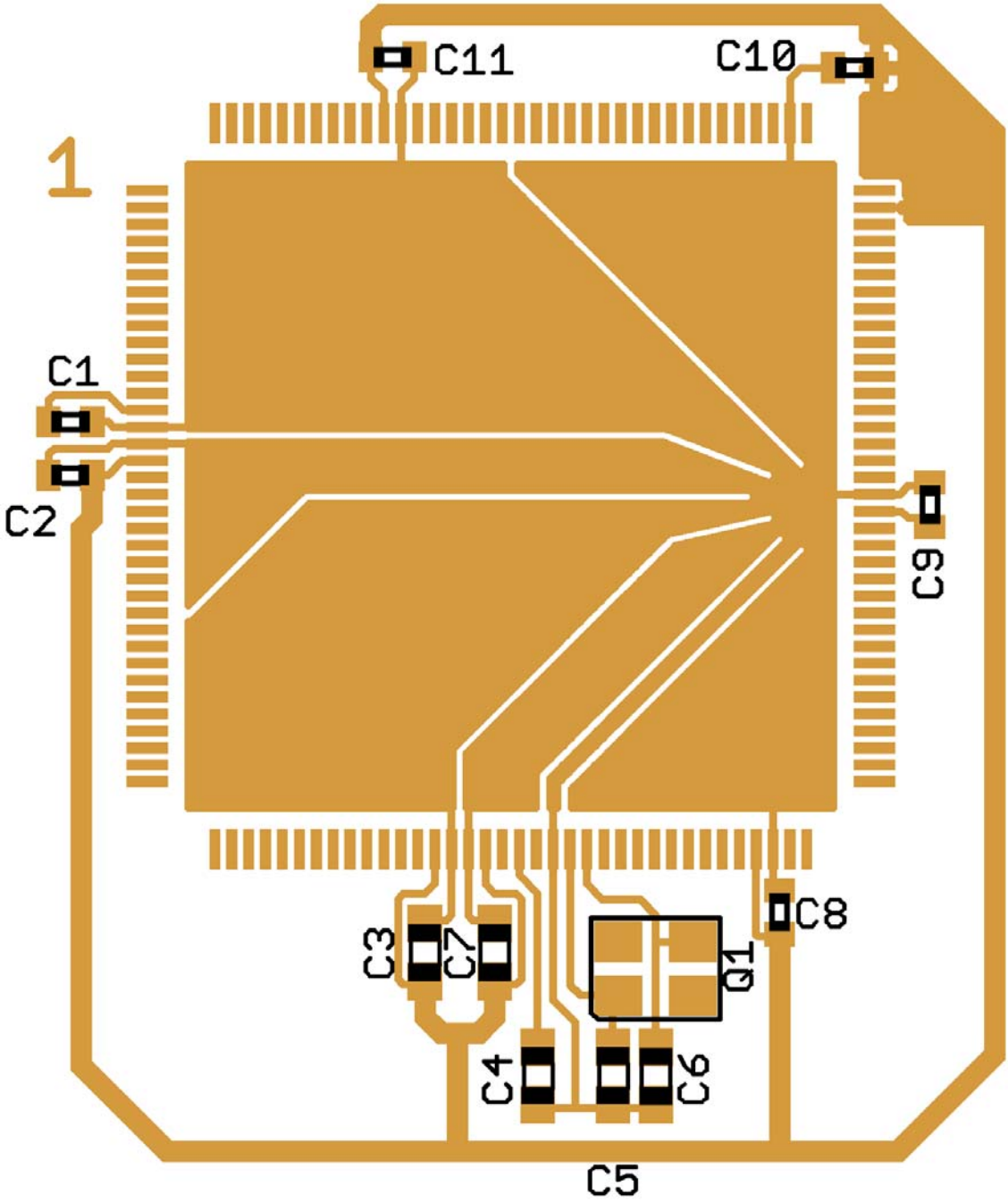


Figure C-1. 144-Pin LQFP Recommended PCB Layout (Loop Controlled Pierce Oscillator)

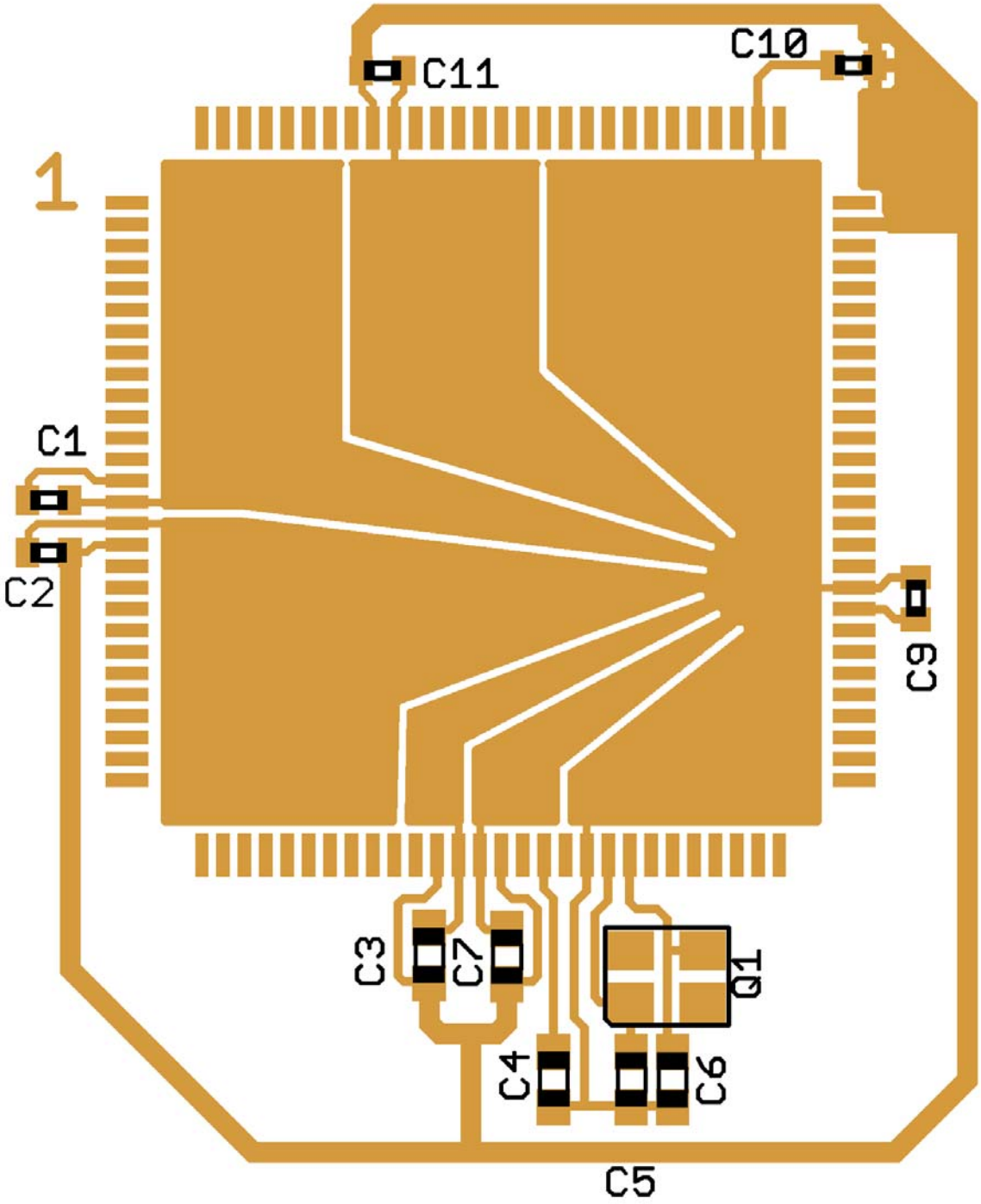


Figure C-2. 112-Pin LQFP Recommended PCB Layout (Loop Controlled Pierce Oscillator)

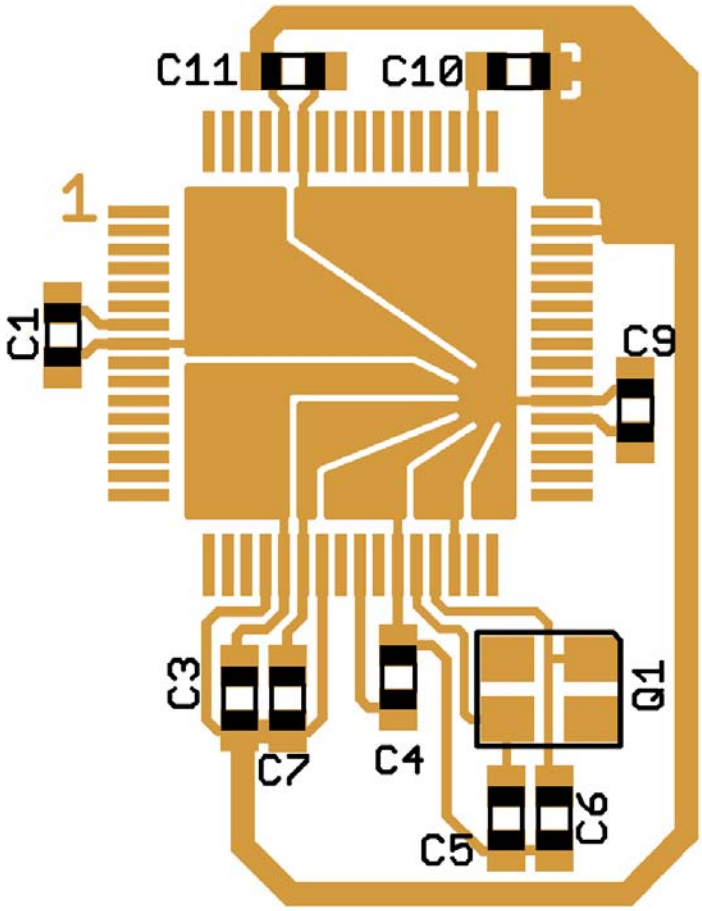


Figure C-3. 64-Pin LQFP Recommended PCB Layout (Loop Controlled Pierce Oscillator)

Appendix D
Derivative Differences

D.1 Memory Sizes and Package Options S12XF - Family

Table D-1. Package and Memory Options of MC9S12XF-Family

Device	Package	Flash	RAM	EEPROM	DATA FLASH
9S12XF512	144 LQFP	512K	32K	4K	32K
	112 LQFP				
	64 LQFP				
9S12XF384	144 LQFP	384K	24K		
	112 LQFP				
	64 LQFP				
9S12XF256	144 LQFP	256K	20K	2K	
	112 LQFP				
	64 LQFP				
9S12XF128	144 LQFP	128K	16K		
	112 LQFP				
	64 LQFP				

D.2 Pinout explanations:

- A/D is the number of modules/total number of A/D channels.
- I/O is the sum of ports capable to act as digital input or output. For details see Table 1-10.

- Versions with 1 CAN module will have CAN0.
- Versions with 2 SPI modules will have SPI0 and SPI1.
- Versions with 1 SPI modules will have SPI0.
- Versions with 2 SCI modules will have SCI0 and SCI1.
- Versions with 1 SCI module will have SCI0.

Appendix E
Detailed Register Address Map

The following tables show the detailed register map of the MC9S12XF512.

0x0000–0x0009 Port Integration Module (PIM) Map 1 of 5

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0000	PORTA	R W	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA 0
0x0001	PORTB	R W	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
0x0002	DDRA	R W	DDRA7	DDRA6	DDRA5	DDRA4	DDRA3	DDRA2	DDRA1	DDRA0
0x0003	DDRB	R W	DDRB7	DDRB6	DDRB5	DDRB4	DDRB3	DDRB2	DDRB1	DDRB0
0x0004	PORTC	R W	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
0x0005	PORTD	R W	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
0x0006	DDRC	R W	DDRC7	DDRC6	DDRC5	DDRC4	DDRC3	DDRC2	DDRC1	DDRC0
0x0007	DDRD	R W	DDRD7	DDRD6	DDRD5	DDRD4	DDRD3	DDRD2	DDRD1	DDRD0
0x0008	PORTE	R W	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
0x0009	DDRE	R W	DDRE7	DDRE6	DDRE5	DDRE4	DDRE3	DDRE2	0	0

0x000A–0x000B Module Mapping Control (S12XMMC) Map 1 of 2

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x000A	MMCCTL0	R W	CS3E1	CS2E1	CS1E1	CS0E1	CS3E0	CS2E0	CS1E0	CS0E0
0x000B	MODE	R W	MODC	MODB	MODA	0	0	0	0	0

0x000C–0x000D Port Integration Module (PIM) Map 2 of 5

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x000C	PUCR	R W	PUPKE	BKPUE	0	PUPEE	PUPDE	PUPCE	PUPBE	PUPAE
0x000D	RDRIV	R W	RDPK	0	0	RDPE	RDPD	RDPC	RDPB	RDPA

0x000E–0x000F External Bus Interface (S12XEBI) Map

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x000E	EBICTL0	R	ITHRS	0	HDBE	ASIZ4	ASIZ3	ASIZ2	ASIZ1	ASIZ0
		W								
0x000F	EBICTL1	R	0	EXSTR12	EXSTR11	EXSTR10	0	EXSTR02	EXSTR01	EXSTR00
		W								

0x0010–0x0017 Module Mapping Control (S12XMMC) Map 2 of 2

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0010	GPAGE	R	0	GP6	GP5	GP4	GP3	GP2	GP1	GP0
		W								
0x0011	DIRECT	R	DP15	DP14	DP13	DP12	DP11	DP10	DP9	DP8
		W								
0x0012	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0013	MMCCTL1	R	TGMRAMON	MGROMON	EEEIFRON	PGMIFRON	RAMHM	EROMON	ROMHM	ROMON
		W								
0x0014	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0015	PPAGE	R	PIX7	PIX6	PIX5	PIX4	PIX3	PIX2	PIX1	PIX0
		W								
0x0016	RPAGE	R	RP7	RP6	RP5	RP4	RP3	RP2	RP1	RP0
		W								
0x0017	EPAGE	R	EP7	EP6	EP5	EP4	EP3	EP2	EP1	EP0
		W								

0x0018–0x001B Miscellaneous Peripheral

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0018	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0019	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x001A	PARTIDH[1]	R	1	1	0	1	0	1	0	0
		W								
0x001B	PARTIDL	R	1	0	0	0	0	0	0	0
		W								

1. Part ID of initial mask set. For the latest Part ID assignment see 1.1.6 Part ID Assignment.

0x001C–0x001D Port Integration Module (PIM) Map 3 of 5

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x001C	ECLKCTL	R	NECLK	NCLKX2	DIV16	EDIV4	EDIV3	EDIV2	EDIV1	EDIV0
		W								
0x001D	Reserved	R	0	0	0	0	0	0	0	0
		W								

0x001E–0x001F Port Integration Module (PIM) Map 3 of 5

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x001E	IRQCR	R	IRQE	IRQEN	0	0	0	0	0	0
		W								
0x001F	Reserved	R	0	0	0	0	0	0	0	0
		W								

0x0020–0x002F Debug Module (S12XDBG) Map

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0020	DBGC1	R	ARM	0	XGSBPE	BDM	DBGBRK		COMRV	
		W		TRIG						
0x0021	DBGSR	R	TBF	EXTF	0	0	0	SSF2	SSF1	SSF0
		W								
0x0022	DBGTCR	R	TSOURCE		TRANGE		TRCMOD		TALIGN	
		W								
0x0023	DBGC2	R	0	0	0	0	CDCM		ABCM	
		W								
0x0024	DBGTBH	R	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8
		W								
0x0025	DBGTBL	R	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
		W								
0x0026	DBGCNT	R	0	CNT						
		W								
0x0027	DBGSCRX	R	0	0	0	0	SC3	SC2	SC1	SC0
		W								
0x0027	DBGMFR	R	0	0	0	0	MC3	MC2	MC1	MC0
		W								
0x0028 (1)	DBGXCTL (COMPA/C)	R	0	NDB	TAG	BRK	RW	RWE	SRC	COMPE
		W								
0x0028 (2)	DBGXCTL (COMPB/D)	R	SZE	SZ	TAG	BRK	RW	RWE	SRC	COMPE
		W								
0x0029	DBGXAH	R	0	Bit 22	21	20	19	18	17	Bit 16
		W								
0x002A	DBGXAM	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x002B	DBGXAL	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x002C	DBGXDH	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x002D	DBGXDL	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x002E	DBGXDHM	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x002F	DBGXDLM	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								

1. This represents the contents if the Comparator A or C control register is blended into this address
2. This represents the contents if the Comparator B or D control register is blended into this address

0x0030–0x0031 Reserved Register Space

0x0030	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0031	Reserved	R	0	0	0	0	0	0	0	0
		W								

0x0032–0x0033 Port Integration Module (PIM) Map 4 of 5

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0032	PORTK	R W	PK7	PK6	PK5	PK4	PK3	PK2	PK1	PK0
0x0033	DDRK	R W	DDRK7	DDRK6	DDRK5	DDRK4	DDRK3	DDRK2	DDRK1	DDRK0

0x0034–0x003F Clock and Reset Generator (CRG) Map

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0034	SYNR	R W	VCOFRQ[1:0]		SYN5	SYN4	SYN3	SYN2	SYN1	SYN0
0x0035	REFDV	R W	REFFRQ[1:0]		REFDV5	REFDV4	REFDV3	REFDV2	REFDV1	REFDV0
0x0036	POSTDIV	R W	0	0	0	POSTDIV[4:0]]				
0x0037	CRGFLG	R W	RTIF	PORF	LVRF	LOCKIF	LOCK	ILAF	SCMIF	SCM
0x0038	CRGINT	R W	RTIE	0	0	LOCKIE	0	0	SCMIE	0
0x0039	CLKSEL	R W	PLLSEL	PSTP	XCLKS	0	PLLWAI	0	RTIWAI	COPWAI
0x003A	PLLCTL	R W	CME	PLLON	FME		FSTWKP	PRE	PCE	SCME
0x003B	RTICTL	R W	RTDEC	RTR6	RTR5	RTR4	RTR3	RTR2	RTR1	RTR0
0x003C	COPCTL	R W	WCOP	RSBCK	0 WRTMASK	0	0	CR2	CR1	CR0
0x003D	FORBYP	R	0	0	0	0	0	0	0	0
		W	Reserved For Factory Test							
0x003E	CTCTL	R	0	0	0	0		0	0	0
		W	Reserved For Factory Test							
0x003F	ARMCOP	R	0	0	0	0	0	0	0	0
		W	Bit 7	6	5	4	3	2	1	Bit 0

0x0040–0x007F Enhanced Capture Timer 16-Bit 8-Channels (ECT) Map (Sheet 1 of 3)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0040	TIOS	R	IOS7	IOS6	IOS5	IOS4	IOS3	IOS2	IOS1	IOS0
		W								
0x0041	CFORC	R	0	0	0	0	0	0	0	0
		W	FOC7	FOC6	FOC5	FOC4	FOC3	FOC2	FOC1	FOC0
0x0042	OC7M	R	OC7M7	OC7M6	OC7M5	OC7M4	OC7M3	OC7M2	OC7M1	OC7M0
		W								
0x0043	OC7D	R	OC7D7	OC7D6	OC7D5	OC7D4	OC7D3	OC7D2	OC7D1	OC7D0
		W								
0x0044	TCNT (hi)	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x0045	TCNT (lo)	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x0046	TSCR1	R	TEN	TSWAI	TSFRZ	TFFCA	PRNT	0	0	0
		W								
0x0047	TTOV	R	TOV7	TOV6	TOV5	TOV4	TOV3	TOV2	TOV1	TOV0
		W								
0x0048	TCTL1	R	OM7	OL7	OM6	OL6	OM5	OL5	OM4	OL4
		W								
0x0049	TCTL2	R	OM3	OL3	OM2	OL2	OM1	OL1	OM0	OL0
		W								
0x004A	TCTL3	R	EDG7B	EDG7A	EDG6B	EDG6A	EDG5B	EDG5A	EDG4B	EDG4A
		W								
0x004B	TCTL4	R	EDG3B	EDG3A	EDG2B	EDG2A	EDG1B	EDG1A	EDG0B	EDG0A
		W								
0x004C	TIE	R	C7I	C6I	C5I	C4I	C3I	C2I	C1I	C0I
		W								
0x004D	TSCR2	R	TOI	0	0	0	TCRE	PR2	PR1	PR0
		W								
0x004E	TFLG1	R	C7F	C6F	C5F	C4F	C3F	C2F	C1F	C0F
		W								
0x004F	TFLG2	R	TOF	0	0	0	0	0	0	0
		W								
0x0050	TC0 (hi)	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x0051	TC0 (lo)	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x0052	TC1 (hi)	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x0053	TC1 (lo)	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x0054	TC2 (hi)	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x0055	TC2 (lo)	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								

0x0040–0x007F Enhanced Capture Timer 16-Bit 8-Channels (ECT) Map (Sheet 2 of 3)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0056	TC3 (hi)	R W	Bit 15	14	13	12	11	10	9	Bit 8
0x0057	TC3 (lo)	R W	Bit 7	6	5	4	3	2	1	Bit 0
0x0058	TC4 (hi)	R W	Bit 15	14	13	12	11	10	9	Bit 8
0x0059	TC4 (lo)	R W	Bit 7	6	5	4	3	2	1	Bit 0
0x005A	TC5 (hi)	R W	Bit 15	14	13	12	11	10	9	Bit 8
0x005B	TC5 (lo)	R W	Bit 7	6	5	4	3	2	1	Bit 0
0x005C	TC6 (hi)	R W	Bit 15	14	13	12	11	10	9	Bit 8
0x005D	TC6 (lo)	R W	Bit 7	6	5	4	3	2	1	Bit 0
0x005E	TC7 (hi)	R W	Bit 15	14	13	12	11	10	9	Bit 8
0x005F	TC7 (lo)	R W	Bit 7	6	5	4	3	2	1	Bit 0
0x0060	PACTL	R W	0	PAEN	PAMOD	PEDGE	CLK1	CLK0	PAOVI	PAI
0x0061	PAFLG	R W	0	0	0	0	0	0	PAOVF	PAIF
0x0062	PACN3 (hi)	R W	Bit 7	6	5	4	3	2	1	Bit 0
0x0063	PACN2 (lo)	R W	Bit 7	6	5	4	3	2	1	Bit 0
0x0064	PACN1 (hi)	R W	Bit 7	6	5	4	3	2	1	Bit 0
0x0065	PACN0 (lo)	R W	Bit 7	6	5	4	3	2	1	Bit 0
0x0066	MCCTL	R W	MCZI	MODMC	RDMCL	0 ICLAT	0 FLMC	MCEN	MCPR1	MCPR0
0x0067	MCFLG	R W	MCZF	0	0	0	POLF3	POLF2	POLF1	POLF0
0x0068	ICPAR	R W	0	0	0	0	PA3EN	PA2EN	PA1EN	PA0EN
0x0069	DLYCT	R W	DLY7	DLY6	DLY5	DLY4	DLY3	DLY2	DLY1	DLY0
0x006A	ICOVW	R W	NOVW7	NOVW6	NOVW5	NOVW4	NOVW3	NOVW2	NOVW1	NOVW0
0x006B	ICSYS	R W	SH37	SH26	SH15	SH04	TFMOD	PACMX	BUFEN	LATQ
0x006C	OCPD	R W	OCPD7	OCPD6	OCPD5	OCPD4	OCPD3	OCPD2	OCPD1	OCPD0

0x0040–0x007F Enhanced Capture Timer 16-Bit 8-Channels (ECT) Map (Sheet 3 of 3)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x006D	TIMTST	R	0	0	0	0	0	0	0	0
		W	Reserved For Factory Test							
0x006E	PTPSR	R	PTPS7	PTPS6	PTPS5	PTPS4	PTPS3	PTPS2	PTPS1	PTPS0
		W								
0x006F	PTMCPSR	R	PTMPS7	PTMPS6	PTMPS5	PTMPS4	PTMPS3	PTMPS2	PTMPS1	PTMPS0
		W								
0x0070	PBCTL	R	0	PBEN	0	0	0	0	PBOVI	0
		W								
0x0071	PBFLG	R	0	0	0	0	0	0	PBOVF	0
		W								
0x0072	PA3H	R	PA3H7	PA3H6	PA3H5	PA3H4	PA3H3	PA3H2	PA3H1	PA3H0
		W								
0x0073	PA2H	R	PA2H7	PA2H6	PA2H5	PA2H4	PA2H3	PA2H2	PA2H1	PA2H0
		W								
0x0074	PA1H	R	PA1H7	PA1H6	PA1H5	PA1H4	PA1H3	PA1H2	PA1H1	PA1H 0
		W								
0x0075	PA0H	R	PA0H7	PA0H6	PA0H5	PA0H4	PA0H3	PA0H2	PA0H1	PA0H0
		W								
0x0076	MCCNT (hi)	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x0077	MCCNT (lo)	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x0078	TC0H (hi)	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x0079	TC0H (lo)	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x007A	TC1H (hi)	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x007B	TC1H (lo)	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x007C	TC2H (hi)	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x007D	TC2H (lo)	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								
0x007E	TC3H (hi)	R	Bit 15	14	13	12	11	10	9	Bit 8
		W								
0x007F	TC3H (lo)	R	Bit 7	6	5	4	3	2	1	Bit 0
		W								

0x0080–0x00AF Analog-to-Digital Converter 12-bit 16-Channels (ATD) Map (Sheet 1 of 3)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0080	ATDCTL0	R	0	0	0	0	WRAP3	WRAP2	WRAP1	WRAP0
		W								
0x0081	ATDCTL1	R	ETRIG SEL	SRES1	SRES0	SMP_DIS	ETRIG CH3	ETRIG CH2	ETRIG CH1	ETRIG CH0
		W								
0x0082	ATDCTL2	R	0	AFFC	ICLKSTP	ETRIGLE	ETRIGP	ETRIGE	ASCIE	ACMPIE
		W								
0x0083	ATDCTL3	R	DJM	S8C	S4C	S2C	S1C	FIFO	FRZ1	FRZ0
		W								
0x0084	ATDCTL4	R	SMP2	SMP1	SMP0	PRS4	PRS3	PRS2	PRS1	PRS0
		W								
0x0085	ATDCTL5	R	0	SC	SCAN	MULT	CD	CC	CB	CA
		W								
0x0086	ATDSTAT0	R	SCF	0	ETORF	FIFOR	CC3	CC2	CC1	CC0
		W								
0x0087	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0088	ATDCMPEH	R	CMPE15	CMPE14	CMPE13	CMPE12	CMPE11	CMPE10	CMPE9	CMPE8
		W								
0x0089	ATDCMPEL	R	CMPE7	CMPE6	CMPE5	CMPE4	CMPE3	CMPE2	CMPE1	CMPE0
		W								
0x008A	ATDSTAT2H	R	CCF15	CCF14	CCF13	CCF12	CCF11	CCF10	CCF9	CCF8
		W								
0x008B	ATDSTATL	R	CCF7	CCF6	CCF5	CCF4	CCF3	CCF2	CCF1	CCF0
		W								
0x008C	ATDDIENH	R	IEN15	IEN14	IEN13	IEN12	IEN11	IEN10	IEN9	IEN8
		W								
0x008D	ATDDIENL	R	IEN7	IEN6	IEN5	IEN4	IEN3	IEN2	IEN1	IEN0
		W								
0x008E	ATDCMPHTH	R	CMPHT15	CMPHT14	CMPHT13	CMPHT12	CMPHT11	CMPHT10	CMPHT9	CMPHT8
		W								
0x008F	ATDCMPHTL	R	CMPHT7	CMPHT6	CMPHT5	CMPHT4	CMPHT3	CMPHT2	CMPHT1	CMPHT0
		W								
0x0090	ATDDR0H	R	Bit15	14	13	12	11	10	9	Bit8
		W								
0x0091	ATDDR0L	R	Bit7	Bit6	0	0	0	0	0	0
		W								
0x0092	ATDDR1H	R	Bit15	14	13	12	11	10	9	Bit8
		W								
0x0093	ATD1DR1L	R	Bit7	Bit6	0	0	0	0	0	0
		W								
0x0094	ATDDR2H	R	Bit15	14	13	12	11	10	9	Bit8
		W								
0x0095	ATDDR2L	R	Bit7	Bit6	0	0	0	0	0	0
		W								
0x0096	ATDDR3H	R	Bit15	14	13	12	11	10	9	Bit8
		W								

0x0080–0x00AF Analog-to-Digital Converter 12-bit 16-Channels (ATD) Map (Sheet 2 of 3)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0097	ATDDR3L	R	Bit7	Bit6	0	0	0	0	0	0
		W								
0x0098	ATDDR4H	R	Bit15	14	13	12	11	10	9	Bit8
		W								
0x0099	ATDDR4L	R	Bit7	Bit6	0	0	0	0	0	0
		W								
0x009A	ATDDR5H	R	Bit15	14	13	12	11	10	9	Bit8
		W								
0x009B	ATDDR5L	R	Bit7	Bit6	0	0	0	0	0	0
		W								
0x009C	ATDDR6H	R	Bit15	14	13	12	11	10	9	Bit8
		W								
0x009D	ATDDR6L	R	Bit7	Bit6	0	0	0	0	0	0
		W								
0x009E	ATDDR7H	R	Bit15	14	13	12	11	10	9	Bit8
		W								
0x009F	ATDDR7L	R	Bit7	Bit6	0	0	0	0	0	0
		W								
0x00A0	ATDDR8H	R	Bit15	14	13	12	11	10	9	Bit8
		W								
0x00A1	ATDDR8L	R	Bit7	Bit6	0	0	0	0	0	0
		W								
0x00A2	ATDDR9H	R	Bit15	14	13	12	11	10	9	Bit8
		W								
0x00A3	ATDDR9L	R	Bit7	Bit6	0	0	0	0	0	0
		W								
0x00A4	ATDDR10H	R	Bit15	14	13	12	11	10	9	Bit8
		W								
0x00A5	ATDDR10L	R	Bit7	Bit6	0	0	0	0	0	0
		W								
0x00A6	ATDDR11H	R	Bit15	14	13	12	11	10	9	Bit8
		W								
0x00A7	ATDDR11L	R	Bit7	Bit6	0	0	0	0	0	0
		W								
0x00A8	ATDDR12H	R	Bit15	14	13	12	11	10	9	Bit8
		W								
0x00A9	ATDDR12L	R	Bit7	Bit6	0	0	0	0	0	0
		W								
0x00AA	ATDDR13H	R	Bit15	14	13	12	11	10	9	Bit8
		W								
0x00AB	ATDDR13L	R	Bit7	Bit6	0	0	0	0	0	0
		W								
0x00AC	ATDDR14H	R	Bit15	14	13	12	11	10	9	Bit8
		W								

0x0080–0x00AF Analog-to-Digital Converter 12-bit 16-Channels (ATD) Map (Sheet 3 of 3)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x00AD	ATDDR14L	R	Bit7	Bit6	0	0	0	0	0	0
		W								
0x00AE	ATDDR15H	R	Bit15	14	13	12	11	10	9	Bit8
		W								
0x00AF	ATDDR15L	R	Bit7	Bit6	0	0	0	0	0	0
		W								

0x00B0–0x00B7 Reserved Register Space

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x00B0 - 0x00C7	Reserved	R	0	0	0	0	0	0	0	0
		W								

0x00C8–0x00CF Asynchronous Serial Interface (SCI0) Map

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x00C8	SCI0BDH(1)	R	IREN	TNP1	TNP0	SBR12	SBR11	SBR10	SBR9	SBR8
		W								
0x00C9	SCI0BDL1	R	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
		W								
0x00CA	SCI0CR11	R	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
		W								
0x00C8	SCI0ASR1(2)	R	RXEDGIF	0	0	0	0	BERRV	BERRIF	BKDIF
		W								
0x00C9	SCI0ACR2	R	RXEDGIE	0	0	0	0	0	BERRIE	BKDIE
		W								
0x00CA	SCI0ACR22	R	0	0	0	0	0	BERRM1	BERRM0	BKDFE
		W								
0x00CB	SCI0CR2	R	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
		W								
0x00CC	SCI0SR1	R	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
		W								
0x00CD	SCI0SR2	R	AMAP	0	0	TXPOL	RXPOL	BRK13	TXDIR	RAF
		W								
0x00CE	SCI0DRH	R	R8	T8	0	0	0	0	0	0
		W								
0x00CF	SCI0DRL	R	R7	R6	R5	R4	R3	R2	R1	R0
		W	T7	T6	T5	T4	T3	T2	T1	T0

1. Those registers are accessible if the AMAP bit in the SCI0SR2 register is set to zero
2. Those registers are accessible if the AMAP bit in the SCI0SR2 register is set to one

0x00D0–0x00D7 Asynchronous Serial Interface (SCI1) Map

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x00D0	SCI1BDH(1)	R	IREN	TNP1	TNP0	SBR12	SBR11	SBR10	SBR9	SBR8
		W								
0x00D1	SCI1BDL1	R	SBR7	SBR6	SBR5	SBR4	SBR3	SBR2	SBR1	SBR0
		W								
0x00D2	SCI1CR11	R	LOOPS	SCISWAI	RSRC	M	WAKE	ILT	PE	PT
		W								
0x00D0	SCI1ASR1(2)	R	RXEDGIF	0	0	0	0	BERRV	BERRIF	BKDIF
		W								
0x00D1	SCI1ACR12	R	RXEDGIE	0	0	0	0	0	BERRIE	BKDIE
		W								
0x00D2	SCI1ACR22	R	0	0	0	0	0	BERRM1	BERRM0	BKDFE
		W								
0x00D3	SCI1CR2	R	TIE	TCIE	RIE	ILIE	TE	RE	RWU	SBK
		W								
0x00D4	SCI1SR1	R	TDRE	TC	RDRF	IDLE	OR	NF	FE	PF
		W								

0x00D0–0x00D7 Asynchronous Serial Interface (SCI1) Map (continued)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x00D5	SCI1SR2	R	AMAP	0	0	TXPOL	RXPOL	BRK13	TXDIR	RAF
		W								
0x00D6	SCI1DRH	R	R8	T8	0	0	0	0	0	0
		W								
0x00D7	SCI1DRL	R	R7	R6	R5	R4	R3	R2	R1	R0
		W	T7	T6	T5	T4	T3	T2	T1	T0

1. Those registers are accessible if the AMAP bit in the SCI1SR2 register is set to zero
2. Those registers are accessible if the AMAP bit in the SCI1SR2 register is set to one

0x00D8–0x00DF Serial Peripheral Interface (SPI0) Map

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x00D8	SPI0CR1	R	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
		W								
0x00D9	SPI0CR2	R	0	XFRW	0	MODFEN	BIDIROE	0	SPISWAI	SPC0
		W								
0x00DA	SPI0BR	R	0	SPPR2	SPPR1	SPPR0	0	SPR2	SPR1	SPR0
		W								
0x00DB	SPI0SR	R	SPIF	0	SPTEF	MODF	0	0	0	0
		W								
0x00DC	SPI0DRH	R	R15	R14	R13	R12	R11	R10	R9	R8
		W	T15	T14	T13	T12	T11	T10	T9	T8
0x00DD	SPI0DRL	R	R7	R6	R5	R4	R3	R2	R1	R0
		W	T7	T6	T5	T4	T3	T2	T1	T0
0x00DE	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x00DF	Reserved	R	0	0	0	0	0	0	0	0
		W								

0x00E0–0x00EF Reserved Register Space

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x00E0 - 0x00EF	Reserved	R	0	0	0	0	0	0	0	0
		W								

0x00F0–0x00F7 Serial Peripheral Interface (SPI1) Map

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x00F0	SPI1CR1	R	SPIE	SPE	SPTIE	MSTR	CPOL	CPHA	SSOE	LSBFE
		W								
0x00F1	SPI1CR2	R	0	XFRW	0	MODFEN	BIDIROE	0	SPISWAI	SPC0
		W								
0x00F2	SPI1BR	R	0	SPPR2	SPPR1	SPPR0	0	SPR2	SPR1	SPR0
		W								
0x00F3	SPI1SR	R	SPIF	0	SPTEF	MODF	0	0	0	0
		W								

0x00F0–0x00F7 Serial Peripheral Interface (SPI1) Map

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x00F4	SPI1DRH	R	R15	R14	R13	R12	R11	R10	R9	R8
		W	T15	T14	T13	T12	T11	T10	T9	T8
0x00F5	SPI1DRL	R	R7	R6	R5	R4	R3	R2	R1	R0
		W	T7	T6	T5	T4	T3	T2	T1	T0
0x00F6	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x00F7	Reserved	R	0	0	0	0	0	0	0	0
		W								

0x00F8–0x00FF Reserved Register Space

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x00F8 - 0x00FF	Reserved	R	0	0	0	0	0	0	0	0
		W								

0x0100–0x0113 NVM Control Register (FTM) Map

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0100	FCLKDIV	R	FDIVLD	FDIV6	FDIV5	FDIV4	FDIV3	FDIV2	FDIV1	FDIV0
		W								
0x0101	FSEC	R	KEYEN1	KEYEN0	RNV5	RNV4	RNV3	RNV2	SEC1	SEC0
		W								
0x0102	FCCOBIX	R	0	0	0	0	0	CCOBIX2	CCOBIX1	CCOBIX0
		W								
0x0103	FECCRIX	R	0	0	0	0	0	ECCRIX2	ECCRIX1	ECCRIX0
		W								
0x0104	FCNFG	R	CCIE	0	0	IGNSF	0	0	FDFD	FSFD
		W								
0x0105	FERCNFG	R	ERSERIE	PGMERIE	0	EPVIOLIE	ERSVIE1	ERSVIE0	DFDIE	SFDIE
		W								
0x0106	FSTAT	R	CCIF	0	ACCERR	FPVIOL	MGBUSY	RSVD	MGSTAT1	MGSTAT0
		W								
0x0107	FERSTAT	R	ERSERIF	PGMERIF	0	EPVIOLIF	ERSVIF1	ERSVIF0	DFDIF	SFDIF
		W								
0x0108	FPROT	R	FPOPEN	RNV6	FPHDIS	FPHS1	FPHS0	FPLDIS	FPLS1	FPLS0
		W								
0x0109	EPROT	R	EPOPEN	RNV6	RNV5	RNV4	EPDIS	EPS2	EPS1	EPS0
		W								
0x010A	FCCOBHI	R	CCOB15	CCOB14	CCOB13	CCOB12	CCOB11	CCOB10	CCOB9	CCOB8
		W								
0x010B	FCCOBLO	R	CCOB7	CCOB6	CCOB5	CCOB4	CCOB3	CCOB2	CCOB1	CCOB0
		W								
0x010C	ETAGHI	R	ETAG15	ETAG14	ETAG13	ETAG12	ETAG11	ETAG10	ETAG9	ETAG8
		W								
0x010D	ETAGLO	R	ETAG7	ETAG6	ETAG5	ETAG4	ETAG3	ETAG2	ETAG1	ETAG0
		W								

0x0100–0x0113 NVM Control Register (FTM) Map (continued)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x010E	FECCRHI	R	ECCR15	ECCR14	ECCR13	ECCR12	ECCR11	ECCR10	ECCR9	ECCR8
		W								
0x010F	FECCRLO	R	ECCR7	ECCR6	ECCR5	ECCR4	ECCR3	ECCR2	ECCR1	ECCR0
		W								
0x0110	FOPT	R	NV7	NV6	NV5	NV4	NV3	NV2	NV1	NV0
		W								
0x0111	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0112	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0113	Reserved	R	0	0	0	0	0	0	0	0
		W								

0x0114–0x011F Memory Protection Unit (MPU) Map

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0114	MPUFLG	R	AEF	WPF	NEXF	0	0	0	0	SVSF
		W								
0x0115	MPUASTAT0	R	0	ADDR[22:16]						
		W								
0x0116	MPUASTAT1	R	ADDR[15:8]							
		W								
0x0117	MPUASTAT2	R	ADDR[7:0]							
		W								
0x0118	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0119	MPUSEL	R	SVSEN	0	0	0	0	SEL[2:0]		
		W								
0x011A	MPUDESC0(1)	R	MSTR0	MSTR1	MSTR2	MSTR3	LOW_ADDR[22:19]			
		W								
0x011B	MPUDESC1[1]	R	LOW_ADDR[18:11]							
		W								
0x011C	MPUDESC2[1]	R	LOW_ADDR[10:3]							
		W								
0x011D	MPUDESC3[1]	R	WP	NEX	0	0	HIGH_ADDR[22:19]			
		W								
0x011E	MPUDESC4[1]	R	HIGH_ADDR[18:11]							
		W								
0x011F	MPUDESC5[1]	R	HIGH_ADDR[10:3]							
		W								

1. The module addresses 0x03C6–0x03CB represent a window in the register map through which different descriptor registers are visible.

0x0120–0x012F Interrupt Module (S12XINT) Map

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0120	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0121	IVBR	R	IVB_ADDR[7:0]							
		W								
0x0122	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0123	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0124	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0125	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0126	INT_XGPRIO	R	0	0	0	0	0	XILVL[2:0]		
		W								
0x0127	INT_CFADDR	R	INT_CFADDR[7:4]				0	0	0	0
		W								

0x0120–0x012F Interrupt Module (S12XINT) Map

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0128	INT_CFDATA0	R	RQST	0	0	0	0	PRIOLVL[2:0]		
		W								
0x0129	INT_CFDATA1	R	RQST	0	0	0	0	PRIOLVL[2:0]		
		W								
0x012A	INT_CFDATA2	R	RQST	0	0	0	0	PRIOLVL[2:0]		
		W								
0x012B	INT_CFDATA3	R	RQST	0	0	0	0	PRIOLVL[2:0]		
		W								
0x012C	INT_CFDATA4	R	RQST	0	0	0	0	PRIOLVL[2:0]		
		W								
0x012D	INT_CFDATA5	R	RQST	0	0	0	0	PRIOLVL[2:0]		
		W								
0x012E	INT_CFDATA6	R	RQST	0	0	0	0	PRIOLVL[2:0]		
		W								
0x012F	INT_CFDATA7	R	RQST	0	0	0	0	PRIOLVL[2:0]		
		W								

0x00130–0x013F Reserved Register Space

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0130 - 0x013F	Reserved	R	0	0	0	0	0	0	0	0
		W								

0x0140–0x017F MSCAN (CAN0) Map

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0140	CAN0CTL0	R	RXFRM	RXACT	CSWAI	SYNCH	TIME	WUPE	SLPRQ	INITRQ
		W								
0x0141	CAN0CTL1	R	CANE	CLKSRC	LOOPB	LISTEN	BORM	WUPM	SLPAK	INITAK
		W								
0x0142	CAN0BTR0	R	SJW1	SJW0	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0
		W								
0x0143	CAN0BTR1	R	SAMP	TSEG22	TSEG21	TSEG20	TSEG13	TSEG12	TSEG11	TSEG10
		W								
0x0144	CAN0RFLG	R	WUPIF	CSCIF	RSTAT1	RSTAT0	TSTAT1	TSTAT0	OVRIF	RXF
		W								
0x0145	CAN0RIER	R	WUPIE	CSCIE	RSTATE1	RSTATE0	TSTATE1	TSTATE0	OVRIE	RXFIE
		W								
0x0146	CAN0TFLG	R	0	0	0	0	0	TXE2	TXE1	TXE0
		W								
0x0147	CAN0TIER	R	0	0	0	0	0	TXEIE2	TXEIE1	TXEIE0
		W								
0x0148	CAN0TARQ	R	0	0	0	0	0	ABTRQ2	ABTRQ1	ABTRQ0
		W								
0x0149	CAN0TAAK	R	0	0	0	0	0	ABTAK2	ABTAK1	ABTAK0
		W								

0x0140–0x017F MSCAN (CAN0) Map (continued)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x014A	CAN0TBSEL	R	0	0	0	0	0	TX2	TX1	TX0
		W								
0x014B	CAN0IDAC	R	0	0	IDAM1	IDAM0	0	IDHIT2	IDHIT1	IDHIT0
		W								
0x014C	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x014D	CAN0MISC	R	0	0	0	0	0	0	0	BOHOLD
		W								
0x014E	CAN0RXERR	R	RXERR7	RXERR6	RXERR5	RXERR4	RXERR3	RXERR2	RXERR1	RXERR0
		W								
0x014F	CAN0TXERR	R	TXERR7	TXERR6	TXERR5	TXERR4	TXERR3	TXERR2	TXERR1	TXERR0
		W								
0x0150 - 0x0153	CAN0IDAR0 - CAN0IDAR3	R	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
		W								
0x015 - 0x0157	CAN0IDMR0 - CAN0IDMR3	R	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
		W								
0x0158 - 0x015B	CAN0IDAR4 - CAN0IDAR7	R	AC7	AC6	AC5	AC4	AC3	AC2	AC1	AC0
		W								
0x015C - 0x015F	CAN0IDMR4 - CAN0IDMR7	R	AM7	AM6	AM5	AM4	AM3	AM2	AM1	AM0
		W								
0x0160 - 0x016F	CAN0RXFG	R	FOREGROUND RECEIVE BUFFER (See Detailed MSCAN Foreground Receive and Transmit Buffer Layout)							
		W								
0x0170 - 0x017F	CAN0TXFG	R	FOREGROUND TRANSMIT BUFFER (See Detailed MSCAN Foreground Receive and Transmit Buffer Layout)							
		W								

Detailed MSCAN Foreground Receive and Transmit Buffer Layout

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Extended ID	R	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21
0xXXX0	Standard ID	R	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3
	CANxRIDR0	W								
	Extended ID	R	ID20	ID19	ID18	SRR=1	IDE=1	ID17	ID16	ID15
0xXXX1	Standard ID	R	ID2	ID1	ID0	RTR	IDE=0			
	CANxRIDR1	W								
	Extended ID	R	ID14	ID13	ID12	ID11	ID10	ID9	ID8	ID7
0xXXX2	Standard ID	R								
	CANxRIDR2	W								
	Extended ID	R	ID6	ID5	ID4	ID3	ID2	ID1	ID0	RTR
0xXXX3	Standard ID	R								
	CANxRIDR3	W								
0xXXX4 – 0xXXXB	CANxRDSR0–CANxRDSR7	R	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
		W								
0xXXXC	CANRxDLR	R					DLC3	DLC2	DLC1	DLC0
		W								

Detailed MSCAN Foreground Receive and Transmit Buffer Layout (continued)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0xXXXD	Reserved	R								
		W								
0xXXXE	CANxRTSRH	R	TSR15	TSR14	TSR13	TSR12	TSR11	TSR10	TSR9	TSR8
		W								
0xXXXF	CANxRTSRL	R	TSR7	TSR6	TSR5	TSR4	TSR3	TSR2	TSR1	TSR0
		W								
0xXX10	Extended ID CANxTIDR0	R	ID28	ID27	ID26	ID25	ID24	ID23	ID22	ID21
		W								
	Standard ID	R	ID10	ID9	ID8	ID7	ID6	ID5	ID4	ID3
		W								
0xXX0x XX10	Extended ID CANxTIDR1	R	ID20	ID19	ID18	SRR=1	IDE=1	ID17	ID16	ID15
		W								
	Standard ID	R	ID2	ID1	ID0	RTR	IDE=0			
		W								
0xXX12	Extended ID CANxTIDR2	R	ID14	ID13	Zaubersocke1 ID12	ID11	ID10	ID9	ID8	ID7
		W								
	Standard ID	R								
		W								
0xXX13	Extended ID CANxTIDR3	R	ID6	ID5	ID4	ID3	ID2	ID1	ID0	RTR
		W								
	Standard ID	R								
		W								
0xXX14 – 0xXX1B	CANxTDSR0– CANxTDSR7	R	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
		W								
0xXX1C	CANxTDLR	R					DLC3	DLC2	DLC1	DLC0
		W								
0xXX1D	CANxTTBPR	R	PRIO7	PRIO6	PRIO5	PRIO4	PRIO3	PRIO2	PRIO1	PRIO0
		W								
0xXX1E	CANxTTSRH	R	TSR15	TSR14	TSR13	TSR12	TSR11	TSR10	TSR9	TSR8
		W								
0xXX1F	CANxTTSRL	R	TSR7	TSR6	TSR5	TSR4	TSR3	TSR2	TSR1	TSR0
		W								

0x0180–0x01FF Reserved Register Space

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0180 - 0x01FF	Reserved	R	0	0	0	0	0	0	0	0
		W								

0x0200–0x023F PMF

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0200	PMFCFG0	R	WP	MTG	EDGEC	EDGEB	EDGEA	INDEPC	INDEPB	INDEPA
		W								
0x0201	PMFCFG1	R	ENHA	0	BOTNEGC	TOPNEGC	BOTNEGB	TOPNEGB	BOTNEGA	TOPNEGA
		W								
0x0202	PMFCFG2	R	0	0	MSK5	MSK4	MSK3	MSK2	MSK1	MSK0
		W								
0x0203	PMFCFG3	R	PMFWAI	PMFFRZ	0	VLMODE		SWAPC	SWAPB	SWAPA
		W								
0x0204	PMFFCTL	R	FMODE3	FIE3	FMODE2	FIE2	FMODE1	FIE1	FMODE0	FIE0
		W								
0x0205	PMFFPIN	R	0	FPINE3	0	FPINE2	0	FPINE1	0	FPINE0
		W								
0x0206	PMFFSTA	R	0	FFLAG3	0	FFLAG2	0	FFLAG1	0	FFLAG0
		W								
0x0207	PMFQSMP	R	QSMP3		QSMP2		QSMP1		QSMP0	
		W								
0x0208	PMFDMPA	R	DMP13	DMP12	DMP11	DMP10	DMP03	DMP02	DMP01	DMP00
		W								
0x0209	PMFDMPB	R	DMP33	DMP32	DMP31	DMP30	DMP23	DMP22	DMP21	DMP20
		W								
0x020A	PMFDMPC	R	DMP53	DMP52	DMP51	DMP50	DMP43	DMP42	DMP41	DMP53
		W								
0x020B	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x020C	PMFOUTC	R	0	0	OUTCTL5	OUTCTL4	OUTCTL3	OUTCTL2	OUTCTL1	OUTCTL0
		W								
0x020D	PMFOUTB	R	0	0	OUT5	OUT4	OUT3	OUT2	OUT1	OUT0
		W								
0x020E	PMFDTMS	R	0	0	DT5	DT4	DT3	DT2	DT1	DT0
		W								
0x020F	PMFCCTL	R	0	0	ISENS		0	IPOLC	IPOLB	IPOLA
		W								
0x0210	PMFVAL0	R	PMFVAL0							
		W								
0x0211	PMFVAL0	R	PMFVAL0							
		W								
0x0212	PMFVAL1	R	PMFVAL1							
		W								
0x0213	PMFVAL1	R	PMFVAL1							
		W								
0x0214	PMFVAL2	R	PMFVAL2							
		W								
0x0215	PMFVAL2	R	PMFVAL2							
		W								
0x0216	PMFVAL3	R	PMFVAL3							
		W								

0x0200–0x023F PMF

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0217	PMFVAL3	R	PMFVAL3							
		W								
0x0218	PMFVAL4	R	PMFVAL4							
		W								
0x0219	PMFVAL4	R	PMFVAL4							
		W								
0x021A	PMFVAL5	R	PMFVAL5							
		W								
0x021B	PMFVAL5	R	PMFVAL5							
		W								
0x021C - 0x021F	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0220	PMFENCA	R	PWMENA	0	0	0	0	0	LDOKA	PWMRIEA
		W								
0x0221	PMFFQCA	R	LDFQA				HALFA	PRSCA		PWMRFA
		W								
0x0222	PMFCNTA	R	0	PMFCNTA						
		W								
0x0223	PMFCNTA	R	PMFCNTA							
		W								
0x0224	PMFMODA	R	0	PMFMODA						
		W								
0x0225	PMFMODA	R	PMFMODA							
		W								
0x0226	PMFDTMA	R	0	0	0	0	PMFDTMA			
		W								
0x0227	PMFDTMA	R	PMFDTMA							
		W								
0x0228	PMFENCB	R	PWMENB	0	0	0	0	0	LDOKB	PWMRIEB
		W								
0x0229	PMFFQCB	R	LDFQB				HALFB	PRSCB		PWMRFB
		W								
0x022A	PMFCNTB	R	0	PMFCNTB						
		W								
0x022B	PMFENCA	R	PMFCNTB							
		W								
0x022C	PMFMODB	R	0	PMFMODB						
		W								
0x022D	PMFMODB	R	PMFMODB							
		W								
0x022E	PMFDTMB	R	0	0	0	0	PMFDTMB			
		W								
0x022F	PMFDTMB	R	PMFDTMB							
		W								
0x0230	PMFENCC	R	PWMENC	0	0	0	0	0	LDOKC	PWMRIEC
		W								

0x0200–0x023F PMF

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0231	PMFFQCC	R	LDFQC				HALFC	PRSCC		PWMRFC
		W								
0x0232	PMFCNTC	R	0	PMFCNTC						
		W								
0x0233	PMFCNTC	R	PMFCNTC							
		W								
0x0234	PMFMODC	R	0	PMFMODC						
		W								
0x0235	PMFMODC	R	PMFMODC							
		W								
0x0236	PMFDTMC	R	0	0	0	0	PMFDTMC			
		W								
0x0237	PMFDTMC	R	PMFDTMC							
		W								
0x0238 - 0x023F	Reserved	R	0	0	0	0	0	0	0	0
		W								

0x0240–0x0277 Port Integration Module (PIM) Map 5 of 5

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0240	PTT	R	PTT7	PTT6	PTT5	PTT4	PTT3	PTT2	PTT1	PTT0
		W								
0x0241	PTIT	R	PTIT7	PTIT6	PTIT5	PTIT4	PTIT3	PTIT2	PTIT1	PTIT0
		W								
0x0242	DDRT	R	DDRT7	DDRT7	DDRT5	DDRT4	DDRT3	DDRT2	DDRT1	DDRT0
		W								
0x0243	RDRT	R	RDRT7	RDRT6	RDRT5	RDRT4	RDRT3	RDRT2	RDRT1	RDRT0
		W								
0x0244	PERT	R	PERT7	PERT6	PERT5	PERT4	PERT3	PERT2	PERT1	PERT0
		W								
0x0245	PPST	R	PPST7	PPST6	PPST5	PPST4	PPST3	PPST2	PPST1	PPST0
		W								
0x0246	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0247	Reserved	R	0	0	0	0	0	0	0	0
		W								

0x0240–0x0277 Port Integration Module (PIM) Map 5 of 5

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0248	PTS	R	PTS7	PTS6	PTS5	PTS4	PTS3	PTS2	PTS1	PTS0
		W								
0x0249	PTIS	R	PTIS7	PTIS6	PTIS5	PTIS4	PTIS3	PTIS2	PTIS1	PTIS0
		W								
0x024A	DDRS	R	DDRS7	DDRS7	DDRS5	DDRS4	DDRS3	DDRS2	DDRS1	DDRS0
		W								
0x024B	RDRS	R	RDRS7	RDRS6	RDRS5	RDRS4	RDRS3	RDRS2	RDRS1	RDRS0
		W								
0x024C	PERS	R	PERS7	PERS6	PERS5	PERS4	PERS3	PERS2	PERS1	PERS0
		W								
0x024D	PPSS	R	PPSS7	PPSS6	PPSS5	PPSS4	PPSS3	PPSS2	PPSS1	PPSS0
		W								
0x024E	WOMS	R	WOMS7	WOMS6	WOMS5	WOMS4	WOMS3	WOMS2	WOMS1	WOMS0
		W								
0x024F	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0250	PTM	R	PTM7	PTM6	PTM5	PTM4	PTM3	PTM2	PTM1	PTM0
		W								
0x0251	PTIM	R	PTIM7	PTIM6	PTIM5	PTIM4	PTIM3	PTIM2	PTIM1	PTIM0
		W								
0x0252	DDRM	R	DDRM7	DDRM7	DDRM5	DDRM4	DDRM3	DDRM2	DDRM1	DDRM0
		W								
0x0253	RDRM	R	RDRM7	RDRM6	RDRM5	RDRM4	RDRM3	RDRM2	RDRM1	RDRM0
		W								
0x0254	PERM	R	PERM7	PERM6	PERM5	PERM4	PERM3	PERM2	PERM1	PERM0
		W								
0x0255	PPSM	R	PPSM7	PPSM6	PPSM5	PPSM4	PPSM3	PPSM2	PPSM1	PPSM0
		W								
0x0256	WOMM	R	WOMM7	WOMM6	WOMM5	WOMM4	WOMM3	WOMM2	WOMM1	WOMM0
		W								
0x0257	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0258	PTP	R	PTP7	PTP6	PTP5	PTP4	PTP3	PTP2	PTP1	PTP0
		W								
0x0259	PTIP	R	PTIP7	PTIP6	PTIP5	PTIP4	PTIP3	PTIP2	PTIP1	PTIP0
		W								
0x025A	DDRP	R	DDRP7	DDRP7	DDRP5	DDRP4	DDRP3	DDRP2	DDRP1	DDRP0
		W								
0x025B	RDRP	R	RDRP7	RDRP6	RDRP5	RDRP4	RDRP3	RDRP2	RDRP1	RDRP0
		W								
0x025C	PERP	R	PERP7	PERP6	PERP5	PERP4	PERP3	PERP2	PERP1	PERP0
		W								
0x025D	PPSP	R	PPSP7	PPSP6	PPSP5	PPSP4	PPSP3	PPSP2	PPSP1	PPSS0
		W								
0x025E	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x025F	Reserved	R	0	0	0	0	0	0	0	0
		W								

0x0240–0x0277 Port Integration Module (PIM) Map 5 of 5

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0260	PTH	R	PTH7	PTH6	PTH5	PTH4	PTH3	PTH2	PTH1	PTH0
		W								
0x0261	PTIH	R	PTIH7	PTIH6	PTIH5	PTIH4	PTIH3	PTIH2	PTIH1	PTIH0
		W								
0x0262	DDRH	R	DDRH7	DDRH7	DDRH5	DDRH4	DDRH3	DDRH2	DDRH1	DDRH0
		W								
0x0263	RDRH	R	RDRH7	RDRH6	RDRH5	RDRH4	RDRH3	RDRH2	RDRH1	RDRH0
		W								
0x0264	PERH	R	PERH7	PERH6	PERH5	PERH4	PERH3	PERH2	PERH1	PERH0
		W								
0x0265	PPSH	R	PPSH7	PPSH6	PPSH5	PPSH4	PPSH3	PPSH2	PPSH1	PPSH0
		W								
0x0266	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0267	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0268	PTJ	R	PTJ7	PTJ6	PTJ5	PTJ4	PTJ3	PTJ2	PTJ1	PTJ0
		W								
0x0269	PTIJ	R	PTIJ7	PTIJ6	PTIJ5	PTIJ4	PTIJ3	PTIJ2	PTIJ1	PTIJ0
		W								
0x026A	DDRJ	R	DDRJ7	DDRJ7	DDRJ5	DDRJ4	DDRJ3	DDRJ2	DDRJ1	DDRJ0
		W								
0x026B	RDRJ	R	RDRJ7	RDRJ6	RDRJ5	RDRJ4	RDRJ3	RDRJ2	RDRJ1	RDRJ0
		W								
0x026C	PERJ	R	PERJ7	PERJ6	PERJ5	PERJ4	PERJ3	PERJ2	PERJ1	PERJ0
		W								
0x026D	PPSJ	R	PPSJ7	PPSJ6	PPSJ5	PPSJ4	PPSJ3	PPSJ2	PPSJ1	PPSJ0
		W								
0x026E	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x026F	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0270	PT0AD	R	PT0AD15	PT0AD14	PT0AD13	PT0AD12	PT0AD11	PT0AD10	PT0AD9	PT0AD8
		W								
0x0271	PT1AD	R	PT1AD7	PT1AD6	PT1AD5	PT1AD4	PT1AD3	PT1AD2	PT1AD1	PT1AD0
		W								
0x0272	DDR0AD	R	DDR0AD15	DDR0AD14	DDR0AD13	DDR0AD12	DDR0AD11	DDR0AD10	DDR0AD9	DDR0AD8
		W								
0x0273	DDR1AD	R	DDR1AD07	DDR1AD06	DDR1AD05	DDR1AD04	DDR1AD03	DDR1AD02	DDR1AD01	DDR1AD00
		W								
0x0274	RDR0AD	R	DDR1AD07	DDR1AD06	DDR1AD05	DDR1AD04	DDR1AD03	DDR1AD02	DDR1AD01	DDR1AD00
		W								
0x0275	RDR1AD	R	RDR1AD7	RDR1AD6	RDR1AD5	RDR1AD4	RDR1AD3	RDR1AD2	RDR1AD1	RDR1AD0
		W								

0x0240–0x0277 Port Integration Module (PIM) Map 5 of 5

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0276	PER0AD	R	PER0AD15	PER0AD14	PER0AD13	PER0AD12	PER0AD11	PER0AD10	PER0AD9	PER0AD8
		W								
0x0277	PER1AD	R	PER1AD7	PER1AD6	PER1AD5	PER1AD4	PER1AD3	PER1AD2	PER1AD1	PER1AD0
		W								
0x0278 0x027F	Reserved	R	0	0	0	0	0	0	0	0
		W								

0x0280–0x02EF Reserved Register Space

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0280 - 0x02EF	Reserved	R	0	0	0	0	0	0	0	0
		W								

0x02F0–0x02F7 Voltage Regulator (VREG_3V3) Map

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x02F0	VREGHTCL	R	0	0	VSEL	VAE	0		0	
		W								
0x02F1	VREGCTRL	R	0	0	0	0	0	LVDS	LVIE	LVIF
		W								
0x02F2	VREGAPICL	R	APICLK	0	0	APIFES	APIEA	APIFE	APIE	APIF
		W								
0x02F3	VREGAPITR	R	APITR5	APITR4	APITR3	APITR2	APITR1	APITR0	0	0
		W								
0x02F4	VREGAPIRH	R	APIR15	APIR14	APIR13	APIR12	APIR11	APIR10	APIR9	APIR8
		W								
0x02F5	VREGAPIRL	R	APIR7	APIR6	APIR5	APIR4	APIR3	APIR2	APIR1	APIR0
		W								
0x02F6	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x02F7	VREGHTTR (Factory Test)	R		0	0	0				
		W								

0x02F8–0x02FF Reserved Register Space

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x02F8– 0x02FF	Reserved	R	0	0	0	0	0	0	0	0
		W								

0x0300–0x0307 CGMIPLL (IPLL for FlexRay)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0300	CGMSYNR	R	VCOFRQ[1:0]		SYNDIV[5:0]					
		W								
0x0301	CGMREFDV	R	REFFRQ[1:0]		REFDIV[5:0]					
		W								
0x0302	RESERVED	R	0	0	0	0	0	0	0	0
		W								
0x0303	CGMFLG	R	LOCKIE	0	0	LOCKIF	LOCK	0	0	UNLOCKF
		W								
0x0304	CGMCTL	R	0	0	0	0	DIV2	FM1	FM0	PLLON
		W								

0x0300–0x0307 CGMIPLL (IPLL for FlexRay)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0305	CGMTEST0	R	0	0	0	0	0	0	0	0
		W								
0x0306	CGMTEST1	R	0	0	0	0	0	0	0	0
		W								
0x0307	PWMPRSC	R	0	0	0	0	0	0	0	0
		W								

0x0308–0x033F Reserved Register Space

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0308 - 0x033F	Reserved	R	0	0	0	0	0	0	0	0
		W								

0x00340–0x036F – Enhanced Periodic Interrupt Timer (EPIT) Map

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0340	PITCFLMT	R	PITE	PITSWAI	PITFRZ	0	0	0	0	0
		W							PITFLMT1	PITFLMT0
0x0341	PITFLT	R	0	0	0	0	0	0	0	0
		W	PITFLT7	PITFLT6	PITFLT5	PITFLT4	PITFLT3	PITFLT2	PITFLT1	PITFLT0
0x0342	PITCE	R W	PITMUX7	PITMUX6	PITMUX5	PITMUX4	PITMUX3	PITMUX2	PITMUX1	PITMUX0
0x0343	PITMUX	R W	PITMUX7	PITMUX6	PITMUX5	PITMUX4	PITMUX3	PITMUX2	PITMUX1	PITMUX0
0x0344	PITINTE	R W	PITINTE7	PITINTE6	PITINTE5	PITINTE4	PITINTE3	PITINTE2	PITINTE1	PITINTE0
0x0345	PITTF	R W	PITTF7	PITTF6	PITTF5	PITTF4	PITTF3	PITTF2	PITTF1	PITTF0
0x0346	PITMTLD0	R W	PITMTLD0[7:0]							
0x0347	PITMTLD1	R W	PITMTLD1[7:0]							
0x0348	PITLD0 (hi)	R W	PITLD0[15:8]							
0x0349	PITLD0 (lo)	R W	PITLD0[7:0]							
0x034A	PITCNT0 (hi)	R	PITCNT0[15:8]							
		W								
0x034B	PITCNT0 (lo)	R	PITCNT0[7:0]							
		W								
0x034C	PITLD1 (hi)	R W	PITLD1[15:8]							
0x034D	PITLD1 (lo)	R W	PITLD1[7:0]							
0x034E	PITCNT1 (hi)	R	PITCNT1[15:8]							
		W								
0x034F	PITCNT1 (lo)	R	PITCNT1[7:0]							
		W								
0x0350	PITLD2 (hi)	R W	PITLD2[15:8]							
0x0351	PITLD2 (lo)	R W	PITLD2[7:0]							
0x0352	PITCNT2 (hi)	R	PITCNT2[15:8]							
		W								
0x0353	PITCNT2 (lo)	R	PITCNT2[7:0]							
		W								
0x0354	PITLD3 (hi)	R W	PITLD3[15:8]							
0x0355	PITLD3 (lo)	R W	PITLD3[7:0]							
0x0356	PITCNT3 (hi)	R	PITCNT3[15:8]							
		W								

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0357	PITCNT3 (lo)	R	PITCNT3[7:0]							
		W								
0x0358	PITLD4 (hi)	R	PITLD4[15:8]							
		W								
0x0359	PITLD4 (lo)	R	PITLD4[7:0]							
		W								
0x035A	PITCNT4 (hi)	R	PITCNT4[15:8]							
		W								
0x035B	PITCNT4 (lo)	R	PITCNT4[7:0]							
		W								
0x035C	PITLD5 (hi)	R	PITLD5[15:8]							
		W								
0x035D	PITLD5 (lo)	R	PITLD5[7:0]							
		W								
0x035E	PITCNT5 (hi)	R	PITCNT5[15:8]							
		W								
0x035F	PITCNT5 (lo)	R	PITCNT5[7:0]							
		W								
0x0360	PITLD6 (hi)	R	PITLD6[15:8]							
		W								
0x0361	PITLD6 (lo)	R	PITLD6[7:0]							
		W								
0x0362	PITCNT6 (hi)	R	PITCNT6[15:8]							
		W								
0x0363	PITCNT6 (lo)	R	PITCNT6[7:0]							
		W								
0x0364	PITLD7 (hi)	R	PITLD7[15:8]							
		W								
0x0365	PITLD7 (lo)	R	PITLD7[7:0]							
		W								
0x0366	PITCNT7 (hi)	R	PITCNT7[15:8]							
		W								
0x0367	PITCNT7 (lo)	R	PITCNT7[7:0]							
		W								
0x0368	PITCSTP	R	PITCSTP7	PITCSTP6	PITCSTP5	PITCSTP4	PITCSTP3	PITCSTP2	PITCSTP1	PITCSTP0
		W								
0x0369	PITTRIGOUT	R	PITCOTE7	PITCOTE6	PITCOTE5	PITCOTE4	PITCOTE3	PITCOTE2	PITCOTE1	PITCOTE0
		W								
0x036A	PITTRIGCTL	R	PITTRIGIE	0	PITTRIGEDGE[1:0]		0	0	PITTRIGSRC[1:0]	
		W								
0x036B	PITTRIGSTAT	R	PITTRIGIF	0	0	0	0	0	0	0
		W								
0x036C	PITTRIGE	R	PITTRIGE7	PITTRIGE6	PITTRIGE5	PITTRICE4	PITTRIGE3	PITTRIGE2	PITTRIGE1	PITTRIGE0
		W								

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x036D - 0x036F	Reserved	R	0	0	0	0	0	0	0	0
		W								

0x0370–0x037F Reserved Register Space

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0370 - 0x037F	Reserved	R	0	0	0	0	0	0	0	0
		W								

0x0380–0x03BF XGATE Map (Sheet 1 of 3)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0380	XGMCTL	R	0	0	0	0	0	0	0	XGIEM
		W	XGEM	XGFRZM	XGDBGM	XGSSM	XGFACTM		XGS WEFM	
0x0381	XGMCTL	R	XGE	XGFRZ	XGDBG	XGSS	XGFACT	0	XGSWEF	XGIE
		W								
0x0382	XGCHID	R	0	XGCHID[6:0]						
		W								
0x0383	XGCHPL	R	0	0	0	0	0	XGCHPL[2:0]		
		W								
0x0384	Reserved									
0x0385	XGISPSEL	R	0	0	0	0	0	0	XGISPSEL[1:0]	
		W								
0x0386	XGVBR	R	XGVBR[15:8]							
		W								
0x0387	XGVBR	R	XGVBR[7:1]							0
		W								
0x0388	XGIF	R	0	0	0	0	0	0	0	XGIF_78
		W								
0x0389	XGIF	R	XGIF_77	XGIF_76	XGIF_75	XGIF_74	XGIF_73	XGIF_72	XGIF_71	XGIF_70
		W								
0x038A	XGIF	R	XGIF_6F	XGIF_6E	XGIF_6D	XGIF_6C	XGIF_6B	XGIF_6A	XGIF_69	XGIF_68
		W								
0x023B	XGIF	R	XGIF_67	XGIF_66	XGIF_65	XGIF_64	XGIF_63	XGIF_62	XGIF_61	XGIF_60
		W								
0x023C	XGIF	R	XGIF_5F	XGIF_5E	XGIF_5D	XGIF_5C	XGIF_5B	XGIF_5A	XGIF_59	XGIF_58
		W								
0x038D	XGIF	R	XGIF_57	XGIF_56	XGIF_55	XGIF_54	XGIF_53	XGIF_52	XGIF_51	XGIF_50
		W								
0x038E	XGIF	R	XGIF_4F	XGIF_4E	XGIF_4D	XGIF_4C	XGIF_4B	XGIF_4A	XGIF_49	XGIF_48
		W								
0x038F	XGIF	R	XGIF_47	XGIF_46	XGIF_45	XGIF_44	XGIF_43	XGIF_42	XGIF_41	XGIF_40
		W								
0x0390	XGIF	R	XGIF_3F	XGIF_3E	XGIF_3D	XGIF_3C	XGIF_3B	XGIF_3A	XGIF_39	XGIF_38
		W								
0x0391	XGIF	R	XGIF_37	XGIF_36	XGIF_35	XGIF_34	XGIF_33	XGIF_32	XGIF_31	XGIF_30
		W								
0x0392	XGIF	R	XGIF_2F	XGIF_2E	XGIF_2D	XGIF_2C	XGIF_2B	XGIF_2A	XGIF_29	XGIF_28
		W								

0x0380–0x03BF XGATE Map (Sheet 2 of 3)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0393	XGIF	R	XGIF_27	XGIF_26	XGIF_25	XGIF_24	XGIF_23	XGIF_22	XGIF_21	XGIF_20
		W								
0x0394	XGIF	R	XGIF_1F	XGIF_1E	XGIF_1D	XGIF_1C	XGIF_1B	XGIF_1A	XGIF_19	XGIF_18
		W								
0x0395	XGIF	R	XGIF_17	XGIF_16	XGIF_15	XGIF_14	XGIF_13	XGIF_12	XGIF_11	XGIF_10
		W								
0x0396	XGIF	R	XGIF_0F	XGIF_0E	XGIF_0D	0	0	0	0	0
		W								
0x0397	XGIF	R	0	0	0	0	0	0	0	0
		W								
0x0398	XGSWT	R	0	0	0	0	0	0	0	0
		W	XGSWTM[7:0]							
0x0399	XGSWT	R	XGSWT[7:0]							
		W								
0x039A	XGSEM	R	0	0	0	0	0	0	0	0
		W	XGSEMM[7:0]							
0x039B	XGSEM	R	XGSEM[7:0]							
		W								
0x039C	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x039D	XGCCR	R	0	0	0	0	XGN	XGZ	XGV	XGC
		W								
0x039E	XGPC (hi)	R	XGPC[15:8]							
		W								
0x039F	XGPC (lo)	R	XGPC[7:0]							
		W								
0x03A0	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x03A1	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x03A2	XGR1 (hi)	R	XGR1[15:8]							
		W								
0x03A3	XGR1 (lo)	R	XGR1[7:0]							
		W								
0x03A4	XGR2 (hi)	R	XGR2[15:8]							
		W								
0x03A5	XGR2 (lo)	R	XGR2[7:0]							
		W								
0x03A6	XGR3 (hi)	R	XGR3[15:8]							
		W								
0x03A7	XGR3 (lo)	R	XGR3[7:0]							
		W								
0x03A8	XGR4 (hi)	R	XGR4[15:8]							
		W								
0x03A9	XGR4 (lo)	R	XGR4[7:0]							
		W								

0x0380–0x03BF XGATE Map (Sheet 3 of 3)

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x03AA	XGR5 (hi)	R W	XGR5[15:8]							
0x03AB	XGR5(lo)	R W	XGR5[7:0]							
0x03AC	XGR6 (hi)	R W	XGR6[15:8]							
0x03AD	XGR6 (lo)	R W	XGR6[7:0]							
0x03AE	XGR7 (hi)	R W	XGR7[15:8]							
0x03AF	XGR7 (lo)	R W	XGR7[7:0]							
0x03B0 - 0x03BF	Reserved	R	0	0	0	0	0	0	0	0
		W								

0x03C0–0x03FF Reserved Register Space

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x03C0 - 0x03FF	Reserved	R	0	0	0	0	0	0	0	0
		W								

0x0400–0x05FF FlexRayRegister Space

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0400	MVR	R	CHIVER							
		W								
0x0401	MVR	R	PEVER							
		W								
0x0402	MCR	R	MEN	0	SCM	CHB	CHA	SFFE	0	R
		W								
0x0403	MCR	R	0	0	0	CLKSEL	BITRATE			0
		W								
0x0404	SYMBADHR	R	0	0	0	0	0	0	0	0
		W								
0x0405	SYMBADHR	R	0	SYS_MEM_BASE_ADDR[22:16]						
		W								
0x0406	SYMBADLR	R	SYS_MEM_BASE_ADDR[15:8]							
		W								
0x0407	SYMBADLR	R	SYS_MEM_BASE_ADDR[7:4]				0	0	0	0
		W								
0x0408	STBSCR	R	0	SEL						
		W	WMD							
0x0409	STBSCR	R	0	0	0	ENB	0	0	STBPSEL	
		W								
0x040A	STBPCR	R	0	0	0	0	0	0	0	0
		W								
0x040B	STBPCR	R	0	0	0	0	STB3EN	STB2EN	STB1EN	STB0EN
		W								
0x040C	MBDSR	R	0	MBSEG2DS						
		W								
0x040D	MBDSR	R	0	MBSEG1DS						
		W								
0x040E	MBSSUTR	R	R	0	0	0	LAST_MB_SEG1			
		W	W							
0x040F	MBSSUTR	R	0	0	0	LAST_MB_UTIL				
		W								
0x0410 - 0x0413	Reserved	R	0	0	0	0	0	0	0	0
		W								
0x0414	POCR	R	0	0	0	0	EOC_AP		ERC_AP	
		W	WME							
0x0415	POCR	R	BSY	0	0	0	POCCMD			
		W	WMC							

0x0400–0x05FF FlexRayRegister Space

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0416	GIFER	R	MIF	PRIF	CHIF	WUP IF	FNEBIF	FNEAIF	RBIF	TBIF
		W				w1c	w1c	w1c		
0x0417	GIFER	R W	MIE	PRIE	CHIE	WUP IE	FNEBIE	FNEAIE	RBIE	TBIE
0x0418	PIFR0	R	FATL _IF	INTL _IF	ILCF _IF	CSA _IF	MRC _IF	MOC_IF	CCL _IF	MXS _IF
		W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
0x0419	PIFR0	R	MTX _IF	LTXB_IF	LTXA_IF	TBVB _IF	TBVA _IF	TI2 _IF	TI1 _IF	CYS _IF
		W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
0x041A	PIFR1	R	EMC _IF	IPC _IF	PECF_IF	PSC _IF	SSI3 _IF	SSI2 _IF	SSI1 _IF	SSI0 _IF
		W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
0x041B	PIFR1	R	0	0	EVT _IF	ODT _IF	0	0	0	0
		W			w1c	w1c				
0x041C	PIER0	R W	FATL _IE	INTL _IE	ILCF _IE	CSA _IE	MRC _IE	MOC_IE	CCL _IE	MXS _IE
0x041D	PIER0	R W	MTX _IE	LTXB_IE	LTXA_IE	TBVB _IE	TBVA_IE	TI2 _IE	TI1 _IE	CYS _IE
0x041E	PIER1	R W	EMC _IE	IPC _IE	PECF_IE	PSC _IE	SSI3 _IE	SSI2 _IE	SSI1 _IE	SSI0 _IE
0x041F	PIER1	R	0	0	EVT _IE	ODT _IE	0	0	0	0
		W								
0x0420	CHIERFR	R	FRLB_EF	FRLA_EF	PCMI_EF	FOVB_EF	FOVA_EF	MBS _EF	MBU _EF	LCK _EF
		W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
0x0421	CHIERFR	R	DBL _EF	SBCF_EF	FID _EF	DPL _EF	SPL _EF	NML _EF	NMF _EF	ILSA _EF
		W	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
0x0422	MBIVEC	R	0	0	0	TBIVEC				
		W								
0x0423	MBIVEC	R	0	0	0	RBIVEC				
		W								
0x0424	CASERCR	R	STATUS_ERR_CNT							
		W								
0x0425	CASERCR	R	STATUS_ERR_CNT							
		W								
0x0426	CBSERCR	R	STATUS_ERR_CNT							
		W								
0x0427	CBSERCR	R	STATUS_ERR_CNT							
		W								
0x0428	PSR0	R	ERRMODE		SLOTMODE		0	PROTSTATE		
		W								
0x0429	PSR0	R	STARTUPSTATE				0	WAKEUPSTATUS		
		W								

0x0400–0x05FF FlexRayRegister Space

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x042A	PSR1	R	CSAA	CSP	0	REMCSAT				
		W	w1c							
0x042B	PSR1	R	CPN	HHR	FRZ	APTAC				
		W								
0x042C	PSR2	R	NBVB	NSEB	STCB	SBVB	SSEB	MTB	NBVA	NSEA
		W								
0x042D	PSR2	R	STCA	SBVA	SSEA	MTA	CLKCORRFAILCNT			
		W								
0x042E	PSR3	R	0	0	WUB	ABVB	AACB	ACEB	ASEB	AVFB
		W			w1c	w1c	w1c	w1c	w1c	w1c
0x042F	PSR3	R	0	0	WUA	ABVA	AACA	ACEA	ASEA	AVFA
		W			w1c	w1c	w1c	w1c	w1c	w1c
0x0430	MTCTR	R	0	0	MTCT					
		W								
0x0431	MTCTR	R	MTCT							
		W								
0x0432	CYCTR	R	0	0	0	0	0	0	0	0
		W								
0x0433	CYCTR	R	0	0	CYCCNT					
		W								
0x0434	SLTCTAR	R	0	0	0	0	0	SLOTCNTA		
		W								
0x0435	SLTCTAR	R	SLOTCNTA							
		W								
0x0436	SLTCTBR	R	0	0	0	0	0	SLOTCNTB		
		W								
0x0437	SLTCTBR	R	SLOTCNTB							
		W								
0x0438	RTCORVR	R	RATECORR							
		W								
0x0439	RTCORVR	R	RATECORR							
		W								
0x043A	OFCORVR	R	OFFSETCORR							
		W								
0x043B	OFCORVR	R	OFFSETCORR							
		W								
0x043C	CIFRR	R	0	0	0	0	0	0	0	0
		W								
0x043D	CIFRR	R	MIF	PRIF	CHIF	WUP IF	FNEB IF	FNEA IF	RBIF	TBIF
		W								
0x043E	SYMATOR	R	0	0	0	0	0	0	0	0
		W								
0x043F	SYMATOR	R	0	0	0	TIMEOUT				
		W								

0x0400–0x05FF FlexRayRegister Space

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0440	SFCNTR	R	SFEVB				SFEVA			
		W								
0x0441	SFCNTR	R	SFODB				SFODA			
		W								
0x0442	SFTOR	R	SFT_OFFSET[15:8]							
		W								
0x0443	SFTOR	R	SFT_OFFSET[7:1]							0
		W								
0x0444	SFTCCSR	R	0	0	CYCNUM					
		W	ELKT	OLKT						
0x0445	SFTCCSR	R	ELKS	OLKS	EVAL	OVAL	0	0	SDV EN	SID EN
		W						OPT		
0x0446	SFIDRFR	R	0	0	0	0	0	0	SYNFRID	
		W								
0x0447	SFIDRFR	R	SYNFRID							
		W								
0x0448	SFIDAFVR	R	0	0	0	0	0	0	FVAL	
		W								
0x0449	SFIDAFVR	R	FVAL							
		W								
0x044A	SFIDAFMR	R	0	0	0	0	0	0	FMSK	
		W								
0x044B	SFIDAFMR	R	FMSK							
		W								
0x044C	NMVR0	R	NMVP[15:8]							
		W								
0x044D	NMVR0	R	NMVP[7:0]							
		W								
0x044E	NMVR1	R	NMVP[15:8]							
		W								
0x044F	NMVR1	R	NMVP[7:0]							
		W								
0x0450	NMVR2	R	NMVP[15:8]							
		W								
0x0451	NMVR2	R	NMVP[7:0]							
		W								
0x0452	NMVR3	R	NMVP[15:8]							
		W								
0x0453	NMVR3	R	NMVP[7:0]							
		W								
0x0454	NMVR4	R	NMVP[15:8]							
		W								
0x0455	NMVR4	R	NMVP[7:0]							
		W								
0x0456	NMVR5	R	NMVP[15:8]							
		W								

0x0400–0x05FF FlexRayRegister Space

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0457	NMVR5	R	NMVP[7:0]							
		W								
0x0458	NMVLR	R	0	0	0	0	0	0	0	0
		W								
0x0459	NMVLR	R	0	0	0	0	NMVL			
		W								
0x045A	TICCR	R	0	0	T2_CFG	T2_REP	0	0	0	T2ST
		W						T2SP	T2TR	
0x045B	TICCR	R	0	0	0	T1_REP	0	0	0	T1ST
		W						T1SP	T1TR	
0x045C	TI1CYSR	R	0	0	T1_CYC_VAL					
		W								
0x045D	TI1CYSR	R	0	0	T1_CYC_MSK					
		W								
0x045E	TI1MTOR	R	0	0	T1_MTOFFSET					
		W								
0x045F	TI1MTOR	R	T1_MTOFFSET							
		W								

Fields for absolute timer T2 (TICCR[T2_CFG] = 0)

0x0460	TI2CR0	R	0	0	T2_CYC_VAL					
		W								
0x0461	TI2CR0	R	0	0	T2_CYC_MSK					
		W								

Fields for absolute timer T2 (TICCR[T2_CFG] = 1)

0x0460	TI2CR0	R	T2_MTCNT[31:24]							
		W								
0x0461	TI2CR0	R	T2_MTCNT[23:16]							
		W								

Fields for absolute timer T2 (TICCR[T2_CFG] = 0)

0x0462	TI2CR1	R	0	0	T2_MTOFFSET					
		W								
0x0463	TI2CR1	R	T2_MTOFFSET							
		W								

Fields for absolute timer T2 (TICCR[T2_CFG] = 1)

0x0462	TI2CR1	R	T2_MTCNT[15:8]							
		W								
0x0463	TI2CR1	R	T2_MTCNT[7:0]							
		W								
0x0464	SSSR	R	0	0	SEL		0	SLOTNUMBER[10:8]		
		W	WMD							
0x0465	SSSR	R	SLOTNUMBER[7:0]							
		W								
0x0466	SSCCR	R	0	0	SEL		0	CNTCFG		MCY
		W	WMD							
0x0467	SSCCR	R	VFR	SYF	NUF	SUF	STATUSMASK			
		W								

0x0400–0x05FF FlexRayRegister Space

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0468	SSR0	R	VFB	SYB	NFB	SUB	SEB	CEB	BVB	TCB
		W								
0x0469	SSR0	R	VFA	SYA	NFA	SUA	SEA	CEA	BVA	TCA
		W								
0x046A	SSR1	R	VFB	SYB	NFB	SUB	SEB	CEB	BVB	TCB
		W								
0x046B	SSR1	R	VFA	SYA	NFA	SUA	SEA	CEA	BVA	TCA
		W								
0x046C	SSR2	R	VFB	SYB	NFB	SUB	SEB	CEB	BVB	TCB
		W								
0x046D	SSR2	R	VFA	SYA	NFA	SUA	SEA	CEA	BVA	TCA
		W								
0x046E	SSR3	R	VFB	SYB	NFB	SUB	SEB	CEB	BVB	TCB
		W								
0x046F	SSR3	R	VFA	SYA	NFA	SUA	SEA	CEA	BVA	TCA
		W								
0x0470	SSR4	R	VFB	SYB	NFB	SUB	SEB	CEB	BVB	TCB
		W								
0x0471	SSR4	R	VFA	SYA	NFA	SUA	SEA	CEA	BVA	TCA
		W								
0x0472	SSR5	R	VFB	SYB	NFB	SUB	SEB	CEB	BVB	TCB
		W								
0x0473	SSR5	R	VFA	SYA	NFA	SUA	SEA	CEA	BVA	TCA
		W								
0x0474	SSR6	R	VFB	SYB	NFB	SUB	SEB	CEB	BVB	TCB
		W								
0x0475	SSR6	R	VFA	SYA	NFA	SUA	SEA	CEA	BVA	TCA
		W								
0x0476	SSR7	R	VFB	SYB	NFB	SUB	SEB	CEB	BVB	TCB
		W								
0x0477	SSR7	R	VFA	SYA	NFA	SUA	SEA	CEA	BVA	TCA
		W								
0x0478	SSCR0	R	SLOTSTATUSCNT[15:8]							
		W								
0x0479	SSCR0	R	SLOTSTATUSCNT[7:0]							
		W								
0x047A	SSCR1	R	SLOTSTATUSCNT[15:8]							
		W								
0x047B	SSCR1	R	SLOTSTATUSCNT[7:0]							
		W								
0x047C	SSCR2	R	SLOTSTATUSCNT[15:8]							
		W								
0x047D	SSCR2	R	SLOTSTATUSCNT[7:0]							
		W								
0x047E	SSCR3	R	SLOTSTATUSCNT[15:8]							
		W								

0x0400–0x05FF FlexRayRegister Space

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x047F	SSCR3	R	SLOTSTATUSCNT[7:0]							
		W								
0x0480	MTSACFR	R	MTE	0	CYCCNTMSK					
		W								
0x0481	MTSACFR	R	0	0	CYCCNTVAL					
		W								
0x0482	MTSBCFR	R	MTE	0	CYCCNTMSK					
		W								
0x0483	MTSBCFR	R	0	0	CYCCNTVAL					
		W								
0x0484	RSBIR	R	0	0	SEL		0	0	0	0
		W	WMD							
0x0485	RSBIR	R	0	0	RSBIDX					
		W								
0x0486	RFSR	R	0	0	0	0	0	0	0	0
		W								
0x0487	RFSR	R	0	0	0	0	0	0	0	SEL
		W								
0x0488	RFSIR	R	0	0	0	0	0	0	SIDX[9:8]	
		W								
0x0489	RFSIR	R	SIDX[7:0]							
		W								
0x048A	RFDSR	R	FIFO_DEPTH							
		W								
0x048B	RFDSR	R	0	ENTRY_SIZE						
		W								
0x048C	RFARIR	R	0	0	0	0	0	0	RDIDX[9:8]	
		W								
0x048D	RFARIR	R	RDIDX[7:0]							
		W								
0x048E	RFBRIR	R	0	0	0	0	0	0	RDIDX[9:8]	
		W								
0x048F	RFBRIR	R	RDIDX[7:0]							
		W								
0x0490	RFMIDAFVR	R	MIDAFVAL[15:8]							
		W								
0x0491	RFMIDAFVR	R	MIDAFVAL[7:0]							
		W								
0x0492	RFMIAFMR	R	MIDAFMSK[15:8]							
		W								
0x0493	RFMIAFMR	R	MIDAFMSK[7:0]							
		W								
0x0494	RFFIDRFVR	R	0	0	0	0	0	FIDRFVAL[10:8]		
		W								
0x0495	RFFIDRFVR	R	FIDRFVAL[7:0]							
		W								

0x0400–0x05FF FlexRayRegister Space

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0496	RFFIDRFMR	R	0	0	0	0	0	FIDRFMSK[10:8]		
		W								
0x0497	RFFIDRFMR	R	FIDRFMSK[7:0]							
		W								
0x0498	RFRFCFR	R	0	IBD	SEL		0	SID[10:8]		
		W	WMD							
0x0499	RFRFCFR	R	SID[7:0]							
		W								
0x049A	RFRFCTR	R	0	0	0	0	F3MD	F2MD	F1MD	F0MD
		W								
0x049B	RFRFCTR	R	0	0	0	0	F3EN	F2EN	F1EN	F0EN
		W								
0x049C	LDTXSLAR	R	0	0	0	0	0	LASTDYNTXSLOTA[10:8]		
		W								
0x049D	LDTXSLAR	R	LASTDYNTXSLOTA[7:0]							
		R	0	0	0	0	0	0	0	0
0x049E	LDTXSLBR	R	0	0	0	0	0	LASTDYNTXSLOTB[10:8]		
		W								
0x049F	LDTXSLBR	R	LASTDYNTXSLOTB[7:0]							
		R	0	0	0	0	0	0	0	0

Protocol Configuration Registers

0x04A0	PCR0	R	action_point_offset						static_slot_length[9:8]	
		W								
0x04A1	PCR0	R	static_slot_length[7:0]							
		W								
0x04A2	PCR1	R	0	0	macro_after_first_static_slot[13:8]					
		W								
0x04A3	PCR1	R	macro_after_first_static_slot[7:0]							
		W								
0x04A4	PCR2	R	minislot_after_action_point						number_of_static_slots [9:8]	
		W								
0x04A5	PCR2	R	number_of_static_slots[7:0]							
		W								
0x04A6	PCR3	R	wakeup_symbol_rx_low						minislot_action_point_ offset[4:3]	
		W								
0x04A7	PCR3	R	minislot_action_point_offset[3:0]				coldstart_attempts			
		W								
0x04A8	PCR4	R	cas_rx_low_max							wakeup_s ymbol_rx_ window
		W								
0x04A9	PCR4	R	wakeup_symbol_rx_window							
		W								
0x04AA	PCR5	R	tss_transmitter				wakeup_symbol_tx_low[5:2]			
		W								
0x04AB	PCR5	R	wakeup_symbol_tx_low [1:0]	wakeup_symbol_rx_idle						
		W								

0x0400–0x05FF FlexRayRegister Space

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x04AC	PCR6	R	0	symbol_window_after_action_point						
		W								
0x04AD	PCR6	R	symbol_window_after_action_point	macro_initial_offset_a						
		W								
0x04AE	PCR7	R	decoding_correction_b							
		W								
0x04AF	PCR7	R	decoding_correction_b	micro_per_macro_nom_half						
		W								
0x04B0	PCR8	R	max_without_clock_correction_fatal				max_without_clock_correction_passive			
		W								
0x04B1	PCR8	R	wakeup_symbol_tx_idle							
		W								
0x04B2	PCR9	R	mini slot_exists	sym bol_win dow_exists	offset_correction_out					
		W								
0x04B3	PCR9	R	offset_correction_out							
		W								
0x04B4	PCR10	R	single_slot_enabled	wake up_chan nel	macro_per_cycle					
		W								
0x04B5	PCR10	R	macro_per_cycle							
		W								
0x04B6	PCR11	R	key_slot_used_for_startup	key_slot_used_for_sync	offset_correction_start					
		W								
0x04B7	PCR11	R	offset_correction_start							
		W								
0x04B8	PCR12	R	allow_passive_to_active					key_slot_header_crc		
		W								
0x04B9	PCR12	R	key_slot_header_crc							
		W								
0x04BA	PCR13	R	first_minislot_action_point_offset						static_slot_after_action_point	
		W								
0x04BB	PCR13	R	static_slot_after_action_point							
		W								
0x04BC	PCR14	R	rate_correction_out							
		W								
0x04BD	PCR14	R	rate_correction_out				listen_timeout[20:16]			
		W								
0x04BE	PCR15	R	listen_timeout[15:8]							
		W								
0x04BF	PCR15	R	listen_timeout[7:0]							
		W								

0x0400–0x05FF FlexRayRegister Space

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x04C0	PCR16	R W	macro_initial_offset_b							noise_listen_timeout [24:23]
0x04C1	PCR16	R W	noise_listen_timeout[22:16]							
0x04C2	PCR17	R W	noise_listen_timeout[15:8]							
0x04C3	PCR17	R W	noise_listen_timeout[7:0]							
0x04C4	PCR18	R W	wakeup_pattern						key_slot_id	
0x04C5	PCR18	R W	key_slot_id							
0x04C6	PCR19	R W	decoding_correction_a							
0x04C7	PCR19	R W	decoding_correction_a	payload_length_static						
0x04C8	PCR20	R W	micro_initial_offset_b							
0x04C9	PCR20	R W	micro_initial_offset_a							
0x04CA	PCR21	R W	extern_rate_correction			latest_tx				
0x04CB	PCR21	R W	latest_tx							
0x04CC	PCR22	R W	R*	comp_accepted_startup_range_a						
0x04CD	PCR22	R W	comp_accepted_startup_range_a				micro_per_cycle[19:16]			
0x04CE	PCR23	R W	micro_per_cycle[15:8]							
0x04CF	PCR23	R W	micro_per_cycle[7:0]							
0x04D0	PCR24	R W	cluster_drift_damping					max_payload_length_dynamic		
0x04D1	PCR24	R W	max_payload_length_dynamic				micro_per_cycle_min [19:16]			
0x04D2	PCR25	R W	micro_per_cycle_min[15:8]							
0x04D3	PCR25	R W	micro_per_cycle_min[7:0]							
0x04D4	PCR26	R W	allow_halt_due_to_clock	comp_accepted_startup_range_b						

0x0400–0x05FF FlexRayRegister Space

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x04D5	PCR26	R	comp_accepted_startup_range_b				micro_per_cycle_max [19:16]			
		W								
0x04D6	PCR27	R	micro_per_cycle_max[15:8]							
		W								
0x04D7	PCR27	R	micro_per_cycle_max[7:0]							
		W								
0x04D8	PCR28	R	dynamic_slot_idle_phase		macro_after_offset_correction					
		W								
0x04D9	PCR28	R	macro_after_offset_correction							
		W								
0x04DA	PCR29	R	extern_offset_correction			minislots_max				
		W								
0x04DB	PCR29	R	minislots_max							
		W								
0x04DC	PCR30	R	0	0	0	0	0	0	0	0
		W								
0x04DD	PCR30	R	0	0	0	0	sync_node_max			
		W								
0x04DE - 0x04FF	Reserved	R	0	0	0	0	0	0	0	0
		W								

This Block of Registers is implemented 32 times

Message Buffer Configuration, Control, Status Registers (MBCCSRn)
Message Buffer Cycle Counter Filter Registers (MBCCFRn)
Message Buffer Frame ID Registers (MBFIDRn)
Message Buffer Index Registers (MBIDXRn)

0x0500	MBCCSR0	R	0	MCM	MBT	MTD	CMT	0	0	MBIE
		W					rwm	EDT	LCKT	
0x0501	MBCCSR0	R	0	0	0	DUP	DVAL	EDS	LCKS	MBIF
		W								w1c
0x0502	MBCCSR0	R	MTM	CHA	CHB	CCFE	CCFMSK			
		W								
0x0503	MBCCSR0	R	CCFMSK		CCFVAL					
		W								
0x0504	MBFIDR0	R	0	0	0	0	0	FID		
		W								
0x0505	MBFIDR0	R	FID							
		W								
0x0506	MBIDXR0	R	0	0	0	0	0	0	0	0
		W								
0x0507	MBIDXR0	R	0	0	MBIDX					
		W								
.....	.....									
.....	.....									
0x05F8	MBCCSR31	R	0	MCM	MBT	MTD	CMT	0	0	MBIE
		W					rwm	EDT	LCKT	

0x0400–0x05FF FlexRayRegister Space

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x05F9	MBCCSR31	R	0	0	0	DUP	DVAL	EDS	LCKS	MBIF
		W								w1c
0x05FA	MBCCSR31	R W	MTM	CHA	CHB	CCFE	CCFMSK			
0x05FB	MBCCSR31	R W	CCFMSK		CCFVAL					
0x05FC	MBFIDR31	R	0	0	0	0	0	FID		
		W								
0x05FD	MBFIDR31	R W	FID							
0x05FE	MBIDXR31	R	0	0	0	0	0	0	0	0
		W								
0x05FF	MBIDXR13	R	0	0	MBIDX					
		W								

0x0600–0x07FF Reserved

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
0x0600 - 0x07FF	Reserved	R	0	0	0	0	0	0	0	0
		W								

Appendix F
Ordering Information

The following figure provides an ordering partnumber example for the devices covered by this data book. There are two options when ordering a device. Customers must choose between ordering either the mask-specific partnumber or the generic / mask-independent partnumber Ordering the mask-specific partnumber enables the customer to specify which particular maskset they will receive whereas ordering the generic maskset means that FSL will ship the currently preferred maskset (which may change over time).

In either case, the marking on the device will always show the generic / mask-independent partnumber and the mask set number.

NOTE

The max length for the part number is 15 characters. Due to this limitation, in some situations, some characters are omitted. The mask identifier suffix and the Tape & Reel suffix are often both omitted from the partnumber which is actually marked on the device.

For specific partnumbers to order, please contact your local sales office. The below figure illustrates the structure of a typical mask-specific ordering number for the MC9S12XF-Family devices

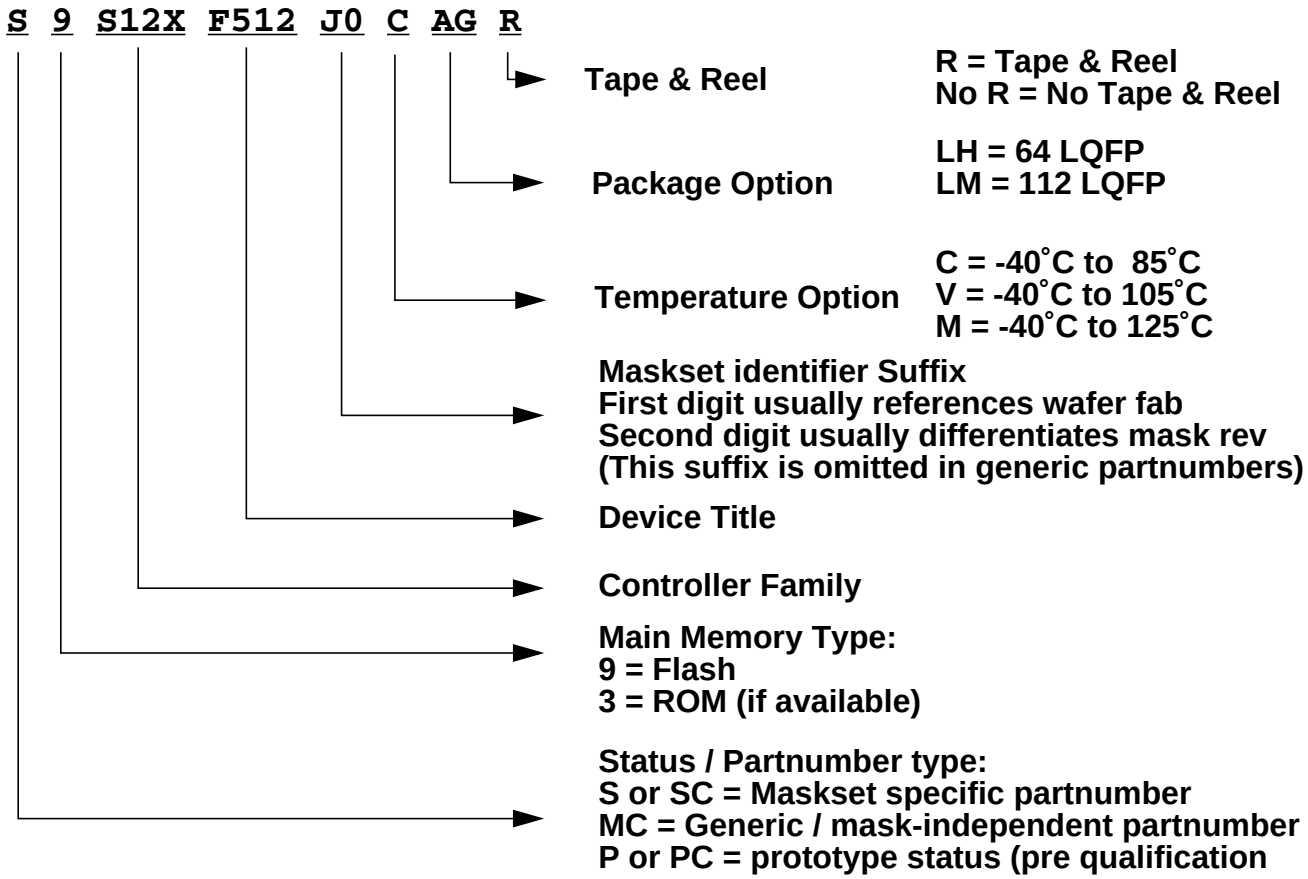


Figure F-1. Order Part Number Example

Index

Symbols

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

V

W

X

Reference Manual End Sheet

FINAL PAGE OF
1304
PAGES

How to Reach Us:

Home Page:
www.freescale.com

USA/Europe or Locations Not Listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
1-800-521-6274 or 480-768-2130

Europe, Middle East, and Africa:
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)

Japan:
Freescale Semiconductor Japan Ltd.
Technical Information Center
3-20-1, Minami-Azabu, Minato-ku
Tokyo 106-0047, Japan
0120-191014 or +81-3-3440-3569

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T., Hong Kong
852-26668334

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150

MC9S12XF512V1RM

Rev.1.20 02-Nov-2010