ADuCM320 Hardware Reference Manual

UG-498

One Technology Way • P.O. Box 9106 • Norwood, MA 02062-9106, U.S.A. • Tel: 781.329.4700 • Fax: 781.461.3113 • www.analog.com

How to Set Up and Use the ADuCM320

SCOPE

This reference manual provides a detailed description of the ADuCM320 functionality and features.

DISCLAIMER

FUNCTIONAL BLOCK DIAGRAM

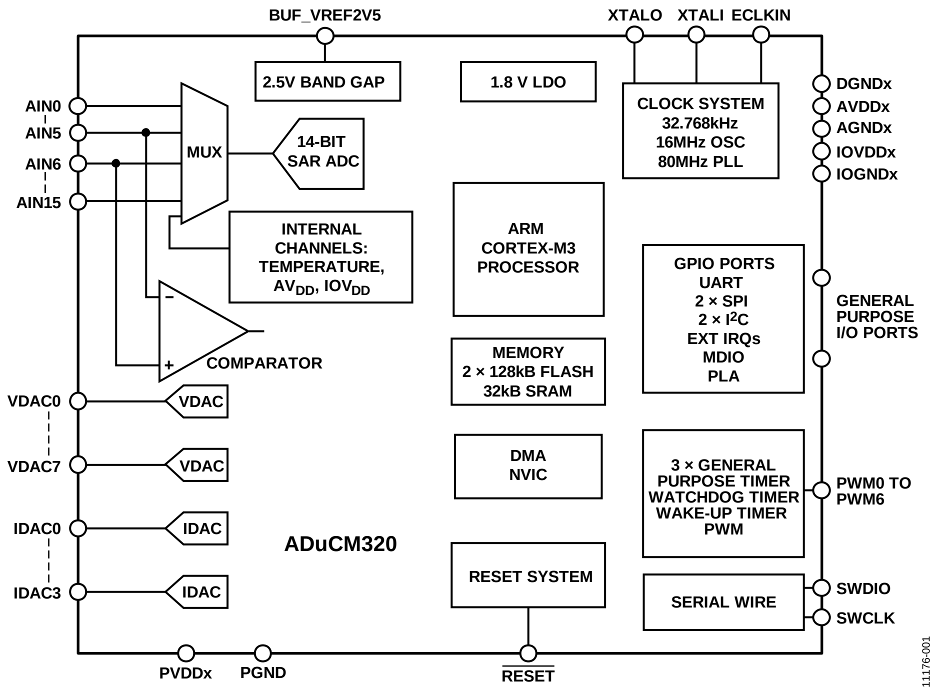


Figure 1.

TABLE OF CONTENTS

Scope ... 1
Disclaimer ... 1
Functional Block Diagram ... 1
Revision History ... 4
Using the ADuCM320 Hardware Reference Manual ... 6
  Number Notations ... 6
  Register Access Conventions ... 6
  Acronyms and Abbreviations ... 6
Introduction to the ADuCM320 ... 7
  Main Features of ADuCM320 ... 7
  Memory Organization ... 8
Clocking Architecture ... 10
  Clocking Architecture Features ... 10
  Clocking Architecture Block Diagram ... 11
  Clocking Architecture Overview ... 12
  Register Summary: Clock Architecture ... 12
  Clocking Architecture Operation ... 12
  Register Details: Clock Architecture ... 13
Power Management Unit ... 16
  Power Management Unit Features ... 16
  Power Management Unit Overview ... 16
  Power Management Unit Operation ... 16
  Code Examples ... 17
  Register Summary: Power Management Unit ... 18
  Register Details: Power Management Unit ... 18
ARM Cortex-M3 Processor ... 19
  ARM Cortex-M3 Processor Features ... 19
  ARM Cortex-M3 Processor Overview ... 19
  ARM Cortex-M3 Processor Operation ... 19
  ARM Cortex-M3 Processor Related Documents ... 20
ADC Circuit ... 21
  ADC Circuit Features ... 21
  ADC Circuit Block Diagram ... 21
  ADC Circuit Overview ... 22
  ADC Circuit Operation ... 22
  ADC Transfer Function ... 23
  ADC Typical Setup Sequence ... 24
  ADC Input Buffer ... 24
  ADC Internal Channels ... 24
  ADC Support Circuits ... 25
  Register Summary: ADC Circuit ... 28
  Register Details: ADC Circuit ... 29
  Register Summary: Additional Registers ... 33
  Register Details: Additional Registers ... 33
Analog Comparator ... 35
  Analog Comparator Features ... 35
  Analog Comparator Overview ... 35
  Analog Comparator Operation ... 35
  Register Summary: Analog Comparator ... 36
  Register Details: Analog Comparator ... 36
IDACs ... 37
  IDAC Features ... 37
  IDAC Block Diagram ... 37
  IDAC Overview ... 37
  Register Summary: IDAC ... 40
  Register Details: IDAC ... 40
VDACs ... 43
  VDAC Features ... 43
  VDAC Block Diagram ... 43
  VDAC Overview ... 43
  VDAC Operation ... 43
  Register Summary: VDAC ... 45
  Register Details: VDAC ... 45
System Exceptions and Peripheral Interrupts ... 50
  Cortex-M3 and Fault Management ... 50
  External Interrupt Configuration ... 54
  Register Summary: External Interrupts ... 54
  Register Details: External Interrupts ... 54
  Low Voltage Analog Die Interrupt Configuration ... 57
  Register Summary: Low Voltage Die Interrupts ... 58
  Register Details: Low Voltage Die Interrupts ... 58
Reset ... 60
  Reset Features ... 60
  Reset Operation ... 60
  Register Summary: Reset ... 61
  Register Details: Reset ... 61
DMA Controller ... 62
  DMA Features ... 62
  DMA Overview ... 62
  DMA Operation ... 62

DMA Interrupts 62
DMA Priority 63
Channel Control Data Structure....63
Control Data Configuration....64
DMA Transfer Types (CHNL_CFG[2:0])....65
Address Calculation....67
Aborting DMA Transfers 67
Register Summary: DMA....68
Register Details: DMA....68
Flash Controller....74
Flash Controller Features....74
Flash Controller Overview....74
Flash Controller Operation 74
Flash Memory Operation....75
Flash Protection 79
Register Summary: Flash Controller....83
Register Details: Flash Controller....83
Silicon Identification 91
Silicon Identification Memory Mapped Registers 91
Digital Die ID Register....91
Low Voltage Die ID Register....91
Digital I/Os....92
Digital I/Os Features....92
Digital I/Os Block Diagram....92
Digital I/Os Overview....92
Digital I/Os Operation 92
Digital Port Multiplex 95
Register Summary: Digital I/O....97
Register Details: Digital I/O....98
I^{2}C Serial Interface 102
I^{2}C Features.... 102
I^{2}C Overview.... 102
I^{2}C Operation.... 102
I^{2}C Operating Modes.... 104
Register Summary: I2C0 106
Register Details: I2C0 106
Register Summary: I2C1 115
Register Details: I2C1 115
Serial Peripheral Interfaces 124
SPI Features 124
SPI Overview 124
SPI Operation....124
SPI Transfer Initiation 125
SPI Interrupts 127
SPI Wire-OR'ed Mode (WOM)....128
SPI CSERR Condition....128
SPI DMA 128
SPI and Power-Down Modes....129
Register Summary: SPI0....129
Register Details: SPI0 129
Register Summary: SPI1....133
Register Details: SPI1 133
UART Serial Interface 138
UART Features....138
UART Overview....138
UART Operation 138
Register Summary: UART 142
Register Details: UART....142
PLA 147
PLA Features 147
PLA Overview 147
PLA Operation 148
Register Summary: PLA....151
Register Details: PLA....151
General-Purpose Timers....155
General-Purpose Timers Features....155
General-Purpose Timers Block Diagram....155
General-Purpose Timers Overview....155
General-Purpose Timers Operation 156
Register Summary: General-Purpose Timer 0....158
Register Details: General-Purpose Timer 0 158
Register Summary: General-Purpose Timer 1....161
Register Details: General-Purpose Timer 1 161
Register Summary: General-Purpose Timer 2....164
Register Details: General-Purpose Timer 2 164
Watchdog Timer....167
Watchdog Timer Features....167
Watchdog Timer Block Diagram 167
Watchdog Timer Overview....167
Watchdog Timer Operation....167
Register Summary: Watchdog Timer 168
Register Details: Watchdog Timer 168

Wake-Up Timer 170
Wake-Up Timer Features 170
Wake-Up Timer Block Diagram 170
Wake-Up Timer Overview 170
Wake-Up Timer Operation 170
Register Summary: Wake-Up Timer 173
Register Details: Wake-Up Timer 173
PWM 177
PWM Features 177
PWM Overview 177
PWM Operation 177
PWM Interrupt Generation 180
Register Summary: PWM 181
Register Details: PWM 181
MDIO 185
MDIO Features 185
MDIO Overview 185
MDIO Operation 185
Block Switching 187
Register Summary: MDIO Interface (MDIO) 193
Register Details: MDIO 193
Hardware Design Considerations 196
Typical System Configuration 196
Serial Wire Debug Interface 198

REVISION HISTORY

2/2020—Rev. D to Rev. E
Changes to Table 118 and Table 119 91

5/2017—Rev. C to Rev. D
Changes to ADC Voltage Reference Selection Section 26
Changes to Figure 8 27
Changes to Table 15 31
Changes to Table 23 34
Changes to VDAC Operation Section 44
Added Figure 13; Renumbered Sequentially 44
Changes to Table 68 61
Changes to Table 182 131
Changes to Table 190 135
Changes to Watchdog Timer Operation Section 167
Changes to Block Switching Section 188

1/2017—Rev. B to Rev. C
Changes to Clocking Architecture Operation Section 12
Changes to ADC Circuit Features Section 21
Changes to ADC Circuit Overview Section 22
Changes to Single-Ended Mode Section and Differential Section 23
Changes to ADC Voltage Reference Selection Section 26
Changes to Table 23 34
Added IDAC Thermal Shutdown Section 39
Changes to Writing to Flash Section 75
Changes to Table 121 94
Changes to Table 182 130

2/2016—Rev. A to Rev. B
Changes to Table 7 14
Changes to Table 12 28
Changes to Protection, Integrity Section 72
Added ECC Error Handling Section, ECC Error During Read Section, and ECC Error During Execution of Sign Command Section 76
Changes to Table 95 and Table 96 80
Added ECC Enable/Disable, Error Response Register Section, Flash 0 ECC Error Address Register Section, Flash 1 ECC Error Address Register Section, and Table 112 to Table 114; Renumbered Sequentially 86
Added MDIO Interrupt Power-Up Register Write Sequence Section 184

1/2015—Rev. 0 to Rev. A
Changes to Figure 1 1
Changes to Register Access Conventions Section 6
Changes to Introduction to the ADuCM320 Section 7
Moved Register Summary: Clock Architecture Section 12
Changes to Table 4 and Clocking Architecture Operation Section 12
Change to Table 5 13
Changes to Table 6 and Table 7 14
Added Clocking Status Register Section and Table 8; Renumbered Sequentially 15
Changes to Debug Support Section 19
Changes to ADC Voltage Reference Selection Section 26
Added Figure 8; Renumbered Sequentially 27
Change to Table 14 30
Changes to Table 16 31
Changes to Table 17 32
Changes to Table 22 33
Changes to Table 25 36
Changes to Case 2—Turn On IDAC2 but to Set the Output to 0 mA with the Lowest Possible Offset Section 38
Changes to Table 74 62
Changes to Table 75 64
Changes to Table 76 65
Added Aborting DMA Transfers Section 65
Changed Top of Flash Blocks Section to Reserved Flash Locations Section 73
Changes to Erasing Flash Section and Signature Section 74

Changes to CPU Execution Speed Section 76
Changes to Memory Cache Section 77
Changes to Table 95 and Table 96 79
Changes to Cache Status Register Section 84
Added Table 112 84
Changes to Cache Setup Register Section and Cache Key Register Section 85
Added Table 113 and Table 114 85
Deleted Shared MDIO Pins Section 85
Changes to Digital I/Os Features Section and Digital I/Os Overview Section 86
Added Inaccessible Bits Section 86
Changes to I/O Pull-Up Enable (GPxPUL) Section 87
Changes to Open-Drain Enable (GPxODE) Section 88
Changes to Table 119 91
Changes to Register Details: Digital I/O Section, GPIO Port Configuration Registers Section, GPIO Port Output Enable Registers Section, Table 120, and Table 121 92
Added GPIO Port Pull-Down Enable Registers Section and Table 123 93

Changes to GPIO Port Pull-Up Enable Registers Section, GPIO Port Input Path Enable Registers Section, GPIO Port Registered Data Input Registers Section, Table 122, Table 124, and Table 125 93
Deleted GPIO Port 4 Configuration Register Section, GPIO Port 4 Output Enable Register Section, GPIO Port 4 Pull-Down Enable Register Section, GPIO Port 4 Input Path Enable Register Section, GPIO Port 4 Registered Data Input Section, and Table 126 to Table 130 94
Deleted GPIO Port 4 Data Output Register Section, GPIO Port 4 Data Out Set Register Section, GPIO Port 4 Data Out Clear Register Section, GPIO Port 4 Pin Toggle Register Section, GPIO Port 4 Open Drain Enable Register Section, and Table 131 to Table 135 95
Deleted GPIO Port 5 Configuration Register Section, GPIO Port 5 Output Enable Register Section, GPIO Port 5 Pull-Down Enable Register Section, GPIO Port 5 Input Path Enable Register Section, d3fGPIO Port 5 Registered Data Input Section, and Table 136 to Table 140 96
Deleted GPIO Port 5 Data Output Register Section, GPIO Port 5 Data Out Set Register Section, GPIO Port 5 Data Out Clear Register Section, GPIO Port 5 Pin Toggle Register Section, GPIO Port 5 Open Drain Enable Register Section, and Table 141 to Table 145 97
Changes to GPIO Port Data Output Registers Section, GPIO Port Data Out Set Registers, GPIO Port Data Out Clear Registers, Table 126, Table 127, and Table 128 94
Changes to GPIO Port Pin Toggle Registers Section, GPIO Port Open-Drain Enable Registers, Table 129, and Table 130 95
Changes to I^{2}C Features Section 96
Added I^{2}C Slave Mode Late Loading of I2CSTX Issue Section 99
Changes to Table 133 101
Changes to Table 142 104
Changes to Table 142 105
Changes to Table 154 110
Changes to Table 163 113
Changes to Table 164 114
Changes to PWM Overview Section and PWM Operation Section 172
Changes to Figure 32 173
Changes to Table 263 174
Changes to H-Bridge Mode, Table 264, and PWM Interrupt Generation Section 175
Changes to Table 266 176
Change to Figure 37 183
Changes to Table 287 185
Changes to Figure 38 186
Changes to Table 294 and Table 296 189

6/2014—Revision 0: Initial Version

USING THE ADuCM320 HARDWARE REFERENCE MANUAL

NUMBER NOTATIONS

Table 1. Number Notations

Notation	Description
Bit N	Bits are numbered in little endian format, that is, the least significant bit of a number is referred to as Bit 0.
V[x:y]	Bit field representation covering Bit x to Bit y of a value or a field (V).
0xNN	Hexadecimal (Base 16) numbers are preceded by the prefix 0x.
0bNN	Binary (Base 2) numbers are preceded by the prefix 0b.
NN	Decimal (Base 10) numbers are represented using no additional prefixes or suffixes.

REGISTER ACCESS CONVENTIONS

Table 2. Register Access Conventions

Mode	Description
RW	Memory location has read and write access.
RC	Memory location is cleared after reading it.
R	Memory location is read access only. A read always returns 0, unless otherwise specified.
W	Memory location is write access only.

MMR bits that are not documented are reserved. When writing to MMRs with reserved bits, the reserved bits should be written with the value in the reset column of the relevant MMR description, unless otherwise noted.

ACRONYMS AND ABBREVIATIONS

Table 3. Acronyms and Abbreviations

Acronym/Abbreviation	Description
ADC	Analog-to-digital converter
DMA	Direct memory access
DDM	Digital diagnostic monitoring
GPIO	General-purpose input and output
LSB	Least significant byte/bit
MDC	Management data input/output clock
MDIO	Management data input/output
MMD	MDIO manageable device (slave)
MMR	Memory mapped register
MSB	Most significant byte/bit
NMI	Nonmaskable interrupt
NVIC	Nested vectored interrupt controller
NVR	Nonvolatile registers
Rx	Receive
SAR	Successive approximation register
SOA	Semiconductor optical amplifier
SPI	Serial peripheral interface
STA	Station management entity (host/master)
SWD	Sync word detect/serial wire debug
Tx	Transmit
UART	Universal asynchronous transmitter
VR	Volatile registers
WDT	Watchdog timer
WUT	Wake-up timer

INTRODUCTION TO THE ADuCM320

The ADuCM320 is a fully integrated single-package device that incorporates high performance analog peripherals together with digital peripherals controlled by an 80 MHz ARM Cortex™-M3 processor and integral flash for code and data.

The ADC on the ADuCM320 provides 14-bit, 1 MSPS data acquisition on up to 16 input pins that can be programmed to be single-ended or differential. Additionally, chip temperature and supply voltages can be measured. The ADC input voltage is 0 V to V_{REF}. A sequencer is provided that allows a user selected set of ADC channels to be measured in sequence without software involvement during the sequence. The sequence can optionally auto repeat at a user-selectable rate.

Up to eight voltage DACs are provided with output ranges programmable to one of two voltage ranges. The DAC outputs have an enhanced feature of being able to retain their output voltage during a watchdog or software reset sequence. On the ADuCM320, four current output DAC sources are provided. The output currents are programmable with ranges of 0 mA to 150 mA. A low drift band gap reference and a voltage comparator complete the analog input peripheral set.

The microcontroller core is a low power ARM Cortex-M3 processor, a 32-bit RISC machine that offers up to 100 MIPS peak performance. Also integrated on chip are two 128 kB Flash/EE memory and 32 kB of SRAM. The flash comprises two separate 128 kB blocks supporting execution from one flash block and simultaneous writing/erasing of the other flash block.

The ADuCM320 operates from an on-chip oscillator or a 16 MHz external crystal and a PLL at 80 MHz. This clock can optionally be divided down to reduce current consumption. Additional low power modes can be set via software. In the normal operating mode, the ADuCM320 digital core consumes about 300 µA/MHz.

The device includes an MDIO interface capable of operating at up to 4 MHz. The capability to simultaneously execute from one flash block and write/erase the other flash block makes the ADuCM320 ideal for 40 G/100 G optical applications. User programming is eased by receiving interrupts after PHYADR, DEVADD, and end of frame and by having PHYADR and DEVADD hardware comparators. In addition, the nonerasable kernel code plus flags in user flash can provide assistance to allow user code to robustly switch between the two blocks of user flash code and data spaces as required for MDIO.

The ADuCM320 also integrates a range of on-chip peripherals that can be configured via software control as required in the application. These peripherals include one UART, two I^{2}Cs, two SPI serial I/O communication controllers, GPIO, 32-element programmable logic array, three general-purpose timers, one wake-up timer, and one system watchdog timer. In addition, 16-bit PWMs with seven output signals are provided.

GPIO pins on the device power up in input mode. In output mode, the software can choose between open-drain mode and push-pull mode. The outputs can drive at least 4 mA. The pull-ups can be disabled and enabled in software. In GPIO mode, the inputs can always be enabled to monitor the pins. The GPIO pins can also be programmed to handle digital or analog peripheral signals, in which case the pin characteristics are matched to the specific requirement.

A large support ecosystem is available for the ARM Cortex-M3 processor, which eases product development of the ADuCM320. Access is via the JTAG serial wire interface. On-chip factory firmware supports in-circuit serial download via MDIO. These features are incorporated in a low cost QuickStart development system supporting this precision analog microcontroller family.

MAIN FEATURES OF ADuCM320

ADC

- Multichannel, 14-bit, 1 MSPS SAR ADC
- Low drift on-chip voltage reference

DACs

- Eight voltage output DACs
  - VDACs are 12-bit monotonic
- Four current output DACs
  - Current DACs are 12-bit monotonic
- Low drift, on-chip 2.5 V voltage reference source
  - Two buffered reference outputs

Communication

- UART
    - Industry standard, 16450 UART peripheral
    - Support for DMA
- Two I^{2}Cs
    - 2-byte transmit and receive FIFOs for the master and slave
    - Support for DMA
- Two SPIs
    - Master or slave mode with separate 4-byte Rx and Tx FIFOs
    - Rx and Tx DMA channels
- 16-bit PWM with seven output channels
- Multiple GPIO pins

Processing

- ARM Cortex-M3 processor, operating from an internal 80 MHz system clock
- Two 128 kB Flash/EE memory, 32 kB SRAM
- In-circuit download and debug via serial wire
- On-chip MDIO download capability

On-Chip Peripherals

- Three general-purpose timers
- Wake-up timer
- Watchdog timer
- 32-element programmable logic array (PLA)

Packages and Temperature Range

- 6 mm × 6 mm, 96-ball BGA package, −40°C to +85°C

Tools

- Low cost development system
- Third-party compiler and emulator tool support

Applications

- Optical networking—10 G, 40 G, and 100 G modules
- Industrial control and automation systems
- Smart sensors, precision instrumentation
- Base station systems

MEMORY ORGANIZATION

The ADuCM320 memory organization is described in this section.

Features

- Cortex-M3 memory system features
    - Predefined memory map.
    - Support for bit-band operation for atomic operations.
    - Unaligned data access.
- ADuCM320 on-chip peripherals are accessed via memory mapped registers, situated in the bit-band region.
- User memory sizes options:
    - 32 kB SRAM
    - Two 128 kB Flash/EE memory
- On-chip kernel for manufacturer data and in-circuit download

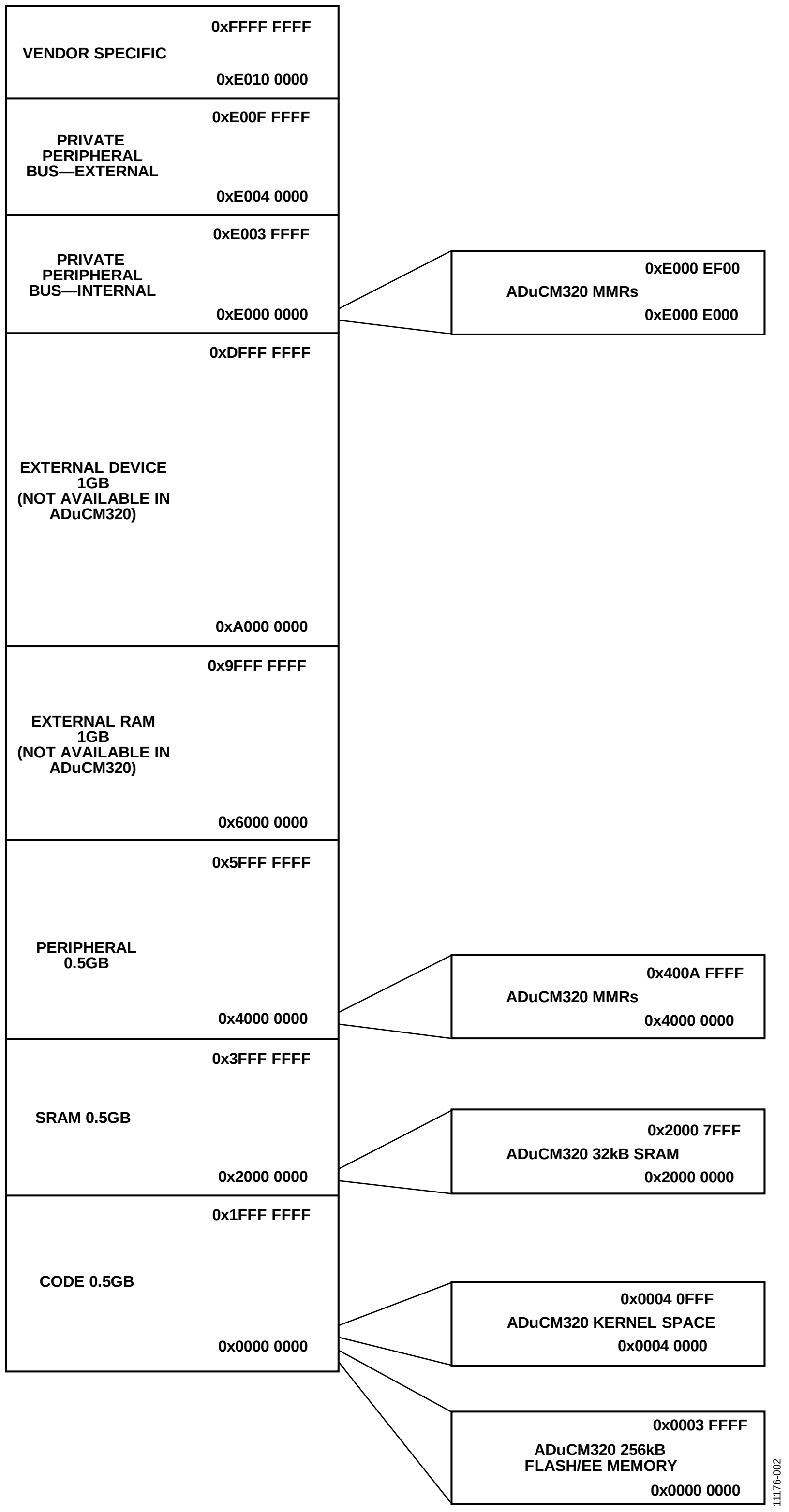


Figure 2. Cortex-M3 Memory Map Diagram

CLOCKING ARCHITECTURE

CLOCKING ARCHITECTURE FEATURES

The ADuCM320 integrates two on-chip oscillators and circuitry for an external crystal and external clock source:

- LFOSC is a 32 kHz low power internal oscillator that is used in low power modes.
- HFOSC is a 16 MHz internal oscillator that is used in active mode. This is the default input to the PLL.
- HFXTAL is a 16 MHz external crystal oscillator.
- External clock input (ECLKIN) via GPIO pin.

CLOCKING ARCHITECTURE BLOCK DIAGRAM

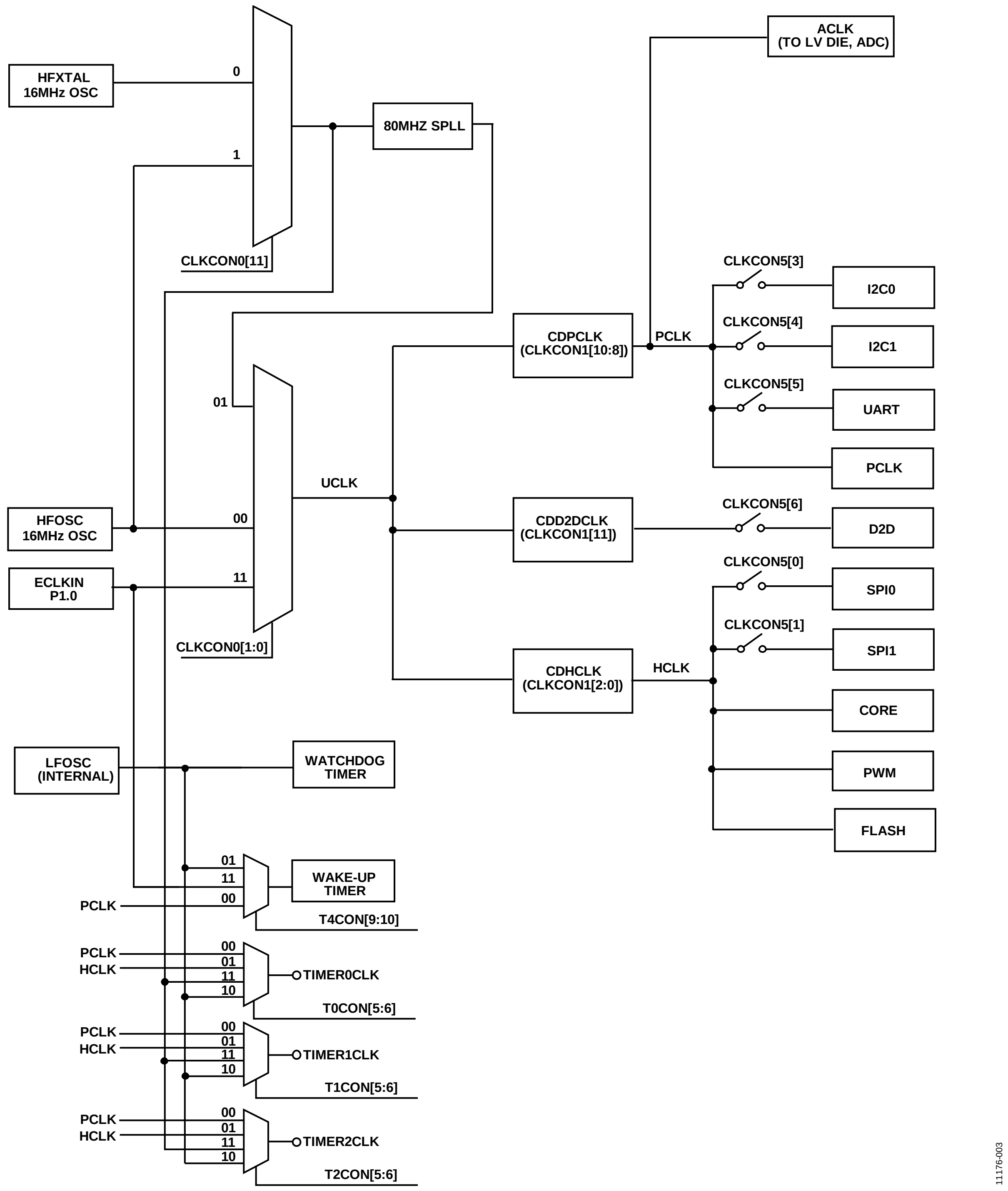


Figure 3. Clocking Architecture Block Diagram

CLOCKING ARCHITECTURE OVERVIEW

The system clock, UCLK, can be selected from a 16 MHz oscillator or from an 80 MHz PLL output (default). An external clock on P1.0 can also be used for test purposes.

Internally, the system clock is divided into separate clocks:

- UCLK system clock
- HCLK for the flash, SRAM, and DMA
- PCLK for most peripherals
- ACLK for the analog section of the chip; this is based on PCLK output and goes to the low voltage analog die

All ADC performance details are based on a 20 MHz ACLK (CLKCON1[10:8] = 0b010). Performance at other clock speeds is not guaranteed; therefore, CLKCON1[10:8] should not be changed when the ADC is being used.

REGISTER SUMMARY: CLOCK ARCHITECTURE

Table 4. Clocking Register Summary

Address	Name	Description	Reset	RW
0x40028000	CLKCON0	Misc clock settings register	0x0041	RW
0x40028004	CLKCON1	Clock dividers register	0x0200	RW
0x40028014	CLKCON5	User clock gating control register	0x0040	RW
0x40028018	CLKSTAT0	Clocking status	0x0000	RW

CLOCKING ARCHITECTURE OPERATION

At power-up, the processor executes at 80 MHz, sourced from the 80 MHz PLL output. The clock source for the 80 MHz PLL is the internal 16 MHz oscillator by default. User code can select the clock source for the system clock and can divide the clock by a factor of 1 to 128, where the clock divider bits are controlled by CLKCON1[2:0]. Slower code execution and reduced power consumption result.

Note that P1.0 must be configured as a clock input before the clock source is switched in the clock control register.

When changing from one clock source to a different clock source, the user code must ensure that both clock sources are kept active for a minimum of five clock cycles to ensure that the clock switching is fully completed without any glitches.

If the clock source for the 80 MHz SPLL needs to be changed from the internal 16 MHz oscillator to the external HFXTAL, observe the following procedure:

1. Check that HFXTAL is stable by reading CLKSTAT0[14:12].
2. Change the system clock to the internal 16 MHz oscillator using CLKCON0[1:0].
3. Wait 5 × 16 MHz clock cycles.
4. Switch the input to the SPLL using CLKCON0[11].
5. Wait until the SPLL has locked by monitoring CLKSTAT0[2:0].
6. Change the system clock back to the SPLL clock.

REGISTER DETAILS: CLOCK ARCHITECTURE

Misc Clock Settings Register

Address: 0x40028000, Reset: 0x0041, Name: CLKCON0

Table 5. Bit Descriptions for CLKCON0

Bits	Bit Name	Description	Reset	Access
15	HFXTALIE	High frequency crystal interrupt enable. 0: an interrupt to the core is not generated on a HFXTAL ok or HFXTAL nok 1: an interrupt to the core is generated on a HFXTAL ok or HFXTAL nok	0x0	RW
14	RESERVED	Reserved.	0x0	RW
13	SPLLIE	SPLL interrupt enable. 0: SPLL interrupt is not generated 1: SPLL interrupt is generated	0x0	RW
12	RESERVED	Reserved.	0x0	R
11	PLLMUX	PLL source selection. 0: internal oscillator is selected (HFOSC) 1: external oscillator is selected (HFXTAL)	0x0	RW
[10:8]	RESERVED	Reserved.	0x0	RW
[7:4]	CLKOUT	GPIO clock out selection. 0000: UCLK 0001: LFOSC (32 kHz) 0010: HFOSC(16 MHz) 0100: Core Clock 0101: PCLK 1011: General Purpose Timer0 clock 1100: Wake-up timer clock 1110: HFXTAL All other combinations are reserved	0x4	RW
[3:2]	RESERVED	Reserved.	0x0	R
[1:0]	CLKMUX	Clock Selection 00: high frequency internal oscillator (HFOSC) 01: SPLL is selected (80 MHz) 10: reserved 11: external GPIO port is selected (ECLKIN)	0x1	RW

Clock Dividers Register

Address: 0x40028004, Reset: 0x0200, Name: CLKCON1

Table 6. Bit Descriptions for CLKCON1

Bits	Bit Name	Description	Reset	Access
[15:12]	RESERVED	Reserved.	0x0	R
11	CDD2DCLK	D2DCLK divide bits. 0: D2D_CLK frequency = HCLK frequency. 1: D2D_CLK frequency = half of HCLK frequency.	0x0	R
[10:8]	CDPCLK	PCLK divide bits. PCLK divide bits. 000: reserved. 001: reserved. 010: DIV4. Divide by 4 (PCLK is quarter the frequency of root clock, 20 MHz). All ADC specifications are based on this setting. Using any other setting may affect ADC performance. 011: DIV8. Divide by 8. 100: DIV16. Divide by 16. 101: DIV32. Divide by 32. 110: DIV64. Divide by 164. 111: DIV128. Divide by 128.	0x2	RW
[7:3]	RESERVED	Reserved. Always returns 0 when read.	0x0	R
[2:0]	CDHCLK	HCLK divide bits. 000: DIV1. Divide by 1 (HCLK is equal to root clock). 001: DIV2. Divide by 2 (HCLK is half the frequency of root clock). 010: DIV4. Divide by 4 (HCLK is quarter the frequency of root clock). 011: DIV8. Divide by 8. 100: DIV16.Divide by 16. 101: DIV32.Divide by 32. 110: DIV64.Divide by 64. 111: DIV128. Divide by 128.	0x0	RW

User Clock Gating Control Register

Address: 0x40028014, Reset: 0x0040, Name: CLKCON5

The user clock gating control register (CLKCON5) is used to control the gates of the peripheral UCLKs.

Table 7. Bit Descriptions for CLKCON5

Bits	Bit Name	Description	Reset	Access
[15:7]	RESERVED	Reserved. Always returns 0 when read.	0x0	R
6	RESERVED	Reserved. Always set to 1.	0x1	RW
5	UCLKUARTOFF	UART clock user control. This bit disables the UCLK_UART clock. It controls the gate on UCLK_UART in Power Mode 0 and Power Mode 1. In Power Mode 2 and Power Mode 3, the UCLK_UART is always off and this bit has no effect. 0: clock on 1: clock off	0x0	RW
4	UCLKI2C1OFF	I2C1 clock user control. This bit disables the PCLK_I2C1 clock. It controls the gate on PCLK_I2C1 in Power Mode 0 and Power Mode 1. In Power Mode 2 and Power Mode 3, the I2C1 PCLK is always off and this bit has no effect. 0: clock on 1: clock off	0x0	RW
3	UCLKI2C0OFF	I2C0 clock user control. This bit disables the PCLK_I2C0 clock. It controls the gate on PCLK_I2C0 in Power Mode 0 and Power Mode 1. In Power Mode 2 and Power Mode 3 the PCLK_I2C0 is always off and this bit has no effect. 0: clock on 1: clock off	0x0	RW

Bits	Bit Name	Description	Reset	Access
2	RESERVED	Reserved.	0x0	R
1	UCLKSPI1OFF	SPI1 clock user control. This bit disables the UCLK_SPI1 clock. It controls the gate on UCLK_SPI1 in Power Mode 0 and Power Mode 1. In Power Mode 2 and Power Mode 3 the UCLK_SPI1 is always off and this bit has no effect. 0: clock on 1: clock off	0x0	RW
0	UCLKSPI0OFF	SPI0 clock user control. This bit disables the UCLK_SPI0 clock. It controls the gate on UCLK_SPI0 in Power Mode 0 and Power Mode 1. In Power Mode 2 and Power Mode 3, the UCLK_SPI0 is always off and this bit has no effect. 0: clock on 1: clock off	0x0	RW

Clocking Status Register

Address: 0x40028018, Reset: 0x0000, Name: CLKSTAT0

The clock status register is used to monitor PLL and oscillator status.

Table 8. Bit Descriptions for CLKSTAT0

Bits	Bit Name	Description	Reset	Access
15	RESERVED	Reserved. Always returns 0 when read.	0x0	R
14	HFXTALNOK	HF crystal not stable. This bit is sticky. It is used to interrupt the core when interrupts are enabled. Write a 1 to this location to clear it. 0: HF crystal stable signal has not been deasserted. 1: HF crystal stable signal has been deasserted.	0x0	RW
13	HFXTALOK	HF crystal stable. This bit is sticky. It is used to interrupt the core when interrupts are enabled. Write a 1 to this location to clear it. 0: HF crystal stable signal has not been asserted. 1: HF crystal stable signal has been asserted.	0x0	RW
12	HFXTALSTATUS	HF crystal status. 0: HF crystal is not stable or not enabled. 1: HF crystal is stable.	0x0	R
[11:3]	RESERVED	Reserved.	0x0	R
2	SPLLUNLOCK	System PLL unlock. This bit is sticky. SPLLUNLOCK is set when the PLL loses its lock. SPLLUNLOCK is used as the interrupt source to signal the core that a lock was lost. Writing a 1 to this bit clears it. SPLLUNLOCK does not set again unless the system PLL gains a lock and subsequently loses it again. 0: no loss of PLL lock was detected. 1: a PLL loss of lock was detected.	0x0	RW
1	SPLLLOCK	System PLL lock. This bit is sticky. SPLLLOCK is set when the PLL locks. SPLLLOCK is used as the interrupt source to signal the core that a lock was detected. Writing a 1 to this bit clears it. SPLLLOCK does not set again unless the system PLL loses lock and subsequently locks again. 0: no PLL lock event was detected. 1: a PLL lock event was detected.	0x0	RW
0	SPLLSTATUS	System PLL status. Indicates the current status of the PLL. Initially the system PLL is unlocked. After a stabilization period, the PLL locks and is ready for use as the system clock source. This is a read only bit. A write has no effect. 0: the PLL is not locked or not properly configured. The PLL is not ready for use as the system clock source. 1: the PLL is locked and is ready for use as the system clock source.	0x0	R

POWER MANAGEMENT UNIT

POWER MANAGEMENT UNIT FEATURES

The power management unit (PMU) controls the different power modes of the ADuCM320.

Four power modes are available:

- Active
- CORE_SLEEP
- SYS_SLEEP
- Hibernate

POWER MANAGEMENT UNIT OVERVIEW

The Cortex-M3 sleep modes are linked to the PMU modes and are described in this section. The PMU is in the always-on section. Each mode gives a power reduction benefit with a corresponding reduction in functionality.

POWER MANAGEMENT UNIT OPERATION

The debug tools can prevent the Cortex-M3 from fully entering its power saving modes by setting bits in the debug logic. Only a power-on reset resets the debug logic. Therefore, the device should be power cycled after using serial wire debug with application code containing the WFI instruction.

Power Mode: Active Mode, Mode 0

The system is fully active. Memories and all user enabled peripherals are clocked, and the Cortex-M3 processor is executing instructions. Note that the Cortex-M3 processor manages its internal clocks and can be in a partial clock gated state. This clock gating affects only the internal Cortex-M3 processing core. Automatic clock gating is used on all blocks and is transparent to the user. User code can use a WFI command to put the Cortex-M3 processor into sleep mode; it is independent of the power mode settings of the PMU.

When the ADuCM320 wakes up from any of the low power modes, the device return to Mode 0.

Power Mode: CORE_SLEEP Mode, Mode 1

In CORE_SLEEP mode, the system gates the clock to the Cortex-M3 core after the Cortex-M3 has entered SLEEP mode. The rest of the system remains active. No instructions can be executed; however, DMA transfers can continue to occur between peripherals and memories. The Cortex-M3 processor FCLK is active, and the device wakes up using the NVIC.

Power Mode: SYS_SLEEP Mode, Mode 2

In SYS_SLEEP mode, the system gates HCLK (system bus clock) and PCLK (peripheral bus clock) after the Cortex-M3 has entered sleep mode. The gating of these clocks stops all AHB attached masters/slaves and all peripherals attached to APB. Peripheral clocks are all off, and they are no longer user programmable. The NVIC (interrupt controller) clock (FCLK) remains active, and the NVIC processes wake-up events.

Power Mode: Hibernate Mode, Mode 3

In hibernate mode, the system disables power to all combinational logic and places sequential logic in retain mode. Because FCLK is stopped, the number of sources capable of waking up the system is restricted. The sources listed in Table 55 are the only sources able to wake up the system.

Power Mode 1 to Power Mode 3 should be entered when the processor is not in an interrupt handler. If Power Mode 1 to Power Mode 3 is entered when the processor is in an interrupt handler, the power-down mode can be exited only by a reset or a higher priority interrupt source.

CODE EXAMPLES

Code Example to Enter Power Saving Modes

```
SCB->SCR = 0x04;                  // sleepdeep mode
pADI_PWRCTL->PWRKEY = 0x4859;     // key1
pADI_PWRCTL->PWRKEY = 0xF27B;     // key2
pADI_PWRCTL->PWRMOD = 0x3;       // Hibernate
__DSB();
__nop();
__nop();
__nop();
__nop();
__nop();
__nop();
__nop();
__nop();
__nop();
__nop();
__nop();
__nop();
__nop();
__nop();
__nop();
__nop();
__WFI();
__nop();
__nop();
__nop();
__nop();
__nop();
__nop();
```

Code Example to Achieve Further Power Savings

```
pADI_ADC->ADCCON = 0;                 // Power off the ADC
pADI_IDAC0->IDACCON = 0x1;            // Turn off IDAC0
pADI_IDAC1->IDACCON = 0x1;            // Turn off IDAC1
pADI_IDAC2->IDACCON = 0x1;            // Turn off IDAC2
pADI_IDAC3->IDACCON = 0x1;            // Turn off IDAC3
pADI_VDAC0->DACCON = 0x100;           // Turn off VDAC0
pADI_VDAC1->DACCON = 0x100;           // Turn off VDAC1
pADI_VDAC2->DACCON = 0x100;           // Turn off VDAC2
pADI_VDAC3->DACCON = 0x100;           // Turn off VDAC3
pADI_VDAC4->DACCON = 0x100;           // Turn off VDAC4
pADI_VDAC5->DACCON = 0x100;           // Turn off VDAC5
pADI_VDAC6->DACCON = 0x100;           // Turn off VDAC6
pADI_VDAC7->DACCON = 0x100;           // Turn off VDAC7
pADI_CLKCTL->CLKCON0 &= 0xFFFC;       // Switch to 16MHz clock
pADI_CLKCTL->CLKCON1 = 0x505;         // Slow down system clocks
pADI_CLKCTL->CLKCON5 = 0x7B;          // Turn off clocks to peripherals
```

REGISTER SUMMARY: POWER MANAGEMENT UNIT

Table 9. Power Management Register Summary

Address	Name	Description	Reset	RW
0x40002400	PWRMOD	Power modes	0x0000	RW
0x40002404	PWRKEY	Key protection for PWRMOD	0x0000	RW

REGISTER DETAILS: POWER MANAGEMENT UNIT

Power Modes Register

Address: 0x40002400, Reset: 0x0000, Name: PWRMOD

Table 10. Bit Descriptions for PWRMOD

Bits	Bit Name	Description	Reset	Access
[14:2]	RESERVED	Reserved. These bits should be written 0 by user code.	0x0	R
[1:0]	PWRMOD	Power modes control bits. When read, these bits contain the last power mode value entered by user code. Note that, to place the Cortex in sleepdeep mode for hibernate, the Cortex-M3 system control register (Address 0xE000ED10) must be configured to 0x4 or 0x06. 00: active mode 01: CORE_SLEEP mode 10: SYS_SLEEP mode 11: hibernate mode	0x0	RW

Key Protection for PWRMOD Register

Address: 0x40002404, Reset: 0x0000, Name: PWRKEY

Table 11. Bit Descriptions for PWRKEY

Bits	Bit Name	Description	Reset	Access
[15:0]	PWRKEY	Power control key register. The PWRMOD register is key-protected. Two writes to the key are necessary to change the value in the PWRMOD register: first 0x4859, then 0xF27B. The PWRMOD register should then be written. A write to any other register before writing to PWRMOD returns the protection to the lock state.	0x0	RW

ARM CORTEX-M3 PROCESSOR

ARM CORTEX-M3 PROCESSOR FEATURES

High Performance

- 1.25 DMIPS/MHz.
- Many instructions, including multiply, are single cycle.
- Separate data and instruction buses allow simultaneous data and instruction accesses to be performed.
- Optimized for single-cycle flash usage.

Low Power

- Low standby current.
- Core implemented using advanced clock gating so that only the actively used logic consumes dynamic power.
- Power-saving mode support (sleep and deep sleep modes). The design has separate clocks to allow unused parts of the processor to be stopped.

Advanced Interrupt Handling

- The nested vectored interrupt controller (NVIC) supports up to 240 interrupts. The ADuCM320 supports 49 of these interrupts. The vectored interrupt feature greatly reduces interrupt latency because there is no need for software to determine which interrupt handler to serve. In addition, there is no need to have software to set up nested interrupt support.
- The ARM Cortex-M3 processor automatically pushes registers onto the stack at the entry interrupt and retrieves them at the exit interrupt. This reduces interrupt handling latency and allows interrupt handlers to be normal C functions.
- Dynamic priority control for each interrupt.
- Latency reduction using late arrival interrupt acceptance and tail-chain interrupt entry.
- Immediate execution of a nonmaskable interrupt request for safety critical applications.

System Features

- Support for bit-band operation and unaligned data access.
- Advanced fault handling features include various exception types and fault status registers.

Debug Support

- Serial wire debug interfaces (SW-DP).
- Flash patch and breakpoint (FPB) unit for implementing breakpoints. Limited to two hardware breakpoints.
- Data watchpoint and trigger (DWT) unit for implementing watchpoints trigger resources and system profiling. Limited to one hardware watchpoint. The DWT does not support data matching for watchpoint generation because it only has one comparator.

ARM CORTEX-M3 PROCESSOR OVERVIEW

The ADuCM320 contains an embedded ARM Cortex-M3 processor, Revision r2p1. The ARM Cortex-M3 processor provides a high performance, low cost platform that meets the system requirements of minimal memory implementation, reduced pin count, and low power consumption while delivering outstanding computational performance and exceptional system response to interrupts.

ARM CORTEX-M3 PROCESSOR OPERATION

Several ARM Cortex-M3 processor components are flexible in their implementation. This section details the actual implementation of these components in the ADuCM320.

Serial Wire Debug (SW/JTAG-DP)

The ADuCM320 only supports the serial wire interface via the SWCLK and SWDIO pins. It does not support the 5-wire JTAG interface.

ROM Table

The ADuCM320 implements the default ROM table.

Nested Vectored Interrupt Controller Interrupts (NVIC)

The ARM Cortex-M3 processor includes a nested vectored interrupt controller (NVIC), which offers several features:

- Nested interrupt support
- Vectored interrupt support
- Dynamic priority changes support
- Interrupt masking

In addition, the NVIC has a nonmaskable interrupt (NMI) input.

The NVIC is implemented on the ADuCM320, and more details are available in the System Exceptions and Peripheral Interrupts section.

Wake-Up Interrupt Controller (WIC)

The ADuCM320 has a modified WIC, which provides the lowest possible power-down current. This feature is transparent to the user and more details are available in the Power Management Unit section. It is not recommended to enter a power saving mode while servicing an interrupt. However, if the part does enter a power saving mode while servicing an interrupt, it can be woken up by only a higher priority interrupt source.

μDMA

The ADuCM320 implements the ARM μDMA. More details are available in the DMA Controller section.

ARM CORTEX-M3 PROCESSOR RELATED DOCUMENTS

- Cortex-M3 Revision r2p1 Technical Reference Manual (DDI 0337)
- ARM Processor Cortex-M3 (AT420) and Cortex-M3 with ETM AT425): Errata Notice
- ARMv7-M Architecture Reference Manual (DDI 0403)
- ARMv7-M Architecture Reference Manual Errata Markup
- ARM Debug Interface v5 Architecture Specification (IHI 0031)
- PrimeCell μDMA Controller (PL230) Technical Reference Manual Revision r0p0 (DDI 0417)

ADC CIRCUIT

ADC CIRCUIT FEATURES

- The ADuCM320 incorporates a fast, multichannel, 16-bit ADC. The ADC is specified to be 14-bit accurate.
- Flexible input multiplexer supporting 16 external inputs and 11 internal channels. The internal channels include the following:
  - Temperature sensor channel.
  - Internal 2.51 V reference.
  - External reference.
  - 4 × IDAC channels. These are the voltage at each of the IDAC output pins.
  - PVDD2 supply voltage.
  - IOVDD/2 supply voltage.
  - AVDD/2 supply voltage.
- Input buffer can be selected for any channel to allow very low input current/input leakage specifications on these input channels.
- High precision, low drift internal 2.51 V reference source.
- An external reference can also be connected to the ADC_REFP and ADC_REFN pins.
- Programmable ADC update rate from 19.55 kSPS to 1 MSPS.
- Internal digital comparator for the AIN4 channel. An interrupt can be generated if the digital comparator detects an ADC result above/below a user defined threshold.
- Each channel has its own distinct data register for its conversion result. For example, when AIN0 is selected, the result appears in ADCDAT0; if AIN7 is selected, the result appears in ADCDAT7. For a differential measurement, the result always appears in the data register of the positive channel.

ADC CIRCUIT BLOCK DIAGRAM

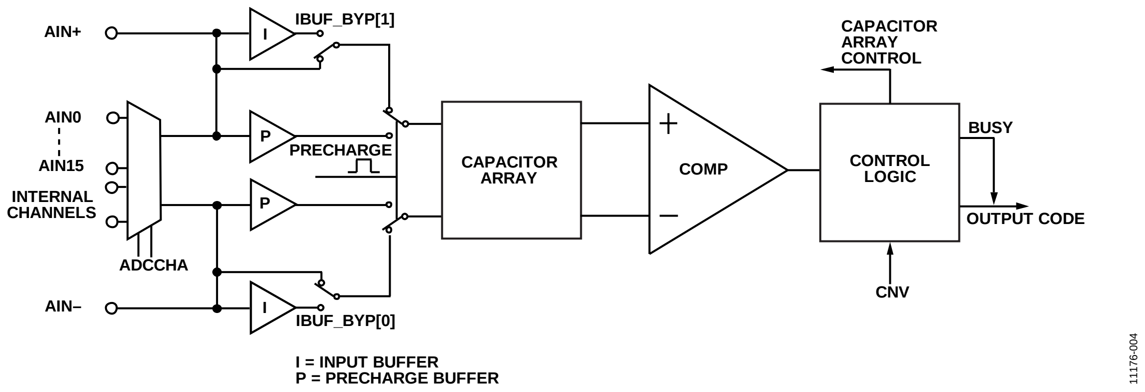


Figure 4. ADC Circuit Block Diagram

ADC CIRCUIT OVERVIEW

The ADuCM320 incorporates a fast, multichannel, 16-bit ADC. The ADC is specified to be 14-bit accurate. It can operate from a 2.9 V to 3.6 V supply and is capable of providing a throughput of up to 1 MSPS. This ADC block provides the user with a multichannel multiplexer, input buffer for high impedance input channels, on-chip reference, and successive approximation register (SAR) ADC.

The SAR ADC circuit is implemented on the low voltage analog die. The ARM Cortex-M3 processor interfaces to the ADC via an internal parallel die-to-die interface.

Depending on the input signal configuration, the ADC can operate in one of the following two modes:

- Differential mode—to measure the difference between two signals.
- Single-ended mode—to measure any signal relative to AGND.

The converter accepts an analog input range of 0 to V_{REF} when operating in single-ended mode. In fully differential mode, the input signal must be balanced around a common-mode voltage (V_{CM}) in the range 0 V to AV_{DD} and with a maximum amplitude of 2 × V_{REF}.

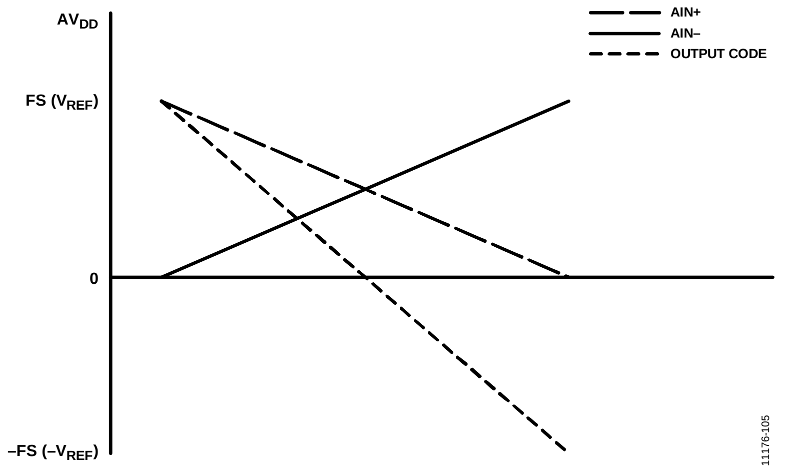


Figure 5. Examples of Balanced Signals for Differential Mode

A high precision, low drift, factory-calibrated 2.51 V reference is provided on chip. An external reference can also be connected to the ADC_REFP and ADC_REFN pins.

Single or continuous conversion modes can be initiated in software. An external pin (alternate function of P2.4), can also be used to generate a repetitive trigger for ADC conversions.

ADC CIRCUIT OPERATION

The SAR ADC is based on a charge redistribution DAC. The capacitive DAC consists of two identical arrays of 18 binary weighted capacitors that are connected to the two inputs of the comparator.

The ADC converts the voltage applied to AIN+ and AIN− in the following three steps:

1. Precharge phase: During this phase, the precharge buffers connect the inputs to the capacitor arrays. This charges the capacitors quickly with minimal loading of the external input source.
2. Acquisition phase: During the acquisition phase, the capacitor arrays are connected directly to the inputs to fully charge the capacitor arrays and eliminate any precharge buffer errors. The timing for the acquisition phase is set by ADCCNVC[25:16]. This value should be set to 500 ns. If the input buffer is not used when measuring AVDD/2, IOVDD/2, or temperature sensor channels, set this value to 1.5 μs.
3. Conversion phase: At the end of the acquisition phase, the internal CNV signal goes high and initiates the conversion phase. The conversion begins with the SW+ and SW− switches being opened. This disconnects the two capacitor arrays from the analog inputs and connects the analog inputs to the AGND (VREF−) input. The conversion is completed by normal successive approximation.

The ADC block operates from an internally generated 20 MHz clock.

The ADC conversion rate is set by ADCCNVC[9:0].

ADC TRANSFER FUNCTION

Single-Ended Mode

In single-ended mode, the input range is 0 to V_{REF}. The output coding is straight binary with

1 LSB = *FS*/65,536; or,

V_{REF}/65,536 = 2.51 V/65,536 = 38.30 μV

The data values in ADCDATx are aligned such that the MSB is in ADCDATx[27] and, therefore, the LSB is in ADCDATx[12]. The ideal code transitions occur midway between successive integer LSB values (that is, 1/2 LSB, 3/2 LSB, 5/2 LSB, …, FS − 3/2 LSB). The ideal input/output transfer characteristic is shown in Figure 6.

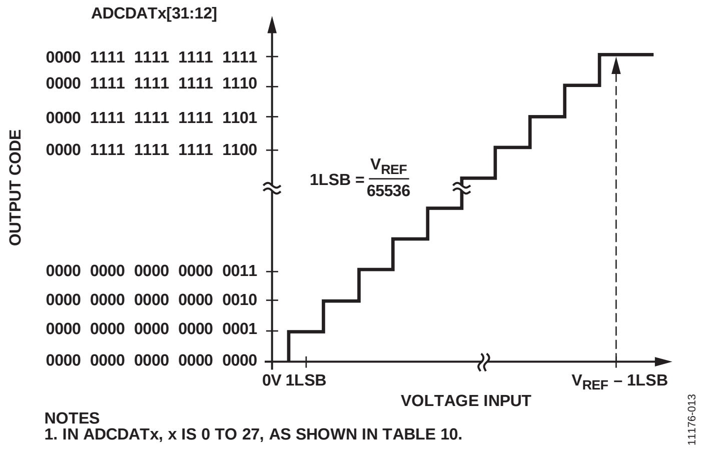


Figure 6. ADC Transfer Function: Single-Ended Mode

Differential Mode

The amplitude of the differential signal is the difference between the signals applied to the AIN+ and AIN− pins (that is, AIN+ − AIN−). The maximum amplitude of the differential signal is, therefore, $-V_{REF}$ to $+V_{REF}$ p-p (2 × V_{REF}). This is regardless of the common mode (CM). The common mode is the average of the two signals (AIN+ + AIN−)/2 and is, therefore, the voltage that the two inputs are centered on. This results in the span of each input being CM ± V_{REF}/2. This voltage must be set up externally, and its range varies with V_{REF}. The voltage at the AIN+ and AIN− pins must be within the allowed input voltage range.

The output coding is twos complement in fully differential mode, with

1 LSB = 2 × V_{REF}/65,536; or,

2 × 2.51 V/65,536 = 76.60 μV

where V_{REF} = 2.51 V.

The data values in ADCDATx are aligned such that the MSB is in ADCDATx[27] and, therefore, the LSB is in ADCDATx[13]. The ideal input/output transfer characteristic is shown in Figure 7.

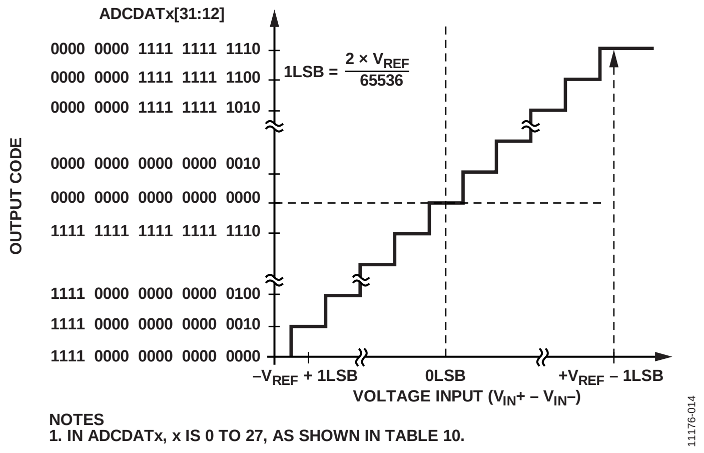


Figure 7. ADC Transfer Function: Differential Mode

ADC TYPICAL SETUP SEQUENCE

After being configured via the ADC control and channel selection registers, the ADC converts the analog input and provides a 16-bit result in the ADC data registers.

The following is an example sequence to set up the ADC and generate a single conversion on AIN0 using a single-ended measurement:

1. Configure the device as follows:

```
ADCCON = 0x280;             // Power up the ADC, enable reference buffer, idle mode.
ADCCHA = 0x1100;            // Select AIN0 as the positive ADC input (AIN+) and ADC_REFN as
the negative ADC input (AIN−).
ADCCNVC = 0xA00C8;          // Select 100 kSPS ADC update rate and 500 ns acquisition time.
ADCCON | = 0x2;             // Enable single conversion.
```

2. Wait for LV die interrupt
3. iADCRESULT = ADCDAT0;   // Read the ADC result.)

Note that if the ADC is set from continuous conversion mode to idle mode after a full ADC sequence is completed, ADCSEQ[31] must be set to 1 before starting another sequence and reconfiguring the ADC back to continuous conversion mode. This ensures that the sequencer restarts with the first selected channel in ADCSEQ.

ADC INPUT BUFFER

An optional input buffer can be enabled for any ADC input channel on the ADuCM320.

The control register IBUFCON controls the input buffer switches as follows:

- IBUF_BYP (IBUFCON[1:0]) controls the bypass switches on the ADC input buffer. If the input buffer is required on either the positive or negative input, the bypass switch must be turned off.
- IBUF_PD (IBUFCON[3:2]) powers up or powers down the ADC input buffer.

ADC INTERNAL CHANNELS

Temperature Sensor Settings

The ADuCM320 provides a voltage output from an on-chip band gap reference that is proportional to the absolute temperature of the low voltage die. This voltage output is routed through the front end of the ADC multiplexer (effectively, an additional ADC channel input), facilitating an internal temperature sensor channel that measures die temperature.

The internal temperature sensor is not designed for use as an absolute ambient temperature calculator. Its intended use is as an approximate indicator of the temperature of the ADuCM320 low voltage analog die.

An ADC temperature sensor conversion differs from a standard ADC voltage. The ADC performance specifications do not apply to the temperature sensor.

When the temperature sensor channel is selected, the ADC update rate should be 80 kSPS.

The ADC automatically changes the ADC update rate to 80 kSPS when the temperature sensor, AVDD/2, or IOVDD/2 input channel is selected. If a different ADC sampling rate is required for other channels after the conversion on any of these three channels is completed, the ADCCNVC register must be updated.

Note that when the sequencer is enabled and includes any of these three channels, the value in the ADCCNVC register does not change and the ADC sampling rate does not change.

The temperature sensor settings are as follows:

Enable the temperature sensor on the ADC; set ADCCHA[12:0] = 0x1116.

To calculate the die temperature, use the following formula:

$$T - T_{REF} = (V_{ADC} - V_{TREF}) \times K$$

where:
T is the temperature result.
T_{REF} is 25°C.
V_{ADC} is the average ADC result from two consecutive conversions.
V_{TREF} is the ADC result in millivolts that corresponds to T_{REF} = 25°C. The user must measure this in their own application because this value varies from device to device. The typical value used for demonstration purposes is 1290 mV.
K is the gain of the ADC in temperature sensor mode. The user should determine the gain by performing a two-point temperature calibration because this value varies from device to device. The typical value used for demonstration purposes only is 4.394 mV/°C.

This corresponds to 1/V TC.

Using the default values from the ADuCM320 data sheet without any calibration, the equation becomes

$$T - 25°C = (V_{ADC} - 1290) \times 1/K.$$

Therefore, assuming V_{ADC} at 25°C = 1290 mV and slope mV/C = 4.394 mV/C,

$$T = ((V_{ADC} - 1290)/4.394) + 25$$

where:
V_{ADC} is in millivolts.

Check the latest version of the ADuCM320 data sheet for the most up to date figures.

For increased accuracy, perform a two-point calibration at a controlled temperature value.

The values used in this example for V_{TREF} and K are not guaranteed values. The values V_{TREF} and K varies from device to device; therefore, the user must derive the appropriate values by performing a calibration at ambient temperature.

AVDD/2 and IOVDD/2 Supply Voltage Channels

These supply voltage channels are measured via internal resistor dividers. Because the resistors used are high impedance and the divided voltage is not buffered, a slower ADC update rate should be used.

The ADC automatically changes the ADC update rate to 80 kSPS when the temperature sensor, AVDD/2, or IOVDD/2 input channel is selected. If a different ADC sampling rate is required for other channels after the conversion on any of these three channels is completed, the ADCCNVC register must be updated.

Note that when the sequencer is enabled and includes any of these three channels, the value in the ADCCNVC register does not change and the ADC sampling rate does not change. At rates above 80 kSPS, the accuracy is reduced if the input buffer is disabled.

ADC SUPPORT CIRCUITS

IDAC Channels

The ADuCM320 allows the voltage on the IDAC output pins to be selected as inputs to the ADC. These channels are useful for determining the power consumed by each IDAC.

ADC Digital Comparator

A digital comparator is provided to allow an interrupt to be triggered if the ADC data result is above or below a programmable threshold. Only the AIN4 external input channel can be used with the digital comparator.

To set up the ADC digital comparator, note the following:

- ADCCMP[17:2] are used to set a 16-bit ADC threshold value.
- ADCCMP[1] configures the comparator to be triggered when the ADC result is above or below the threshold value.
- To enable the ADC comparator interrupt, set INTSEL[2] = 1 to enable the digital comparator to the Low Voltage Die Interrupt 1 signal.
- Similarly, set INTSEL[10] = 1 to enable the digital comparator interrupt to the Low Voltage Die Interrupt 0 signal.
- The comparator output is asserted when the value in ADCDAT4[27:12] rises above the value in ADCCMP[17:2] if ADCCMP[1] = 1. If ADCDAT4[27:12] remains above ADCCMP[17:2], no further comparator interrupts occur. The interrupt only occurs when the comparator circuit detects a rise above the threshold.
- Similarly, if ADCCMP[1] = 0, the comparator output is asserted when the value in ADCDAT4[27:12] falls below the value in ADCCMP[17:2]. If ADCDAT4[27:12] remains below ADCCMP[17:2], no further comparator interrupts occur. The interrupt only occurs when the comparator circuit detects a fall below the threshold value.

ADC Channel Sequencer

An ADC sequencer is provided to reduce the processor overhead of sampling and reading individual channels. The ADC sequencer allows a user to select the number and order of ADC input channels that the ADC samples and provides a single interrupt source that is asserted when the sequence ends. The sequencer can also be programmed to restart automatically without a delay or with a programmable delay between the end and start of sequences.

Some additional details about the sequencer include the following:

- The sequencer reads the ADCSEQ[0:27] register to determine which channels need to be included and which need to be excluded from the execution sequence.
- ADCSEQ corresponds to the ADCCHA[4:0] for the list of ADC input channels. For example, to include AIN9, set ADCSEQ[9].
- To enable the sequencer as the Low Voltage Die Interrupt 1 source, set INTSEL[1] = 1. To enable the sequencer as the Low Voltage Die Interrupt 0 source, set INTSEL[9] = 1.
- To start the sequencer, set ADCSEQ[31:30] = 0x3.
- The ADCSEQC[27:20] register bits are used to set the delay between finishing one sequence of channels and starting another sequence.
- Normally, single-ended measurements are assumed by the ADC with AGND as the negative reference. However, for Channel 0, Channel 2, Channel 4, and Channel 6, a differential measurement can be selected by configuring the appropriate bits in ADCSEQC[19:0]. For example, ADCSEQC[4:0] selects the negative input when AIN0 is the positive. For single-ended measurements using the sequencer and AIN0, ADCSEQC[4:0] should be set to 0x11 for VREFN_NADC (ADC_REFN pin).
- Care should be taken when using the sequencer if the input buffer is enabled. The IBUFCON register controls the input buffer. If the input buffer is enabled, all channels sampled in a sequence will be sampled with the input buffer enabled. It is recommended to split sequences into the following:
  - Sample unbuffered channels together in one sequence.
  - Sample buffered channels in a separate sequence.
  - If full accuracy results are required for the AVDD/2, IOVDD/2, or temperature channels, then care must be taken when measured with the sequencer.
    - With the input buffer disabled, the acquisition time should be set to 1.5 µs via ADCCONV[25:16] = 0x1E.
    - Or, alternatively, enable the input buffer.

ADC DMA (Direct Memory Access)

The ADC or the ADC sequencer can be selected as the source channel for the DMA controller. This reduces processor overhead by moving ADC results directly into SRAM with a single interrupt asserted when the required number of ADC conversions has been completely logged to memory.

When using the ADC sequencer with the DMA controller, it is recommended to use DMA autorequest transfer types rather than basic transfer types.

ADC Voltage Reference Selection

The ADuCM320 integrates a low drift, 2.5 V ADC reference source. By default, this internal reference is enabled and selected as the reference source for the ADC. When using the internal 2.5 V voltage reference, ensure the following:

- ADCCON[7] = 1 to power up the internal reference buffer
- AFEREFC[3] = 0 to select the internal reference as the ADC reference source

It is also possible to select an external reference source through the ADC_REFP pin.

To select an external voltage source as the ADC reference source, ensure the following:

- ADCCON[7] = 0 to power down the internal reference buffer
- AFEREFC[3] = 1 to select the external reference as the ADC reference source

The external reference source must be capable of driving the 4.7 µF capacitor on the ADC_REFP pin.

If switching from the external to internal reference voltage source, note that there is a power-on time specification given in the ADuCM320 data sheet for the ADC reference buffer to fully power up after ADCCON[7] is set to 1.

Figure 8 shows the block diagram of how the analog references are provided.

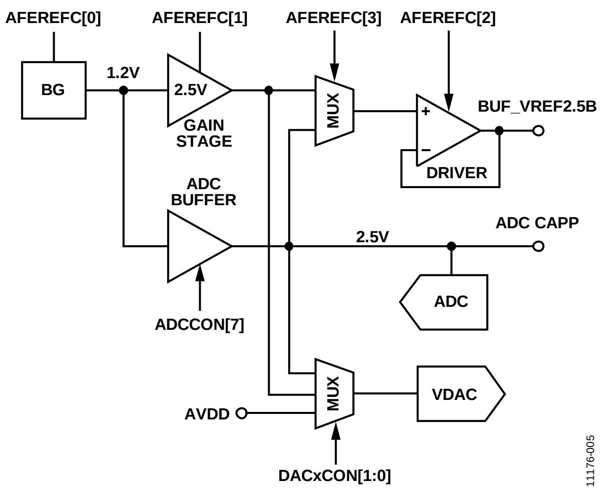


Figure 8. System Reference Voltage Block Diagram

REGISTER SUMMARY: ADC CIRCUIT

The CPU accesses the ADC circuit over a die to die interface (D2D) which increases the execution times of ldr and str instructions. 32 bit MMRs have addresses 0x40086xxx and take 8 CPU cycles at 80 MHz to execute. 16 bit MMRs have addresses 0x40082xxx and take 6 CPU cycles at 80 MHz to execute.

Table 12. ADC Circuit Register Summary

Address	Name	Description	Reset	RW
0x40082174	ADCCON	ADC configuration	0x0280	RW
0x40086000	ADCDAT0	ADC0 data and flags	Undefined	R
0x40086004	ADCDAT1	ADC1 data and flags	Undefined	R
0x40086008	ADCDAT2	ADC2 data and flags	Undefined	R
0x4008600C	ADCDAT3	ADC3 data and flags	Undefined	R
0x40086010	ADCDAT4	ADC4 data and flags	Undefined	R
0x40086014	ADCDAT5	ADC5 data and flags	Undefined	R
0x40086018	ADCDAT6	ADC6 data and flags	Undefined	R
0x4008601C	ADCDAT7	ADC7 data and flags	Undefined	R
0x40086020	ADCDAT8	ADC8 data and flags	Undefined	R
0x40086024	ADCDAT9	ADC9 data and flags	Undefined	R
0x40086028	ADCDAT10	ADC10 data and flags	Undefined	R
0x4008602C	ADCDAT11	ADC11 data and flags	Undefined	R
0x40086030	ADCDAT12	ADC12 data and flags	Undefined	R
0x40086034	ADCDAT13	ADC13 data and flags	Undefined	R
0x40086038	ADCDAT14	ADC14 data and flags	Undefined	R
0x4008603C	ADCDAT15	ADC15 data and flags	Undefined	R
0x40086040	ADCDAT16	ADC16 data and flags	Undefined	R
0x40086044	ADCDAT17	ADC17 data and flags	Undefined	R
0x40086048	ADCDAT18	ADC18 data and flags	Undefined	R
0x4008604C	ADCDAT19	ADC19 data and flags	Undefined	R
0x40086050	ADCDAT20	ADC20 data and flags	Undefined	R
0x40086054	ADCDAT21	ADC21 data and flags	Undefined	R
0x40086058	ADCDAT22	ADC22 data and flags	Undefined	R
0x4008605C	ADCDAT23	ADC23 data and flags	Undefined	R
0x40086060	ADCDAT24	ADC24 data and flags	Undefined	R
0x40086064	ADCDAT25	ADC25 data and flags	Undefined	R
0x40086068	ADCDAT26	ADC26 data and flags	Undefined	R
0x4008606C	ADCDAT27	ADC27 data and flags	Undefined	R
0x40086080	ADCCHA	ADC channel select	0x111F	RW
0x40086088	ADCSEQ	ADC sequencer control	0x00000000	RW
0x4008608C	ADCSEQC	ADC sequencer configuration	0x0008C631	RW
0x40086090	RESERVED	Reserved	Not applicable	Not applicable
0x40086094	RESERVED	Reserved	Not applicable	Not applicable
0x40086098	ADCCMP	Digital comparator configuration	0x00000	RW
0x4008609C	ADCCNVC	ADC conversion configuration	0x000A00C8	RW

REGISTER DETAILS: ADC CIRCUIT

ADC Configuration Register

Address: 0x40082174, Reset: 0x0280, Name: ADCCON

Table 13. Bit Descriptions for ADCCON

Bits	Bit Name	Description	Reset	Access
[15:11]	RESERVED	Reserved.	0x0	R
10	SOFT_RESET	Software reset ADC.	0x0	W
9	PUP	ADC power up. 0: power down 1: power up	0x1	RW
8	RESERVED	Reserved.	0x0	R
7	REFB_PUP	ADC reference buffer power up. 0: power down 1: power up Must be set to 1 for the ADC to operate normally	0x1	RW
6	RESTART_ADC	Restart ADC, reset analog part of ADC. Active high. 0: normal ADC operation. 1: reset the ADC.	0x0	W
5	RESERVED	Reserved.	0x0	R
4	SEQ_DMA	DMA request enable for ADC sequence conversion. 0: disable ADC sequencer DMA access 1: enable ADC sequencer DMA access	0x0	RW
3	CNV_DMA	DMA request enable for ADC non-sequence conversion. 0: disable ADC DMA access 1: enable ADC DMA access	0x0	RW
[2:0]	C_TYPE	ADC conversion type. 00: no conversion 01: DIO pin starts conversion (P2.4) 10: single conversion 11: continuous conversion (use this mode for the sequencer) 100: PLA conversion	0x0	RW

ADCx Data and Flags Register

Address: 0x40086000 to 0x4008606C (Increments of 0x4), Reset: 0x00000000, Name: ADCDAT0 to ADCDAT27

At the end of each conversion, the ADC writes the data to the appropriate ADCDATx MMR, where x is 0 to 27. This process takes 2 ADC clock cycles, which at 20 MHz means 100 ns. During this time, the value in ADCDATx cannot be read reliably by the CPU. For this reason during this time ADCDATx is forced to zero and specifically Bit ADCDATx[3] is zero. Therefore, if ADCDATx is read at random times, ADCDATx[3] should be checked, and if it is zero ADCDATx should be read again. This second read must be at least 100 ns later, which is basically guaranteed by the time used to check the bit plus the time required to read the value via the D2D interface. Make sure that the second read does not coincide with any further conversion on that channel. Alternately, perform repeated reads until the read is successful. At 1 MSPS conversion speed, the read is valid 90% of the time, while at 100 kSPS, it is valid 99% of the time. When using interrupts, this problem does not occur unless the read happens exactly when a subsequent ADC conversion completes on that channel. This behavior is valid for all conversion modes (single conversions, repeated conversions, and sequencer conversions).

Table 14. Bit Descriptions for ADCDAT0 to ADCDAT27

Bits	Bit Name	Description	Reset	Access
[31:4]	DAT	ADCx data. The numeric value of the conversion is stored in bits 12 to 27. Bit 28 to Bit 31 are the extended sign bits. Bit 4 to Bit 11 are always zero. The format is twos complement (signed int).	0x0	RW
3	VALID	Flag indicating if data is valid. 0: data is invalid 1: data is valid	0x0	R
2	OLD	Flag data has already been read. 0: last data has not been read 1: last data already read	0x0	RW
[1:0]	RESERVED	Reserved.	0x0	RW

ADC Channel Select Register

Address: 0x40086080, Reset: 0x111F, Name: ADCCHA

ADC channel select register for non-sequence operation.

Table 15. Bit Descriptions for ADCCHA

Bits	Bit Name	Description	Reset	Access
[15:13]	RESERVED	Reserved.	0x0	R
[12:8]	ADCCN	Selects channel for ADC negative input. 0x00: AIN0. 0x01: AIN1. 0x02: AIN2. 0x03: AIN3. 0x04: AIN4. 0x05: AIN5. 0x06: AIN6. 0x07: AIN7. 0x08: AIN8. 0x09: AIN9. 0x0A: AIN10. 0x0B: AIN11. 0x0C: AIN12. 0x0D: AIN13. 0x0E: AIN14. 0x0F: AIN15. 0x10: VREFP_NADC: connect ADC_REFP to negative input. 0x11: VREFN_NADC: connect ADC_REFN to negative input. Use this setting for single-ended measurements. 0x12: AGND. 0x13: PGND. 0x14 to 0x1F: reserved	0x11	RW

Bits	Bit Name	Description	Reset	Access
[7:5]	RESERVED	Reserved	0x0	R
[4:0]	ADCCP	Select ADC channel. 0x0: AIN0. 0x1: AIN1. 0x2: AIN2. 0x3: AIN3. 0x4: AIN4. 0x5: AIN5. 0x6: AIN6. 0x7: AIN7. 0x8: AIN8. 0x9: AIN9. 0xA: AIN10. 0xB: AIN11/BUF_VREF2V5. Note that, to measure BUF_VREF2V5, it must be first enabled in the AFEREFC[2] register. 0xC: AIN12. 0xD: AIN13. 0xE: AIN14. 0xF: AIN15. 0x10: reserved. 0x11: reserved. 0x12: IDAC3. 0x13: IDAC1. 0x14: IDAC0. 0x15: IDAC2. 0x16: TEMP_SENSOR. 0x17: VREFP_PADC: Connect ADC_REFP to positive input. Note that this pin should not be measured relative to AGND. This selection is intended for measuring the differential voltage between the negative input and ADC_REFP. 0x18: PVDD_IDAC2: use this to measure the PVDD supply voltage for IDAC2. 0x19: IOVDD_2: use this to measure half of the IOVDD supply voltage. 0x1A: AVDD_2: use this to measure half of the AVDD supply voltage. 0x1B: VREFN_PADC: connect ADC_REFN to positive input. 0x1C to 0x1F: reserved.	0x1F	RW

ADC Sequencer Control Register

Address: 0x40086088, Reset: 0x00000000, Name: ADCSEQ

Table 16. Bit Descriptions for ADCSEQ

Bits	Bit Name	Description	Reset	Access
31	ST	Sequence restart, used to force sequence to start at first channel when sequence is working. 1: set to 1 to restart the sequencer. Cleared after writing 1.	0x0	W
30	EN	Sequence enable. 1: set to 1 to enable the sequencer	0x0	W
29	RESERVED	Reserved.	0x0	R
[28:0]	CH	Select channels included in sequence operation. For each channel: 0: channel is skipped. 1: channel is included in the sequence. Each bit corresponds to an ADC channel as defined by ADCCHA[4:0]. For example, a value of 0x33 (00110011) includes AIN0, AIN1, AIN4, and AIN5 in the sequence and excludes all other channels.	0x0	RW

ADC Sequencer Configuration Register

Address: 0x4008608C, Reset: 0x0008C631, Name: ADCSEQC

Table 17. Bit Descriptions for ADCSEQC

Bits	Bit Name	Description	Reset	Access
[31:28]	RESERVED	Reserved.	0x0	R
[27:20]	T	Define programmable delay of 0 to 254 between sequences. A delay of 255 causes a halt after one sequence. Set ADCSEQ[30] if another sequence is required.	0x0	RW
[19:15]	DIF6	Selects differential mode negative input for AIN6 in the sequence. See ADCCHA[12:8] for list of channels. 0x11: Channel 6 is single ended	0x11	RW
[14:10]	DIF4	Selects differential mode negative input for AIN4 in the sequence. See ADCCHA[12:8] for list of channels. 0x11: Channel 4 is single ended	0x11	RW
[9:5]	DIF2	Selects differential mode negative input for AIN2 in the sequence. See ADCCHA[12:8] for list of channels. 0x11: Channel 2 is single ended	0x11	RW
[4:0]	DIF0	Selects differential mode negative input for AIN0 in the sequence. See ADCCHA[12:8] for list of channels. 0x11: Channel 0 is single ended	0x11	RW

Digital Comparator Configuration Register

Address: 0x40086098, Reset: 0x00000, Name: ADCCMP

Table 18. Bit Descriptions for ADCCMP

Bits	Bit Name	Description	Reset	Access
[17:2]	THR	Digital compare threshold. Value to compare to Channel 4 data.	0x0000	RW
1	DIR	Select digital comparator direction. 0: ADCTH less than Channel 4 data 1: ADCTH larger than Channel 4 data	0x0	RW
0	EN	Digital comparator enable. 0: Disable 1: Enable	0x0	RW

ADC Conversion Configuration Register

Address: 0x4008609C, Reset: 0x000A00C8, Name: ADCCNVC

Note that, when ADCCP is set to 22 (temp sensor) or 25 (IOVDD/2) or 26 (AVDD/2), the ADCCNVC register automatically changes to 0x7D00FA − (80 kSPS) for single conversions. ADCCNVC should be set to the required conversion rate after sampling these 3× channels if a different sample rate is required for other ADC input channels.

Note that, when the sequencer is enabled and includes any of these 3× channels, the value in ADCCNVC does not change and the ADC sampling rate does not change.

Table 19. Bit Descriptions for ADCCNVC

Bits	Bit Name	Description	Reset	Access
[31:26]	RESERVED	Do not overwrite	0x0	RW
[25:16]	CNVD	Configure ADC acquisition time and sampling time Acquisition time = CNVD/20 MHz Default acquisition time is 500 ns For best SNR results, ensure that the acquisition time is set to ≥500 ns for all ADC conversion rates	0xA	RW
[15:10]	RESERVED	Do not overwrite	0x00	RW
[9:0]	CNVC	Configure conversion frequency. Conversion frequency = 20 MHz/CNVC	0xC8	RW

REGISTER SUMMARY: ADDITIONAL REGISTERS

The CPU accesses these additional registers over a die to die interface (D2D) which increases the execution times of ldr and str instructions. The 32 bit MMRs have addresses of 0x40087xxx and take 8 CPU cycles at 80 MHz to execute. The 8 bit MMRs have addresses of 0x40081xxx and take 5 CPU cycles at 80 MHz to execute.

Table 20. Register Summary

Address	Name	Description	Reset	RW
0x40081400	IBUFCON	InBuf control bit	0x000F	RW
0x40087830	AFETEMPC	Temperature sensor configuration register	0x00	RW
0x40087834	AFEREFC	Reference configuration register	0x00	RW

REGISTER DETAILS: ADDITIONAL REGISTERS

InBuf Control Bit Register

Address: 0x40081400, Reset: 0x000F, Name: IBUFCON

Table 21. Bit Descriptions for IBUFCON

Bits	Bit Name	Description	Reset	Access
[15:4]	RESERVED	Reserved.	0x0	RW
[3:2]	IBUF_PD	Power down P/N InBuf separately. 00: both sides powered on 01: N side powered down 10: P side powered down 11: both sides powered down	0x3	RW
[1:0]	IBUF_BYP	Bypass P/N InBuf separately. 00: bypass none sided 01: N side bypassed 10: P side bypassed 11: bypass both	0x3	RW

Temperature Sensor Configuration Register

Address: 0x40087830, Reset: 0x00, Name: AFETEMPC

Table 22. Bit Descriptions for AFETEMPC

Bits	Bit Name	Description	Reset	Access
[7:2]	RESERVED	Reserved.	0x0	R
1	CHOP	Temperature sensor chopping enable. Do not use chopping mode together with the sequencer. 0: disable chopping mode 1: enable chopping mode	0x0	RW
0	PD	Temperature sensor power down. 0: power up temperature sensor 1: power down temperature sensor	0x0	RW

Reference Configuration Register

Address: 0x40087834, Reset: 0x00, Name: AFEREFC

Table 23. Bit Descriptions for AFEREFC

Bits	Bit Name	Description	Reset	Access
[7:4]	RESERVED	Reserved.	0x0	R
3	REF	Bypass the internal reference, and select the external reference. 0: select internal 2.51 V reference 1: select external 2.51 V reference	0x0	RW
2	B2MA_PDB	Power down the reference 1.2 mA output driving Buffer B, which is on the AIN11/BUF_VREF2V5 pin. 0: power down 2.5 V reference output driving BUF_VREF2V5 1: power up 2.5 V reference output driving BUF_VREF2V5	0x0	RW
1	B2V5R_PD	2.5 V reference buffer power down. 0: power up 2.5 V reference buffer 1: power down 2.5 V reference buffer	0x0	RW
0	BG_PD	Band gap power down. 0: power up 1.2 V band gap 1: power down 1.2 V band gap	0x0	RW

ANALOG COMPARATOR

ANALOG COMPARATOR FEATURES

The analog comparator compares two analog signals and gives an output indicating which of the input signals is bigger. This output can generate an interrupt

ANALOG COMPARATOR OVERVIEW

The positive input of the comparator is shared with AIN6.

The negative input of the comparator can be set by software to AVDD/2, AIN5, or DAC7.

The comparator output is connected to the interrupt logic and can be used as described in the System Exceptions and Peripheral Interrupts section.

ANALOG COMPARATOR OPERATION

If required, change the hysteresis with AFECOMP[0], the comparator speed with AFECOMP[1:2], and the output polarity with AFECOMP[3].

Select the input source with AFECOMP[6:7].

Power up and enable the comparator with AFECOMP[8] and AFECOMP[4:5].

REGISTER SUMMARY: ANALOG COMPARATOR

The CPU accesses the ADC circuit over a die to die interface (D2D) which increases the execution times of ldr and str instructions. Accessing AFECOMP takes 8 CPU cycles at 80 MHz to execute.

Table 24. Analog Comparator Register Summary

Address	Name	Description	Reset	RW
0x40087838	AFECOMP	Analog comparator configuration register	0x0000	RW

REGISTER DETAILS: ANALOG COMPARATOR

Analog Comparator Configuration Register

Address: 0x40087838, Reset: 0x0000, Name: AFECOMP

Table 25. Bit Descriptions for AFECOMP

Bits	Bit Name	Description	Reset	Access
[15:9]	RESERVED	Reserved.	0x0	R
8	EN	Powers up and enables comparator. 0: power down and disable comparator 1: power up and enable comparator	0x0	RW
[7:6]	INNEG	Selects comparator negative input signal. 00: AVDD/2 01: AIN5 10: DAC7 11: unused	0x0	RW
[5:4]	OUT	Connects comparator output to interrupt logic. 0: do not connect output 1: connect output to interrupt logic	0x0	RW
3	INV	Selects output logic state. 0: output is high if +ve input is above −ve input. 1: output is high if +ve input is below −ve input.	0x0	RW
[2:1]	SPEED	Selects comparator speed to falling output. 00: 6 µs 01: 4 µs 10: 4 µs 11: 3 µs Response time to rising output is 6 µs typical.	0x0	RW
0	HYS	Enables comparator hysteresis. 0: disable hysteresis 1: enable hysteresis	0x0	RW

IDACs

IDAC FEATURES

The ADuCM320 provides four IDACs. These are low noise, low drift current source outputs.

- IDAC0, IDAC1, IDAC2 and IDAC3: 0 mA to 150 mA full-scale output, bias current setting for optical laser.

IDAC BLOCK DIAGRAM

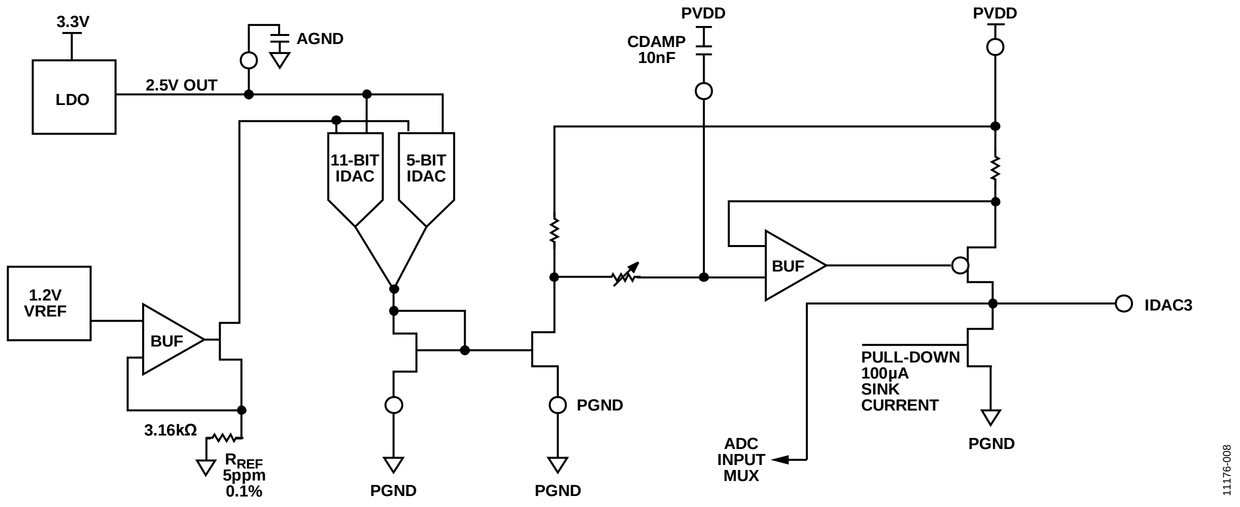


Figure 9. Example IDAC Circuit—IDAC3

IDAC OVERVIEW

Precision Current Generation and Fault Protection

The reference current for the IDACs is generated by a precision internal band gap voltage reference ($V_{BANDGAP}$)) and an external precision resistor (R_{REF}, 5 ppm, 0.1%). The reference current is equal to $V_{BANDGAP} \div R_{REF}$. The band gap voltage reference is a low drift, high accuracy voltage source that helps to minimize the overall IDAC gain error and gain error drift. The noise of the IDAC outputs is limited by the low-pass filter on the output stage; each IDAC requires a 10 nF capacitor between PVDD and its CDAMP pin.

Figure 9 shows the typical architecture of the IDAC. The parallel 11-bit and 5-bit IDACs set the output current. The output of these IDACs are summed together and fed to a current mirror and then are gained up at the output stage.

Production trimming of the LDO band gap reference aids performance. In addition, gain trimming and scaling of the current mirror and output stages are also included in the ATE test program.

IDAC Shutdown

IDAC0, IDAC1, IDAC2, and IDAC3 also have a small current sink capability to minimize the offset current when the data register is set to 0. The IDACxCON[1] bit can be used to enable a pull-down current source to PGND. This pull-down current is typically 100 µA.

IDAC Output Filter

Each IDAC has a filter on the output stage to minimize noise. Each IDAC requires an external 10 nF capacitor between PVDD and its CDAMP pin as per the ADuCM320 data sheet. The on-chip, programmable resistor is controlled by the IDACxCON[5:2] bits.

Table 26. IDAC Filter Bandwidth Control Settings

IDACxCON[5:2]	R Value	Cutoff Frequency (f_C)
0000	60 Ω	262 kHz
0101	5.6 kΩ	2.8 kHz
0110	11.2 kΩ	1.4 kHz
0111	22.2 kΩ	715 Hz
1000	44.4 kΩ	357 Hz
1001	104 kΩ	153 Hz
All other options are reserved		

IDAC Data Register

The IDAC output is controlled by an internal 11-bit and 5-bit DAC.

The 11-bit DAC (IDACxDAT[27:17]) controls the most significant bits. The 5-bit DAC (IDACxDAT[16:12]) controls the LSBs. The two MSBs of the 5-bit DAC (IDACxDAT[16:15]) overlap the two LSBs of the 11-bit DAC (IDACxDAT[18:17]) as shown in Figure 10.

14-BIT IDAC OUTPUT		13	12	11	10	9	8	7	6	5	4	3	2	1	0
11-BIT DAC	IDACxDAT	27	26	25	24	23	22	21	20	19	18	17			
5-BIT DAC											16	15	14	13	12

11176-009

Figure 10. 14-Bit IDAC Output

The 11-bit DAC and the 5-bit DAC are guaranteed monotonic as individual DACs. This combination makes it possible to reach 14-bit resolution—up to 16,384 unique output values. However, monotonicity is only guaranteed for 11 bits (DNL < −1 LSB).

IDACs—Common Use Cases

Case 1—Setting the Output Current of IDAC1 to Quarter Scale

- Set up IDAC1CON to 10xxxx00b:
  - IDAC1CON[7] = 1: enable writes to the IDAC1DAT register
  - IDAC1CON[0] = 0: power up IDAC1
  - IDAC1CON[5:2] as per Table 26: set up the filter bandwidth as required
  - IDAC1CON[1] = 0: disable the IDAC1 pull-down current source
  - IDAC1DAT[3] = 0: clear the IDAC1 sync bit to allow immediate updating of the IDAC
  - IDAC1CON[6] = 0: disable the overtemperature shutdown feature
- Set up IDAC1DAT to give a current output of quarter scale:
  - If IDAC1 is used in an open loop system or in a set and forget type operation, set IDAC1DAT = 0x03FE0000
    - Set IDAC1DAT[27:17] = 0x1FF
- Set IDAC1DAT[16:12] = 0x00
  - If IDAC1 is used in a closed loop system, set IDAC1DAT = 0x03FCF000
    - Set IDAC1DAT[27:17] = 0x1FE
    - Set IDAC1DAT[16:12] = [01111]b
    - Adjust the 5-bit IDAC (IDAC1DAT[16:12]) up or down accordingly to attain the correct setting.

Case 2—Turn On IDAC2 but to Set the Output to 0 mA with the Lowest Possible Offset

Before powering up the IDACs, ensure that the internal reference is fully powered on.

- Set up the IDAC2CON register to 10xxxx00b:
  - IDAC2CON[7] = 1: enable writes to the IDAC2DAT register
  - IDAC2CON[0] = 0: power up IDAC2
  - IDAC2CON[5:2] as per Table 26: set up the filter bandwidth as required
  - IDAC2CON[1] = 0: enable the IDAC2 pull-down current sink
  - IDAC2CON[6] = 0: disable the overtemperature shutdown feature
- Set up the IDAC2DAT register
  - IDAC2DAT[27:12] = 0x0000: set the IDAC to zero scale
- If any or all the IDACs are not used, connect the pins as follows:
  - IREF: If no IDACs are used, connect a low cost 3.3 kΩ resistor to ground.
  - If no PVDD supply is available, connect PVDDx of all IDACs to AVDD_REG1 (Ball F10).
    - Make sure not to power up such IDACs to avoid loading AVDD_REG1 unnecessarily.
  - CDAMPx of any IDAC not used, leave unconnected.
  - IDACx of individual IDAC not used:
    - Power down IDACx using PD bit (IDACxCON = 1; it is already powered down after reset).
    - Set IDACxDAT = 0 (zero current; it is already 0 after reset).
    - Connect IDACx pin to PGND.

IDAC Thermal Shutdown

The ADuCM320 has an internal temperature sensor that monitors the die temperature. This temperature sensor can be monitored as an ADC input channel; the measured voltage is proportional to die temperature. See the Temperature Sensor Settings section for more information.

Internally, the die temperature is compared to a fixed voltage, proportional to approximately 130°C die temperature. If the die temperature exceeds 130°C, there is a risk of damaging the die because the absolute maximum junction temperature rating is 150°C. Because the IDACs potentially consume the most power, shut off the IDACs to reduce power and therefore to reduce the die temperature. Two options to enable shutdown include the following:

- Enable a thermal interrupt by setting either INTSEL[12] or INTSEL[4]. If the die temperature exceeds the threshold of approximately 130°C, this interrupt triggers and user code takes the appropriate action. It is recommended to use this procedure.
- Enable automatic shutdown of the IDACs by setting the individual thermal shutdown bits for each IDAC via IDACxCON[6]. If this bit is set in the appropriate IDACxCON register, the IDAC output current reduces to 0 mA, which reduces the power consumption of the device and the die temperature of the device.

Note that the internal temperature sensor accuracy can be up to ±20°C, and there is no way of calibrating the thermal shutdown trip point. Therefore, it is recommended that the automatic thermal shutdown feature (IDACxCON[6] = 0) not be enabled.

REGISTER SUMMARY: IDAC

The CPU accesses the IDAC circuit over a die to die interface (D2D) which increases the execution times of ldr and str instructions. The 32-bit MMRs have addresses of 0x40086xxx and take 8 CPU cycles at 80 MHz to execute.

Table 27. IDAC Register Summary

Address	Name	Description	Reset	RW
0x40086800	IDAC0DAT	IDAC0 data register	0x00000000	RW
0x40086804	IDAC0CON	IDAC0 control register	0x01	RW
0x40086808	IDAC1DAT	IDAC1 data register	0x00000000	RW
0x4008680C	IDAC1CON	IDAC1 control register	0x01	RW
0x40086810	IDAC2DAT	IDAC2 data register	0x00000000	RW
0x40086814	IDAC2CON	IDAC2 control register	0x01	RW
0x40086818	IDAC3DAT	IDAC3 data register	0x00000000	RW
0x4008681C	IDAC3CON	IDAC3 control register	0x01	RW

REGISTER DETAILS: IDAC

IDAC0 Data Register

Address: 0x40086800, Reset: 0x00000000, Name: IDAC0DAT

Table 28. Bit Descriptions for IDAC0DAT

Bits	Bit Name	Description	Reset	Access
[31:28]	RESERVED	Reserved. Write 0.	0x0	R
[27:17]	DATH	IDAC0 high data.	0x0	RW
[16:12]	DATL	IDAC0 low data.	0x0	RW
[11:4]	RESERVED	Reserved.	0x0	R
3	SYNC3	IDAC3 sync bit. Setting the SYNC3 bits of all IDAC channels to 1 prevents IDAC3 from updating. When the SYNC3 bit of any of the IDAC channels is 0, IDAC3 updates immediately when it is written.	0x0	RW
2	SYNC2	IDAC2 sync bit. Setting the SYNC2 bits of all IDAC channels to 1 prevents IDAC2 from updating. When the SYNC2 bit of any of the IDAC channels is 0, IDAC2 updates immediately when it is written.	0x0	RW
1	SYNC1	IDAC1 sync bit. Setting the SYNC1 bits of all IDAC channels to 1 prevents IDAC1 from updating. When the SYNC1 bit of any of the IDAC channels is 0, IDAC1 updates immediately when it is written.	0x0	RW
0	SYNC0	IDAC0 sync bit. Setting the SYNC0 bits of all IDAC channels to 1 prevents IDAC0 from updating. When the SYNC0 bit of any of the IDAC channels is 0, IDAC0 updates immediately when it is written.	0x0	RW

IDAC0 Control Register

Address: 0x40086804, Reset: 0x01, Name: IDAC0CON

Table 29. Bit Descriptions for IDAC0CON

Bits	Bit Name	Description	Reset	Access
7	CLRB	IDAC0 clear bit. 0: clear IDAC0DAT 1: enable write	0x0	RW
6	SHT_EN	IDAC0 shutdown enable. Enables automatic shutdown in case of overtemperature. 0: disable this function 1: enable this function	0x0	RW
[5:2]	BW	IDAC0 bandwidth. See the IDAC Output Filter section for more details.	0x0	RW
1	PUL	IDAC0 pull-down. 0: disable the pull-down current source 1: enable the pull-down current source	0x0	RW
0	PD	IDAC0 power down. 0: powers up IDAC0 1: powers down IDAC0	0x1	RW

IDAC1 Data Register

Address: 0x40086808, Reset: 0x00000000, Name: IDAC1DAT

Table 30. Bit Descriptions for IDAC1DAT

Bits	Bit Name	Description	Reset	Access
[31:28]	RESERVED	Reserved. Write 0.	0x0	R
[27:17]	DATH	IDAC1 high data.	0x0	RW
[16:12]	DATL	IDAC1 low data.	0x0	RW
[11:4]	RESERVED	Reserved.	0x0	R
3	SYNC3	IDAC3 sync bit. Setting the SYNC3 bits of all IDAC channels to 1 prevents IDAC3 from updating. When the SYNC3 bit of any of the IDAC channels is 0, IDAC3 updates immediately when it is written.	0x0	RW
2	SYNC2	IDAC2 sync bit. Setting the SYNC2 bits of all IDAC channels to 1 prevents IDAC2 from updating. When the SYNC2 bit of any of the IDAC channels is 0, IDAC2 updates immediately when it is written.	0x0	RW
1	SYNC1	IDAC1 sync bit. Setting the SYNC1 bits of all IDAC channels to 1 prevents IDAC1 from updating. When the SYNC1 bit of any of the IDAC channels is 0, IDAC1 updates immediately when it is written.	0x0	RW
0	SYNC0	IDAC0 sync bit. Setting the SYNC0 bits of all IDAC channels to 1 prevents IDAC0 from updating. When the SYNC0 bit of any of the IDAC channels is 0, IDAC0 updates immediately when it is written.	0x0	RW

IDAC1 Control Register

Address: 0x4008680C, Reset: 0x01, Name: IDAC1CON

Table 31. Bit Descriptions for IDAC1CON

Bits	Bit Name	Description	Reset	Access
7	CLRB	IDAC1 clear bit. 0: clear IDAC1DAT 1: enable write	0x0	RW
6	SHT_EN	IDAC1 shutdown enable. Enables automatic shutdown in case of overtemperature. 0: disable this function 1: enable this function	0x0	RW
[5:2]	BW	IDAC1 bandwidth. See the IDAC Output Filter section for more details.	0x0	RW
1	PUL	IDAC1 pull-down. 0: disable the pull-down current source 1: enable the pull-down current source	0x0	RW
0	PD	IDAC1 power down. 0: powers up IDAC1 1: powers down IDAC1	0x1	RW

IDAC2 Data Register

Address: 0x40086810, Reset: 0x00000000, Name: IDAC2DAT

Table 32. Bit Descriptions for IDAC2DAT

Bits	Bit Name	Description	Reset	Access
[31:28]	RESERVED	Reserved. Write 0.	0x0	R
[27:17]	DATH	IDAC2 high data.	0x0	RW
[16:12]	DATL	IDAC2 low data.	0x0	RW
[11:4]	RESERVED	Reserved.	0x0	R
3	SYNC3	IDAC3 sync bit. Setting the SYNC3 bits of all IDAC channels to 1 prevents IDAC3 from updating. When the SYNC3 bit of any of the IDAC channels is 0, IDAC3 updates immediately when it is written.	0x0	RW
2	SYNC2	IDAC2 sync bit. Setting the SYNC2 bits of all IDAC channels to 1 prevents IDAC2 from updating. When the SYNC2 bit of any of the IDAC channels is 0, IDAC2 updates immediately when it is written.	0x0	RW
1	SYNC1	IDAC1 sync bit. Setting the SYNC1 bits of all IDAC channels to 1 prevents IDAC1 from updating. When the SYNC1 bit of any of the IDAC channels is 0, IDAC1 updates immediately when it is written.	0x0	RW
0	SYNC0	IDAC0 sync bit. Setting the SYNC0 bits of all IDAC channels to 1 prevents IDAC0 from updating. When the SYNC0 bit of any of the IDAC channels is 0, IDAC0 updates immediately when it is written.	0x0	RW

IDAC2 Control Register

Address: 0x40086814, Reset: 0x01, Name: IDAC2CON

Table 33. Bit Descriptions for IDAC2CON

Bits	Bit Name	Description	Reset	Access
7	CLR	IDAC2 clear bit. 0: clear IDAC2DAT 1: enable write	0x0	RW
6	SHT_EN	IDAC2 shutdown enable. Enables automatic shutdown in case of overtemperature. 0: disable this function 1: enable this function	0x0	RW
[5:2]	BW	IDAC2 bandwidth. See the IDAC Output Filter section for more details.	0x0	RW
1	PUL	IDAC2 pull-down. 0: disable the pull-down current source 1: enable the pull-down current source	0x0	RW
0	PD	IDAC2 power down. 0: powers up IDAC2 1: powers down IDAC2	0x1	RW

IDAC3 Data Register

Address: 0x40086818, Reset: 0x00000000, Name: IDAC3DAT

Table 34. Bit Descriptions for IDAC3DAT

Bits	Bit Name	Description	Reset	Access
[31:28]	RESERVED	Reserved. Write 0.	0x0	R
[27:17]	DATH	IDAC3 high data.	0x0	RW
[16:12]	DATL	IDAC3 low data.	0x0	RW
[11:4]	RESERVED	Reserved.	0x0	R
3	SYNC3	IDAC3 sync bit. Setting the SYNC3 bits of all IDAC channels to 1 prevents IDAC3 from updating. When the SYNC3 bit of any of the IDAC channels is 0, IDAC3 updates immediately when it is written.	0x0	RW
2	SYNC2	IDAC2 sync bit. Setting the SYNC2 bits of all IDAC channels to 1 prevents IDAC2 from updating. When the SYNC2 bit of any of the IDAC channels is 0, IDAC2 updates immediately when it is written.	0x0	RW
1	SYNC1	IDAC1 sync bit. Setting the SYNC1 bits of all IDAC channels to 1 prevents IDAC1 from updating. When the SYNC1 bit of any of the IDAC channels is 0, IDAC1 updates immediately when it is written.	0x0	RW
0	SYNC0	IDAC0 sync bit. Setting the SYNC0 bits of all IDAC channels to 1 prevents IDAC0 from updating. When the SYNC0 bit of any of the IDAC channels is 0, IDAC0 updates immediately when it is written.	0x0	RW

IDAC3 Control Register

Address: 0x4008681C, Reset: 0x01, Name: IDAC3CON

Table 35. Bit Descriptions for IDAC3CON

Bits	Bit Name	Description	Reset	Access
7	CLR	IDAC3 clear bit. 0: clear IDAC3DAT 1: enable write	0x0	RW
6	SHT_EN	IDAC3 shutdown enable. Enables automatic shutdown in case of overtemperature. 0: disable this function 1: enable this function	0x0	RW
[5:2]	BW	IDAC3 bandwidth. See the IDAC Output Filter section for more details.	0x0	RW
1	PUL	IDAC3 pull-down. 0: disable the pull-down current source 1: enable the pull-down current source	0x0	RW
0	PD	IDAC3 power down. 0: powers up IDAC3 1: powers down IDAC3	0x1	RW

VDACs

VDAC FEATURES

The ADuCM320 has eight VDACs. The specified load resistance is greater than 5 kΩ, and the specified capacitance is less than 100 pF.

VDAC BLOCK DIAGRAM

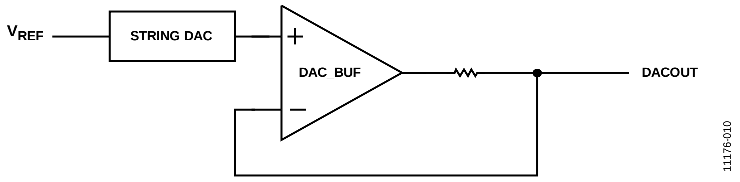


Figure 11. Output Mode Capacitor Load ≤ 100 pF

VDAC OVERVIEW

The ADuCM320 has eight VDACs specified to drive 5 kΩ load, 500 µA maximum.

The VDACs can select from two reference sources:

- 0 V to internal reference, V_{REF} (0 V to 2.5 V)
- 0 V to AV_{DD} (3.3 V)

VDAC OPERATION

The DAC is configurable through a control register and a data register. The on-chip DAC architecture consists of a resistor string DAC followed by an output buffer amplifier, as shown in Figure 11.

The linearity specification of the DAC when driving a 5 kΩ resistive load to ground is guaranteed through the full transfer function except for Code 0 to Code 100; in 0-to-AV_{DD} mode, the linearity specification is also not guaranteed for Code 3995 to Code 4095. Linearity degradation near ground and AV_{DD} is caused by saturation of the output amplifier; a general representation of its effects (neglecting offset and gain error) is shown in Figure 12.

The dotted line in Figure 12 indicates the ideal transfer function. The solid line represents what the transfer function may look like with endpoint nonlinearities due to saturation of the output amplifier. Figure 12 represents a transfer function in 0-to-AV_{DD} mode only. In 0-to-V_{REF} mode, the lower nonlinearity is similar. However, the upper portion of the transfer function follows the ideal line all the way to the end, showing no signs of endpoint linearity errors.

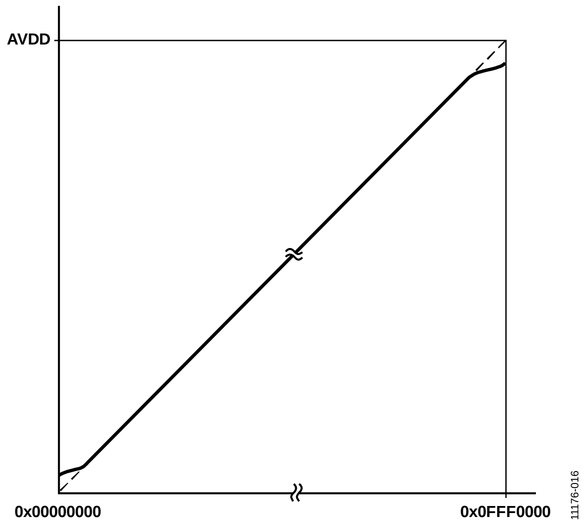


Figure 12. DAC Endpoint Nonlinearities Due to Amplifier Saturation

During power-on reset, all VDAC channels ramp up and return to normal state at 1 ms, as shown in Figure 13.

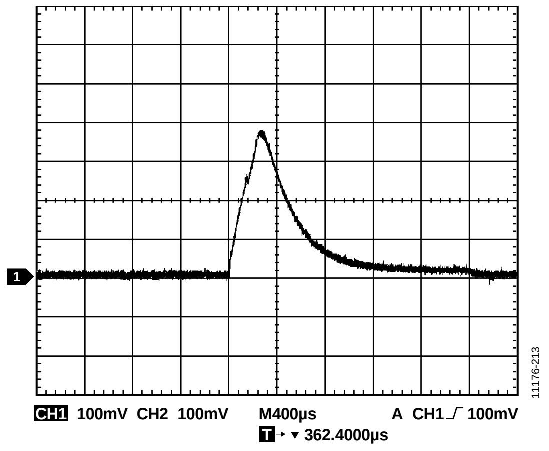


Figure 13. VDAC Channels Response at Power-Up

REGISTER SUMMARY: VDAC

The CPU accesses the VDAC circuit over a die to die interface (D2D), which increases the execution times of ldr and str instructions. The 32-bit MMRs have addresses of 0x40086xxx and take 8 CPU cycles at 80 MHz to execute. The 16-bit MMRs have addresses of 0x40082xxx and take 6 CPU cycles at 80 MHz to execute.

Table 36. VDAC Register Summary

Address	Name	Description	Reset	RW
0x40082400	DAC0CON	DAC0 control register	0x0100	RW
0x40082404	DAC1CON	DAC1 control register	0x0100	RW
0x40082408	DAC2CON	DAC2 control register	0x0100	RW
0x4008240C	DAC3CON	DAC3 control register	0x0100	RW
0x40082410	DAC4CON	DAC4 control register	0x0100	RW
0x40082414	DAC5CON	DAC5 control register	0x0100	RW
0x40082418	DAC6CON	DAC6 control register	0x0100	RW
0x4008241C	DAC7CON	DAC7 control register	0x0100	RW
0x40086404	DAC0DAT	DAC0 data register	0x00000000	RW
0x40086408	DAC1DAT	DAC1 data register	0x00000000	RW
0x4008640C	DAC2DAT	DAC2 data register	0x00000000	RW
0x40086410	DAC3DAT	DAC3 data register	0x00000000	RW
0x40086414	DAC4DAT	DAC4 data register	0x00000000	RW
0x40086418	DAC5DAT	DAC5 data register	0x00000000	RW
0x4008641C	DAC6DAT	DAC6 data register	0x00000000	RW
0x40086420	DAC7DAT	DAC7 data register	0x00000000	RW

REGISTER DETAILS: VDAC

DAC0 Control Register

Address: 0x40082400, Reset: 0x0100, Name: DAC0CON

Table 37. Bit Descriptions for DAC0CON

Bits	Bit Name	Description	Reset	Access
[15:9]	RESERVED	Reserved.	0x0	R
8	PD	DAC0 power down. 0: DAC0 is powered up 1: DAC0 is powered down and output is floating	0x1	RW
[7:5]	RESERVED	Reserved.	0x0	RW
4	EN	DAC0 enable. Must be set to 1. 0: DAC disable. Clear DAC data immediately 1: DAC enable.	0x0	RW
[3:2]	RESERVED	Reserved.	0x0	RW
[1:0]	RN	DAC0 reference selection. These bits set the DAC range. A write to these bits has immediate effect on the DAC. 00: internal reference 01: reserved 10: reserved 11: AVDD/AGND	0x0	RW

DAC1 Control Register

Address: 0x40082404, Reset: 0x0100, Name: DAC1CON

Table 38. Bit Descriptions for DAC1CON

Bits	Bit Name	Description	Reset	Access
[15:9]	RESERVED	Reserved.	0x0	R
8	PD	DAC1 power down. 0: DAC1 is powered up 1: DAC1 is powered down and output is floating	0x1	RW

Bits	Bit Name	Description	Reset	Access
[7:5]	RESERVED	Reserved.	0x0	RW
4	EN	DAC1 enable. Must be set to high. 0: DAC disable. Clear DAC data immediately. 1: DAC enable.	0x0	RW
[3:2]	RESERVED	Reserved.	0x0	RW
[1:0]	RN	DAC1 reference selection. These bits set the DAC range. A write to these bits has immediate effect on the DAC. 00: internal reference 01: reserved 10: reserved 11: AVDD/AGND	0x0	RW

DAC2 Control Register

Address: 0x40082408, Reset: 0x0100, Name: DAC2CON

Table 39. Bit Descriptions for DAC2CON

Bits	Bit Name	Description	Reset	Access
[15:9]	RESERVED	Reserved.	0x0	R
8	PD	DAC2 power down. 0: DAC2 is powered up 1: DAC2 is powered down and output is floating	0x1	RW
[7:5]	RESERVED	Reserved.	0x0	RW
4	EN	DAC2 enable. Must be set to high. 0: DAC disable. Clear DAC data immediately 1: DAC enable.	0x0	RW
[3:2]	RESERVED	Reserved.	0x0	RW
[1:0]	RN	DAC2 reference selection. These bits set the DAC range. A write to these bits has immediate effect on the DAC. 00: internal reference 01: reserved 10: reserved 11: AVDD/AGND	0x0	RW

DAC3 Control Register

Address: 0x4008240C, Reset: 0x0100, Name: DAC3CON

Table 40. Bit Descriptions for DAC3CON

Bits	Bit Name	Description	Reset	Access
[15:9]	RESERVED	Reserved.	0x0	R
8	PD	DAC3 power down. 0: DAC3 is powered up 1: DAC3 is powered down and output is floating	0x1	RW
[7:5]	RESERVED	Reserved.	0x0	RW
4	EN	DAC3 enable. Must be set to high. 0: DAC disable. Clear DAC data immediately 1: DAC enable.	0x0	RW
[3:2]	RESERVED	Reserved.	0x0	RW
[1:0]	RN	DAC3 reference selection. These bits set the DAC range. A write to these bits has immediate effect on the DAC. 00: internal reference 01: reserved 10: reserved 11: AVDD/AGND	0x0	RW

DAC4 Control Register

Address: 0x40082410, Reset: 0x0100, Name: DAC4CON

Table 41. Bit Descriptions for DAC4CON

Bits	Bit Name	Description	Reset	Access
[15:9]	RESERVED	Reserved.	0x0	R
8	PD	DAC4 power down. 0: DAC4 is powered up 1: DAC4 is powered down and output is floating	0x1	RW
[7:5]	RESERVED	Reserved.	0x0	RW
4	EN	DAC4 enable. Must be set to high. 0: DAC disable. Clear DAC data immediately 1: DAC enable.	0x0	RW
[3:2]	RESERVED	Reserved.	0x0	RW
[1:0]	RN	DAC4 reference selection. These bits set the DAC range. A write to these bits has immediate effect on the DAC. 00: internal reference 01: reserved 10: reserved 11: AVDD/AGND	0x0	RW

DAC5 Control Register

Address: 0x40082414, Reset: 0x0100, Name: DAC5CON

Table 42. Bit Descriptions for DAC5CON

Bits	Bit Name	Description	Reset	Access
[15:9]	RESERVED	Reserved.	0x0	R
8	PD	DAC5 power down. 0: DAC5 is powered up 1: DAC5 is powered down and output is floating	0x1	RW
[7:5]	RESERVED	Reserved.	0x0	RW
4	EN	DAC5 enable. Must be set to high. 0: DAC disable. Clear DAC data immediately 1: DAC enable.	0x0	RW
[3:2]	RESERVED	Reserved.	0x0	RW
[1:0]	RN	DAC5 reference selection. These bits set the DAC range. A write to these bits has immediate effect on the DAC. 00: internal reference 01: reserved 10: reserved 11: AVDD/AGND	0x0	RW

DAC6 Control Register

Address: 0x40082418, Reset: 0x0100, Name: DAC6CON

Table 43. Bit Descriptions for DAC6CON

Bits	Bit Name	Description	Reset	Access
[15:9]	RESERVED	Reserved.	0x0	R
8	PD	DAC6 power down. 0: DAC6 is powered up 1: DAC6 is powered down and output is floating	0x1	RW
[7:5]	RESERVED	Reserved.	0x0	RW

Bits	Bit Name	Description	Reset	Access
4	EN	DAC6 enable. Must be set to high. 0: DAC disable. Clear DAC data immediately 1: DAC enable.	0x0	RW
[3:2]	RESERVED	Reserved.	0x0	RW
[1:0]	RN	DAC6 reference selection. These bits set the DAC range. A write to these bits has immediate effect on the DAC. 00: internal reference 01: reserved 10: reserved 11: AVDD/AGND	0x0	RW

DAC7 Control Register

Address: 0x4008241C, Reset: 0x0100, Name: DAC7CON

Table 44. Bit Descriptions for DAC7CON

Bits	Bit Name	Description	Reset	Access
[15:9]	RESERVED	Reserved.	0x0	R
8	PD	DAC7 power down. 0: DAC7 is powered up 1: DAC7 is powered down and output is floating	0x1	RW
[7:5]	RESERVED	Reserved.	0x0	RW
4	EN	DAC7 enable. Must be set to high. 0: DAC disable. Clear DAC data immediately 1: DAC enable.	0x0	RW
[3:2]	RESERVED	Reserved.	0x0	RW
[1:0]	RN	DAC7 reference selection. These bits set the DAC range. A write to these bits has immediate effect on the DAC. 00: internal reference 01: reserved 10: reserved 11: AVDD/AGND	0x0	RW

DAC0 Data Register

Address: 0x40086404, Reset: 0x00000000, Name: DAC0DAT

Table 45. Bit Descriptions for DAC0DAT

Bits	Bit Name	Description	Reset	Access
[31:28]	RESERVED	Reserved. Write 0.	0x0	R
[27:16]	DAT	DAC0 data.	0x0	RW
[15:0]	RESERVED	Reserved. Write 0.	0x0	R

DAC1 Data Register

Address: 0x40086408, Reset: 0x00000000, Name: DAC1DAT

Table 46. Bit Descriptions for DAC1DAT

Bits	Bit Name	Description	Reset	Access
[31:28]	RESERVED	Reserved. Write 0.	0x0	R
[27:16]	DAT	DAC1 data.	0x0	RW
[15:0]	RESERVED	Reserved. Write 0.	0x0	R

DAC2 Data Register

Address: 0x4008640C, Reset: 0x00000000, Name: DAC2DAT

Table 47. Bit Descriptions for DAC2DAT

Bits	Bit Name	Description	Reset	Access
[31:28]	RESERVED	Reserved. Write 0.	0x0	R
[27:16]	DAT	DAC2 data.	0x0	RW
[15:0]	RESERVED	Reserved. Write 0.	0x0	R

DAC3 Data Register

Address: 0x40086410, Reset: 0x00000000, Name: DAC3DAT

Table 48. Bit Descriptions for DAC3DAT

Bits	Bit Name	Description	Reset	Access
[31:28]	RESERVED	Reserved. Write 0.	0x0	R
[27:16]	DAT	DAC3 data.	0x0	RW
[15:0]	RESERVED	Reserved. Write 0.	0x0	R

DAC4 Data Register

Address: 0x40086414, Reset: 0x00000000, Name: DAC4DAT

Table 49. Bit Descriptions for DAC4DAT

Bits	Bit Name	Description	Reset	Access
[31:28]	RESERVED	Reserved. Write 0.	0x0	R
[27:16]	DAT	DAC4 data.	0x0	RW
[15:0]	RESERVED	Reserved. Write 0.	0x0	R

DAC5 Data Register

Address: 0x40086418, Reset: 0x00000000, Name: DAC5DAT

Table 50. Bit Descriptions for DAC5DAT

Bits	Bit Name	Description	Reset	Access
[31:28]	RESERVED	Reserved. Write 0.	0x0	R
[27:16]	DAT	DAC5 data.	0x0	RW
[15:0]	RESERVED	Reserved. Write 0.	0x0	R

DAC6 Data Register

Address: 0x4008641C, Reset: 0x00000000, Name: DAC6DAT

Table 51. Bit Descriptions for DAC6DAT

Bits	Bit Name	Description	Reset	Access
[31:28]	RESERVED	Reserved. Write 0.	0x0	R
[27:16]	DAT	DAC6 data.	0x0	RW
[15:0]	RESERVED	Reserved. Write 0.	0x0	R

DAC7 Data Register

Address: 0x40086420, Reset: 0x00000000, Name: DAC7DAT

Table 52. Bit Descriptions for DAC7DAT

Bits	Bit Name	Description	Reset	Access
[31:28]	RESERVED	Reserved. Write 0.	0x0	R
[27:16]	DAT	DAC7 data.	0x0	RW
[15:0]	RESERVED	Reserved. Write 0.	0x0	R

SYSTEM EXCEPTIONS AND PERIPHERAL INTERRUPTS

CORTEX-M3 AND FAULT MANAGEMENT

The ADuCM320 integrates an ARM Cortex-M3 processor, which supports several of system exceptions and interrupts generated by peripherals. Table 53 lists the ARM Cortex-M3 processor system exceptions.

Table 53. System Exceptions

Number	Type	Priority	Description
1	Reset	−3 (highest)	Any reset.
2	NMI	−2	Nonmaskable interrupt not connected on the ADuCM320.
3	Hard fault	−1	All fault conditions if the corresponding fault handler is not enabled.
4	Memory management fault	Programmable	Memory management fault; access to invalid locations.
5	Bus fault	Programmable	Prefetch fault, memory access fault, data abort, and other address/memory related faults.
6	Usage fault	Programmable	Same as undefined instruction executed or invalid state transition attempt.
7 to 10	Reserved	N/A	
11	SVCall	Programmable	System service call with SVC instruction. Used for system function calls.
12	Debug monitor	Programmable	Debug monitor (breakpoint, watchpoint, or external debug requests).
13	Reserved	N/A	
14	PendSV	Programmable	Pendable request for system service. Used for queuing system calls until other tasks and interrupts are serviced.
15	SYSTICK	Programmable	System tick timer.

The peripheral interrupts are controlled by the NVIC and are listed in Table 54. All interrupt sources can wake up the device from Mode 1. Only a limited number of interrupts can wake up the processor from the low power modes (Mode 2 and Mode 3) as shown in Table 54. When the device is woken up from Mode 2 or Mode 3, it returns to Mode 0. If the processor enters any power mode from Mode 1 to Mode 3 while the processor is in an interrupt handler, only an interrupt source with a higher priority than the current interrupt can wake up the device (higher value in IPRx registers).

Two steps are usually required to configure an interrupt

- Configuring a peripheral to generate an interrupt request to the NVIC.
- Configuring the NVIC for that peripheral request.

Table 54. Interrupt Vector Table

Position No.	Vector	Wake Up Processor from Mode 1	Wake Up Processor from Mode 2 or Mode 3
0	Wake-up timer	Yes	Yes
1	External Interrupt 0	Yes	Yes
2	External Interrupt 1	Yes	Yes
3	External Interrupt 2	Yes	Yes
4	Reserved		
5	External Interrupt 4	Yes	Yes
6	External Interrupt 5	Yes	Yes
7	Reserved		
8	External Interrupt 7	Yes	Yes
9	External Interrupt 8	Yes	Yes
10	Watchdog timer	Yes	Yes
11	Reserved		
12	Reserved		
13	LV Die Interrupt 0	Yes	No
14	MDIO	Yes	No
15	GP Timer 0	Yes	No
16	GP Timer 1	Yes	No
17	Flash controller	Yes	No
18	UART	Yes	No
19	SPI0	Yes	No

Position No.	Vector	Wake Up Processor from Mode 1	Wake Up Processor from Mode 2 or Mode 3
20	SPI1	Yes	No
21	I2C0 slave	Yes	No
22	I2C0 master	Yes	No
23	PLA 0	Yes	No
24	PLA 1	Yes	No
25	DMA error	Yes	No
26	DMA Channel 0 (SPI0 Tx) done	Yes	No
27	DMA Channel 1 (SPI0 Rx) done	Yes	No
28	DMA Channel 2 (SPI1 Tx) done	Yes	No
29	DMA Channel 3 (SPI1 Rx) done	Yes	No
30	DMA Channel 4 (UART Tx) done	Yes	No
31	DMA Channel 5 (UART Rx) done	Yes	No
32	DMA Channel 6 (I2C0 slave Tx) done	Yes	No
33	DMA Channel 7 (I2C0 slave Rx) done	Yes	No
34	DMA Channel 8 (I2C0 master) done	Yes	No
35	DMA Channel 9 (I2C1 slave Tx) done	Yes	No
36	DMA Channel 10 (I2C1 slave Rx) done	Yes	No
37	DMA Channel 11 (I2C1 master) done	Yes	No
38	DMA Channel 12 (ADC) done	Yes	No
39	DMA Channel 13 (flash) done	Yes	No
40	Reserved		
41	Reserved		
42	Reserved		
43	Reserved		
44	I2C1 slave	Yes	No
45	I2C1 master	Yes	No
46	PLA 2	Yes	No
47	PLA 3	Yes	No
48	GP Timer 2	Yes	No
49	LV Die Interrupt 1	Yes	No
50	PWM trip	Yes	No
51	PWM PAIR0	Yes	No
52	PWM PAIR1	Yes	No
53	PWM PAIR2	Yes	No
54	PWM PAIR3	Yes	No

Internal to the ARM Cortex-M3 processor, the highest user-programmable priority (0) is treated as fourth priority—after a reset, an NMI, and a hard fault. The ADuCM320 implements three priority bits, which means that eight priority levels are available as programmable priorities. Note that 0 is the default priority for all the programmable priorities. If the same priority level is assigned to two or more interrupts, their hardware priority (the lower the position number) determines the order in which the processor activates them. For example, if both SPI0 and SPI1 are Priority Level 1, then SPI0 has higher priority.

To enable an interrupt for any peripheral listed from 0 to 31 in Table 54, set the appropriate bit in the ISER0 register. ISER0 is a 32-bit register, and each bit corresponds to the first 32 entries in Table 54.

For example, to enable External Interrupt 4 interrupt source in the NVIC, set ISER0[5] = 1. Similarly, to disable External Interrupt 4, set ICER0[5] = 1.

To enable an interrupt for any peripheral listed from 32 to 54 in Table 54, set the appropriate bit in the ISER1 register. ISER1 is a 32-bit register, and ISER1 Bit 0 to Bit 22 correspond to the entries 32 to 54 in Table 54.

For example, to enable the PWM PAIR0 interrupt source in the NVIC, set ISER1[20] = 1. Similarly, to disable the PWM PAIR0 interrupt, set ICER1[20] = 1.

Alternatively, CMSIS provides a number of useful NVIC functions in the core_cm3.h file. The function NVIC_EnableIRQ(PWM_PAIR0_IRQn) enables the PWM PAIR0 interrupt. The interrupt can be disabled by calling the NVIC_DisableIRQ(PWM_PAIR0_IRQn) function.

To set the priority of a peripheral interrupt, the IPRx bits can be set appropriately or, alternatively, the NVIC_SetPriority() function can be called. For example, NVIC_SetPriority(TIMER0_IRQn, 2) configures the GP Timer 0 interrupt with a priority level of 2.

Table 55 lists the registers to enable and disable relevant interrupts and set the priority levels. The registers in Table 55 are defined in the CMSIS core_cm3.h file that is shipped with tools from third party vendors.

Table 55. NVIC Registers

Address	Analog Devices Header File Name	Description	Access
0xE000E004	ICTR	Shows the number of interrupt lines that the NVIC supports.	R
0xE000E010	STCSR	SYSTICK control and status register.	RW
0xE000E014	STRVR	SYSTICK reload value register.	RW
0xE000E018	STCVR	SYSTICK current value register.	RW
0xE000E01C	STCR	SYSTICK calibration value register.	R
0xE000E100	ISER0	Set IRQ0 to IRQ31 enable. Each bit corresponds to Interrupt 0 to Interrupt 31 in Table 54.	RW
0xE000E104	ISER1	Set IRQ32 to IRQ54 enable. Each bit corresponds to interrupt 32 to Interrupt 54 in Table 54.	RW
0xE000E180	ICER0	Clear IRQ0 to IRQ31 by setting appropriate bit. Each bit corresponds to Interrupt 0 to Interrupt 31 in Table 54.	RW
0xE000E184	ICER1	Clear IRQ32 to IRQ54 by setting appropriate bit. Each bit corresponds to Interrupt 32 to Interrupt 54 in Table 54.	RW
0xE000E200	ISPR0	Set IRQ0 to IRQ31 pending. Each bit corresponds to Interrupt 32 to Interrupt 38 in Table 54.	RW
0xE000E204	ISPR1	Set IRQ32 to IRQ54 pending. Each bit corresponds to Interrupt 32 to Interrupt 54 in Table 54.	RW
0xE000E280	ICPR0	Clear IRQ0 to IRQ31 pending. Each bit corresponds to Interrupt 32 to Interrupt 38 in Table 54.	RW
0xE000E284	ICPR1	Clear IRQ32 to IRQ54 pending. Each bit corresponds to Interrupt 32 to Interrupt 54 in Table 54.	RW
0xE000E300	IABR0	IRQ0 to IRQ31 active bits.	RW
0xE000E304	IABR1	IRQ32 to IRQ54 active bits.	RW
0xE000E400	IPR0	IRQ0 to IRQ3 priority.	RW
0xE000E404	IPR1	IRQ4 to IRQ7 priority.	RW
0xE000E408	IPR2	IRQ8 to IRQ11 priority.	RW
0xE000E40C	IPR3	IRQ12 to IRQ15 priority.	RW
0xE000E410	IPR4	IRQ16 to IRQ19 priority.	RW
0xE000E414	IPR5	IRQ20 to IRQ23 priority.	RW
0xE000E418	IPR6	IRQ24 to IRQ27 priority.	RW
0xE000E41C	IPR7	IRQ28 to IRQ31 priority.	RW
0xE000E420	IPR8	IRQ32 to IRQ35 priority.	RW
0xE000E424	IPR9	IRQ36 to IRQ39 priority.	RW
0xE000E428	IPR10	IRQ40 to IRQ43 priority.	RW
0xE000E42C	IPR11	IRQ44 to IRQ47 priority.	RW
0xE000E430	IPR12	IRQ48 to IRQ51 priority.	RW
0xE000E434	IPR13	IRQ52 to IRQ54 priority.	RW
0xE000ED00	CPUID	CPUID base register.	R
0xE000ED04	ICSR	Interrupt control and status register.	RW
0xE000ED08	VTOR	Vector table offset register.	RW
0xE000ED0C	AIRCR	Application interrupt/reset control register.	RW
0xE000ED10	SCR	System control register.	RW
0xE000ED14	CCR	Configuration control register.	RW
0xE000ED18	SHPR1	System Handlers Register 1.	RW
0xE000ED1C	SHPR2	System Handlers Register 2.	RW
0xE000ED20	SHPR3	System Handlers Register 3.	RW
0xE000ED24	SHCRS	System handler control and state.	RW
0xE000ED28	CFSR	Configurable fault status.	RW
0xE000ED2C	HFSR	Hard fault status.	RW
0xE000ED34	MMAR	Memory manage fault address register.	RW
0xE000ED38	BFAR	Bus fault address.	RW
0xE000EF00	STIR	Software trigger interrupt register.	W

EXTERNAL INTERRUPT CONFIGURATION

Seven external interrupts are implemented. These seven external interrupts can be separately configured to detect any combination of the following type of events:

- Edge: rising edge, falling edge, or both rising and falling edges. An interrupt signal (pulse) is sent to the NVIC upon detecting a transition from low to high, high to low, or on either high to low or low to high.
- Level: high or low. An interrupt signal is generated and remains asserted in the NVIC until the conditions generating the interrupt deassert. The level must be maintained for a minimum of one core clock cycle to be detected.

The external interrupt detection unit block is in the always-on section and allows external interrupt to wake up the device when in hibernate mode.

Ensure that the associated GPxIE register bit is enabled for the required external interrupt input. The GPxIE register enables the input path circuit for the external interrupt.

For example, for External Interrupt 0, the following code disables the P0.3 output and enables the input path. The appended code also enables the External Interrupt 0 NVIC interrupt source:

```
pADI_GP0->GPOE &= 0xf7;               //Disable P0.3 output.
pADI_GP0->GPIE |= 0x8;                //Enable input path for P0.3 input.
pADI_INTERRUPT->EI0CFG |= 0x8;        //External IRQ0 enabled.
NVIC_EnableIRQ(EINT0_IRQn);           //Enable External Interrupt 0 source.
```

REGISTER SUMMARY: EXTERNAL INTERRUPTS

Table 56. External Interrupts Register Summary

Address	Name	Description	Reset	RW
0x40002420	EI0CFG	External Interrupt Configuration 0	0x0000	RW
0x40002424	EI1CFG	External Interrupt Configuration 1	0x0000	RW
0x40002428	EI2CFG	External Interrupt Configuration 2	0x0000	RW
0x40002430	EICLR	External interrupt clear	0x0000	RW

REGISTER DETAILS: EXTERNAL INTERRUPTS

External Interrupt Configuration Register 0

Address: 0x40002420, Reset: 0x0000, Name: EI0CFG

Table 57. Bit Descriptions for EI0CFG

Bits	Bit Name	Description	Reset	Access
[15:12]	RESERVED	Reserved.	0	
11	IRQ2EN	External Interrupt 2 enable bit. 0: External Interrupt 2 disabled 1: External Interrupt 2 enabled	0x0	RW
[10:8]	IRQ2MDE	External Interrupt 2 mode registers. 000: rising edge 001: falling edge 010: rising or falling edge 011: high level 100: low level 101: falling edge (same as 001) 110: rising or falling edge (same as 010) 111: high level (same as 011)	0x0	RW
7	IRQ1EN	External Interrupt 1 enable bit. 0: External Interrupt 0 disabled 1: External Interrupt 0 enabled	0x0	RW

Bits	Bit Name	Description	Reset	Access
[6:4]	IRQ1MDE	External Interrupt 1 mode registers. 000: rising edge 001: falling edge 010: rising or falling edge 011: high level 100: low level 101: falling edge (same as 001) 110: rising or falling edge (same as 010) 111: high level (same as 011)	0x0	RW
3	IRQOEN	External Interrupt 0 enable bit. 0: External Interrupt 0 disabled 1: External Interrupt 0 enabled	0x0	RW
[2:0]	IRQ0MDE	External Interrupt 0 mode registers. 000: rising edge 001: falling edge 010: rising or falling edge 011: high level 100: low level 101: falling edge (same as 001) 110: rising or falling edge (same as 010) 111: high level (same as 011)	0x0	RW

External Interrupt Configuration Register 1

Address: 0x40002424, Reset: 0x0000, Name: EI1CFG

Table 58. Bit Descriptions for EI1CFG

Bits	Bit Name	Description	Reset	Access
15	IRQ7EN	External Interrupt 7 enable bit. 0: External Interrupt 7 disabled 1: External Interrupt 7 enabled	0x0	RW
[14:12]	IRQ7MDE	External Interrupt 7 mode registers. 000: rising edge 001: falling edge 010: rising or falling edge 011: high level 100: low level 101: falling edge (same as 001) 110: rising or falling edge (same as 010) 111: high level (same as 011)	0x0	RW
[11:8]	RESERVED	Reserved.	0	
7	IRQ5EN	External Interrupt 5 enable bit. 0: External Interrupt 5 disabled 1: External Interrupt 5 enabled	0x0	RW
[6:4]	IRQ5MDE	External Interrupt 5 mode registers. 000: rising edge 001: falling edge 010: rising or falling edge 011: high level 100: low level 101: falling edge (same as 001) 110: rising or falling edge (same as 010) 111: high level (same as 011)	0x0	RW

Bits	Bit Name	Description	Reset	Access
3	IRQ4EN	External Interrupt 4 enable bit. 0: External Interrupt 4 disabled 1: External Interrupt 4 enabled	0x0	RW
[2:0]	IRQ4MDE	External Interrupt 4 mode registers. 000: rising edge 001: falling edge 010: rising or falling edge 011: high level 100: low level 101: falling edge (same as 001) 110: rising or falling edge (same as 010) 111: high level (same as 011)	0x0	RW

External Interrupt Configuration Register 2

Address: 0x40002428, Reset: 0x0000, Name: EI2CFG

Table 59. Bit Descriptions for EI2CFG

Bits	Bit Name	Description	Reset	Access
15:4	RESERVED	Reserved.	0x0	
3	IRQ8EN	External Interrupt 8 enable bit. 0: External Interrupt 8 disabled 1: External Interrupt 8 enabled	0x0	RW
[2:0]	IRQ8MDE	External Interrupt 8 mode registers. 000: rising edge 001: falling edge 010: rising or falling edge 011: high level 100: low level 101: falling edge (same as 001) 110: rising or falling edge (same as 010) 111: high level (same as 011)	0x0	RW

External Interrupt Clear Register

Address: 0x40002430, Reset: 0x0000, Name: EICLR

Table 60. Bit Descriptions for EICLR

Bits	Bit Name	Description	Reset	Access
[15:9]	RESERVED	Reserved.	0x0	RW
8	IRQ8	External interrupt 8. Set to 1 to clear an internal interrupt flag. Cleared automatically by hardware.	0x0	RW
7	IRQ7	External interrupt 7. Set to 1 to clear an internal interrupt flag. Cleared automatically by hardware.	0x0	RW
6	RESERVED	Reserved.	0x0	RW
5	IRQ5	External interrupt 5. Set to 1 to clear an internal interrupt flag. Cleared automatically by hardware.	0x0	RW
4	IRQ4	External interrupt 4. Set to 1 to clear an internal interrupt flag. Cleared automatically by hardware.	0x0	RW
3	RESERVED		0x0	RW
2	IRQ2	External interrupt 2. Set to 1 to clear an internal interrupt flag. Cleared automatically by hardware.	0x0	RW
1	IRQ1	External interrupt 1. Set to 1 to clear an internal interrupt flag. Cleared automatically by hardware.	0x0	RW
0	IRQ0	External interrupt 0. Set to 1 to clear an internal interrupt flag. Cleared automatically by hardware.	0x0	RW

LOW VOLTAGE ANALOG DIE INTERRUPT CONFIGURATION

Two interrupt lines are available between the low voltage analog die and the interrupt controller on the digital die.

These two interrupt lines are the outputs of two multiplexers of multiple interrupt sources from the low voltage analog die.

The full list of interrupt sources from the low voltage analog die are as follows:

- ADC software conversion complete interrupt. This is asserted at the end of an ADC conversion when this interrupt source is enabled.
- ADC sequencer complete interrupt. This is the interrupt asserted by the ADC sequencer.
- Analog comparator interrupt. If the input signal is outside the selected threshold, this interrupt is asserted.
- Digital comparator interrupt. If the ADC result is outside the selected threshold, this interrupt is asserted.
- IDAC thermal shutdown interrupt.
- IDAC external reference resistor status interrupt.
- Read ECC interrupt source. Error correction and checking is available on the interface between the digital and analog die. If a read error occurs (for example, an error on the ADC result), this interrupt is asserted.
- Write ECC interrupt source. If the ECC returns an error on a value written to the LV die, this interrupt is asserted.

Low Voltage Die Interrupt 1 is more flexible than Low Voltage Die Interrupt 0. The key differences are as follows:

- Low Voltage Die Interrupt 1 allows all seven different interrupt sources as configured by INTSEL[7:0] to be enabled. In the interrupt handler, the LV1 interrupt source can be determined by the INTSTA register
- Low Voltage Die Interrupt 0 allows only one of the possible seven interrupt sources selected by INTSEL[15:8] to be enabled at a given time. The INTSTA register is not valid for LV Interrupt 0.

To clear an interrupt, set the appropriate bit in the INTCLR register.

Note that there is a delay period required after writing to INTCLR before the associated status bit in the INTSTA register is updated.

If polling is used of the INTSTA register, the following example code can be used:

```
pADI_LV_INT->INTCLR = 0x1; // Clear Irq source
delay(10);
ucLVIrqStatus = pADI_LV_INT->INTSTA;

// Simple delay routine
void delay (long int length)
{
 while (length >0)
 length--;
}
```

REGISTER SUMMARY: LOW VOLTAGE DIE INTERRUPTS

Table 61. Low Voltage Die Interrupts Register Summary

Address	Name	Description	Reset	RW
0x40083004	INTCLR	Interrupt clear register	0x0000	W
0x40083008	INTSEL	Interrupt mask register	0x0000	RW
0x4008300C	INTSTA	Interrupt status register	0x0000	R

REGISTER DETAILS: LOW VOLTAGE DIE INTERRUPTS

Interrupt Clear Register

Address: 0x40083004, Reset: 0x0000, Name: INTCLR

Table 62. Bit Descriptions for INTCLR

Bits	Bit Name	Description	Reset	Access
7	CLR_WRECC_ERR	Write 1 to this bit to clear the write ECC error interrupt flag.	0x0	W
6	CLR_RDECC_ERR	Write 1 to this bit to clear the read ECC error interrupt flag.	0x0	W
5	CLR_IDAC_EXTRESLOW	Write 1 to this bit to clear the IDAC EXTRESLOW interrupt flag.	0x0	W
4	CLR_IDAC_TSHUT	Write 1 to this bit to clear the IDAC TSHUT interrupt flag.	0x0	W
3	CLR_ACOMP	Write 1 to this bit to clear the analog compare interrupt flag.	0x0	W
2	CLR_DCOMP	Write 1 to this bit to clear the digital compare interrupt flag.	0x0	W
1	CLR_ADC_SEQ	Write 1 to this bit to clear the ADC sequence conversion interrupt flag.	0x0	W
0	CLR_ADC_SOFTCONV	Write 1 to this bit to clear the ADC software conversion interrupt flag.	0x0	W

Interrupt Mask Register

Address: 0x40083008, Reset: 0x0000, Name: INTSEL

Table 63. Bit Descriptions for INTSEL

Bits	Bit Name	Description	Reset	Access
15	SEL_WRECC_ERR_0	Write 1 to this bit to enable write ECC error interrupt for Interrupt Pin 0.	0x0	RW
14	SEL_RDECC_ERR_0	Write 1 to this bit to enable read ECC error interrupt for Interrupt Pin 0.	0x0	RW
13	SLE_IDAC_EXTRESLOW_0	Write 1 to this bit to enable IDAC EXTRESLOW interrupt for Interrupt Pin 0.	0x0	RW
12	SEL_IDAC_TSHUT_0	Write 1 to this bit to enable IDAC TSHUT interrupt for Interrupt Pin 0.	0x0	RW
11	SEL_ACOMP_0	Write 1 to this bit to enable analog comparator interrupt for Interrupt Pin 0.	0x0	RW
10	SEL_DCOMP_0	Write 1 to this bit to enable digital comparator interrupt for Interrupt Pin 0.	0x0	RW
9	SEL_ADC_SEQ_0	Write 1 to this bit to enable ADC sequence conversion interrupt for Interrupt Pin 0.	0x0	RW
8	SEL_ADC_SOFTCONV_0	Write 1 to this bit to enable ADC software conversion interrupt for Interrupt Pin 0.	0x0	RW
7	SEL_WRECC_ERR_1	Write 1 to this bit to enable write ECC error interrupt for Interrupt Pin 1.	0x0	RW
6	SEL_RDECC_ERR_1	Write 1 to this bit to enable read ECC error interrupt for Interrupt Pin 1.	0x0	RW
5	SLE_IDAC_EXTRESLOW_1	Write 1 to this bit to enable IDAC EXTRESLOW interrupt for Interrupt Pin 1.	0x0	RW
4	SEL_IDAC_TSHUT_1	Write 1 to this bit to enable IDAC TSHUT interrupt for Interrupt Pin 1.	0x0	RW
3	SEL_ACOMP_1	Write 1 to this bit to enable analog comparator interrupt for Interrupt Pin 1.	0x0	RW
2	SEL_DCOMP_1	Write 1 to this bit to enable digital comparator interrupt for Interrupt Pin 1.	0x0	RW
1	SEL_ADC_SEQ_1	Write 1 to this bit to enable ADC sequence conversion interrupt for Interrupt Pin 1.	0x0	RW
0	SEL_ADC_SOFTCONV_1	Write 1 to this bit to enable ADC software conversion interrupt for Interrupt Pin 1.	0x0	RW

Interrupt Status Register

Address: 0x4008300C, Reset: 0x0000, Name: INTSTA

Table 64. Bit Descriptions for INTSTA

Bits	Bit Name	Description	Reset	Access
7	WRECC_ERR	Write data ECC error interrupt status.	0x0	R
6	RDECC_ERR	Read data ECC error interrupt status.	0x0	R
5	IDAC_EXTRESLOW	IDAC EXTRESLOW interrupt status.	0x0	R
4	IDAC_TSHUT	IDAC temperature TSHT interrupt status.	0x0	R
3	ACOMP	Analog comparator interrupt status.	0x0	R
2	DCOMP	Digital comparator interrupt status.	0x0	R
1	ADC_SEQ	ADC sequence interrupt status.	0x0	R
0	ADC_SOFTCONV	ADC software conversion interrupt status.	0x0	R

RESET

RESET FEATURES

There are four kinds of resets:

- External reset
- Power-on reset
- Watchdog timeout
- Software system reset

RESET OPERATION

The software system reset is provided as part of the Cortex-M3 processor. To generate a software system reset, the NVIC_SystemReset() function must be called. This effectively writes 0x05FA to the top 16 bits of an AIRCR NVIC register. This function along with other useful functions are defined in the CMSIS header files that are shipped with the tools from third party vendors. The NVIC_SystemReset() function is defined in the core_cm3.h file.

Analog peripherals have the option of maintaining their state after a software or watchdog reset. This function is enabled by default. It can be disabled using the LVRST register. Note that while debugging, the software tools generally only issue a software reset; therefore, an external reset is needed to return registers to their default values.

The GPIO pins and PLA also have the option of maintaining their state after a software or watchdog reset. By default, this function is enabled. Writing a value of 0x1 to RSTCFG configures the GPIO pins and PLA to reset after a software or watchdog reset. Before writing to this register, 0x2009 must be written to RSTKEY followed by 0x0426. After the two keys are written to RSTKEY, RSTCFG must be immediately written.

The RSTSTA register stores the cause for the reset until it is cleared by writing the RSTSTA register. RSTSTA can be used during a reset exception service routine to identify the source of the reset.

The watchdog timer is enabled by default after a reset. The default timeout period is approximately 32 seconds.

User code should disable the watchdog timer at the start of user code when debugging or if the watchdog timer is not required.

```
pADI_WDT->T3CON = 0x00;                        // Disable watchdog timer
```

Table 65. Device Reset Implications

	Impact					
Reset	**Reset External Pins to Default State**	**Execute Kernel**	**Reset All MMRs Except RSTSTA**	**Reset All Peripherals**	**Valid SRAM**	**RSTSTA After Reset Event**
SWRST	Yes[1]/No[2]	Yes	Yes/No[2]	Yes/No[2]	Yes/No[3]	RSTSTA[3] = 1
WDRST	Yes[1]/No[2]	Yes	Yes/No[2]	Yes/No[2]	Yes/No[3]	RSTSTA[2] = 1
External Reset Pin	Yes[1]	Yes	Yes	Yes	Yes/No[3]	RSTSTA[1] = 1
POR	Yes[1]	Yes	Yes	Yes	No	RSTSTA[0] = 1

[1] P2.2 returns to its default state, that is, POR output. It is low only in the case of a POR event; in all other cases, it remains high.
[2] GPIOs, PLA, and analog peripherals have the option of retaining their state during a watchdog or software reset.
[3] RAM is not valid in the case of a reset following an MDIO download.

REGISTER SUMMARY: RESET

Table 66. Reset Register Summary

Address	Name	Description	Reset	RW
0x40002408	RSTCFG	Reset configuration	0x0000	RW
0x4000240C	RSTKEY	Key protection for RSTCFG	0x0000	RW
0x40002440	RSTSTA	Reset status	0x0000	RW
0x40082C34	LVRST	LV die reset configuration	0x0000	RW

REGISTER DETAILS: RESET

Reset Status Register

Address: 0x40002440, Reset: 0x0000, Name: RSTSTA

Table 67. Bit Descriptions for RSTSTA

Bits	Bit Name	Description	Reset	Access
[15:4]	RESERVED		0x0	R
3	SWRST	Software reset. Software reset. Set automatically to 1 when the Cortex-M3 system reset is generated. Cleared by writing 1 to the bit.	0x0	RW1C
2	WDRST	Watchdog timeout. Set automatically to 1 when a watchdog timeout occurs. Cleared by writing 1 to the bit.	0x0	RW1C
1	EXTRST	External reset. Set automatically to 1 when an external reset occurs. Cleared by writing 1 to the bit.	0x0	RW1C
0	POR	Power-on reset. Set automatically when a power-on reset occurs. Cleared by writing 1 to the bit.	0x0	RW1C

Reset Configuration Register

Address: 0x40002408, Reset: 0x0000, Name: RSTCFG

Table 68. Bit Descriptions for RSTCFG

Bits	Bit Name	Description	Reset	Access
0	GPIO_PLA_RETAIN	GPIO/PLA retain their status after WDT and software reset. Note that, if GPIO_PLA_RETAIN = 0, P3.4 goes high-Z for approximately 260 μs after software/watchdog reset. 1: GPIO/PLA do not retain status after watchdog or software reset. 0: GPIO/PLA retain status after watchdog or software reset.	0x0	RW

Key Protection for RSTCFG Register

Address: 0x4000240C, Reset: 0x0000, Name: RSTKEY

Table 69. Bit Descriptions for RSTKEY

Bits	Bit Name	Description	Reset	Access
[15:0]	RSTKEY	Reset configuration key register. The RSTCFG register is key-protected. Two writes to the key are necessary to change the value in the RSTCFG register: first 0x2009, then 0x0426. The RSTCFG register should then be written. A write to any other register on the APB bus before writing to RSTCFG returns the protection to the lock state.	0x0	RW

LV Die Reset Configuration Register

Address: 0x40082C34, Reset: 0x0000, Name: LVRST

Table 70. Bit Descriptions for LVRST

Bits	Bit Name	Description	Reset	Access
[15:1]	RESERVED	Reserved.	0x0	R
0	RETAIN	LV retains status after WDT and software reset. 0: LV die retains status after watchdog or software reset. 1: LV die does not retain status after watchdog or software reset.	0x0	RW

DMA CONTROLLER

DMA FEATURES

- 14 dedicated and independent DMA channels.
- Two programmable priority levels for each DMA channel.
  - Each priority level arbitrates using a fixed priority that is determined by the DMA channel number.
  - Channels with lower numbers have higher priority. For example, SPI0 transmit has the highest priority, and the next highest is the SPI0 receive.
- Each DMA channel can access a primary and/or alternate channel control structure.
- Supports multiple DMA transfer types.
  - Memory to memory
  - Memory to peripheral
  - Peripheral to memory

DMA OVERVIEW

Direct memory access (DMA) is used to provide high speed data transfer between peripherals and memory. Data can be moved quickly by DMA without any processor actions. This keeps processor resources free for other operations.

The DMA controller has 14 channels in total. The 14 channels used are dedicated to managing DMA requests from specific peripherals. Channels are assigned as shown in Table 71.

Table 71. DMA Channel Assignment

Channel	Peripheral
0	SPI0 Tx
1	SPI0 Rx
2	SPI1 Tx
3	SPI1 Rx
4	UART Tx
5	UART Rx
6	I2C0 slave Tx
7	I2C0 slave Rx
8	I2C0 master
9	I2C1 slave Tx
10	I2C1 slave Rx
11	I2C1 master
12	ADC
13	Flash

The channels are connected to dedicated hardware DMA requests; a software trigger is also supported on each channel. This configuration is done by software. Each DMA channel has a programmable priority level: default or high. Within a priority level, arbitration is performed using a fixed priority that is determined by the DMA channel number. Channels with lower numbers have higher priority. For example, SPI0 transmit has the highest priority, and the next highest is the SPI0 receive.

The DMA controller supports multiple DMA transfer data widths—independent source and destination transfer size (byte, half word, and word). Source/destination addresses must be aligned on the data size.

The DMA controller supports peripheral-to-memory, memory-to-peripheral, and memory-to-memory transfers and access to flash or SRAM as source and destination.

DMA OPERATION

The DMA controller performs direct memory transfer by sharing the system bus with the Cortex-M3 processor. The DMA request may stall the processor access to the system bus for some bus cycles when the processor and DMA are targeting the same destination (memory or peripheral).

DMA INTERRUPTS

An interrupt can be produced for each DMA channel when a transfer is complete. Separate interrupt enable bits are available in the NVIC for each of the DMA channels.

The DMA controller fetches channel control data structures located in the SRAM memory to perform data transfers. When enabled to use DMA operation, the DMA-capable peripherals request the DMA controller for transfer. At the end of the programmed number of DMA transfers for a channel, the DMA controller generates an interrupt corresponding to that channel. This interrupt indicates the completion of the DMA transfer.

DMA PRIORITY

The priority of a channel is determined by its number and priority level. Each channel can have two priority levels: default or high. All channels at high priority level have higher priority than all channels at default priority level. At the same priority level, a channel with a lower channel number has higher priority than a channel with a higher channel number. The DMA channel priority levels can be changed by writing into the appropriate bit in the DMAPRISET register.

CHANNEL CONTROL DATA STRUCTURE

Every channel has two control data structures associated with it: primary data structure and an alternate data structure. For simple transfer modes, the DMA controller uses either the primary or the alternate data structure. For more complex data transfer modes, such as ping-pong or scatter-gather, the DMA controller uses both the primary and alternate data structures. Each control data structure (primary or alternate) occupies four 32-bit locations in the memory as shown in Table 72. The entire channel control data structure is shown in Table 73.

Table 72. Channel Control Data Structure

Offset	Name	Description
0x00	SRC_END_PTR	Source end pointer
0x04	DST_END_PTR	Destination end pointer
0x08	CHNL_CFG	Control data configuration
0x0C	Reserved	Reserved

Before the controller can perform a DMA transfer, the data structure related to the DMA channel must be programmed at the designated location in system memory, SRAM.

- The source end pointer memory location contains the end address of the source data.
- The destination end pointer memory location contains the end address of the destination data.
- The control data configuration memory location contains the channel configuration control data.

The programming determines the source and destination data size, number of transfers, and the number of arbitrations.

Table 73. Memory Map of Primary and Alternate DMA Structures

	Primary Structures		Alternate Structures	
Channel	Register Description	Offset Address	Register Description	Offset Address
Channel 13	Reserved; set to 0	0x0DC	Reserved; set to 0	0x1DC
	Control	0x0D8	Control	0x1D8
	Destination end pointer	0x0D4	Destination end pointer	0x1D4
	Source end pointer	0x0D0	Source end pointer	0x1D0
…	…	…	…	…
Channel 1	Reserved; set to 0	0x01C	Reserved; set to 0	0x11C
	Control	0x018	Control	0x118
	Destination end pointer	0x014	Destination end pointer	0x114
	Source end pointer	0x010	Source end pointer	0x110
Channel 0	Reserved; set to 0	0x00C	Reserved; set to 0	0x10C
	Control	0x008	Control	0x108
	Destination end pointer	0x004	Destination end pointer	0x104
	Source end pointer	0x000	Source end pointer	0x100

The user must define DMA structures in their source code as shown in the examples in the Example Code: Define DMA Structures section. After the structure has been defined, its start address must be assigned to the DMA base address pointer register, DMAPDBPTR.

Each register for each DMA channel is then at the offset address, as specified in Table 73, plus the value in the DMAPDBPTR register.

Example Code: Define DMA Structures

To define DMA structures,

```
memset(dmaChanDesc,0x0, sizeof(dmaChanDesc));  // Set up the DMA base address pointer register.
uiBasPtr = (unsigned int)&dmaChanDesc;         // Set up the DMA base pointer.
pADI_DMA->DMACFG = 1;                          // Enable DMA controller
pADI_DMA->DMAPDBPTR = uiBasPtr;
```

CONTROL DATA CONFIGURATION

For each DMA transfer, the CHNL_CFG memory location provides the control information for the DMA transfer to the controller.

Table 74. Control Data Configuration

Bits	Name	Description		
[31:30]	DST_INC	Destination address increment. The address increment depends on the source data width as follows:		
		Source Data Width	**DST_INC**	**Destination Address Increment**
		Byte	00	Byte.
			01	Half word.
			10	Word.
			11	No increment. Address remains set to the value that the DST_END_PTR memory location contains.
		Half Word	00	Reserved.
			01	Half word.
			10	Word.
			11	No increment. Address remains set to the value that the DST_END_PTR memory location contains.
		Word	00	Reserved.
			01	Reserved.
			10	Word.
			11	No increment. Address remains set to the value that the DST_END_PTR memory location contains.
[29:28]	DST_SIZE	Size of the destination data. Must match SRC_SIZE. 00: byte. 01: half word. 10: word. 11: reserved.		
[27:26]	SRC_INC	Source address increment. The address increment depends on the source data width as follows:		
		Source Data Width	**DST_INC**	**Source Address Increment**
		Byte	00	Byte.
			01	Half word.
			10	Word.
			11	No increment. Address remains set to the value that the SRC_END_PTR memory location contains.
		Half Word	00	Reserved.
			01	Half word.
			10	Word.
			11	No increment. Address remains set to the value that the SRC_END_PTR memory location contains.
		Word	00	Reserved.
			01	Reserved.
			10	Word.
			11	No increment. Address remains set to the value that the SRC_END_PTR memory location contains.

Bits	Name	Description
[25:24]	SRC_SIZE	Size of the source data. 00: byte. 01: half word. 10: word. 11: reserved.
[23:18]	RESERVED	Undefined. Write as 0.
[17:14]	R_POWER	Set these bits to control how many DMA transfers can occur before the controller re-arbitrates. Must be set to 0000 for all DMA transfers involving peripherals. Note that the operation of the DMA is indeterminate if a value other than 0000 is programmed in this location for DMA transfers involving peripherals.
[13:4]	N_MINUS_1	The number of configured transfers minus 1 for that channel. The 10-bit value indicates the number of DMA transfers (not the total number of bytes) minus one. The possible values are 0x000: 1 DMA transfer 0x001: 2 DMA transfers 0x002: 3 DMA transfers … 0x3FF: 1024 DMA transfers.
3	RESERVED	Undefined. Write as 0.
[2:0]	CYCLE_CTRL	The transfer types of the DMA cycle. 000: stop (invalid) 001: basic 010: autorequest 011: ping-pong 100: memory scatter-gather primary 101: memory scatter-gather alternate 110: peripheral scatter-gather primary 111: peripheral scatter-gather alternate

During the DMA transfer process, but before arbitration, CHNL_CFG is written back to system memory with the N_MINUS_1 field changed to reflect the number of transfers yet to be completed.

When the DMA cycle is complete, the CYCLE_CTRL bits are made invalid to indicate the completion of the transfer.

DMA TRANSFER TYPES (CHNL_CFG[2:0])

The DMA controller supports five types of DMA transfers. The various types are selected by programming the appropriate values into the CYCLE_CTRL bits (Bits[2:0]) in the CHNL_CFG location of the control data structure.

Invalid (CHNL_CFG[2:0] = 000)

This means no DMA transfer is enabled for the channel. After the controller completes a DMA cycle, it sets the cycle type to invalid to prevent it from repeating the same DMA cycle.

Basic (CHNL_ CFG[2:0] = 001)

In this mode, the controller can be configured to use either the primary or alternate data structure. The peripheral must present a request for every data transfer. After the channel is enabled, when the controller receives a request, it performs the following operations:

1. The controller performs a transfer. If the number of transfers remaining is zero, the flow continues at Step 3.
2. The controller arbitrates
   a. If a higher priority channel is requesting service, then the controller services that channel.
   b. If the peripheral or software signals a request to the controller, then it continues at Step 1.
3. At the end of the transfer, the controller generates the corresponding DMA channel interrupt in the NVIC

Autorequest (CHNL_CFG[2:0] = 010)

When the controller operates in this mode, it is only necessary for the controller to receive a single request to enable it to complete the entire DMA cycle. This allows a large data transfer to occur without significantly increasing the latency for servicing higher priority requests or requiring multiple requests from the processor or peripheral. This mode is very useful for a memory-to-memory copy application.

Autorequest is not suitable for peripheral use, except for the ADC sequencer mode, where a number of peripheral operations need to be completed.

In this mode, the controller can be configured to use either the primary or alternate data structure. After the channel is enabled, when the controller receives a request, it performs the following operations:

1. The controller performs min(2^{R_POWER}, N) transfers for the channel, where R_POWER is Bits[17:14] of the control data configuration register and N is the number of transfers. If the number of transfers remaining is zero, the flow continues at Step 3.
2. A request for the channel is automatically generated. The controller arbitrates. If the channel has the highest priority, the DMA cycle continues at Step 1.
3. At the end of the transfer, the controller generates an interrupt for the corresponding DMA channel.

Ping-Pong (CHNL_CFG[2:0] = 011)

In ping-pong mode, the controller performs a DMA cycle using one of the data structures and then performs a DMA cycle using the other data structure. The controller continues to alternate between using the primary and alternate data structures until it either reads a data structure that is invalid or the host processor disables the channel.

This mode is useful for transferring data from peripheral to memory using different buffers in the memory. In a typical application, the host must configure both primary and alternate data structures before starting the transfer. As the transfer progresses, the host can subsequently configure primary or alternate control data structures in the interrupt service routine when the corresponding transfer ends.

The DMA controller interrupts the processor after the completion of transfers associated with each control data structure. The individual transfers using either the primary or alternate control data structure work exactly the same as a basic DMA transfer.

Memory Scatter-Gather (CHNL_CFG[2:0] = 100 or 101)

In memory scatter-gather mode, the controller must be configured to use both the primary and alternate data structures. The controller uses the primary data structure to program the control configuration for the alternate data structure. The alternate data structure is used for actual data transfers, which are similar to an autorequest DMA transfer. The controller arbitrates after every primary transfer. The controller needs only one request to complete the entire transfer. This mode is used when performing multiple memory-to-memory copy tasks. The processor can configure all of the tasks simultaneously and does not need to intervene in between each task. The controller generates the corresponding DMA channel interrupt in the NVIC when the entire scatter-gather transaction completes using a basic cycle.

In this mode, the controller receives an initial request and then performs four DMA transfers using the primary data structure to program the control structure of the alternate data structure. After this transfer completes, the controller starts a DMA cycle using the alternate data structure. After the cycle completes, the controller performs another four DMA transfers using the primary data structure. The controller continues to alternate between using the primary and alternate data structures until either the processor configures the alternate data structure for a basic cycle or the DMA reads an invalid data structure.

Table 75 lists the fields of the CHNL_CFG memory location for the primary data structure, which must be programmed with constant values for the memory scatter-gather mode.

Table 75. CHNL_CFG for Primary Data Structure in Memory Scatter-Gather Mode, CHNL_CFG[2:0] = 100

Bits	Name	Description
[31:30]	DST_INC	10: configures the controller to use word increments for the address.
[29:28]	DST_SIZE	10: configures the controller to use word transfers.
[27:26]	SRC_INC	10: configures the controller to use word increments for the address.
[25:24]	SRC_SIZE	10: configures the controller to use word transfers.
[23:18]	RESERVED	Undefined. Write as 0.
[17:14]	R_POWER	0010: indicates that the DMA controller is to perform four transfers.
[13: 4]	N_MINUS_1	Configures the controller to perform N DMA transfers, where N is a multiple of 4.
3	RESERVED	Undefined. Write as 0.
[2:0]	CYCLE_CTRL	100: configures the controller to perform a memory scatter-gather DMA cycle.

Peripheral Scatter-Gather (CHNL_CFG[2:0] = 110 or 111)

In peripheral scatter-gather mode, the controller must be configured to use both the primary and alternate data structure. The controller uses the primary data structure to program the control structure of the alternate data structure. The alternate data structure is used for actual data transfers, and each transfer takes place using the alternate data structure with a basic DMA transfer. The controller does not arbitrate after every primary transfer. This mode is used when there are multiple peripheral-to-memory DMA tasks to be performed. The Cortex-M3 can configure all of the tasks simultaneously and does not need to intervene in between each task. This is very similar to memory scatter-gather mode except for arbitration and request requirements. The controller generates the corresponding DMA channel interrupt in the NVIC when the entire scatter-gather transaction completes using a basic cycle.

In peripheral scatter-gather mode, the controller receives an initial request from a peripheral and then performs four DMA transfers using the primary data structure to program the alternate control data structure. The controller then immediately starts a DMA cycle using the alternate data structure without rearbitrating.

After this cycle completes, the controller rearbitrates, and if it receives a request from the peripheral that has the highest priority, it performs another four DMA transfers using the primary data structure. It then immediately starts a DMA cycle using the alternate data structure without rearbitrating. The controller continues to alternate between using the primary and alternate data structures until either the processor configures the alternate data structure for a basic cycle or the DMA reads an invalid data structure.

Table 76 lists the fields of the CHNL_CFG memory location for the primary data structure, which must be programmed with constant values for the peripheral scatter-gather mode.

Table 76. CHNL_CFG for Primary Data Structure in Peripheral Scatter-Gather Mode, CHNL_CFG[2:0] = 110

Bits	Name	Description
[31:30]	DST_INC	10: configures the controller to use word increments for the address.
[29:28]	DST_SIZE	10: configures the controller to use word transfers.
[27:26]	SRC_INC	10: configures the controller to use word increments for the address.
[25:24]	SRC_SIZE	10: configures the controller to use word transfers.
[23:18]	RESERVED	Undefined. Write as 0.
[17:14]	R_POWER	0010: indicates that the DMA controller performed four transfers without rearbitration.
[13: 4]	N_MINUS_1	Configures the controller to perform N DMA transfers, where N is a multiple of 4.
3	RESERVED	Undefined. Write as 0.
[2:0]	CYCLE_CTRL	110: configures the controller to perform a memory scatter-gather DMA cycle.

ADDRESS CALCULATION

The DMA controller calculates the source read address based on the content of SRC_END_PTR, the source address increment setting in CHNL_CFG, and the current value of N_MINUS_1 (CHNL_CFG[13:4]).

Similarly, the destination write address is calculated based on the content of DST_END_PTR, the destination address increment setting in CHNL_CFG, and the current value of N_MINUS_1 (CHNL_CFG[13:4]).

```
Source Read Address = SRC_END_PTR – (N_MINUS_1 << (SRC_INC)) for SRC_INC = 0, 1, 2
Source Read Address = SRC_END_PTR for SRC_INC = 3
Destination Write Address = DST_END_PTR – (N_MINUS_1 << (DST_INC)) for DST_INC = 0, 1, 2
Destination Write Address = DST_END_PTR for DST_INC = 3
```

where *N_MINUS_1* is the number of configured transfers minus 1 for that channel.

ABORTING DMA TRANSFERS

It is possible to abort a DMA transfer that is in progress by writing to the bit in the DMAENCLR register corresponding to the channel that needs to be aborted. Do not set DMACFG = 0 because this can corrupt the DMA structures.

REGISTER SUMMARY: DMA

Table 77. DMA Register Summary

Address	Name	Description	Reset	RW
0x40010000	DMASTA	DMA status	0x000F0000	R
0x40010004	DMACFG	DMA configuration	0x00000000	W
0x40010008	DMAPDBPTR	DMA channel primary control data base pointer	0x00000000	RW
0x4001000C	DMAADBPTR	DMA channel alternate control data base pointer	0x00000100	R
0x40010014	DMASWREQ	DMA channel software request	0x00000000	W
0x40010020	DMARMSKSET	DMA channel request mask set	0x00000000	RW
0x40010024	DMARMSKCLR	DMA channel request mask clear	0x00000000	W
0x40010028	DMAENSET	DMA channel enable set	0x00000000	RW
0x4001002C	DMAENCLR	DMA channel enable clear	0x00000000	W
0x40010030	DMAALTSET	DMA channel primary-alternate set	0x00000000	RW
0x40010034	DMAALTCLR	DMA channel primary-alternate clear	0x00000000	W
0x40010038	DMAPRISET	DMA channel priority set	0x00000000	RW
0x4001003C	DMAPRICLR	DMA channel priority clear	0x00000000	W
0x4001004C	DMAERRCLR	DMA per channel bus error	0x00000000	RW
0x40010800	DMABSSET	DMA channel bytes swap enable set	0x00000000	RW
0x40010804	DMABSCLR	DMA channel bytes swap enable clear	0x00000000	W

REGISTER DETAILS: DMA

DMA Status Register

Address: 0x40010000, Reset: 0x000F0000, Name: DMASTA

Table 78. Bit Descriptions for DMASTA

Bits	Bit Name	Description	Reset	Access
[31:21]	RESERVED	Reserved.	0x0	R
[20:16]	CHNLSM1	Number of available DMA channels minus 1. Number of available DMA channels minus one. With 8 channels available, the register reads back 0x07.	0xF	R
[15:8]	RESERVED	Reserved. Undefined.	0x0	R
[7:4]	STATE	Current state of DMA controller. Current state of the DMA control state machine. Provides insight into the operation performed by the DMA at the time this register is read. 0000: idle 0001: reading channel controller data 0010: reading source data end pointer 0011: reading destination end pointer 0100: reading source data 0101: writing destination data 0110: waiting for DMA request to clear 0111: writing channel controller data 1000: stalled 1001: done 1010: peripheral scatter-gather transition 1011: undefined ... 1111: undefined	0x0	R
[3:1]	RESERVED	Reserved. Undefined.	0x0	R
0	MENABLE	Enable status of the controller. 0: controller is disabled 1: controller is enabled	0x0	R

DMA Configuration Register

Address: 0x40010004, Reset: 0x00000000, Name: DMACFG

Table 79. Bit Descriptions for DMACFG

Bits	Bit Name	Description	Reset	Access
[31:1]	RESERVED	Reserved. Undefined.	0x0	W
0	MENABLE	Controller enable. 0: disable controller 1: enable controller	0x0	W

DMA Channel Primary Control Data Base Pointer Register

Address: 0x40010008, Reset: 0x00000000, Name: DMAPDBPTR

The DMAPDBPTR register must be programmed to point to the primary channel control base pointer in the system memory. The amount of system memory that must be assigned to the DMA controller depends on the number of DMA channels used and whether the alternate channel control data structure is used. This register cannot be read when the DMA controller is in the reset state.

Table 80. Bit Descriptions for DMAPDBPTR

Bits	Bit Name	Description	Reset	Access
[31:0]	CTRLBASEPTR	Pointer to the base address of the primary data structure. 5 + log(2) M LSBs are reserved and must be written 0. M is the number of channels.	0x0	RW

DMA Channel Alternate Control Data Base Pointer Register

Address: 0x4001000C, Reset: 0x00000100, Name: DMAADBPTR

The DMAADBPTR read-only register returns the base address of the alternate channel control data structure. This register removes the necessity for application software to calculate the base address of the alternate data structure. This register cannot be read when the DMA controller is in the reset state.

Table 81. Bit Descriptions for DMAADBPTR

Bits	Bit Name	Description	Reset	Access
[31:0]	ALTCBPTR	Base address of the alternate data structure.	0x100	R

DMA Channel Software Request Register

Address: 0x40010014, Reset: 0x00000000, Name: DMASWREQ

The DMASWREQ register enables the generation of software DMA request. Each bit of the register represents the corresponding channel number in the DMA controller. M is the number of DMA channels

Table 82. Bit Descriptions for DMASWREQ

Bits	Bit Name	Description	Reset	Access
[31:14]	RESERVED	Reserved.	0x0	W
[13:0]	CHSWREQ	Generate software request. Set the appropriate bit to generate a software DMA request on the corresponding DMA channel. Bit 0 corresponds to DMA Channel 0, and Bit M-1 corresponds to DMA Channel M-1. When written: Bit [C] = 0, does not create a DMA request for Channel C. Bit [C] = 1, generates a DMA request for Channel C. These bits are automatically cleared by the hardware after the corresponding software request completes.	0x0	W

DMA Channel Request Mask Set Register

Address: 0x40010020, Reset: 0x00000000, Name: DMARMSKSET

Table 83. Bit Descriptions for DMARMSKSET

Bits	Bit Name	Description	Reset	Access
[31:14]	RESERVED	Reserved. Reserved, reads back 0.	0x0	R
[13:0]	CHREQMSET	Mask requests from DMA channels. This register disables DMA requests from peripherals. Each bit of the register represents the corresponding channel number in the DMA controller. Set the appropriate bit to mask the request from the corresponding DMA channel. Bit 0 corresponds to DMA Channel 0, and Bit M-1 corresponds to DMA Channel M-1. When read: Bit [C] = 0, Requests are enabled for Channel C. Bit [C] = 1, Requests are disabled for Channel C. When written: Bit [C] = 0, no effect. Use the DMARMSKCLR register to enable DMA requests. Bit [C] = 1, disables peripheral associated with Channel C from generating DMA requests.	0x0	RW

DMA Channel Request Mask Clear Register

Address: 0x40010024, Reset: 0x00000000, Name: DMARMSKCLR

Table 84. Bit Descriptions for DMARMSKCLR

Bits	Bit Name	Description	Reset	Access
[31:14]	RESERVED	Reserved.	0x0	R
[13:0]	CHREQMCLR	Clear REQ_MASK_SET bits in DMARMSKSET. This register enables DMA requests from peripherals by clearing the mask set in DMARMSKSET register. Each bit of the register represents the corresponding channel number in the DMA controller. Set the appropriate bit to clear the corresponding REQ_MASK_SET bit in DMARMSKSET register. Bit 0 corresponds to DMA Channel 0, and Bit M-1 corresponds to DMA Channel M-1. When written: Bit [C] = 0, no effect. Use the DMARMSKSET register to disable DMA requests. Bit [C] = 1, enables peripheral associated with Channel C to generate DMA requests.	0x0	W

DMA Channel Enable Set Register

Address: 0x40010028, Reset: 0x00000000, Name: DMAENSET

Table 85. Bit Descriptions for DMAENSET

Bits	Bit Name	Description	Reset	Access
[31:14]	RESERVED	Reserved.	0x0	R
[13:0]	CHENSET	Enable DMA channels. This register allows for the enabling of DMA channels. Reading the register returns the enable status of the channels. Each bit of the register represents the corresponding channel number in the DMA controller. Set the appropriate bit to enable the corresponding channel. Bit 0 corresponds to DMA Channel 0, and Bit M-1 corresponds to DMA Channel M-1. When read: Bit [C] = 0, Channel C is disabled. Bit [C] = 1, Channel C is enabled. When written: Bit [C] = 0, no effect. Use the DMAENCLR register to disable the channel. Bit [C] = 1, enables Channel C.	0x0	RW

DMA Channel Enable Clear Register

Address: 0x4001002C, Reset: 0x00000000, Name: DMAENCLR

Table 86. Bit Descriptions for DMAENCLR

Bits	Bit Name	Description	Reset	Access
[31:14]	RESERVED	Reserved. Undefined.	0x0	R
[13:0]	CHENCLR	Disable DMA channels. This register allows for the disabling of DMA channels. Reading the register returns the enable status of the channels. Each bit of the register represents the corresponding channel number in the DMA controller. Note that the controller disables a channel automatically, by setting the appropriate bit, when it completes the DMA cycle. Set the appropriate bit to disable the corresponding channel. Bit 0 corresponds to DMA Channel 0, and Bit M-1 corresponds to DMA Channel M-1. When written: Bit [C] = 0, no effect. Use the DMAENSET register to enable the channel. Bit [C] = 1, disables Channel C.	0x0	W

DMA Channel Primary-Alternate Set Register

Address: 0x40010030, Reset: 0x00000000, Name: DMAALTSET

The DMAALTSET register enables the user to configure the appropriate DMA channel to use the alternate control data structure. Reading the register returns the status of which data structure is in use for the corresponding DMA channel. Each bit of the register represents the corresponding channel number in the DMA controller.

Note that the DMA controller sets/clears these bits automatically as necessary for ping-pong, memory scatter-gather, and peripheral scatter-gather transfers.

Table 87. Bit Descriptions for DMAALTSET

Bits	Bit Name	Description	Reset	Access
[31:14]	RESERVED	Reserved. Undefined.	0x0	R
[13:0]	CHPRIALTSET	Control structure status/select alt struct. Returns the channel control data structure status, or selects the alternate data structure for the corresponding DMA channel. Bit 0 corresponds to DMA Channel 0, and Bit M-1 corresponds to DMA Channel M-1. When read: Bit [C] = 0, DMA Channel C is using the primary data structure. Bit [C] = 1, DMA Channel C is using the alternate data structure. When written: Bit [C] = 0, no effect. Use the DMAALTCLR register to set Bit [C] to 0. Bit [C] = 1, selects the alternate data structure for Channel C.	0x0	RW

DMA Channel Primary-Alternate Clear Register

Address: 0x40010034, Reset: 0x00000000, Name: DMAALTCLR

The DMAALTCLR write-only register enables the user to configure the appropriate DMA channel to use the primary control data structure. Each bit of the register represents the corresponding channel number in the DMA controller.

Note that the DMA controller sets/clears these bits automatically as necessary for ping-pong, memory scatter-gather and peripheral scatter-gather transfers.

Table 88. Bit Descriptions for DMAALTCLR

Bits	Bit Name	Description	Reset	Access
[31:14]	RESERVED	Reserved. Undefined.	0x0	R
[13:0]	CHPRIALTCLR	Select primary data struct. Set the appropriate bit to select the primary data structure for the corresponding DMA channel. Bit 0 corresponds to DMA Channel 0, and Bit M-1 corresponds to DMA Channel M-1. When written: Bit [C] = 0, no effect. Use the DMAALTSET register to select the alternate data structure. Bit [C] = 1, selects the primary data structure for Channel C.	0x0	W

DMA Channel Priority Set Register

Address: 0x40010038, Reset: 0x00000000, Name: DMAPRISET

Table 89. Bit Descriptions for DMAPRISET

Bits	Bit Name	Description	Reset	Access
[31:14]	RESERVED	Reserved. Undefined.	0x0	R
[13:0]	CHPRISET	Configure channel for high priority. This register enables the user to you to configure a DMA channel to use the high priority level. Reading the register returns the status of the channel priority mask. Each bit of the register represents the corresponding channel number in the DMA controller. Returns the channel priority mask status, or sets the channel priority to high. Bit 0 corresponds to DMA Channel 0, and Bit M-1 corresponds to DMA Channel M-1. When read: Bit [C] = 0, DMA Channel C is using the default priority level. Bit [C] = 1, DMA Channel C is using a high priority level. When written: Bit [C] = 0, no effect. Use the DMAPRICLR register to set Channel C to the default priority level. Bit [C] = 1, Channel C uses the high priority level.	0x0	RW

DMA Channel Priority Clear Register

Address: 0x4001003C, Reset: 0x00000000, Name: DMAPRICLR

Table 90. Bit Descriptions for DMAPRICLR

Bits	Bit Name	Description	Reset	Access
[31:14]	RESERVED	Reserved. Undefined.	0x0	R
[13:0]	CHPRICLR	Configure channel for default priority level. The DMAPRICLR write-only register enables the user to configure a DMA channel to use the default priority level. Each bit of the register represents the corresponding channel number in the DMA controller. Set the appropriate bit to select the default priority level for the specified DMA channel. Bit 0 corresponds to DMA Channel 0, and Bit M-1 corresponds to DMA Channel M-1. When written: Bit [C] = 0, no effect. Use the DMAPRISET register to set Channel C to the high priority level. Bit [C] = 1, Channel C uses the default priority level.	0x0	W

DMA Per Channel Bus Error Register

Address: 0x4001004C, Reset: 0x00000000, Name: DMAERRCLR

Table 91. Bit Descriptions for DMAERRCLR

Bits	Bit Name	Description	Reset	Access
[31:14]	RESERVED	Reserved. Undefined.	0x0	R
[13:0]	ERRCLR	Bus error status. This register is used to read and clear the DMA bus error status. The error status is set if the controller encountered a bus error while performing a transfer or when it reads an invalid descriptor (whose cycle control is 3'b000). If a bus error occurs or invalid cycle control is read on a channel, that channel is automatically disabled by the controller. The other channels are unaffected. Write one to clear bits. Bit 0 corresponds to DMA Channel 0, and Bit M-1 corresponds to DMA Channel M-1. When read: Bit [C] = 0: no bus error/invalid cycle control occurred. Bit [C] = 1: a bus error/invalid cycle control is pending. When written: Bit [C] = 0: no effect. Bit [C] = 1: bit is cleared.	0x0	RW1C

DMA Channel Bytes Swap Enable Set Register

Address: 0x40010800, Reset: 0x00000000, Name: DMABSSET

Table 92. Bit Descriptions for DMABSSET

Bits	Bit Name	Description	Reset	Access
[31:14]	RESERVED	Reserved. Undefined.	0x0	R
[13:0]	CHBSWAPSET	Byte swap status. This register is used to configure a DMA channel to use byte. Each bit of the register represents the corresponding channel number in the DMA controller. Bit 0 corresponds to DMA Channel 0, and Bit M-1 corresponds to DMA Channel M-1. When read: Bit [C] = 0, Channel C byte swap is disabled. Bit [C] = 1, Channel C byte swap is enabled. When written: Bit [C] = 0, no effect. Use the DMABSCLR register to disable byte swap on Channel C. Bit [C] = 1, enables byte swap on Channel C.	0x0	RW

DMA Channel Bytes Swap Enable Clear Register

Address: 0x40010804, Reset: 0x00000000, Name: DMABSCLR

Table 93. Bit Descriptions for DMABSCLR

Bits	Bit Name	Description	Reset	Access
[31:14]	RESERVED	Reserved. Undefined.	0x0	R
[13:0]	CHBSWAPCLR	Disable byte swap. The DMABSCLR write-only register enables the user to configure a DMA channel to not use byte swapping and use the default operation. Each bit of the register represents the corresponding channel number in the DMA controller. Bit 0 corresponds to DMA Channel 0, and Bit M-1 corresponds to DMA Channel M-1. When written: Bit [C] = 0, no effect. Use the DMABSSET register to enable byte swap on Channel C. Bit [C] = 1, disables byte swap on Channel C.	0x0	W

FLASH CONTROLLER

FLASH CONTROLLER FEATURES

- 256 kB Flash/EE memory in two blocks of 128 kB each (Flash 0 and Flash 1).
- 4 kB information space, which contains factory code.
- Supports flash block switching for MDIO. For MDIO specific details, see the MDIO section.

FLASH CONTROLLER OVERVIEW

- Supports read on one flash block and erase/write operation on the other block.
- Peripheral DMA support for flash keyhole-based write.
- Kernel present in information space.
- Supports buffered read, executing code from a 64-bit read while fetching the next 64 bits.
- 32-bit interface for MMR access.
- Flash program, erase timing controlled via the fixed 16 MHz reference clock.
- Keyhole open for access, command fail, command complete status bits.
- Cache provided to speed up execution.

Commands

- Write command: 64 bits per write.
- Page erase commands.
- Mass erase commands for each flash block.
- Generation of signatures for single or multiple pages.
- Command abort supported. This is possible by writing to command MMR or by a system interrupt.
- Keys required for running commands such as a mass erase and the test commands.

Protection, Integrity

- Write/read protection for user space.
- Read and write protection for information space.
- Ability to lock the SW-DP interface.
- Automatic signature check of information space on reset.
- User signature check of user space and information space.
- 8-bit ECC
- 1-bit ECC error correction
- 1-bit ECC errors and 2-bit or greater ECC errors can be configured to generate a flash ECC interrupt or a system exception

FLASH CONTROLLER OPERATION

User Space

Flash blocks (Flash 0 and Flash 1) of 128 kB each are available for user code and data. Generally, this can be considered a 256 kB block, from 0 to 0x3FFFF, except that it is not possible to execute from one flash block while erasing or writing parts of the same block.

The top 24 bytes of user space in each flash block are reserved for a signature, the user write protection pattern, and the user flash failure analysis key (USERFAAKEY).

If a user tries to read a portion of memory that is not available, a bus error is returned. If a user tries to write via keyhole access to a portion of memory that is not available, an appropriate error flag is set.

Information Space

Information space of Flash 0 and Flash 1 is located at Address 0x40000 to Address 0x40FFF and is divided up between kernel space, test space, and calibration space. Information space is reserved for use by Analog Devices. Upon a reset, the hardware forces the part to execute from the start of information space to copy calibration and configuration values to appropriate MMRs. When the kernel completes, it passes code execution to the start of user code.

The hardware automatically checks the integrity of the kernel after reset. In the event of a failure, FEESTA[13] is set and user code cannot run. This bit can only be read via a serial wire read if the serial wire interface is enabled.

The kernel code cannot be accessed by the user. A user can read 16 bytes of Flash 0 information space at Address 0x407E8 to Address 0x407F7. These locations contain ManfID0, ManfID1, and the next eight bytes, which are reserved. ManfID0 and ManfID1 contain traceability information to uniquely identify every part sold.

The top two bytes at 0x407F4 identifies the silicon version and the kernel revision. The first hexadecimal digit in the two bytes translates to the silicon revision, with 0x1 being the first silicon and each future revision incrementing by 1. The next two hexadecimal digits are the ASCII encoded version of the kernel. Prerelease versions start at Y; after release, this changes to the ASCII character 0 and increment upwards if any changes are necessary. The fourth hexadecimal digit represents the kernel minor revision. This starts at 0xE and is decremented for every minor change to the kernel. For example, 0x159A translates as follows: 1 indicates first silicon, 59 indicates kernel revision Y in ASCII code, and A indicates minor revision A.

There are also hardware registers that identify the version of each silicon die. For more information, see the Silicon Identification section.

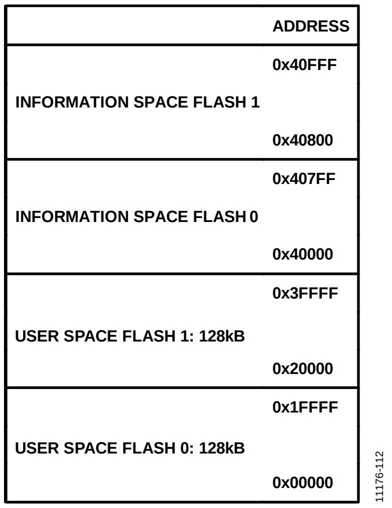

ADDRESS
0x40FFF
INFORMATION SPACE FLASH 1
0x40800
0x407FF
INFORMATION SPACE FLASH 0
0x40000
0x3FFFF
USER SPACE FLASH 1: 128kB
0x20000
0x1FFFF
USER SPACE FLASH 0: 128kB
0x00000
11176-112

Figure 14. Information and User Space Memory Map

Keys

The value 0xF123F456 must be written to the FEEKEY register to run certain user commands, to write to certain locations in flash, or to enable write access to the user setup register (FEECON1).

FLASH MEMORY OPERATION

Keyhole Access

Writing to flash is through keyhole access.

Keyhole access consists of

- Flash address
- Flash data MMR
- Command MMR

Reserved Flash Locations

The top six words of each flash block have special functionality, as listed in Figure 15 and Figure 16. Therefore, normal code or data cannot be placed in this space.

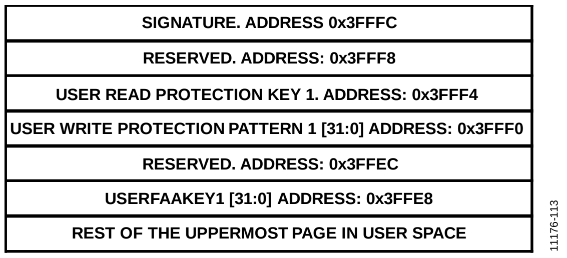

SIGNATURE. ADDRESS 0x3FFFC
RESERVED. ADDRESS: 0x3FFF8
USER READ PROTECTION KEY 1. ADDRESS: 0x3FFF4
USER WRITE PROTECTION PATTERN 1 [31:0] ADDRESS: 0x3FFF0
RESERVED. ADDRESS: 0x3FFEC
USERFAAKEY1 [31:0] ADDRESS: 0x3FFE8
REST OF THE UPPERMOST PAGE IN USER SPACE
11176-113

Figure 15. Uppermost Page in User Flash 1 Space

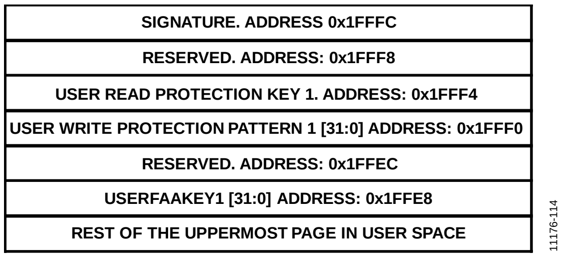

SIGNATURE. ADDRESS 0x1FFFC
RESERVED. ADDRESS: 0x1FFF8
USER READ PROTECTION KEY 1. ADDRESS: 0x1FFF4
USER WRITE PROTECTION PATTERN 1 [31:0] ADDRESS: 0x1FFF0
RESERVED. ADDRESS: 0x1FFEC
USERFAAKEY1 [31:0] ADDRESS: 0x1FFE8
REST OF THE UPPERMOST PAGE IN USER SPACE
11176-114

Figure 16. Uppermost Page in User Flash 0 Space

Writing to Flash

Each write programs 64 bits of data.

To write to a flash location, the following sequence is required:

1. Write the address of the flash location to FEEFLADR.
2. Write the 64 bits of data to FEEFLDATA0 and FEEFLDATA1.
3. Write the write command to FEECMD.

After the write command is given, the controller writes to flash. CMDDONE (FEESTA[2]) indicates that the command is completed.

In addition, note that a 64-bit location can be written to only once, unless it is erased again because a second write to the same 64-bit location will corrupt the 8-bit ECC. With the ECC disabled, it is possible to write to a 64-bit location multiple times as long as subsequent writes clear more bits to 0 than the previously ones because a page erase or mass erase is needed to change bits from 0 to 1. It is only possible to write to a 64-bit twice without doing an erase and remain within ADuCM320 data sheet specifications. If the flash is written to more than twice without an erase, there is the risk of corrupting data.

Do not write to flash from within interrupts because interrupts can occur at any point during normal program execution and an existing write operation to flash could be interrupted and the flash control registers could be corrupted resulting in unexpected behavior. If an interrupt function needs to store code to flash, the data must be stored in a queue in SRAM, and the write should be performed after the interrupt exits. An alternative to this is to implement a mutex to ensure that two or more sections of code cannot attempt to write to flash before all previous flash operations are completed.

Customers must have a single function that performs all the writes to flash needed by the customer application. It must only be possible to call this function once until the function has exited. This function must be protected so that it is not possible under a fault condition for the CPU to jump to the code and to execute random writes. One of the techniques recommended for implementing such protection is to include code before and after the function that traps the CPU there or that jumps to a fault handler.

The flash can only be programmed and erased a number of times specified in the ADuCM320 data sheet before data will corrupt. If customers need to perform more write operations than what is specified in the ADuCM320 data sheet, store the data using a wear levelling scheme where the data is stored across various locations in flash over multiple pages. For example, if 64 bits of data need to be stored and four flash pages are used, the data can be updated 128 times, and the four flash pages will only be subjected to a single program/erase cycle. When implementing this type of wear levelling scheme, it is necessary to have a dedicated location where pointers to the valid data is stored, where the next data is to be programmed, and a counter of the total number of writes to ensure that the flash is not over cycled.

Erasing Flash

User code can call three flash erase commands:

- MASSERASE0: This command erases the entire user Flash 0 memory. After entering the user protection key into FEEKEY, write the MASSERASE0 command to FEECMD.
- MASSERASE1: This command erases the entire user Flash 1 memory. After entering the user protection key into FEEKEY, write the MASSERASE1 command to FEECMD.
- PAGEERASE: This command erases 2 kB of flash. The page is selected by FEEADR0. After entering the user protection key into FEEKEY, load FEEADR0 with the page address to be erased. Finally, write the page erase command to FEECMD. CMDDONE (FEESTA[2]) indicates that the command is completed.

During a page or mass erase sequence, the flash controller and flash block consume extra current for the duration of the flash erase sequence.

Signature

The signature is used to check the integrity of the flash device. The signature is calculated from the lowest 32-bit word to the second highest 32-bit word in the selected block. The signature is a 24-bit CRC with an initial value of 0xFFFFFF and the following polynomial:

$x^{24} + x^{23} + x^6 + x^5 + x + 1$

The data is pushed into the CRC polynomial until the specified end address is reached. A block can be a single page or multiple pages. The hardware assumes that the signature for a block is stored in the upper four bytes of the most significant page of a block; therefore, these 32 bits are not included when generating the signature. While the signature is being computed for a particular flash, all other accesses to the same flash are stalled.

Note that FEEADR0/FEEADR1 addresses are byte addresses but only pages need to be identified because the lower 11 bits are ignored by the hardware. Also, ensure that the addresses written to FEEADR0 or FEEADR1 are both either in Flash 0 or Flash 1.

The following code illustrates how the CRC is calculated and how to compare it to the result of the SIGN command.

```
int FeeCrc(int iLen,int *aiData)
   {
   int i1,i2,iCrc;

   iCrc = 0xffffffff;            //Seed value.
   for(i1=0; i1<iLen; i1++)      //Starting at lowest address.
      {
      for(i2=31; i2>=0; i2--)    //MSB first.
         {
         iCrc <<= 1;             //Left shift.
         if((*(aiData+i1))&(1<<i2)) iCrc ^= 0x00800063;  //^= Polynomial.
         if(iCrc&(1<<24))           iCrc ^= 0x00800063;
         }
      }
   return(iCrc&0x00ffffff);      //Return 24 bits.
   }

int FeeSign(unsigned long ulStartAddr, unsigned long ulEndAddr)
   {
   if((pADI_FEE->FEESTA&1)!=0) return 0;
   pADI_FEE->FEEADR0 = ulStartAddr;
   pADI_FEE->FEEADR1 = ulEndAddr;
   pADI_FEE->FEEKEY =  0xF123F456;
   pADI_FEE->FEECMD = 0x2;
   return 1;
   }

 FeeSign(0x00800,0x00900);       //SIGN for page1.
   if(FeeCrc(511,(int *)0x00800) != pADI_FEE->FEESIG)
      FlagError();
   Else FlagSuccess();
```

ECC Error Handling

During the signature check, the error checking and correcting (ECC) is checked on each 72-bit flash read (64-bit flash read and 8-bit ECC). If errors are corrected by the ECC, the ERRDETECTED flag in the status register, FEESTA, is set after the signature check is completed. If errors are detected and cannot be corrected by ECC, the ERRDETECTED flag in FEESTA is set. A signature check is treated as a failure when the computed signature is not equal to the stored signature.

During a read of the flash, if there is a 1-bit error, the error is corrected by default; however, neither ECC interrupts nor system exceptions are enabled. If interrupts or system exceptions are not enabled by the user, the appropriate flags in FEESTA are not set in the event of an ECC error.

A 1-bit ECC interrupt or system exception can be enabled in the ECC enable/disable register (FEEECCCONFIG), if required. If the appropriate interrupts or system exceptions are enabled in the FEEECCCONFIG register, the appropriate flags are set in the status register.

If there is a 2-bit ECC error and if interrupts or system exceptions are enabled in the FEEECCCONFIG register, an error is issued by the controller. If the appropriate interrupts or system exceptions are enabled in the FEEECCCONFIG register, the appropriate flags are set in the status register.

An ECC error is signaled by the ECC error detection/correction hardware when a flash location is read. Depending on from which flash (Flash 0 or Flash 1) the read happens, the appropriate flags are set in the status register (ECCREADERRFLSH0, ECCREADERRFLSH1, and so on). Note that 1-bit errors corrected meet full data sheet specification.

If a system exception is enabled, the device vectors to a hard fault or bus fault in the event of an ECC error. See the SHCSR register in the ARM Cortex-M3 processor documentation to enable a bus fault.

ECC Error During Read

Two separate ECCREADERR flags are present in the status register: FEESTA[10:9] and FEESTA[12:11] for Flash 0 And Flash 1. If the interrupt is configured to be generated when an ECC error occurs, the address at which the error is detected is available for the user. If a system exception is configured, the BFAR register contains the address for which ECC error is detected.

ECC Error During Execution of Sign Command

If there is an ECC error during signature check, registers are not updated. After the command is complete, ECCERRCMD flags in FEESTA[8:7] are updated. No interrupt or system exception is generated.

FLASH PROTECTION

There are three types of protection implemented:

- Key protection
- Read protection
- Write protection

Flash Protection: Key Protection

Some of the flash controller MMRs are key-protected to avoid accidental writes to these MMRs.

The user key is 0xF123F456. This key must be entered to run certain user commands, to write to certain locations in flash, or to enable write access to FEECON1. Once entered, the key remains asserted unless a command is written to FEECMD. When the command starts, the key clears automatically. If this key is entered to enable write access to FEECON1 or to enable writes to certain locations in flash, it needs to be cleared by user code afterwards. To clear the key, write any value other than 0xF123F456 to FEEKEY.

Flash Protection: User Read Protection

User space read protection is provided by disabling serial wire access. A user can disable serial wire debug access by writing 0 to Bit 0 of FEECON1. Serial wire debug access is disabled while the kernel is running; otherwise, serial wire debug access may prevent the kernel from running to completion. When the kernel exits to user code, it enables serial wire access unless either of the keys at 0x3FFF4 or 0x1FFF4 is set to 0x0000003A. This means that the part is always read protected after either key is in place and that no debug access can occur.

Flash Protection: User Write Protection

User write protection is provided to prevent accidental writes to pages in user space and to protect blocks of user code when downloading extra code to flash. If a write or erase of a protected location is detected, the flash controller generates an interrupt if the command error/complete interrupt are enabled. The write protection for each block is stored near the top of each block. The top four bytes are for a signature, and the next eight bytes are reserved. The next 32-bit flash location contains the protection pattern, which is copied to FEEPRO0 and FEEPRO1 at startup, with each bit protecting a block of 4 kB of flash. If no protection is specified, protection can be set by writing to FEEPRO0 and FEEPRO1.

Flash Failure Analysis Key

It may be necessary to perform failure analysis on parts that are returned by a user even though read protection is enabled. A method has been provided to allow failure analysis of protected memory by a user flash failure analysis key (USERFAAKEY.).

The user must set the key as two 32-bit values near the top of each user flash block. Supplying this key to Analog Devices allows access to user code for debug purposes. See Figure 15 and Figure 16 for details.

Flash Controller Abort

Commands (erase, sign, or mass verify) and writes can be aborted upon receipt of an interrupt, as listed in Table 54. Aborts are also possible by writing an abort command to the FEECMD register. However, if flash is being programmed and the routine controlling the programming is in flash, it is not possible to use the abort command to abort the cycle because instructions cannot be read. Therefore, the ability to abort a cycle on the assertion of any system interrupt is provided. The FEEAENx register is used to enable aborts upon receipt of an interrupt. Each bit in the FEEAENx registers corresponds to an interrupt listed in Table 54. Setting a bit in the FEEAENx register enables the corresponding interrupt to abort flash operations.

When a command or write is aborted via a system interrupt, FEESTA[5:4] indicates an abort (FEESTA[5:4] = 11).

Depending on the state that a write cycle is in when the abort asserts, the write cycle may or may not complete. If the write or erase cycle did not complete successfully, a fail status of aborted can be read in the status register.

If an immediate response to an interrupt is required during an erase or program cycle, the interrupt service routine and the interrupt vector table must be moved to SRAM or must be in the other flash block for the duration of the cycle.

If the DMA engine is set up to write a block of data to flash, an interrupt can be set up to abort the current write; however, the DMA engine starts the next write immediately. The interrupt causing the abort stays asserted so that there is a number of aborted write cycles in this case before the processor gains access to flash.

When an abort is triggered by an interrupt, all commands are repeatedly aborted until the appropriate FEEAENx bit is cleared or the interrupt source is cleared.

CPU Execution Speed

The basic execution speed of the ADuCM320 is one CPU cycle per clock cycle. The default clock speed is 80 MHz, which is achieved when running from cache, but is slightly less when running directly from flash. An average execution speed of over 70 MHz is typically achieved for typical C code. For more details and how to achieve full speed operation for critical code, see the AN-1322 Application Note, *ADuCM320 Code Execution Speed*.

Memory Cache

A memory cache is provided on chip to speed up program execution. The instruction cache is configured and set up by default. If the user writes code to the flash, the user should perform a chip reset to ensure that the old cached data is cleared and that the new code can be executed. If a chip reset is not possible, the following code can be used to clear the cache. iCache should be 0x10001 or (CACHESETUP_IINIT_EN|CACHESETUP_DINIT_EN) to clear both the instruction and data cache.

```
int FeeCacheClr(int iCache)
   {
   unsigned int   ui1;

   ui1 = pADI_FEE->CACHESETUP;
   pADI_FEE->CACHEKEY = 0xF123F456;
   pADI_FEE->CACHESETUP = ui1|(iCache&(CACHESETUP_IINIT_EN|CACHESETUP_DINIT_EN));
   while(pADI_FEE->CACHESTAT&(iCache&(CACHESETUP_IINIT_EN|CACHESETUP_DINIT_EN)));
   pADI_FEE->CACHEKEY = 0xF123F456;
   pADI_FEE->CACHESETUP = ui1;
   return 1;
   }
```

Most programming tools clear the cache before downloading code to a device.

Flash DMA Support

Flash controller operations can be supported by DMA. This feature is software configurable. The two flash blocks are independent, meaning that the user can continue executing from one block while programming another block. The DMA is very useful for this because the core only needs to initiate the write to flash, and then the DMA finishes executing the code in the background, triggering an interrupt when the operation is complete. The following code can be used for writing to flash using the DMA:

```
void FLASHDMAINIT(void)
{
pADI_DMA->DMACFG = 0x1;                        // Enable DMA mode in DMA controller
Dma_Init();
NVIC_EnableIRQ(DMA_FLASH_IRQn);                // Enable Flash DMA IRQ
FLASHDMAWRITE(uxFlashData, 64);
pADI_DMA->DMAENSET = 0x2000;
pADI_FEE->FEEFLADR = uiAdr;
pADI_FEE->FEEKEY = 0xF123F456;
pADI_FEE->FEECON1 |= (FEECON1_KHDMA_EN);       // Enable Flash DMA mode
}

void FLASHDMAWRITE (unsigned char * pucTX_DMA, unsigned int iNumVals)
{
DmaDesc Desc;
// Common configuration of all the descriptors used here
Desc.ctrlCfg.Bits.cycle_ctrl = DMA_BASIC;
desc.ctrlcfg.bits.next_useburst = 0x0;
desc.ctrlcfg.bits.r_power = 1;
desc.ctrlcfg.bits.src_prot_ctrl = 0x0;
Desc.ctrlCfg.Bits.dst_prot_ctrl = 0x0;
Desc.ctrlCfg.Bits.src_size = DMA_SIZE_WORD;
Desc.ctrlCfg.Bits.dst_size = DMA_SIZE_WORD;
// TX Primary Descriptor
Desc.srcEndPtr = (unsigned int)(pucTX_DMA+ 4*(iNumVals - 0x1));
Desc.destEndPtr = (unsigned int)&(pADI_FEE->FEEFLDATA1);
Desc.ctrlCfg.Bits.n_minus_1 = iNumVals - 0x1;
Desc.ctrlCfg.Bits.src_inc = DMA_SRCINC_WORD;
Desc.ctrlCfg.Bits.dst_inc = DMA_DSTINC_NO;
*Dma_GetDescriptor(Flash_C) = Desc;
}

void DMA_Flsh_Int_Handler()
{
 pADI_FEE->FEEKEY = 0xF123F456;
 pADI_FEE->FEECON1 &= (~FEECON1_KHDMA_EN);     // Disable Flash DMA mode
 dma_done = 1;
}
```

Flash Controller Performance and Command Duration

All flash functions are slower than the CPU execution speed. The CPU Execution Speed section details the slight penalty of slower flash reads. All other flash operations are significantly slower, as detailed in Table 94.

Table 94. Typical Flash Execution Times

Operation	Time (Typical)	Comment
Write 64-bit location	75 µs	
Mass erase one flash block	18 ms	
Page erase one page	18 ms	
Sign Flash 0/Flash 1 information space	33 µs	512 cycles, 2 kB
Sign Flash 0/Flash 1 user space	2.1 ms	32k cycles, 128 kB

In general, these timings should be used as a guideline only, and software should use the flash status information or the interrupt system to detect when flash operations are complete. If one of the operations in Table 94 is executed in the same block as the block from which the CPU fetches instructions, the CPU stalls until the operation is complete.

REGISTER SUMMARY: FLASH CONTROLLER

Table 95. Flash Controller Register Summary

Address	Name	Description	Reset	RW
0x40018000	FEESTA	Status register	0x00000000	R
0x40018004	FEECON0	Command control register: interrupt enable register	0x00000000	RW
0x40018008	FEECMD	Command register	0x00000000	RW
0x4001800C	FEEFLADR	Flash address keyhole register	0x00000000	RW
0x40018010	FEEFLDATA0	Flash data register: keyhole interface lower 32 bits	0x00000000	RW
0x40018014	FEEFLDATA1	Flash data register: keyhole interface upper 32 bits	0x00000000	RW
0x40018018	FEEADR0	Lower page address register	0x00000000	RW
0x4001801C	FEEADR1	Upper page address register	0x00000000	RW
0x40018020	FEEKEY	Key register	0x00000000	W
0x40018028	FEEPRO0	Write protection register for Flash 0	0xFFFFFFFF	RW
0x4001802C	FEEPRO1	Write protection register for Flash 1	0xFFFFFFFF	RW
0x40018034	FEESIG	Upper half-word of signature	0x0000000X	R
0x40018038	FEECON1	User setup register	0x0000001X	RW
0x40018040	FEEWRADDRA	Write abort address register	0x0000000X	R
0x40018048	FEEAEN0	Interrupt abort enable register—Interrupt 31 to Interrupt 0	0x00000000	RW
0x4001804C	FEEAEN1	Interrupt abort enable register—Interrupt 54 to Interrupt 32	0x000000	RW
0x40018064	FEEECCCONFIG	ECC enable/disable, error response	0x00000000	RW
0x40018074	FEEECCADDR0	Flash 0 ECC error address	0x00000000	R
0x40018078	FEEECCADDR1	Flash 1 ECC error address	0x00000000	R
0x400180C0	CACHESTAT	Cache status register	0x2	R
0x400180C4	CACHESETUP	Cache setup register	0x2	RW
0x400180C8	CACHEKEY	Cache key register	0x0	W

REGISTER DETAILS: FLASH CONTROLLER

Status Register

Address: 0x40018000, Reset: 0x00000000, Name: FEESTA

Table 96. Bit Descriptions for FEESTA

Bits	Bit Name	Description	Reset	Access
[31:29]	RESERVED	Reserved.	0x0	R
[28:27]	ECCREADERRIBUS	Instruction bus ECC error during a read of flash if a system exception is enabled. **Bits / Name / Description** 00 NOERR: No error. Successful read from Flash 1. 01 ERRDETECTED: 2-bit error detected in one or more flash locations during a read from Flash 1. The errors are not corrected. 10 ERRCORRECTED: 1-bit error detected for one flash location while during read from Flash 1. The error is corrected. 11 ERR1BIT_2Bit: During the read, 1-bit error and 2-bit errors are detected in Flash 1.	0x0	RC
[26:25]	ECCREADERRDBUS	Data bus ECC error during a read of flash if a system exception is enabled. **Bits / Name / Description** 00 NOERR: No error. Successful read from Flash 1. 01 ERRDETECTED: 2-bit error detected in one or more flash locations during a read from Flash 1. The errors are not corrected. 10 ERRCORRECTED: 1-bit error detected for one flash location while during read from Flash 1. The error is corrected. 11 ERR1BIT_2Bit: During the read, 1-bit error and 2-bit errors are detected in Flash 1.	0x0	RC
[24:22]	ECCCOUNTFLASH1	This is a 3-bit counter that reflects the number of 1-bit ECC read errors in Flash 1 after FEESTA[12:11] = 0x2 and before FEESTA is read. This counter does not count on ECC 2-bit errors. The counter is cleared when FEESTA is read by the user.	0x0	R
[21:20]	RESERVED	Reserved.	0x0	R

Bits	Bit Name	Description	Reset	Access
[19:17]	ECCCOUNTFLASH 0	This is a 3-bit counter that reflects the number of 1-bit ECC read errors in Flash 0 after FEESTA[10:9] = 0x2 and before FEESTA is read. This counter does not count on ECC 2 bit errors. The counter is cleared when FEESTA is read by the user.	0x0	RC
[16:15]	ECCERRSIGN	ECC error during initial signature check.	0x0	R
		Bits | **Name** | **Description**		
		00 | NOERR | No error. Successful flash read operation during initial signature check or page signature check.		
		01 | ERRDETECTED | During initial signature check, 2-bit errors are detected, and not corrected for at least one flash location.		
		10 | ERRCORRECTED | 1-bit error is corrected for one flash location during a signature command.		
		11 | ERR1BIT_2Bit | During the initial signature command, 1-bit errors and 2-bit errors are detected on one or more flash locations.		
13	SIGNERR	Information space signature check on reset error. After a reset, the flash controller automatically checks the Info space signature. If the signature check fails this bit is asserted. User can check if this bit is set via serial wire only. User code does not execute if this bit is set. The bit is cleared if the correct signature is programmed to the MS long word in information space.	0x0	R
[12:11]	ECCREADERRFLSH1	ECC errors during a read of Flash 1 if interrupt is enabled	0x0	RC
		Bits | **Name** | **Description**		
		00 | NOERR | No error. Successful read from Flash 1.		
		01 | ERRDETECTED | 2-bit error detected in one or more flash locations during a read from Flash 1. The errors are not corrected.		
		10 | ERRCORRECTED | 1-bit error detected for one flash location while during read from Flash 1. The error is corrected.		
		11 | ERR1BIT_2Bit | During the read, 1-bit error and 2-bit errors are detected in Flash 1.		
[10:9]	ECCREADERRFLSH0	ECC errors during read of Flash 0 if interrupt is enabled	0x0	RC
		Bits | **Name** | **Description**		
		00 | NOERR | No error. Successful read from Flash 0.		
		01 | ERRDETECTED | 2-bit error detected in one or more flash locations during a read from Flash 0. The errors are not corrected.		
		10 | ERRCORRECTED | 1-bit error detected for one flash location while during read from Flash 0. The error is corrected.		
		11 | ERR1BIT_2Bit | During the read, 1-bit error and 2-bit errors are detected in Flash 0.		
[8:7]	ECCERRCMD	ECC errors during signature commands	0x0	RC
		Bits | **Name** | **Description**		
		00 | NOERR | No error. Successful flash read operation during the signature check.		
		01 | ERRDETECTED | 2-bit error detected in one or more flash locations during the signature command. The errors are not corrected.		
		10 | ERRCORRECTED | 1-bit error detected for one flash location while doing a signature check. The error is corrected.		
		11 | ERR1BIT_2Bit | During the signature command, 1-bit error and 2-bit errors are detected on one or more flash locations.		
6	RESERVED	Reserved.	0x0	R
[5:4]	CMDRES	These two bits indicate the status of a command on completion or the status of a write. If multiple commands are executed or there are multiple writes without a read of the status register, the first error encountered is stored. Cleared to 0 when read.	0x0	RC
		00: successful completion of a command or a write		
		01: attempted signcheck, write, or erase of a protected location		
		10: read verify error. After an erase, the controller reads the corresponding word(s) to verify that the transaction completed successfully. If data read is not all Fs, this is the resulting status. If the sign command is executed and the resulting signature does not match the data in the upper 4 bytes of the upper page in a block, this is the resulting status.		
		11: indicates that a command or a write was aborted by an abort command or a system interrupt has caused an abort		

Bits	Bit Name	Description	Reset	Access
3	WRALMOSTDONE	Write almost complete. Keyhole registers open for access. This bit flags the earliest point at which the flash controller data and address may be updated for the next command without affecting an active flash command operation. 0: cleared to 0 when read 1: set to 1 when a write completes	0x0	RC
2	CMDDONE	This bit asserts when a command completes. If there are multiple commands, this status bit asserts after the first command completes and stays asserted until read. 0: cleared to 0 when read 1: set to 1 when a command completes	0x0	RC
1	WRCLOSE	This bit is asserted when the user has written all keyhole registers for flash write and the controller has started the write. If this bit is high, all keyhole registers (FEEFLADR, FEEFLDATA0, FEEFLDATA1) except the command register (FEECMD) are closed for write.	0x0	R
0	CMDBUSY	Command busy. This bit is asserted when the flash block is executing any command entered via the command register.	0x0	R

Command Control Register: Interrupt Enable Register

Address: 0x40018004, Reset: 0x00000000, Name: FEECON0

Table 97. Bit Descriptions for FEECON0

Bits	Bit Name	Description	Reset	Access
[31:3]	RESERVED	Returns 0 when read.	0x0	R
2	IENERR	Command fail interrupt enable. If this bit is set, an interrupt is generated when a command or flash write completes with an error status. 0: disable 1: enable	0x0	RW
1	IWRALCOMP	Write almost complete interrupt enable. Returns 0 when read. 0: disable 1: enable	0x0	RW
0	IENCMD	Command complete interrupt enable. When set, an interrupt is generated when a command or flash write completes. 0: disable 1: enable	0x0	RW

Command Register

Address: 0x40018008, Reset: 0x00000000, Name: FEECMD

Table 98. Bit Descriptions for FEECMD

Bits	Bit Name	Description	Reset	Access
[31:5]	RESERVED	Returns 0x0. Always returns 0 when read.	0x0	RW
[4:0]	CMD	00000: IDLE. No command executed. 00001: PAGEERASE. Write the address of the page to be erased to the FEEADR0 register, then write this code to the FEECMD, and the flash erases the page. When the erase has completed the flash reads every location in the page to verify all words in the page are erased. If there is a read verify error, it is indicated in the status register. To erase multiple pages wait until a previous page erase has completed, check the status, then issue a command to start the next page erase. Before entering this command, 0xF123F456 must be written to the FEEKEY register. 00010: SIGN. Use this command to generate a signature for a block of data. The signature is generated on a page by page basis. To generate a signature the address of the first page of the block is entered in the FEEADR0 register, the address of the last page is written to the FEEADR1 register, then write this code to the FEECMD register. When the command has completed the signature is available for reading in the sign register. The last 4 bytes of the last page in a block is reserved for storing the signature. Before entering this command 0xF123F456 must be written to the FEEKEY register.	0x0	RW

Bits	Bit Name	Description	Reset	Access
		00100: WRITE. Use this command to write to flash locations. This command needs a user key for writing into write protection location and user flash failure analysis key (USERFAAKEY) location. No key is required for other flash locations. This command takes the address and data from the FEEADR and FEEFLDATA keyhole registers. 00101: MASSERASE0. Erase all of Flash 0 user space. To enable this operation 0xF123F456 must be written to the FEEKEY register (this is to prevent accidental erases). When the mass erase has completed, the controller reads every location to verify that all locations are 0xFFFFFFFFFFFFFFFF. If there is a read verify error, it is indicated in the status register. 00110: MASSERASE1. Erase all of Flash 1 user space. To enable this operation 0xF123F456 must be written to the FEEKEY register (this is to prevent accidental erases). When the mass erase has completed the controller reads every location to verify that all locations are 0xFFFFFFFFFFFFFFFF. If there is a read verify error, it is indicated in the status register. 01000: ABORT. If this command is issued, any command currently in progress is stopped. The status indicates command completed with an error status (FEESTA[5:4] = 0x3). Note that this is the only command that can be issued while another command is already in progress. This command can also be used to stop a write that may be in progress. If a write or erase is aborted, the flash timing is violated and it is not possible to determine if the write or erase completed successfully. To enable this operation, 0xF123F456 must be written to the FEEKEY register (this is to prevent accidental aborts). All other combinations are reserved.		

Flash Address Keyhole Register

Address: 0x4001800C, Reset: 0x00000000, Name: FEEFLADR

Table 99. Bit Descriptions for FEEFLADR

Bits	Bit Name	Description	Reset	Access
[31:19]	RESERVED	Returns 0x0 if read.	0x0	R
[18:3]	FLADDR	Memory mapped address for the flash location. Used to specify flash address for write command. LSB 3 bits always reads zero.	0x0	RW
[2:0]	RESERVED	Returns 0x0 if read.	0x0	R

Flash Data Register: Keyhole Interface Lower 32 Bits

Address: 0x40018010, Reset: 0x00000000, Name: FEEFLDATA0

Table 100. Bit Descriptions for FEEFLDATA0

Bits	Bit Name	Description	Reset	Access
[31:0]	FLDATA0	FLDATA0 forms the lower 32 bits of 64 bit data to be written to flash.	0x0	RW

Flash Data Register: Key-Hole Interface Upper 32 Bits

Address: 0x40018014, Reset: 0x00000000, Name: FEEFLDATA1

Table 101. Bit Descriptions for FEEFLDATA1

Bits	Bit Name	Description	Reset	Access
[31:0]	FLDATA1	FLDATA1 forms the upper 32 bits of 64 bit data to be written to flash.	0x0	RW

Lower Page Address Register

Address: 0x40018018, Reset: 0x00000000, Name: FEEADR0

Table 102. Bit Descriptions for FEEADR0

Bits	Bit Name	Description	Reset	Access
[31:19]	RESERVED	Return 0 when read.	0x0	RW
[18:11]	PAGEADDR0	Used by SIGN and PAGEERASE commands for specifying page address. See the description of these commands in FEECMD.	0x0	RW
[10:0]	RESERVED	Reserved.	0x0	R

Upper Page Address Register

Address: 0x4001801C, Reset: 0x00000000, Name: FEEADR1

Table 103. Bit Descriptions for FEEADR1

Bits	Bit Name	Description	Reset	Access
[31:19]	RESERVED	Return 0 when read.	0x0	RW
[18:11]	PAGEADDR1	Used by SIGN command for specifying the endpage address. See the description of this command in FEECMD.	0x0	RW
[10:0]	RESERVED	Reserved.	0x0	R

Key Register

Address: 0x40018020, Reset: 0x00000000, Name: FEEKEY

Table 104. Bit Descriptions for FEEKEY

Bits	Bit Name	Description	Reset	Access
[31:0]	KEY	Enter 0xF123F456 to allow key protected operations. Returns 0x00 if read.	0x0	W

Write Protection Register for Flash 0

Address: 0x40018028, Reset: 0xFFFFFFFF, Name: FEEPRO0

Table 105. Bit Descriptions for FEEPRO0

Bits	Bit Name	Description	Reset	Access
[31:0]	WRPROT0	Write protection for Flash 0 – 32 bits. Each bit corresponds to a 4 kB flash section. Writing 0 to a bit protects the corresponding section of flash. This register is read-only if the write protection in flash has been programmed.	0xFFFFFFFF	RW

Write Protection Register for Flash 1

Address: 0x4001802C, Reset: 0xFFFFFFFF, Name: FEEPRO1

Table 106. Bit Descriptions for FEEPRO1

Bits	Bit Name	Description	Reset	Access
[31:0]	WRPROT1	Write protection for Flash 1 – 32 bits. Each bit corresponds to a 4 kB flash section. Writing 0 to a bit protects the corresponding section of flash. This register is read-only if the write protection in flash has been programmed.	0xFFFFFFFF	RW

Upper Half-Word of Signature Register

Address: 0x40018034, Reset: 0x0000000X, Name: FEESIG

Table 107. Bit Descriptions for FEESIG

Bits	Bit Name	Description	Reset	Access
[31:24]	RESERVED	Returns 0x0 if read.	0x0	R
[23:0]	SIGN	24-bit signature.	0xx	R

User Setup Register

Address: 0x40018038, Reset: 0x0000001X, Name: FEECON1

This register is key-protected, so the key (0xF123F456) must be entered in FEEKEY. After writing to FEECON1, a value other than 0xF123F456 must be written again to FEEKEY to reassert the key protection.

Table 108. Bit Descriptions for FEECON1

Bits	Bit Name	Description	Reset	Access
[31:5]	RESERVED	Returns 0 when read.	0x0	R
4	MDIO	MDIO mode. This bit is for read-only purpose. If this bit is set, MDIO address swapping can be enabled.	0x1	R
3	SWAP	Swap program code for MDIO mode. 0: disable address swap for Userspace Flash 0 and Flash 1. 1: enable address swap for Userspace Flash 0 and Flash 1.	0xX	RW
2	INCR	Auto increment FEEFLAADR for non-DMA operation. 0: disable auto address increment 1: enable auto address increment	0x0	RW
1	KHDMA	Keyhole DMA enable. 0: disable DMA mode 1: enable DMA mode	0x0	RW
0	DBG	JTAG debug enable. If this bit is 1, access via the serial wire debug interface is enabled. If this bit is 0, access via the serial wire debug interface is disabled. The kernel set this bit to 1 when it has finished executing, thus enabling debug access to a user. 0: disable JTAG access 1: enable JTAG access	0xX	RW

Write Abort Address Register

Address: 0x40018040, Reset: 0x0000000X, Name: FEEWRADDRA

Table 109. Bit Descriptions for FEEWRADDRA

Bits	Bit Name	Description	Reset	Access
[31:0]	WRABORTADDR	If a write is aborted, this register contains the address of the location being written when the write was aborted. This register has appropriate value if command abort happened. This has to be read after the command is aborted, and before any other command is given. After reset, the value is random.	0xx	R

Interrupt Abort Enable Register—Interrupt 31 to Interrupt 0

Address: 0x40018048, Reset: 0x00000000, Name: FEEAEN0

Table 110. Bit Descriptions for FEEAEN0

Bits	Bit Name	Description	Reset	Access
[31:0]	SYSIRQABORTEN	Lower 32 bits of system interrupt abort enable. To allow a system interrupt to abort a command (write, erase, sign or mass verify), write a 1 to the appropriate bit in this register. Each bit corresponds to 1 interrupt listed in the interrupt vector table.	0x0	RW

Interrupt Abort Enable Register—Interrupt 54 to Interrupt 32

Address: 0x4001804C, Reset: 0x000000, Name: FEEAEN1

Table 111. Bit Descriptions for FEEAEN1

Bits	Bit Name	Description	Reset	Access
[22:0]	SYSIRQABORTEN	Upper 23 bits of system interrupt abort enable. To allow a system interrupt to abort a command (write, erase, sign or mass verify) then write a 1 to the appropriate bit in this register. Each bit corresponds to 1 interrupt listed in the interrupt vector table.	0x0	RW

ECC Enable/Disable, Error Response Register

Address: 0x40018064, Reset: 0x00000000, Name: FEEECCCONFIG

This register is key-protected, so the key (0x5ECCACCE) must be entered in FEEKEY. After writing to FEECCCONFIG the key is cleared.

Table 112. Bit Descriptions for FEEECCCONFIG

Bits	Bit Name	Description		Reset	Access
[31:5]	RESERVED	Reserved.		0x0	R
[4:3]	ECCCMDINTEN	Interrupt enabled (Flash Interrupt) when an ECC error happens during a read.		0x0	RW
		Bits	**Description**		
		00	Interrupt is not generated if an ECC error occurs while reading from flash.		
		01	Interrupt enable d only if a 2-bit error is detected during a read from Flash 0 or Flash 1.		
		10	Interrupt enable d only if a 1-bit error is detected during a read from Flash 0 or Flash 1.		
		11	Interrupt enable d if either a 2-bit error or 1-bit error is detected during a read from Flash 0 or Flash 1.		
[2:1]	ECCCMDAHBEN	Generates a system exception (Bus Fault) when an ECC error happens during a read.		0x0	RW
		Bits	**Description**		
		00	Exception is not generated if an ECC error occurs while reading from flash.		
		01	Exception enable d only if a 2-bit error is detected during a read from Flash 0 or Flash 1.		
		10	Exception enable d only if a 1-bit error is detected during a read from Flash 0 or Flash 1.		
		11	Exception enable d if either a 2-bit error or 1-bit error is detected during a read from Flash 0 or Flash 1.		
0	ECCDISABLE	Setting this bit to 1 disables ECC. When ECC is disabled, ECC module is bypassed. When a read to a flash location is carried out, corresponding to the requested address, LSB 32-bit or MSB 32-bit raw data is returned to the bus.		0x0	RW

Flash 0 ECC Error Address Register

Address: 0x40018074, Reset: 0x00000000, Name: FEEECCADDR0

Table 113. Bit Descriptions for FEEECCADDR0

Bits	Bit Name	Description	Reset	Access
[31:19]	RESERVED	Reserved.	0x0	R
[18:0]	VALUE	This register has the address of flash0 for which ECC error is detected.	0x0	R

Flash 1 ECC Error Address Register

Address: 0x40018078, Reset: 0x00000000, Name: FEEECCADDR1

Table 114. Bit Descriptions for FEEECCADDR1

Bits	Bit Name	Description	Reset	Access
[31:19]	RESERVED	Reserved.	0x0	R
[18:0]	VALUE	This register has the address of flash0 for which ECC error is detected.	0x0	R

Cache Status Register

Address: 0x400180C0, Reset: 0x00000002, Name: CACHESTAT

Table 115. Bit Descriptions for CACHESTAT

Bits	Bit Name	Description	Reset	Access
[31:20]	RESERVED	Reserved.	0x0	R
18	DLOCK	This bit is set when D-Cache is locked and cleared when D-Cache is unlocked.	0x0	R
17	DEN	If this bit is set then D-Cache is enabled and when cleared D-Cache is disabled. This is also cleared when CACHESTAT[16] is set.	0x0	R
16	DINIT	It is set when D-cache memory initialization starts and clears when initialization is done. D-Cache is disabled when this bit is set.	0x0	R
[15:4]	RESERVED	Reserved.	0x0	R
2	ILOCK	This bit is set when I-Cache is locked and cleared when I-cache is unlocked.	0x0	R
1	IEN	If this bit is set then I-Cache is enabled and when cleared I-Cache is disabled. This is also cleared when CACHESTAT[0] is set.	0x1	R
0	IINIT	It is set when I-cache memory initialization starts and clears when initialization is done. I-Cache is disabled when this bit is set.	0x0	R

Cache Setup Register

Address: 0x400180C4, Reset: 0x00000002, Name: CACHESETUP

This register is key-protected; therefore, the key (0xF123F456) must be entered in CACHEKEY.

Table 116. Bit Descriptions for CACHESETUP

Bits	Bit Name	Description	Reset	Access
[31:20]	RESERVED	Reserved.	0x0	RW
19	DWRBUF	If this bit is set, for every AHB access, hit from write buffer is not checked.	0x0	RW
18	DLOCK	If this bit is set, D-cache contents are locked. Any new misses are not replaced in D-Cache. This bit is cleared when CACHESETUP[16] is set.	0x0	RW
17	DEN	If this bit set, D-Cache is enabled for AHB accesses. If 0, D-cache is disabled, and all AHB accesses are via Flash memory. This bit is cleared when CACHESETUP[16] is set.	0x0	RW
16	DINIT	If this bit is set, the D-cache contents are initialized to all zeros. This bit is cleared when the initialization starts.	0x0	RW
[15:5]	RESERVED	Reserved.	0x0	RW
4	IRDBUF	If this bit is set, for every AHB access, hit from read buffer is not checked.	0x0	RW
3	IWRBUF	If this bit is set, for every AHB access, hit from write buffer is not checked.	0x0	RW
2	ILOCK	If this bit is set, I-cache contents are locked. Any new misses are not replaced in I-Cache. This bit is cleared when CACHESETUP[0] is set.	0x0	RW
1	IEN	If this bit set, I-Cache is enabled for AHB accesses. If 0, then I-cache is disabled, and all AHB accesses are via Flash memory. This bit is cleared when CACHESETUP[0] is set.	0x1	RW
0	IINIT	If this bit is set, the I-cache contents are initialized to all zeros. This bit is cleared when the initialization starts.	0x0	RW

Cache Key Register

Address: 0x400180C8, Reset: 0x00000000, Name: CACHEKEY

Table 117. Bit Descriptions for CACHEKEY

Bits	Bit Name	Description	Reset	Access
[31:0]	KEY	Cache key register. Enter 0xF123F456 to allow key protected operations. Returns 0x0 if read. The key is cleared automatically after writing to the setup register.	0x0	W

SILICON IDENTIFICATION

The ADuCM320 has two silicon die, and each die has a register that identifies the silicon.

The CHIPID register contains the digital die silicon version in Bits[3:0] and the part identification in Bits[15:4].

The LVID register contains the low voltage die silicon version.

SILICON IDENTIFICATION MEMORY MAPPED REGISTERS

Table 118. Silicon ID Register Summary

Address	Name	Description	Reset	Access
0x40002024	CHIPID	Digital die ID	0x562	R
0x40082C30	LVID	Low voltage die ID	0x0073	R

DIGITAL DIE ID REGISTER

Address: 0x40002024, Reset: 0x0561, Name: CHIPID

Table 119. Bit Descriptions for CHIPID

Bits	Bit Name	Description	Reset	Access
[15:4]	PARTID	Digital die part identifier.	0x56	R
[3:0]	REV	Digital die silicon revision number.	0x2[1]	R

[1] These values are based on initial released silicon. Previous/future revisions of silicon have a different reset value.

LOW VOLTAGE DIE ID REGISTER

Address: 0x40082C30, Reset: 0x0073, Name: LVID

Table 120. Bit Descriptions for LVID

Bits	Bit Name	Description	Reset	Access
[15:4]	LVID	Low voltage die part identifier.	0x7	R
[3:0]	LVREV	Low voltage die silicon revision number.	0x3[1]	

[1] These values are based on initial released silicon. Previous/future revisions of silicon have a different reset value.

DIGITAL I/Os

DIGITAL I/Os FEATURES

The ADuCM320 features multiple general-purpose bidirectional digital input/output (GPIO) pins. Most of the GPIO pins have multiple functions, configurable by user code. At power up, all but one of these pins are configured as GPIOs; one pin reflects the state of the POR. This pin can also be configured by user code to be used as a GPIO. On power-up, these pins are configured as inputs with their corresponding pull-up or pull-down disabled. There are five ports 8 bits wide, but not all bits on some ports are accessible. Inaccessible bits must be ignored.

DIGITAL I/Os BLOCK DIAGRAM

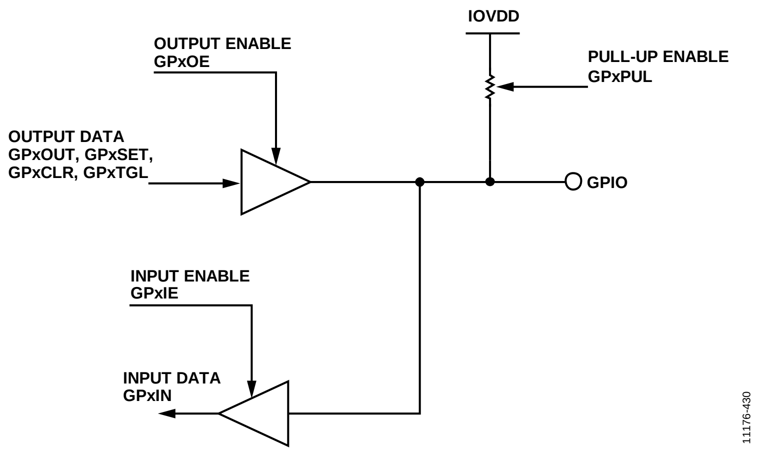


Figure 17. GPIO Structure for Port P0 to Port P3

The pin circuit of Port P0 to Port P3 is shown in Figure 17. Port P4 and Port P5 are essentially the same, but instead of the pull-ups, there are pull-downs to IOGND.

DIGITAL I/Os OVERVIEW

The GPIOs are grouped into six ports: Port 0 to Port 5. Each GPIO can be configured as input, output, or fully open circuit. In input mode, the internal pull-up/pull-down can be enabled by software. All I/O pins, except P3.0 to P3.6 in MDIO mode, are functional over the full supply range (IOV_{DD} = 3.1 V to 3.6 V (maximum)), and the logic input voltages are specified as percentages of the supply as follows:

$$V_{INL} = 0.25 \times IOV_{DD}\ max$$

$$V_{INH} = 0.58 \times IOV_{DD}\ min$$

The absolute maximum input voltage is IOV_{DD} + 0.3 V. The typical leakage current of the GPIOs configured as input or open circuit is 50 nA per GPIO. When the ADuCM320 enters a power saving mode, the GPIO pins retain their states. Note that in power save mode, a driving peripheral cannot drive the pin. That is, if the UART is driving the pin upon entry to deep sleep, it is isolated from the pin and power is gated. Its state and control are restored upon wake-up.

Inaccessible Bits

Some of the bits of P2, P3 and P4 are not brought out of the package. The pin definitions in Table 121 indicate which are accessible. The inaccessible bits are still implemented. Therefore, the pullups/pulldowns for these should be disabled using the GPxPUL MMRs so they do not waste power. Also, such outputs should be disabled using the GPxOE MMRs. These settings are the default at power up. On the other hand, P5.4 to P5.7 are not implemented at all.

DIGITAL I/Os OPERATION

Each digital IO is configured, read, and written independent of the other bits.

GP Input Data (GPxIN)

GPxIN contains the pin input levels if enabled as inputs by GPxIE.

GP Output Data (GPxOUT)

The values of GPxOUT are output on the GPIO pins when configured as outputs by GPxOE.

I/O Data Out Enable (GPxOE)

GPxOE enables the values of GPxOUT to be output on the GPIO pins.

I/O Pull-Up Enable (GPxPUL)

In input mode, GPxPUL enables/disables internal pull-ups/pull-downs. All Port 0 to Port 3 pins have internal pull-ups, and the Port 4 and Port 5 pins have pull-downs. The pull-ups/pull-downs are implemented as MOS devices, with typical performance shown in Figure 18 and Figure 19.

If a pin is configured as an output, the internal pull-up/pull-down is disabled even in open-drain mode.

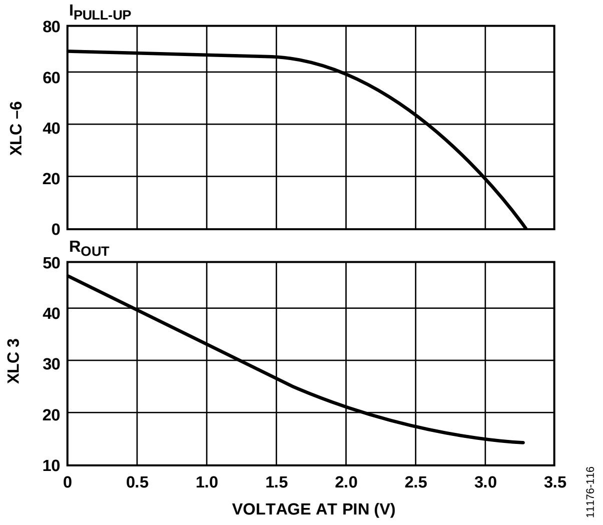


Figure 18. Typical P0 to P3 Pull-Up Characteristics

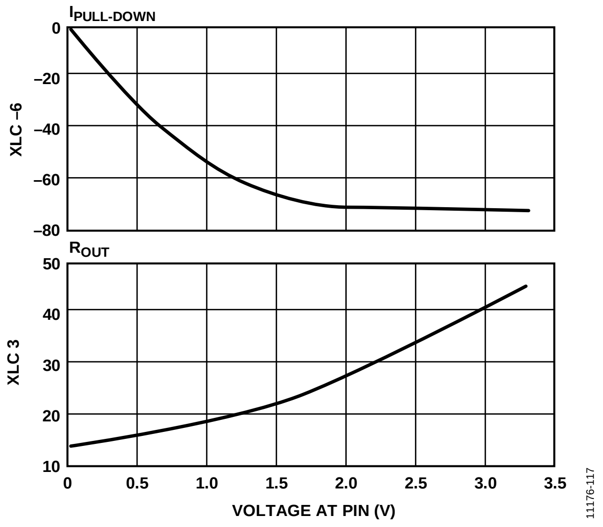


Figure 19. Typical P4 to P5 Pull-Down Characteristics

I/O Data In Enable (GPxIE)

GPxIE enables the GPIO pin input levels to be available in GPxIN.

Open-Drain Enable (GPxODE)

GPxODE configures pins in output mode to open-drain mode. For P0 to P3 in this mode, the outputs can sink current if the corresponding bit in GPxOUT.y is low. If the bit in GPxOUT.y is high, the pin is high impedance. For P4 and P5 in this mode, the outputs can source current if the corresponding bit in GPxOUT.y is high. If the bit in GPxOUT.y is low, the pin is high impedance.

To enable a pin as an open-drain output, set the appropriate bits in GPxOEN and GPxODE.

If a pin is configured as an output, the internal pull-up is disabled even in open-drain mode regardless of GPxPULy.

If internal pull-ups are required in open-drain mode, it is possible to configure the GPIOs in pseudo open-drain mode by setting the corresponding bits of GPxOUT and GPxODE to 0b0 and the corresponding bit of GPxPUL to 0b1. To change between a low output and open-drain high with a pull-up, the corresponding bit GPxOE can be changed from 0b1 to 0b0.

Bit Set

Bit set mode is used to set one or more GPIO data outputs without affecting other outputs within a port. Only the GPIO corresponding with the write data bit equal to 1 is set; the remaining GPIOs are unaffected.

Bit Clear

Bit clear mode is used to clear one or more GPIO data outputs without affecting other outputs within a port. Only the GPIO corresponding with the write data bit equal to 1 is cleared; the remaining GPIOs are unaffected.

Bit Toggle

Bit toggle mode is used to toggle one or more GPIO data outputs without affecting other outputs within a port. Only the GPIO corresponding to the write data bit equal to 1 is toggled; the remaining GPIOs are unaffected.

DIGITAL PORT MULTIPLEX

This block provides control over the GPIO functionality of specified pins because some of the pins offer the choice to work as a GPIO or to have other specific functions.

Table 121. GPIO Multiplex Table

	Configuration Modes			
GPIO	**00**	**01**	**10**	**11**
GP0—GP0CON Controls These Bits				
P0.0	GPIO (GP0CON[1:0] = 0x0)	SPI0 SCLK (GP0CON[1:0] = 0x1)		PLAI[0] (GP0CON[1:0] = 0x3)
P0.1	GPIO (GP0CON[3:2] = 0x0)	SPI0 MISO (GP0CON[3:2] = 0x1)		PLAI[1] (GP0CON[3:2] = 0x3)
P0.2	GPIO (GP0CON[5:4] = 0x0)	SPI0 MOSI (GP0CON[5:4] = 0x1)		PLAI[2] (GP0CON[5:4] = 0x3)
P0.3	GPIO/IRQ0 (GP0CON[7:6] = 0x0)	SPI0 CS (GP0CON[7:6] = 0x1)	PLACLK0 (GP0CON[7:6] = 0x2)	PLAI[3] (GP0CON[7:6] = 0x3)
P0.4	GPIO (GP0CON[9:8] = 0x0)	I2C0 SCL (GP0CON[9:8] = 0x1)		PLAO[2] (GP0CON[9:8] = 0x3)
P0.5	GPIO (GP0CON[11:10] = 0x0)	I2C0 SDA (GP0CON[11:10] = 0x1)		PLAO[3] (GP0CON[11:10] = 0x1)
P0.6	GPIO (GP0CON[13:12] = 0x0)	I2C1 SCL (GP0CON[13:12] = 0x1)		PLAO[4] (GP0CON[13:12] = 0x3)
P0.7[1]	GPIO (GP0CON[15:14] = 0x0)	I2C1 SDA (GP0CON[15:14] = 0x1)		PLAO[5] (GP0CON[15:14] = 0x3)
GP1—GP1CON Controls These Bits				
P1.0	GPIO (GP1CON[1:0] = 0x0)	UART SIN (GP1CON[1:0] = 0x1)	ECLKIN (GP1CON[1:0] = 0x2)	PLAI[4] (GP1CON[1:0] = 0x3)
P1.1	GPI0 (GP1CON[3:2] = 0x0)	UART SOUT (GP1CON[3:2] = 0x1)	PLACLK1 (GP1CON[3:2] = 0x2)	PLAI[5] (GP1CON[3:2] = 0x3)
P1.2	GPIO (GP1CON[5:4] = 0x0)	PWM0 (GP1CON[5:4] = 0x1)		PLAI[6] (GP1CON[5:4] = 0x3)
P1.3	GPIO (GP1CON[7:6] = 0x0)	PWM1 (GP1CON[7:6] = 0x1)		PLAI[7] (GP1CON[7:6] = 0x3)
P1.4	GPIO (GP1CON[9:8] = 0x0)	PWM2 (GP1CON[9:8] = 0x1)	SPI1 SCLK (GP1CON[9:8] = 0x2)	PLAO[10] (GP1CON[9:8] = 0x3)
P1.5	GPIO (GP1CON[11:10] = 0x0)	PWM3 (GP1CON[11:10] = 0x1)	SPI1 MISO (GP1CON[11:10] = 0x2)	PLAO[11] (GP1CON[11:10] = 0x3)
P1.6	GPIO (GP1CON[13:12] = 0x0)	PWM4 (GP1CON[13:12] = 0x1)	SPI1 MOSI (GP1CON[13:12] = 0x2)	PLAO[12] (GP1CON[13:12] = 0x3)
P1.7	GPIO/IRQ1 (GP1CON[15:14] = 0x0)	PWM5 (GP1CON[15:14] = 0x1)	SPI1 CS (GP1CON[15:14] = 0x2)	PLAO[13] (GP1CON[15:14] = 0x3)
GP2—GP2CON Controls These Bits				
P2.0	GPIO/IRQ2 (GP2CON[1:0] = 0x0)	PWMTRIP (GP2CON[1:0] = 0x1)	PLACLK2 (GP2CON[1:0] = 0x2)	PLAI[8] (GP2CON[1:0] = 0x3)
P2.1[2]				
P2.2	GPIO/IRQ4 (GP2CON[5:4] = 0x0)	PORB (GP2CON[5:4] = 0x1)	CLKOUT (GP2CON[5:4] = 0x2)	PLAI[10] (GP2CON[5:4] = 0x3)
P2.3	GPIO/BM (GP2CON[7:6] = 0x0)			
P2.4	GPIO/IRQ5 (GP2CON[9:8] = 0x0)	ADCCONV (GP2CON[9:8] = 0x1)	PWM6 (GP2CON[9:8] = 0x2)	PLAO[18] GP2CON[9:8] = 0x3)

GPIO	Configuration Modes 00	01	10	11
P2.5[2]				
P2.6	GPIO/IRQ7 (GP2CON[13:12] = 0x0)			PLAO[20] (GP2CON[13:12] = 0x3)
P2.7	GPIO/IRQ8 (GP2CON[15:14] = 0x0)			PLAO[21] (GP2CON[15:14] = 0x3)
GP3—GP3CON Controls These Bits				
P3.0	GPIO (GP3CON[1:0] = 0x0)	PRTADDR0 (GP3CON[1:0] = 0x1)		PLAI[12] (GP3CON[1:0] = 0x3)
P3.1	GPIO (GP3CON[3:2] = 0x0)	PRTADDR1 (GP3CON[3:2] = 0x1)		PLAI[13] (GP3CON[3:2] = 0x3)
P3.2	GPIO (GP3CON[5:4] = 0x0)	PRTADDR2 (GP3CON[5:4] = 0x1)		PLAI[14] (GP3CON[5:4] = 0x3)
P3.3	GPIO (GP3CON[7:6] = 0x0)	PRTADDR3 (GP3CON[7:6] = 0x1)		PLAI[15] (GP3CON[7:6] = 0x3)
P3.4	GPIO (GP3CON[9:8] = 0x0)	PRTADDR4 (GP3CON[9:8] = 0x1)		PLAO[26] (GP3CON[9:8] = 0x3)
P3.5	GPIO (GP3CON[11:10] = 0x0)	MCLK (GP3CON[11:10] = 0x1)		PLAO[27] (GP3CON[11:10] = 0x3)
P3.6		MDIO (GP3CON[13:12] = 0x1)		
P3.7[3, 4]	GPIO (GP3CON[15:14] = 0x0)	VDAC2 (GP3CON[15:14] = 0x1)		PLAO[29] (GP3CON[15:14] = 0x3)
GP4—GP4CON Controls These Bits				
P4.2	GPIO (GP4CON[5:4] = 0x0)	AIN8 (GP4CON[5:4] = 0x1)		
P4.3	GPIO (GP4CON[7:6] = 0x0)	AIN9 (GP4CON[7:6] = 0x1)		
P4.4	GPIO (GP4CON[9:8] = 0x0)	AIN12 (GP4CON[9:8] = 0x1)		
P4.5	GPIO (GP4CON[11:10] = 0x0)	AIN13 (GP4CON[11:10] = 0x1)		
P4.6	GPIO (GP4CON[13:12] = 0x0)	AIN14 (GP4CON[13:12] = 0x1)		
P4.7	GPIO (GP4CON[15:14] = 0x0)	AIN15 (GP4CON[15:14] = 0x1)		
GP5—GP5CON Controls These Bits				
P5.0[3, 4]	GPIO (GP5CON[1:0] = 0x0)	VDAC3 (GP5CON[1:0] = 0x1)		
P5.1[3, 4]	GPIO (GP5CON[3:2] = 0x0)	VDAC6 (GP5CON[3:2] = 0x1)		
P5.2[3, 4]	GPIO (GP5CON[5:4] = 0x0)	VDAC7 (GP5CON[5:4] = 0x1)		
P5.3[3, 4]	GPIO (GP5CON[7:6] = 0x0)	VDAC0 (GP5CON[7:6] = 0x1)		

[1] During the power-on reset, the ADuCM320 dives the pin low for up to 200 µs.
[2] Not available as an external pin. Internal PLA elements connected to these pins can be used.
[3] Never configure this pin as an output if associated VDAC output is enabled.
[4] During the power-on reset, the ADuCM320 can enable a pull-down current of 160 µA on this pin.

REGISTER SUMMARY: DIGITAL I/O

Table 122. GPIO Register Summary

Address	Name	Description	Reset	RW
0x40020000	GP0CON	GPIO Port 0 configuration	0x0000	RW
0x40020004	GP0OE	GPIO Port 0 output enable	0x00	RW
0x40020008	GP0PUL	GPIO Port 0 pull-up enable	0x00	RW
0x4002000C	GP0IE	GPIO Port 0 input path enable	0xFF	RW
0x40020010	GP0IN	GPIO Port 0 registered data input	0xXX	R
0x40020014	GP0OUT	GPIO Port 0 data output	0x00	RW
0x40020018	GP0SET	GPIO Port 0 data out set	0x00	W
0x4002001C	GP0CLR	GPIO Port 0 data out clear	0x00	W
0x40020020	GP0TGL	GPIO Port 0 pin toggle	0x00	W
0x40020024	GP0ODE	GPIO Port 0 open-drain enable	0x00	RW
0x40020040	GP1CON	GPIO Port 1 configuration	0x0000	RW
0x40020044	GP1OE	GPIO Port 1 output enable	0x00	RW
0x40020048	GP1PUL	GPIO Port 1 pull-up enable	0x00	RW
0x4002004C	GP1IE	GPIO Port 1 input path enable	0xFF	RW
0x40020050	GP1IN	GPIO Port 1 registered data input	0xXX	R
0x40020054	GP1OUT	GPIO Port 1 data output	0x00	RW
0x40020058	GP1SET	GPIO Port 1 data out set	0x00	W
0x4002005C	GP1CLR	GPIO Port 1 data out clear	0x00	W
0x40020060	GP1TGL	GPIO Port 1 pin toggle	0x00	W
0x40020064	GP1ODE	GPIO Port 1 open-drain enable	0x00	RW
0x40020080	GP2CON	GPIO Port 2 configuration	0x0010	RW
0x40020084	GP2OE	GPIO Port 2 output enable	0x00	RW
0x40020088	GP2PUL	GPIO Port 2 pull-up enable	0x08	RW
0x4002008C	GP2IE	GPIO Port 2 input path enable	0xFF	RW
0x40020090	GP2IN	GPIO Port 2 registered data input	0xXX	R
0x40020094	GP2OUT	GPIO Port 2 data output	0x00	RW
0x40020098	GP2SET	GPIO Port 2 data out set	0x00	W
0x4002009C	GP2CLR	GPIO Port 2 data out clear	0x00	W
0x400200A0	GP2TGL	GPIO Port 2 pin toggle	0x00	W
0x400200A4	GP2ODE	GPIO Port 2 open-drain enable	0x00	RW
0x400200C0	GP3CON	GPIO Port 3 configuration	0x0000	RW
0x400200C4	GP3OE	GPIO Port 3 output enable	0x00	RW
0x400200C8	GP3PUL	GPIO Port 3 pull-up enable	0x00	RW
0x400200CC	GP3IE	GPIO Port 3 input path enable	0xFF	RW
0x400200D0	GP3IN	GPIO Port 3 registered data input	0xXX	R
0x400200D4	GP3OUT	GPIO Port 3 data output	0x00	RW
0x400200D8	GP3SET	GPIO Port 3 data out set	0x00	W
0x400200DC	GP3CLR	GPIO Port 3 data out clear	0x00	W
0x400200E0	GP3TGL	GPIO Port 3 pin toggle	0x00	W
0x400200E4	GP3ODE	GPIO Port 3 open-drain enable	0x00	RW
0x40020100	GP4CON	GPIO Port 4 configuration	0x0000	RW
0x40020104	GP4OE	GPIO Port 4 output enable	0x00	RW
0x40020108	GP4PUL	GPIO Port 4 pull-down enable	0x00	RW
0x4002010C	GP4IE	GPIO Port 4 input path enable	0xFF	RW
0x40020110	GP4IN	GPIO Port 4 registered data input	0xXX	R
0x40020114	GP4OUT	GPIO Port 4 data output	0x00	RW
0x40020118	GP4SET	GPIO Port 4 data out set	0x00	W
0x4002011C	GP4CLR	GPIO Port 4 data out clear	0x00	W
0x40020120	GP4TGL	GPIO Port 4 pin toggle	0x00	W
0x40020124	GP4ODE	GPIO Port 4 open-drain enable	0x00	RW
0x40020240	GP5CON	GPIO Port 5 configuration	0x00	RW

Address	Name	Description	Reset	RW
0x40020244	GP5OE	GPIO Port 5 output enable	0x00	RW
0x40020248	GP5PUL	GPIO Port 5 pull-down enable	0x0	RW
0x4002024C	GP5IE	GPIO Port 5 input path enable	0xF	RW
0x40020250	GP5IN	GPIO Port 5 registered data input	0xXX	R
0x40020254	GP5OUT	GPIO Port 5 data output	0x0	RW
0x40020258	GP5SET	GPIO Port 5 data out set	0x0	W
0x4002025C	GP5CLR	GPIO Port 5 data out clear	0x0	W
0x40020260	GP5TGL	GPIO Port 5 pin toggle	0x0	W
0x40020264	GP5ODE	GPIO Port 5 open-drain enable	0x0	RW

REGISTER DETAILS: DIGITAL I/O

Note that not all bits are accessible to the user on some ports. Inaccessible bits are reserved and must be ignored. See Table 121 for more details on the accessible bits.

GPIO Port Configuration Registers

Address: 0x40020000, Reset: See Table 122**, Name: GP0CON**

Address: 0x40020040, Reset: See Table 122**, Name: GP1CON**

Address: 0x40020080, Reset: See Table 122**, Name: GP2CON**

Address: 0x400200C0, Reset: See Table 122**, Name: GP3CON**

Address: 0x40020100, Reset: See Table 122**, Name: GP4CON**

Address: 0x40020240, Reset: See Table 122**, Name: GP5CON**

Table 123. Bit Descriptions for GP0CON, GP1CON, GP2CON, GP3CON, GP4CON, and GP5CON

Bits	Bit Name	Description	Reset	Access
[15:14]	CON7	Configuration bits for Port x.7. See Table 121.[1]	See Table 122	RW
[13:12]	CON6	Configuration bits for Port x.6. See Table 121.	See Table 122	RW
[11:10]	CON5	Configuration bits for Port x.5. See Table 121.	See Table 122	RW
[9:8]	CON4	Configuration bits for Port x.4. See Table 121.	See Table 122	RW
[7:6]	CON3	Configuration bits for Port x.3. See Table 121.	See Table 122	RW
[5:4]	CON2	Configuration bits for Port x.2. See Table 121.	See Table 122	RW
[3:2]	CON1	Configuration bits for Port x.1. See Table 121.	See Table 122	RW
[1:0]	CON0	Configuration bits for Port x.0. See Table 121.	See Table 122	RW

[1] Where x is 0 for Port 0, 1 for Port 1, 2 for Port 2, and 3 for Port 3.

GPIO Port Output Enable Registers

Address: 0x40020004, Reset: 0x00, Name: GP0OE

Address: 0x40020044, Reset: 0x00, Name: GP1OE

Address: 0x40020084, Reset: 0x00, Name: GP2OE

Address: 0x400200C4, Reset: 0x00, Name: GP3OE

Address: 0x40020104, Reset: 0x00, Name: GP4OE

Address: 0x40020244, Reset: 0x00, Name: GP5OE

Table 124. Bit Descriptions for GP0OE, GP1OE, GP2OE, GP3OE, GP4OE, and GP5OE

Bits	Bit Name	Description	Reset	Access
[7:0]	OE	Pin output drive enable 0: disable the output on the corresponding GPIO 1: enable the output on the corresponding GPIO	See Table 122	RW

GPIO Port Pull-Up Enable Registers

Address: 0x40020008, Reset: 0x00, Name: GP0PUL

Address: 0x40020048, Reset: 0x00, Name: GP1PUL

Address: 0x40020088, Reset: 0x08, Name: GP2PUL

Address: 0x400200C8, Reset: 0x00, Name: GP3PUL

Table 125. Bit Descriptions for GP0PUL, GP1PUL, GP2PUL, GP3PUL

Bits	Bit Name	Description	Reset	Access
[7:0]	PUL	Pin pull-up enable in input mode. 0: Disable the pull up on the corresponding GPIO. 1: Enable the pull up on the corresponding GPIO.	See Table 122	RW

GPIO Port Pull-Down Enable Registers

Address: 0x40020108, Reset: 0x00, Name: GP4PUL

Address: 0x40020248, Reset: 0x00, Name: GP5PUL

Table 126. Bit Descriptions for GP4PUL, GP5PUL

Bits	Bit Name	Description	Reset	Access
[7:0]	PUL	Pin pull-down enable in input mode. 0: disable the pull-down on the corresponding GPIO. 1: enable the pull-down on the corresponding GPIO.	See Table 122	RW

GPIO Port Input Path Enable Registers

Address: 0x4002000C, Reset: 0xFF, Name: GP0IE

Address: 0x4002004C, Reset: 0xFF, Name: GP1IE

Address: 0x4002008C, Reset: 0xFF, Name: GP2IE

Address: 0x400200CC, Reset: 0xFF, Name: GP3IE

Address: 0x4002010C, Reset: 0xFF, Name: GP4IE

Address: 0x4002024C, Reset: 0xF, Name: GP5IE

Table 127. Bit Descriptions for GP0IE, GP1IE, GP2IE, GP3IE, GP4IE, and GP5IE

Bits	Bit Name	Description	Reset	Access
[7:0]	IEN	Input path enable. 0: disable the input path on the corresponding GPIO. 1: enable the input path on the corresponding GPIO. Must be set for external interrupts and to read the pin value.	See Table 122	RW

GPIO Port Registered Data Input Registers

Address: 0x40020010, Reset: 0x0X, Name: GP0IN

Address: 0x40020050, Reset: 0x0X, Name: GP1IN

Address: 0x40020090, Reset: 0x0X, Name: GP2IN

Address: 0x400200D0, Reset: 0x0X, Name: GP3IN

Address: 0x40020110, Reset: 0x0X, Name: GP4IN

Address: 0x40020250, Reset: 0x0X, Name: GP5IN

Table 128. Bit Descriptions for GP0IN, GP1IN, GP2IN, GP3IN, GP4IN, and GP5IN

Bits	Bit Name	Description	Reset	Access
[7:0]	IN	Registered data input. Each bit reflects the state of the GPIO pin.	See Table 122	R

GPIO Port Data Output Registers

Address: 0x40020014, Reset: 0x0000, Name: GP0OUT

Address: 0x40020054, Reset: 0x0000, Name: GP1OUT

Address: 0x40020094, Reset: 0x0000, Name: GP2OUT

Address: 0x400200D4, Reset: 0x0000, Name: GP3OUT

Address: 0x40020114, Reset: 0x0000, Name: GP4OUT

Address: 0x40020254, Reset: 0x0000, Name: GP5OUT

Table 129. Bit Descriptions for GP0OUT, GP1OUT, GP2OUT, GP3OUT, GP4OUT, and GP5OUT

Bits	Bit Name	Description	Reset	Access
[7:0]	OUT	Data out. The bit-band alias addresses should not be used for this register. 0: cleared by user to drive the corresponding GPIO low. 1: set by user code to drive the corresponding GPIO high.	See Table 122	RW

GPIO Port Data Out Set Registers

Address: 0x40020018, Reset: 0x00, Name: GP0SET

Address: 0x40020058, Reset: 0x00, Name: GP1SET

Address: 0x40020098, Reset: 0x00, Name: GP2SET

Address: 0x400200D8, Reset: 0x00, Name: GP3SET

Address: 0x40020118, Reset: 0x00, Name: GP4SET

Address: 0x40020258, Reset: 0x00, Name: GP5SET

Table 130. Bit Descriptions for GP0SET, GP1SET, GP2SET, GP3SET, GP4SET, and GP5SET

Bits	Bit Name	Description	Reset	Access
[7:0]	SET	Set the output high. The bit-band alias addresses should not be used for this register. 0: clearing this bit has no effect. 1: set by user code to drive the corresponding GPIO high.	See Table 122	W

GPIO Port Data Out Clear Registers

Address: 0x4002001C, Reset: 0x00, Name: GP0CLR

Address: 0x4002005C, Reset: 0x00, Name: GP1CLR

Address: 0x4002009C, Reset: 0x00, Name: GP2CLR

Address: 0x400200DC, Reset: 0x00, Name: GP3CLR

Address: 0x4002011C, Reset: 0x00, Name: GP4CLR

Address: 0x4002021C, Reset: 0x00, Name: GP5CLR

Table 131. Bit Descriptions for GP0CLR, GP1CLR, GP2CLR, GP3CLR, GP4CLR, and GP5CLR

Bits	Bit Name	Description	Reset	Access
[7:0]	CLR	Set the output low. The bit-band alias addresses should not be used for this register. 0: Clearing this bit has no effect. 1: Each bit is set to drive the corresponding GPIO pin low.	See Table 122	W

GPIO Port Pin Toggle Registers

Address: 0x40020020, Reset: 0x00, Name: GP0TGL

Address: 0x40020060, Reset: 0x00, Name: GP1TGL

Address: 0x400200A0, Reset: 0x00, Name: GP2TGL

Address: 0x400200E0, Reset: 0x00, Name: GP3TGL

Address: 0x40020120, Reset: 0x00, Name: GP4TGL

Address: 0x40020260, Reset: 0x00, Name: GP5TGL

Table 132. Bit Descriptions for GP0TGL, GP1TGL, GP2TGL, GP3TGL, GP4TGL, and GP5TGL

Bits	Bit Name	Description	Reset	Access
[7:0]	TGL	Toggle the output of the port pin. The bit-band alias addresses should not be used for this register. 0: clearing this bit has not effect. 1: set by user code to invert the corresponding GPIO pin.	See Table 122	W

GPIO Port Open-Drain Enable Registers

Address: 0x40020024, Reset: 0x00, Name: GP0ODE

Address: 0x40020064, Reset: 0x00, Name: GP1ODE

Address: 0x400200A4, Reset: 0x00, Name: GP2ODE

Address: 0x400200E4, Reset: 0x00, Name: GP3ODE

Address: 0x40020124, Reset: 0x00, Name: GP4ODE

Address: 0x40020264, Reset: 0x00, Name: GP5ODE

Table 133. Bit Descriptions for GP0ODE, GP1ODE, GP2ODE, GP3ODE, GP4ODE, and GP5ODE

Bits	Bit Name	Description	Reset	Access
[7:0]	ODE	Open-drain enable. 0: disable the open-drain output mode on corresponding GPIO. 1: enable the open-drain output mode on corresponding GPIO.	See Table 122	RW

I^{2}C SERIAL INTERFACE

I^{2}C FEATURES

- Master or slave mode with 2-byte transmit and receive FIFOs
- Supports
  - 7-bit and 10-bit addressing modes
  - Four 7-bit device addresses or one 10-bit address and two 7-bit addresses in the slave
  - Repeated starts in master and slave modes
  - Clock stretching can be enabled by other devices on the bus without causing any issues with the ADuCM320; however, the ADuCM320 cannot enable clock stretching
  - Master arbitration
  - Continuous read mode for the master or up to 512 bytes fixed read
  - Internal and external loopback
- Support for DMA in master and slave modes
- Software control on the slave of NACK signal

I^{2}C OVERVIEW

The I^{2}C data transfer uses a serial clock pin (SCL) and a serial data pin (SDA). The pins are configured in a wired-AND'ed format that allows arbitration in a multimaster system.

The transfer sequence of an I^{2}C system consists of a master device initiating a transfer by generating a start condition while the bus is idle. The master transmits the slave device address and the direction of the data transfer during the initial address transfer. If the master does not lose arbitration and the slave acknowledges the initial address transfer, the data transfer is initiated. This continues until the master issues a stop condition and the bus becomes idle. Figure 20 shows a typical I^{2}C transfer.

A master device can be configured to generate the serial clock. The frequency is programmed by the user in the serial clock divisor register, I2CxDIV (where x is 0 for I2C0 and 1 for I2C1). The master channel can be set to operate in fast mode (400 kHz) or standard mode (100 kHz).

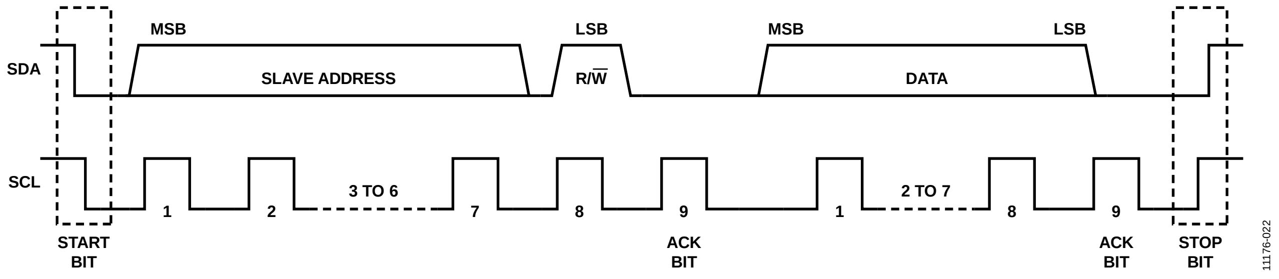


Figure 20. Typical I^{2}C Transfer Sequence

The I^{2}C bus peripheral address in the I^{2}C bus system is programmed by the user. This ID can be modified any time a transfer is not in progress. The user can set up to four slave addresses that are recognized by the peripheral. The peripheral is implemented with a 2-byte FIFO for each transmit and receive shift register. The IRQ and status bits in the control registers are available to signal to the processor core when the FIFOs need to be serviced.

I^{2}C OPERATION

I^{2}C Startup

The following steps are required to run the I^{2}C peripheral:

1. Configure the I^{2}C clock in CLKCON1[10:8], CLKCON5[4] for I^{2}C1, and CLKCON5[3] for I^{2}C0.
2. Configure digital pins (P0.4/P0.5, P0.6/P0.7) for I^{2}C operation via the GP0CON register.
3. Configure I^{2}C registers as required for slave or master operation.
4. Enable the I^{2}C slave or master interrupt source as required.

Note that the user should disable the internal pull-up resistors on the I^{2}C pins via the GP0PUL register when using I^{2}C.

Table 134. GPIO Multiplex

GPIO	Configuration Mode (01)
P0.4, P0.6	SCL
P0.5, P0.7	SDA

Addressing Modes

7-Bit Addressing

The I2CxID0, I2CxID1, I2CxID2, and I2CxID3 registers contain the slave device IDs. The device compares the four I2CxIDx registers to the address byte. To be correctly addressed, the seven MSBs of either ID register must be identical to that of the seven MSBs of the first received address byte. The LSB of the ID registers (the transfer direction bit) is ignored in the process of address recognition.

The master addresses a device using the I2CxADR0 register.

10-Bit Addressing

This feature is enabled by setting I2CxSCON[1] for master and slave mode.

The 10-bit address of the slave is stored in I2CxID0 and I2CxID1, where I2CxID0 contains the first byte of the address, and the R/$\overline{W}$ bit and the upper five bits must be programmed to 11110 as shown in Figure 21. I2CxID1 contains the remaining eight bits of the 10-bit address. I2CxID2 and I2CxID3 can still be programmed with 7-bit addresses.

The master communicates to a 10-bit address slave using the I2CxADR0 and I2CxADR1 registers. The format is described in Figure 21. To perform a read from a slave with a 10-bit address, the master must first send a 10-bit address with the read/write bit cleared, and then it must generate a repeated start and send only the first byte of the address with the read/write bit set. A repeated start is generated by writing to I2CxADR0 while the master is still busy.

I2CADR0 AND I2CID0								I2CADR1 AND I2CID1							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
1	1	1	1	0	2 MSB		R/$\overline{W}$	8 LSB							

11176-023

Figure 21. 10-Bit Address Format

A repeated start condition occurs when a second start condition is sent to a slave without a stop condition being sent in between. This allows the master to reverse the direction of the transfer by changing the R/$\overline{W}$ bit without having to give up control of the bus.

An example of a transfer sequence is shown in Figure 22. This sequence is generally used in cases where the first data sent to the part sets up the register address to be read from.

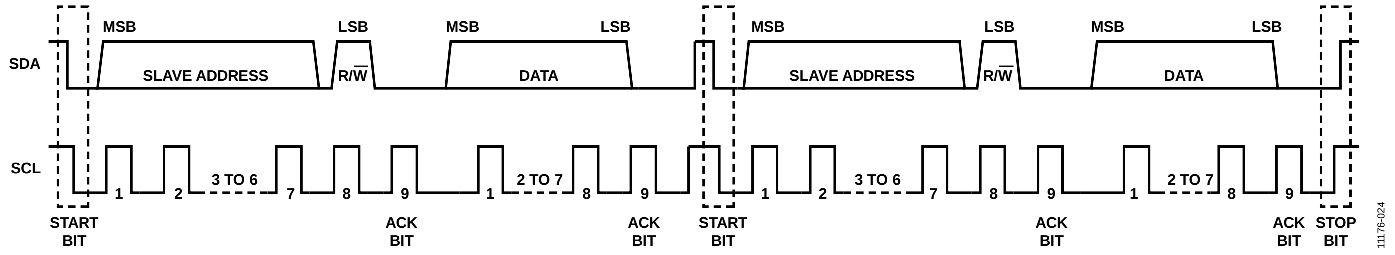


Figure 22. I²C Repeated Start Sequence

On the slave side, an interrupt is generated (if enabled in the I2CxSCON register) when a repeated start and a slave address are received. This can be differentiated from receiving a start and slave address by using the START and REPSTART status bits in the I2CxSSTA MMR.

On the master side, the master generates a repeated start if the I2CxADR0 register is written while the master is still busy with a transaction. After the state machine has started to transmit the device address, it is safe to write to the I2CxADR0 register.

For example, if a transaction involving a write, a repeated start, and then a read/write is required, write to the I2CxADR0 register either after the state machine starts to transmit the device address or after the first MTXREQ interrupt is received. When the transmit FIFO empties, a repeated start is generated.

Similarly, if a transaction involving a read, a repeated start, and then a read/write is required, write to the first master address byte register, I2CxADR1, either after the state machine starts to transmit the device address or after the first MRXREQ interrupt is received. When the requested receive count is reached, a repeated start is generated.

I²C Clock Control

The I²C peripherals are clocked by a gated 20 MHz system clock (PCLK). The CLKCON5[3] bit must be cleared to enable the clock to the I²C0 block. Similarly, the CLKCON5[4] bit must be cleared to enable the clock to the I²C1 block. The CLKCON1[10:8] bits allow the I²C block to be clocked with a slower clock by allowing the 20 MHz clock to be divided, which helps to reduce power.

The I²C master in the system generates the serial clock for a transfer. The master channel can be configured to operate in fast mode (400 kHz) or standard mode (100 kHz).

The bit rate is defined in the I2CxDIV MMR as follows:

$$f_{SCL} = f_{I2CCLK}/(LOW + HIGH + 3)$$

where:

$f_{I2CCLK} = f_{PCLK}/I2CCD$.
f_{PCLK} is the peripheral clock, 20 MHz.
I2CCD is the clock divide value and is set by the CLKCON1[10:8] bits.
HIGH is the high period of the clock, I2CxDIV[15:8] = (REQD_HIGH_TIME/PCLK_PERIOD) − 2.
LOW is the low period of the clock, I2CxDIV[7:0] = (REQD_LOW_TIME/PCLK_PERIOD) − 1.

For 100 kHz SCL operation, with a low time of 5 μs, a high time of 5 μs, and a PCLK frequency of 20 MHz,

$$HIGH = (5\ \mu s/(1/20{,}000{,}000)) - 2 = 98 = 0x62$$

$$LOW = (5\ \mu s/(1/20{,}000{,}000)) - 1 = 99 = 0x63$$

$$f_{SCL} = 20{,}000{,}000/(98 + 99 + 3) = 100\ kHz$$

Resetting the I²C block

Three steps are needed to reset the I²C block.

In master mode,

1. Clear I2CxMCON[0] to 0. This disables the I²C master.
2. Set I2CxSHCON[0] to 1. This is a write only register. Writing to this bit resets the start/stop detection circuits of the I²C block and clears the LINEBUSY status bit (I2CxMSTA[10]).
3. Set I2CxMCON[0] = 1 to reenable the I²C master.

In slave mode,

1. Clear I2CxSCON[0] to 0. This disables the I²C slave.
2. Set I2CxSHCON[0] to 1. This is a write only register. Writing to this bit resets the start/stop detection circuits of the I²C block.
3. Set I2CxSCON[0] to 1 to reenable the I²C slave

Do not reset the I²C peripheral on two consecutive communication sequences.

I²C OPERATING MODES

Master Transfer Initiation

If the master enable bit (I2CxMCON[0], MASEN) is set, a master transfer sequence is initiated by writing a value to the I2CxADRx register. If there is valid data in the I2CxMTX register, it is the first byte transferred in the sequence after the address byte during a write sequence.

Slave Transfer Initiation

If the slave enable bit (I2CxSCON[0], SLVEN) is set, a slave transfer sequence is monitored for the device address in Register I2CxID0, Register I2CxID1, Register I2CxID2, or Register I2CxID3. If the device address is recognized, the part participates in the slave transfer sequence.

Note that a slave operation always starts with the assertion of one of three interrupt sources—read request (MRXREQ/ SRXREQ), write request (MTXREQ, STXREQ), or general call (GCINT) interrupt—and the software should always look for a stop interrupt to ensure that the transaction has completed correctly and to deassert the stop interrupt status bit.

Rx/Tx Data FIFOs

The transmit datapath consists of a master and slave Tx FIFO, I2CxMTX and I2CxSTX, that are each two bytes deep and a transmit shifter. The transmit status bits in I2CxMSTA[1:0] and I2CxSSTA[0] denote whether there is valid data in the Tx FIFO. Data from the Tx FIFO is loaded into the Tx shifter when a serial byte begins transmission. If the Tx FIFO is not full during an active transfer sequence, the transmit request bit (I2CxMSTA[2] or I2CxSSTA[2]) asserts.

In the slave, if there is no valid data to transmit when the Tx shifter is loaded, the transmit underflow status bit asserts (I2CxMSTA[12], ISCxSSTA[1]).

The master generates a stop condition if there is no data in the transmit FIFO and the master is writing data.

The receive datapath consists of a master and slave Rx FIFO, I2CxMRX and I2CxSRX, that are each two bytes deep. The receive request interrupt bit (I2CxMSTA[3] or I2CxSSTA[3]) indicates whether there is valid data in the Rx FIFO. Data is loaded into the Rx FIFO after each byte is received. If valid data in the Rx FIFO is overwritten by the Rx shifter, the receive overflow status bit (I2CxMSTA[9] or I2CxSSTA[4]) is asserted.

I^{2}C Slave Mode Late Loading of I2CSTX Issue

If a byte with the MSB equal to 0 is loaded into I2CxSTX just after the ninth rising edge of SCL during a read operation, the I^{2}C slave pulls the SDA pin low and holds it indefinitely. I2CxSTX can be loaded before the rising edge of the ninth clock by preloading it in advance or during the preceding Rx interrupt. The SDA pin can be released by performing a chip reset, or the master can generate more clocks until the slave releases the SDA pin, and then the master should generate a stop condition.

Master NACK

When receiving data, the master responds with a NACK if its FIFO is full and an attempt is made to write another byte to the FIFO. This last byte received is not written to the FIFO and is lost.

No Acknowledge from the Slave

If the slave does not want to acknowledge a read access, then simply not writing data into the slave transmit FIFO results in a NACK.

If the slave does not want to acknowledge a master write, assert the NACK bit in the slave control register, I2CxSCON[7].

Normally, the slave acknowledges all bytes written into the receive FIFO. If the receive FIFO fills up, the slave cannot write further bytes to it, and it does not acknowledge subsequent bytes not written to the FIFO. The master should then stop the transaction.

The slave does not acknowledge a matching device address if the read/write bit is set and the transmit FIFO is empty. Therefore, there is very little time for the microcontroller to respond to a slave transmit request and the assertion of ACK. It is recommended that EARLYTXR (I2CxSCON[5]) be asserted for this reason.

General Call

An I^{2}C general call is for addressing every device on the I^{2}C bus. A general call address is 0x00 or 0x01. The first byte, address byte is followed by a command byte.

If the address byte is 0x00, then Byte 2, the command byte, can be one of the following:

- 0x6: the I^{2}C interface (master and slave) is reset. The general call interrupt status asserts, and the general call ID bits, GCID (I2CxSSTA[9:8]), are 0x1. User code should take corrective action to reset the entire system or simply reenable the I^{2}C interface.
- 0x4: the general call interrupt status bit is asserted, and the general call ID bits (GCID) are 0x2.

If the address byte is 0x01, a hardware general call is issued.

- Byte 2 in this case is the hardware master address.

The general call interrupt status bit is set on any general call after the second byte is received, and user code should take corrective action to reprogram the device address.

If GCEN is asserted, the slave always acknowledges the first byte of a general call. It acknowledges the second byte of a general call if the second byte is 0x04 or 0x06 or if the second byte is a hardware general call and HGCEN (I2CxSCON[3]) is asserted.

The I2CxALT register contains the alternate device ID for a hardware general call sequence. If the hardware general call enable bit (HGCEN), the general call enable bit (GCEN), and the slave enable bit (SLVEN) are all set, the device recognizes a hardware general call. When a general call sequence is issued and the second byte of the sequence is identical to ALT, the hardware call sequence is recognized for the device.

I^{2}C Reset Mode

The slave state machine is reset when SLVEN is written to 0.

The master state machine is reset when MASEN is written to 0.

I^{2}C Test Modes

The device can be placed in an internal loopback mode by setting the LOOPBACK bit (I2CxMCON[2]). There are four FIFOs (master Tx and Rx, and slave Tx and Rx); therefore the I^{2}C peripheral can, in effect, be set up to talk to itself. External loopback can be performed if the master is set up to address the slave address.

I^{2}C Low Power Mode

If the master and slave are both disabled (MASEN = SLVEN = 0), the I^{2}C section is off. To fully power down the I^{2}C block, the clock to the I^{2}C section of the chip should be disabled by setting CLKCON5[4:3] = 0x3

DMA Requests

Four DMA channels are required to service the I^{2}C master and slave. DMA enable bits are provided in the slave control register and in the master control register.

REGISTER SUMMARY: I2C0

Table 135. I2C0 Register Summary

Address	Name	Description	Reset	RW
0x40003000	I2C0MCON	Master control register	0x0000	RW
0x40003004	I2C0MSTA	Master status register	0x6000	R
0x40003008	I2C0MRX	Master receive data register	0x0000	R
0x4000300C	I2C0MTX	Master transmit data register	0x0000	RW
0x40003010	I2C0MRXCNT	Master receive data count register	0x0000	RW
0x40003014	I2C0MCRXCNT	Master current receive data count register	0x0000	R
0x40003018	I2C0ADR0	1st master address byte register	0x0000	RW
0x4000301C	I2C0ADR1	2nd master address byte register	0x0000	RW
0x40003024	I2C0DIV	Serial clock period divisor register	0x1F1F	RW
0x40003028	I2C0SCON	Slave control register	0x0000	RW
0x4000302C	I2C0SSTA	Slave I2C0 status/error/IRQ register	0x0001	R
0x40003030	I2C0SRX	Slave receive register	0x0000	R
0x40003034	I2C0STX	Slave transmit register	0x0000	RW
0x40003038	I2C0ALT	Hardware general call ID register	0x0000	RW
0x4000303C	I2C0ID0	1st slave address device ID register	0x0000	RW
0x40003040	I2C0ID1	2nd slave address device ID register	0x0000	RW
0x40003044	I2C0ID2	3rd slave address device ID register	0x0000	RW
0x40003048	I2C0ID3	4th slave address device ID register	0x0000	RW
0x4000304C	I2C0FSTA	Master and slave FIFO status register	0x0000	RW
0x40003050	I2C0SHCON	Master and slave shared control register	0x0000	W

REGISTER DETAILS: I2C0

Master Control Register

Address: 0x40003000, Reset: 0x0000, Name: I2C0MCON

Table 136. Bit Descriptions for I2C0MCON

Bits	Bit Name	Description	Reset	Access
[15:12]	RESERVED	Reserved.	0x0	R
11	MTXDMA	Enable master Tx DMA request. 0: disable DMA mode 1: enable I2C0 master DMA Tx requests.	0x0	W
10	MRXDMA	Enable master Rx DMA request. 0: disable DMA mode 1: enable I2C0 master DMA Rx requests.	0x0	W
9	RESERVED	Reserved.	0x0	RW
8	IENCMP	Transaction completed (or stop detected) interrupt enable. 0: an interrupt is not generated when a STOP is detected. 1: an interrupt is generated when a STOP is detected.	0x0	RW
7	IENACK	ACK not received interrupt enable. 0: ACK not received interrupt disable 1: ACK not received interrupt enable	0x0	RW
6	IENALOST	Arbitration lost interrupt enable. 0: arbitration lost interrupt disable 1: arbitration lost interrupt enable	0x0	RW
5	IENMTX	Transmit request interrupt enable. 0: transmit request interrupt disable 1: transmit request interrupt enable	0x0	RW
4	IENMRX	Receive request interrupt enable. 0: receive request interrupt disable 1: receive request interrupt enable	0x0	RW

Bits	Bit Name	Description	Reset	Access
3	RESERVED	Reserved. A value of 0 should be written to this bit.	0x0	RW
2	LOOPBACK	Internal loopback enable. Note that is also possible for the master to loop back a transfer to the slave as long as the device address corresponds, that is, external loopback. 0: SCL and SDA out of the device are not muxed onto their corresponding inputs. 1: SCL and SDA out of the device are muxed onto their corresponding inputs.	0x0	RW
1	COMPETE	Start back-off disable. Setting this bit enables the device to compete for ownership even if another device is currently driving a START condition.	0x0	RW
0	MASEN	Master enable. The master should be disabled when not in use, because this gates the clock to the master and saves power. This bit should not be cleared until a transaction has completed (see the TCOMP bit in the master status register). 0: master is disabled 1: master is enabled	0x0	RW

Master Status Register

Address: 0x40003004, Reset: 0x6000, Name: I2C0MSTA

Table 137. Bit Descriptions for I2C0MSTA

Bits	Bit Name	Description	Reset	Access
15	RESERVED	Reserved.	0x0	R
14	SCL_FILTERED	State of SCL line. This bit is the output of the glitch-filter on SCL. SCL is always pulled high when undriven.	0x1	R
13	SDA_FILTERED	State of SDA line. This bit is the output of the glitch-filter on SDA. SDA is always pulled high when undriven.	0x1	R
12	MTXUFLOW	Master transmit underflow. Asserts when the I2C0 master ends the transaction due to Tx-FIFO empty condition. This bit is asserted only when the IENMTX bit is set.	0x0	RC
11	MSTOP	STOP driven by this I2C0 Master. Asserts when this I2C0 master drives a STOP condition on the I2C0 bus. This bit, when asserted, can indicate a Transaction completion, Tx-underflow, Rx-overflow or a NACK by the slave. This is different from the TCOMP as this bit is not asserted when the STOP condition occurs due to any other I2C0 master. No interrupt is generated for the assertion of this bit. However, if IENCMP is 1, every STOP condition generates an interrupt and this bit can be read. When this bit is read, it clears status.	0x0	RC
10	LINEBUSY	Line is busy. Asserts when a START is detected on the I2C0 bus. Deasserts when a STOP is detected on the I2C0 bus.	0x0	R
9	MRXOF	Master receive FIFO overflow. Asserts when a byte is written to the receive FIFO when the FIFO is already full. When the bit is read, it clears status.	0x0	RC
8	TCOMP	Transaction complete or stop detected. Transaction complete. This bit asserts when a STOP condition is detected on the I2C0 bus. If IENCMP is 1, an interrupt is generated when this bit asserts. This bit only asserts if the master is enabled (MASEN = 1). This bit should be used to determine when it is safe to disable the master. It can also be used to wait for another master transaction to complete on the I2C0 bus when this master loses arbitration. When this bit is read, it clears status. This bit can drive an interrupt.	0x0	RC
7	NACKDATA	ACK not received in response to data write. This bit asserts when an ACK is not received in response to a data write transfer. If IENACK is 1, an interrupt is generated when this bit asserts. This bit can drive an interrupt. This bit is cleared on a read of the I2C0MSTA register.	0x0	RC
6	MBUSY	Master busy. This bit indicates that the master state machine is servicing a transaction. It is cleared if the state machine is idle or another device has control of the I2C0 bus.	0x0	R
5	ALOST	Arbitration lost. This bit asserts if the master loses arbitration. If IENALOST is 1, an interrupt is generated when this bit asserts. This bit is cleared on a read of the I2C0MSTA register. This bit can drive an interrupt.	0x0	RC

Bits	Bit Name	Description	Reset	Access
4	NACKADDR	ACK not received in response to an address. This bit asserts if an ACK is not received in response to an address. If IENACK is 1, an interrupt is generated when this bit asserts. This bit is cleared on a read of the I2C0MSTA register. This bit can drive an interrupt.	0x0	RC
3	MRXREQ	Master receive request. This bit asserts when there is data in the receive FIFO. If IENMRX is 1, an interrupt is generated when this bit asserts. This bit can drive an interrupt.	0x0	R
2	MTXREQ	Master transmit request. This bit asserts when the direction bit is 0 and the transmit FIFO is either empty or not full. If IENMTX is 1, an interrupt is generated when this bit asserts. This bit can drive an interrupt.	0x0	R
[1:0]	MTXFSTA	Master transmit FIFO status. These 2 bits show the master transmit FIFO status and can be decoded as follows: 00 = FIFO empty 10 = 1 byte in FIFO 11 = FIFO full.	0x0	R

Master Receive Data Register

Address: 0x40003008, Reset: 0x0000, Name: I2C0MRX

Table 138. Bit Descriptions for I2C0MRX

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
[7:0]	ICMRX	Master receive register. This register allows access to the receive data FIFO. The FIFO can hold 2 bytes.	0x0	R

Master Transmit Data Register

Address: 0x4000300C, Reset: 0x0000, Name: I2C0MTX

Table 139. Bit Descriptions for I2C0MTX

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
[7:0]	I2C0MTX	Master transmit register. For test and debug purposes, when read, this register returns the byte that is currently being transmitted by the master. That is a byte written to the transmit register can be read back some time later when that byte is being transmitted on the line. This register allows access to the transmit data FIFO. The FIFO can hold 2 bytes.	0x0	RW

Master Receive Data Count Register

Address: 0x40003010, Reset: 0x0000, Name: I2C0MRXCNT

Table 140. Bit Descriptions for I2C0MRXCNT

Bits	Bit Name	Description	Reset	Access
[15:9]	RESERVED	Reserved.	0x0	R
8	EXTEND	Extended read. Use this bit if greater than 256 bytes are required on a read. For example, to receive 412 bytes, write 0x100 (EXTEND = 1) to the I2C0MRXCNT register. Wait for the first byte to be received, then check the I2C0MCRXCNT register for every byte received thereafter. When COUNT returns to 0, 256 bytes have been received. Then write 0x09C to the I2C0MRXCNT register.	0x0	RW
[7:0]	COUNT	Receive count. Program the number of bytes required minus one to this register. If just 1 byte is required, write 0 to this register. If greater than 256 bytes are required, use EXTEND.	0x0	RW

Master Current Receive Data Count Register

Address: 0x40003014, Reset: 0x0000, Name: I2C0MCRXCNT

Table 141. Bit Descriptions for I2C0MCRXCNT

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
[7:0]	COUNT	Current receive count. This register gives the total number of bytes received so far. If 256 bytes are requested, this register reads 0 when the transaction has completed.	0x0	R

First Master Address Byte Register

Address: 0x40003018, Reset: 0x0000, Name: I2C0ADR0

Table 142. Bit Descriptions for I2C0ADR0

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
[7:0]	ADR0	Address byte 0. If a 7-bit address is required, Bit 7 to Bit 1 of ADR0 are programmed with the address and Bit 0 of ADR0 is programmed with the direction (0 = write, 1 = read). If a 10-bit address is required, Bit 7 to Bit 3 of ADR0 are programmed with 11110, Bit 2 to Bit 1 of ADR0 are programmed with the 2 MSBs of the address, and Bit 0 of ADR0 is programmed to 0.	0x0	RW

Second Master Address Byte Register

Address: 0x4000301C, Reset: 0x0000, Name: I2C0ADR1

Table 143. Bit Descriptions for I2C0ADR1

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
[7:0]	ADR1	Address byte 1. This register is only required when addressing a slave with a 10-bit address. Bit 7 to Bit 0 of ADR1 are programmed with the lower 8 bits of the address.	0x0	RW

Serial Clock Period Divisor Register

Address: 0x40003024, Reset: 0x1F1F, Name: I2C0DIV

Table 144. Bit Descriptions for I2C0DIV

Bits	Bit Name	Description	Reset	Access
[15:8]	HIGH	Serial clock high time. This register controls the clock high time. The timer is driven by the core clock (PCLK). Use the following equation to derive the required high time. HIGH = (REQD_HIGH_TIME/PCLK_PERIOD) − 2 For example, to generate a 400kHz SCL with a low time of 1300 ns and a high time of 1200 ns, with a core clock frequency of 50 MHz: LOWTIME = 1300 ns/20 ns − 1 = 0x40 (64 decimal) HIGH = 1200ns/20ns − 2 = 0x3A (58 decimal). This register is reset to 0x1F, which gives an SCL high time of 33 PCLK ticks. $t_{HD:STA}$ is also determined by the HIGH. $t_{HD:STA}$ = (HIGH − 1) × PCLK_PERIOD. As $t_{HD:STA}$ must be 600 ns; with PCLK = 50 MHz, the minimum value for HIGH is 31. This gives an SCL high time of 660 ns.	0x1F	RW
[7:0]	LOW	Serial clock low time. This register controls the clock low time. The timer is driven by the core clock (PCLK). Use the following equation to derive the required low time. LOW = (REQD_LOW_TIME/PCLK_PERIOD) − 1 This register is reset to 0x1F, which gives an SCL low time of 32 PCLK ticks.	0x1F	RW

Slave Control Register

Address: 0x40003028, Reset: 0x0000, Name: I2C0SCON

Table 145. Bit Descriptions for I2C0SCON

Bits	Bit Name	Description	Reset	Access
15	RESERVED	Reserved.	0x0	R
14	STXDMA	Enable slave Tx DMA request. Set to 1 by user code to enable I2C0 slave DMA Rx requests. Cleared by user code to disable DMA mode.	0x0	RW
13	SRXDMA	Enable slave Rx DMA request. Set to 1 by user code to enable I2C0 slave DMA Rx requests. Cleared by user code to disable DMA mode.	0x0	RW
12	IENREPST	Repeated start interrupt enable. If 1, an interrupt is generated when the REPSTART status bit asserts. If 0, an interrupt is not generated when the REPSTART status bit asserts.	0x0	RW
11	RESERVED	Reserved.	0x0	RW
10	IENSTX	Slave transmit request interrupt enable.	0x0	RW
9	IENSRX	Slave receive request interrupt enable.	0x0	RW
8	IENSTOP	Stop condition detected interrupt enable.	0x0	RW
7	NACK	NACK next communication. If this bit is set the next communication is NACK'ed. This could be used for example if during a 24xx style access, an attempt was made to write to a read-only or nonexisting location in system memory. That is the indirect address in a 24xx style write pointed to an unwritable memory location.	0x0	RW
6	RESERVED	Reserved. A value of 0 should be written to this bit.	0x0	RW
5	EARLYTXR	Early transmit request mode. Setting this bit enables a transmit request just after the positive edge of the direction bit SCL clock pulse.	0x0	RW
4	GCSBCLR	General call status bit clear. The general call status and general call ID bits are cleared when a 1 is written to this bit. The general call status and general call ID bits are not reset by anything other than a write to this bit or a full reset.	0x0	W
3	HGCEN	Hardware general call enable. When this bit and the general call enable bit are set the device after receiving a general call, address 00h and a data byte checks the contents of the ALT against the receive shift register. If they match the device has received a hardware general call. This is used if a device needs urgent attention from a master device without knowing which master it needs to turn to. This is a call "to whom it may concern." The device that requires attention embeds its own address into the message. The LSB of the ALT register should always be written to a 1, as per I2C0 January 2000 specification.	0x0	RW
2	GCEN	General call enable. This bit enables the I2C0 slave to ACK an I2C0 general call, Address 0x00 (Write).	0x0	RW
1	ADR10EN	Enabled 10-bit addressing. If this bit is clear, the slave can support four slave addresses, programmed in Register I2C0ID0 to Register I2C0ID3. When this bit is set, 10-bit addressing is enabled. One 10-bit address is supported by the slave and is stored in I2C0ID0 and I2C0ID1, where I2C0ID0 contains the first byte of the address and the upper 5 bits must be programmed to 11110. I2C0ID3 and I2C0ID4 can be programmed with 7-bit addresses at the same time.	0x0	RW
0	SLVEN	Slave enable. When 1, the slave is enabled. When 0, all slave state machine flops are held in reset and the slave is disabled.	0x0	RW

Slave I2C0 Status/Error/IRQ Register

Address: 0x4000302C, Reset: 0x0001, Name: I2C0SSTA

Table 146. Bit Descriptions for I2C0SSTA

Bits	Bit Name	Description	Reset	Access
15	RESERVED	Reserved.	0x0	R
14	START	Start and matching address. This bit is asserted if a start is detected on SCL/SDA and the device address matched, or a general call (address = 0000_0000) code is received and GC is enabled, or a high speed (address = 0000_1XXX) code is received, or a start byte (0000_0001) is received. It is cleared on receipt of either a stop or start condition.	0x0	R
13	REPSTART	Repeated start and matching address. This bit is asserted if START is already asserted and then a repeated start is detected. It is cleared when read or on receipt of a STOP condition. This bit can drive an interrupt.	0x0	RC
[12:11]	IDMAT	Device ID matched. 00: received address matched ID register 0 01: received address matched ID register 1 10: received address matched ID register 2 11: received address matched ID register 3	0x0	R
10	STOP	Stop after start and matching address. Gets set by hardware if the slave device received a STOP condition after a previous START condition and a matching address. Cleared by a read of the status register. If STOPINTEN in the slave control register is asserted, the slave interrupt request asserts when this bit is set. This bit can drive an interrupt.	0x0	RC
[9:8]	GCID	General ID. GCID is cleared when the GCSBCLR is written to 1. These status bits are not cleared by a general call reset. 00: no general call 01: general call reset and program address 10: general call program address 11: general call matching alternative ID	0x0	R
7	GCINT	General call interrupt. This bit always drives an interrupt. The bit is asserted if the slave device receives a general call of any type. To clear, write 1 to the GCSBCLR in the slave control register. If it was a general call reset, all registers are at their default values. If it was a hardware general call, the Rx FIFO holds the second byte of the general call, and this can be compared with the ALT register.	0x0	R
6	SBUSY	Slave busy. Set by hardware if the slave device receives a I2C0 START condition. Cleared by hardware when the address does not match an ID register, the slave device receives a I2C0 STOP condition, or if a repeated start address does not match.	0x0	R
5	NOACK	ACK not generated by the slave. When asserted, it indicates that the slave responded to its device address with a NOACK. It is asserted if there was no data to transmit and sequence was a slave read or if the NACK bit was set in the slave control register and the device was addressed. This bit is cleared on a read of the I2C0SSTA register.	0x0	RC
4	SRXOF	Slave receive FIFO overflow. Asserts when a byte is written to the slave receive FIFO when the FIFO is already full.	0x0	RC
3	SRXREQ	Slave receive request. SRXREQ asserts whenever the slave receive FIFO is not empty. Read or flush the slave receive FIFO to clear this bit. This bit asserts on the falling edge of the SCL clock pulse that clocks in the last data bit of a byte. This bit can drive an interrupt.	0x0	RC

Bits	Bit Name	Description	Reset	Access
2	STXREQ	Slave transmit request. If EARLYTXR = 0, STXREQ is set when the direction bit for a transfer is received high. Thereafter, as long as the transmit FIFO is not full, this bit remains asserted. Initially, it is asserted on the negative edge of the SCL pulse that clocks in the direction bit (if the device address matched also). If EARLYTXR = 1, STXREQ is set when the direction bit for a transfer is received high. Thereafter, as long as the transmit FIFO is not full this bit remains asserted. Initially, it is asserted after the positive edge of the SCL pulse that clocks in the direction bit (if the device address matched also). This bit is cleared on a read of the I2C0SSTA register.	0x0	RC
1	STXUR	Slave transmit FIFO underflow. Is set if a master requests data from the device, and the Tx FIFO is empty for the rising edge of SCL.	0x0	RC
0	STXFSEREQ	Slave Tx FIFO status or early request. If EARLYTXR = 0, this bit is asserted whenever the slave Tx FIFO is empty. If EARLYTXR = 1, TXFSEREQ is set when the direction bit for a transfer is received high. It asserts on the positive edge of the SCL clock pulse that clocks in the direction bit (if the device address matched also). It only asserts once for a transfer. It is cleared when read if EARLYTXR is asserted.	0x1	RW

Slave Receive Register

Address: 0x40003030, Reset: 0x0000, Name: I2C0SRX

Table 147. Bit Descriptions for I2C0SRX

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved	0x0	R
[7:0]	I2C0SRX	Slave receive register	0x0	R

Slave Transmit Register

Address: 0x40003034, Reset: 0x0000, Name: I2C0STX

Table 148. Bit Descriptions for I2C0STX

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved	0x0	R
[7:0]	I2C0STX	Slave transmit register	0x0	RW

Hardware General Call ID Register

Address: 0x40003038, Reset: 0x0000, Name: I2C0ALT

Table 149. Bit Descriptions for I2C0ALT

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
[7:0]	ALT	Slave Alt. This register is used in conjunction with I2C0SCON[3] to match a master generating a hardware general call. It is used in the case where a master device cannot be programmed with a slave's address and instead the slave must recognize the master's address.	0x0	RW

First Slave Address Device ID Register

Address: 0x4000303C, Reset: 0x0000, Name: I2C0ID0

Table 150. Bit Descriptions for I2C0ID0

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
[7:0]	ID0	Slave device ID 0. I2C0ID0[7:1] is programmed with the device ID. I2C0ID0[0] is don't care. See the ADR10EN bit in the slave control register to see how this register is programmed with a 10-bit address.	0x0	RW

Second Slave Address Device ID Register

Address: 0x40003040, Reset: 0x0000, Name: I2C0ID1

Table 151. Bit Descriptions for I2C0ID1

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
[7:0]	ID1	Slave device ID 1. I2C0ID1[7:1] is programmed with the device ID. I2C0ID1[0] is don't care. See the ADR10EN bit in the slave control register to see how this register is programmed with a 10-bit address.	0x0	RW

Third Slave Address Device ID Register

Address: 0x40003044, Reset: 0x0000, Name: I2C0ID2

Table 152. Bit Descriptions for I2C0ID2

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
[7:0]	ID2	Slave device ID 2. I2C0ID2[7:1] is programmed with the device ID. I2C0ID2[0] is don't care. See the ADR10EN bit in the slave control register to see how this register is programmed with a 10-bit address.	0x0	RW

Fourth Slave Address Device ID Register

Address: 0x40003048, Reset: 0x0000, Name: I2C0ID3

Table 153. Bit Descriptions for I2C0ID3

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
[7:0]	ID3	Slave device ID 3. I2C0ID3[7:1] is programmed with the device ID. I2C0ID3[0] is don't care. See the ADR10EN bit in the slave control register to see how this register is programmed with a 10-bit address.	0x0	RW

Master and Slave FIFO Status Register

Address: 0x4000304C, Reset: 0x0000, Name: I2C0FSTA

Table 154. Bit Descriptions for I2C0FSTA

Bits	Bit Name	Description	Reset	Access
[15:10]	RESERVED	Reserved.	0x0	RW
9	MFLUSH	Flush the master transmit FIFO. 0: clearing to 0 has no effect. 1: set to 1 to flush the master transmit FIFO. The master transmit FIFO must be flushed if arbitration is lost or a slave responds with a NACK.	0x0	W
8	SFLUSH	Flush the slave transmit FIFO. 0: clearing to 0 has no effect. 1: set to 1 to flush the slave transmit FIFO.	0x0	W
[7:6]	MRXFSTA	Master receive FIFO status. The status is a count of the number of bytes in a FIFO. 00: FIFO empty 01: 1 bytes in the FIFO 10: 2 bytes in the FIFO 11: reserved	0x0	R
[5:4]	MTXFSTA	Master transmit FIFO status. The status is a count of the number of bytes in a FIFO. 00: FIFO empty 01: 1 bytes in the FIFO 10: 2 bytes in the FIFO 11: reserved	0x0	R
[3:2]	SRXFSTA	Slave receive FIFO status. The status is a count of the number of bytes in a FIFO. 00: FIFO empty 01: 1 bytes in the FIFO 10: 2 bytes in the FIFO 11: reserved	0x0	R
[1:0]	STXFSTA	Slave transmit FIFO status. The status is a count of the number of bytes in a FIFO. 00: FIFO empty 01: 1 bytes in the FIFO 10: 2 bytes in the FIFO 11: reserved	0x0	R

Master and Slave Shared Control Register

Address: 0x40003050, Reset: 0x0000, Name: I2C0SHCON

Table 155. Bit Descriptions for I2C0SHCON

Bits	Bit Name	Description	Reset	Access
[15:1]	RESERVED	Reserved.	0x0000	RW
0	RESET	Write a 1 to this bit to reset the I^{2}C start and stop detection circuits. Setting this bit resets the LINEBUSY status bit.	0x0	W

REGISTER SUMMARY: I2C1

Table 156. I2C1 Register Summary

Address	Name	Description	Reset	RW
0x40003400	I2C1MCON	Master control register	0x0000	RW
0x40003404	I2C1MSTA	Master status register	0x6000	R
0x40003408	I2C1MRX	Master receive data register	0x0000	R
0x4000340C	I2C1MTX	Master transmit data register	0x0000	RW
0x40003410	I2C1MRXCNT	Master receive data count register	0x0000	RW
0x40003414	I2C1MCRXCNT	Master current receive data count register	0x0000	R
0x40003418	I2C1ADR0	1st master address byte register	0x0000	RW
0x4000341C	I2C1ADR1	2nd master address byte register	0x0000	RW
0x40003424	I2C1DIV	Serial clock period divisor register	0x1F1F	RW
0x40003428	I2C1SCON	Slave control register	0x0000	RW
0x4000342C	I2C1SSTA	Slave I^{2}C status/error/IRQ register	0x0001	R
0x40003430	I2C1SRX	Slave receive register	0x0000	R
0x40003434	I2C1STX	Slave transmit register	0x0000	RW
0x40003438	I2C1ALT	Hardware general call ID register	0x0000	RW
0x4000343C	I2C1ID0	1st slave address device ID register	0x0000	RW
0x40003440	I2C1ID1	2nd slave address device ID register	0x0000	RW
0x40003444	I2C1ID2	3rd slave address device ID register	0x0000	RW
0x40003448	I2C1ID3	4th slave address device ID register	0x0000	RW
0x4000344C	I2C1FSTA	Master and slave FIFO status register	0x0000	RW
0x40003450	I2C1SHCON	Master and slave shared control register	0x0000	W

REGISTER DETAILS: I2C1

Master Control Register

Address: 0x40003400, Reset: 0x0000, Name: I2C1MCON

Table 157. Bit Descriptions for I2C1MCON

Bits	Bit Name	Description	Reset	Access
[15:12]	RESERVED	Reserved.	0x0	R
11	MTXDMA	Enable master Tx DMA request. 0: disable DMA mode 1: enable I^{2}C master DMA Tx requests	0x0	W
10	MRXDMA	Enable master Rx DMA request. 0: disable DMA mode 1: enable I^{2}C master DMA Rx requests	0x0	W
9	RESERVED	Reserved.	0x0	RW
8	IENCMP	Transaction completed (or stop detected) interrupt enable. 0: an interrupt is not generated when a STOP is detected. 1: an interrupt is generated when a STOP is detected.	0x0	RW
7	IENACK	ACK not received interrupt enable. 0: ACK not received interrupt disable 1: ACK not received interrupt enable	0x0	RW
6	IENALOST	Arbitration lost interrupt enable. 0: arbitration lost interrupt disable 1: arbitration lost interrupt enable	0x0	RW
5	IENMTX	Transmit request interrupt enable. 0: transmit request interrupt disable 1: transmit request interrupt enable	0x0	RW
4	IENMRX	Receive request interrupt enable. 0: receive request interrupt disable 1: receive request interrupt enable	0x0	RW

Bits	Bit Name	Description	Reset	Access
3	RESERVED	Reserved. A value of 0 should be written to this bit.	0x0	RW
2	LOOPBACK	Internal loopback enable. Note that is also possible for the master to loop back a transfer to the slave as long as the device address corresponds, i.e. external loopback. 0: SCL and SDA out of the device are not muxed onto their corresponding inputs. 1: SCL and SDA out of the device are muxed onto their corresponding inputs.	0x0	RW
1	COMPETE	Start back-off disable. Setting this bit enables the device to compete for ownership even if another device is currently driving a START condition.	0x0	RW
0	MASEN	Master enable. The master should be disabled when not in use as this gates the clock to the master and saves power. This bit should not be cleared until a transaction has completed; see the TCOMP bit in the master status register. 0: master is disabled 1: master is enabled	0x0	RW

Master Status Register

Address: 0x40003404, Reset: 0x6000, Name: I2C1MSTA

Table 158. Bit Descriptions for I2C1MSTA

Bits	Bit Name	Description	Reset	Access
15	RESERVED	Reserved.	0x0	R
14	SCL_FILTERED	State of SCL line. This bit is the output of the glitch-filter on SCL. SCL is always pulled high when undriven.	0x1	R
13	SDA_FILTERED	State of SDA line. This bit is the output of the glitch-filter on SDA. SDA is always pulled high when undriven.	0x1	R
12	MTXUFLOW	Master transmit underflow. Asserts when the I^{2}C master ends the transaction due to Tx-FIFO empty condition. This bit is asserted only when the IENMTX bit is set.	0x0	RC
11	MSTOP	STOP driven by this I^{2}C Master. Asserts when this I2C master drives a STOP condition on the I^{2}C bus. This bit, when asserted, can indicate a transaction completion, Tx-underflow, Rx-overflow or a NACK by the slave. This is different from the TCOMP as this bit is not asserted when the STOP condition occurs due to any other I^{2}C master. No interrupt is generated for the assertion of this bit. However, if IENCMP is 1, every STOP condition generates an interrupt and this bit can be read. When this bit is read, it clears status.	0x0	RC
10	LINEBUSY	Line is busy. Asserts when a START is detected on the I^{2}C bus. Deasserts when a STOP is detected on the I^{2}C bus.	0x0	R
9	MRXOF	Master Receive FIFO overflow. Asserts when a byte is written to the receive FIFO when the FIFO is already full. When the bit is read it clears status.	0x0	RC
8	TCOMP	Transaction complete or stop detected. Transaction complete. This bit asserts when a STOP condition is detected on the I^{2}C bus. If IENCMP is 1, an interrupt is generated when this bit asserts. This bit only asserts if the master is enabled (MASEN = 1). This bit should be used to determine when it is safe to disable the master. It can also be used to wait for another master transaction to complete on the I^{2}C bus when this master loses arbitration. When this bit is read, it clears status. This bit can drive an interrupt.	0x0	RC
7	NACKDATA	ACK not received in response to data write. This bit asserts when an ACK is not received in response to a data write transfer. If IENACK is 1, an interrupt is generated when this bit asserts. This bit can drive an interrupt. This bit is cleared on a read of the I2C1MSTA register.	0x0	RC
6	MBUSY	Master busy. This bit indicates that the master state machine is servicing a transaction. It is clear if the state machine is idle or another device has control of the I^{2}C bus.	0x0	R
5	ALOST	Arbitration lost. This bit asserts if the master loses arbitration. If IENALOST is 1, an interrupt is generated when this bit asserts. This bit is cleared on a read of the I2C1MSTA register. This bit can drive an interrupt.	0x0	RC

Bits	Bit Name	Description	Reset	Access
4	NACKADDR	ACK not received in response to an address. This bit asserts if an ACK is not received in response to an address. If IENACK is 1, an interrupt is generated when this bit asserts. This bit is cleared on a read of the I2CMSTA register. This bit can drive an interrupt.	0x0	RC
3	MRXREQ	Master receive request. This bit asserts when there is data in the receive FIFO. If IENMRX is 1, an interrupt is generated when this bit asserts. This bit can drive an interrupt.	0x0	R
2	MTXREQ	Master transmit request. This bit asserts when the direction bit is 0 and the transmit FIFO is either empty or not full. If IENMTX is 1, an interrupt is generated when this bit asserts. This bit can drive an interrupt.	0x0	R
[1:0]	MTXFSTA	Master Transmit FIFO status. These 2 bits show the master transmit FIFO status and can be decoded as follows: 00 = FIFO empty 10 = 1 byte in FIFO 11 = FIFO full	0x0	R

Master Receive Data Register

Address: 0x40003408, Reset: 0x0000, Name: I2C1MRX

Table 159. Bit Descriptions for I2C1MRX

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
[7:0]	ICMRX	Master receive register. This register allows access to the receive data FIFO. The FIFO can hold 2 bytes.	0x0	R

Master Transmit Data Register

Address: 0x4000340C, Reset: 0x0000, Name: I2C1MTX

Table 160. Bit Descriptions for I2C1MTX

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
[7:0]	I2CMTX	Master transmit register. For test and debug purposes, when read, this register returns the byte that is currently being transmitted by the master. That is a byte written to the transmit register can be read back some time later when that byte is being transmitted on the line. This register allows access to the transmit data FIFO. The FIFO can hold 2 bytes.	0x0	RW

Master Receive Data Count Register

Address: 0x40003410, Reset: 0x0000, Name: I2C1MRXCNT

Table 161. Bit Descriptions for I2C1MRXCNT

Bits	Bit Name	Description	Reset	Access
[15:9]	RESERVED	Reserved.	0x0	R
8	EXTEND	Extended read. Use this bit if greater than 256 bytes are required on a read. For example, to receive 412 bytes, write 0x100 (EXTEND = 1) to the I2CMRXCNT register. Wait for the first byte to be received, then check the I2CMCRXCNT register for every byte received thereafter. When COUNT returns to 0, 256 bytes have been received. Then write 0x09C to the I2CMRXCNT register.	0x0	RW
[7:0]	COUNT	Receive count. Program the number of bytes required minus one to this register. If just 1 byte is required, write 0 to this register. If greater than 256 bytes are required, use EXTEND.	0x0	RW

Master Current Receive Data Count Register

Address: 0x40003414, Reset: 0x0000, Name: I2C1MCRXCNT

Table 162. Bit Descriptions for I2C1MCRXCNT

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
[7:0]	COUNT	Current receive count. This register gives the total number of bytes received so far. If 256 bytes are requested, this register reads 0 when the transaction has completed.	0x0	R

First Master Address Byte Register

Address: 0x40003418, Reset: 0x0000, Name: I2C1ADR0

Table 163. Bit Descriptions for I2C1ADR0

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
[7:0]	ADR0	Address byte 0. If a 7-bit address is required, Bit 7 to Bit 1 of ADR0 are programmed with the address and Bit 0 of ADR0 is programmed with the direction (0 = write, 1 = read). If a 10-bit address is required, Bit 7 to Bit 3 of ADR0 are programmed with 11110, Bit 2 to Bit 1 of ADR0 are programmed with the 2 MSBs of the address, and Bit 0 of ADR0 is programmed to 0.	0x0	RW

Second Master Address Byte Register

Address: 0x4000341C, Reset: 0x0000, Name: I2C1ADR1

Table 164. Bit Descriptions for I2C1ADR1

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
[7:0]	ADR1	Address byte 1. This register is only required when addressing a slave with a 10-bit address. Bit 7 to Bit 0 of ADR1 are programmed with the lower 8 bits of the address.	0x0	RW

Serial Clock Period Divisor Register

Address: 0x40003424, Reset: 0x1F1F, Name: I2C1DIV

Table 165. Bit Descriptions for I2C1DIV

Bits	Bit Name	Description	Reset	Access
[15:8]	HIGH	Serial clock high time. This register controls the clock high time. The timer is driven by the core clock (PCLK). Use the following equation to derive the required high time. HIGH = (REQD_HIGH_TIME/PCLK_PERIOD) − 2. For example, to generate a 400 kHz SCL with a low time of 1300 ns and a high time of 1200 ns, with a core clock frequency of 50 MHz: LOWTIME = 1300 ns/20 ns − 1 = 0x40 (64 decimal). HIGH = 1200 ns/20 ns − 2 = 0x3A (58 decimal). This register is reset to 0x1F, which gives an SCL high time of 33 PCLK ticks. $t_{HD:STA}$ is also determined by the HIGH. $t_{HD:STA}$ = (HIGH − 1) × PCLK_PERIOD. As $t_{HD:STA}$ must be 600 ns, with PCLK = 50 MHz, the minimum value for HIGH is 31. This gives an SCL high time of 660 ns.	0x1F	RW
[7:0]	LOW	Serial clock low time. This register controls the clock low time. The timer is driven by the core clock (PCLK). Use the following equation to derive the required low time. LOW = (REQD_LOW_TIME/PCLK_PERIOD) − 1 This register is reset to 0x1F, which gives an SCL low time of 32 PCLK ticks.	0x1F	RW

Slave Control Register

Address: 0x40003428, Reset: 0x0000, Name: I2C1SCON

Table 166. Bit Descriptions for I2C1SCON

Bits	Bit Name	Description	Reset	Access
15	RESERVED	Reserved.	0x0	R
14	STXDMA	Enable slave Tx DMA request. Set to 1 by user code to enable I^{2}C slave DMA Rx requests. Cleared by user code to disable DMA mode.	0x0	RW
13	SRXDMA	Enable slave Rx DMA request. Set to 1 by user code to enable I^{2}C slave DMA Rx requests. Cleared by user code to disable DMA mode.	0x0	RW
12	IENREPST	Repeated start interrupt enable. If 1, an interrupt is generated when the REPSTART status bit asserts. If 0, an interrupt is not generated when the REPSTART status bit asserts.	0x0	RW
11	RESERVED	Reserved.	0x0	RW
10	IENSTX	Slave transmit request interrupt enable.	0x0	RW
9	IENSRX	Slave receive request interrupt enable.	0x0	RW
8	IENSTOP	Stop condition detected interrupt enable.	0x0	RW
7	NACK	NACK next communication. If this bit is set, the next communication is NACK 'ed. This could be used for example if during a 24xx style access, an attempt was made to write to a read-only or nonexisting location in system memory. That is the indirect address in a 24xx style write pointed to an unwritable memory location.	0x0	RW
6	RESERVED	Reserved. A value of 0 should be written to this bit.	0x0	RW
5	EARLYTXR	Early transmit request mode. Setting this bit enables a transmit request just after the positive edge of the direction bit SCL clock pulse.	0x0	RW
4	GCSBCLR	General call status bit clear. The general call status and general call ID bits are cleared when a 1, is written to this bit. The general call status and general call ID bits are not reset by anything other than a write to this bit or a full reset.	0x0	W
3	HGCEN	Hardware general call enable. When this bit and the general call enable bit are set the device after receiving a general call, Address 00h and a data byte checks the contents of the ALT against the receive shift register. If they match the device has received a hardware general call. This is used if a device needs urgent attention from a master device without knowing which master it needs to turn to. This is a call "to whom it may concern." The device that requires attention embeds its own address into the message. The LSB of the ALT register should always be written to a 1, as per I2C January 2000 specification.	0x0	RW
2	GCEN	General call enable. This bit enables the I^{2}C slave to ACK an I^{2}C general call, Address 0x00 (Write).	0x0	RW
1	ADR10EN	Enabled 10-bit addressing. If this bit is clear, the slave can support four slave addresses, programmed in Register I2CID0 to Register I2CID3. When this bit is set, 10-bit addressing is enabled. One 10-bit address is supported by the slave and is stored in I2CID0 and I2CID1, where I2CID0 contains the first byte of the address and the upper 5 bits must be programmed to 11110. I2CID3 and I2CID4 can be programmed with - bit addresses at the same time.	0x0	RW
0	SLVEN	Slave enable. When 1, the slave is enabled. When 0, all slave state machine flops are held in reset and the slave is disabled.	0x0	RW

Slave I²C Status/Error/IRQ Register

Address: 0x4000342C, Reset: 0x0001, Name: I2C1SSTA

Table 167. Bit Descriptions for I2C1SSTA

Bits	Bit Name	Description	Reset	Access
15	RESERVED	Reserved.	0x0	R
14	START	Start and matching address. This bit is asserted if a start is detected on SCL/SDA and the device address matched, or a general call (GC) (address = 0000_0000) code is received and GC is enabled, or a high speed (address = 0000_1XXX) code is received, or a start byte (0000_0001) is received. It is cleared on receipt of either a stop or start condition.	0x0	R
13	REPSTART	Repeated start and matching address. This bit is asserted if START is already asserted and then a repeated start is detected. It is cleared when read or on receipt of a STOP condition. This bit can drive an interrupt.	0x0	RC
[12:11]	IDMAT	Device ID matched. 00: received address matched ID register 0 01: received address matched ID register 1 10: received address matched ID register 2 11: received address matched ID register 3	0x0	R
10	STOP	Stop after start and matching address. Gets set by hardware if the slave device received a STOP condition after a previous START condition and a matching address. Cleared by a read of the status register. If STOPINTEN in the slave control register is asserted, the slave interrupt request asserts when this bit is set. This bit can drive an interrupt.	0x0	RC
[9:8]	GCID	General ID. GCID is cleared when the GCSBCLR is written to 1. These status bits are not cleared by a general call reset. 00: no general call 01: general call reset and program address 10: general call program address 11: general call matching alternative ID	0x0	R
7	GCINT	General call interrupt. This bit always drives an interrupt. The bit is asserted if the slave device receives a general call of any type. To clear, write 1 to the GCSBCLR in the slave control register. If it was a general call reset, all registers are at their default values. If it was a hardware general call, the Rx FIFO holds the second byte of the general call, and this can be compared with the ALT register.	0x0	R
6	SBUSY	Slave busy. Set by hardware if the slave device receives a I²C START condition. Cleared by hardware when the address does not match an ID register, the slave device receives a I²C STOP condition, or if a repeated start address does not match.	0x0	R
5	NOACK	ACK not generated by the slave. When asserted, it indicates that the slave responded to its device address with a NOACK. It is asserted if there was no data to transmit and sequence was a slave read or if the NACK bit was set in the slave control register and the device was addressed. This bit is cleared on a read of the I2CSSTA register.	0x0	RC
4	SRXOF	Slave receive FIFO overflow. Asserts when a byte is written to the slave receive FIFO when the FIFO is already full.	0x0	RC
3	SRXREQ	Slave receive request. SRXREQ asserts whenever the slave receive FIFO is not empty. Read or flush the slave receive FIFO to clear this bit. This bit asserts on the falling edge of the SCL clock pulse that clocks in the last data bit of a byte. This bit can drive an interrupt.	0x0	RC

Bits	Bit Name	Description	Reset	Access
2	STXREQ	Slave transmit request. If EARLYTXR = 0, STXREQ is set when the direction bit for a transfer is received high. Thereafter, as long as the transmit FIFO is not full, this bit remains asserted. Initially, it is asserted on the negative edge of the SCL pulse that clocks in the direction bit (if the device address matched also). If EARLYTXR = 1, STXREQ is set when the direction bit for a transfer is received high. Thereafter, as long as the transmit FIFO is not full this bit remains asserted. Initially, it is asserted after the positive edge of the SCL pulse that clocks in the direction bit (if the device address matched also). This bit is cleared on a read of the I2CSSTA register.	0x0	RC
1	STXUR	Slave transmit FIFO underflow. Is set if a master requests data from the device, and the Tx FIFO is empty for the rising edge of SCL.	0x0	RC
0	STXFSEREQ	Slave Tx FIFO status or early request. If EARLYTXR = 0, this bit is asserted whenever the slave Tx FIFO is empty. If EARLYTXR = 1, TXFSEREQ is set when the direction bit for a transfer is received high. It asserts on the positive edge of the SCL clock pulse that clocks in the direction bit (if the device address matched also). It only asserts once for a transfer. It is cleared when read if EARLYTXR is asserted.	0x1	RW

Slave Receive Register

Address: 0x40003430, Reset: 0x0000, Name: I2C1SRX

Table 168. Bit Descriptions for I2C1SRX

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved	0x0	R
[7:0]	I2CSRX	Slave receive register	0x0	R

Slave Transmit Register

Address: 0x40003434, Reset: 0x0000, Name: I2C1STX

Table 169. Bit Descriptions for I2C1STX

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved	0x0	R
[7:0]	I2CSTX	Slave transmit register	0x0	RW

Hardware General Call ID Register

Address: 0x40003438, Reset: 0x0000, Name: I2C1ALT

Table 170. Bit Descriptions for I2C1ALT

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
[7:0]	ALT	Slave Alt. This register is used in conjunction with I2CSCON[3] to match a master generating a hardware general call. It is used in the case where a master device cannot be programmed with a slave's address and instead the slave has to recognize the master's address.	0x0	RW

First Slave Address Device ID Register

Address: 0x4000343C, Reset: 0x0000, Name: I2C1ID0

Table 171. Bit Descriptions for I2C1ID0

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
[7:0]	ID0	Slave device ID 0. I2CID0[7:1] is programmed with the device ID. I2CID0[0] is don't care. See the ADR10EN bit in the slave control register to see how this register is programmed with a 10-bit address.	0x0	RW

Second Slave Address Device ID Register

Address: 0x40003440, Reset: 0x0000, Name: I2C1ID1

Table 172. Bit Descriptions for I2C1ID1

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
[7:0]	ID1	Slave device ID 1. I2CID1[7:1] is programmed with the device ID. I2CID1[0] is don't care. See the ADR10EN bit in the slave control register to see how this register is programmed with a 10-bit address.	0x0	RW

Third Slave Address Device ID Register

Address: 0x40003444, Reset: 0x0000, Name: I2C1ID2

Table 173. Bit Descriptions for I2C1ID2

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
[7:0]	ID2	Slave device ID 2. I2CID2[7:1] is programmed with the device ID. I2CID2[0] is don't care. See the ADR10EN bit in the slave control register to see how this register is programmed with a 10-bit address.	0x0	RW

Fourth Slave Address Device ID Register

Address: 0x40003448, Reset: 0x0000, Name: I2C1ID3

Table 174. Bit Descriptions for I2C1ID3

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
[7:0]	ID3	Slave device ID 3. I2CID3[7:1] is programmed with the device ID. I2CID3[0] is don't care. See the ADR10EN bit in the slave control register to see how this register is programmed with a 10-bit address.	0x0	RW

Master and Slave FIFO Status Register

Address: 0x4000344C, Reset: 0x0000, Name: I2C1FSTA

Table 175. Bit Descriptions for I2C1FSTA

Bits	Bit Name	Description	Reset	Access
[15:10]	RESERVED	Reserved.	0x0	RW
9	MFLUSH	Flush the master transmit FIFO. 0: clearing to 0 has no effect. 1: set to 1 to flush the master transmit FIFO. The master transmit FIFO must be flushed if arbitration is lost or a slave responds with a NACK.	0x0	W
8	SFLUSH	Flush the slave transmit FIFO. 0: clearing to 0 has no effect. 1: set to 1 to flush the slave transmit FIFO.	0x0	W
[7:6]	MRXFSTA	Master receive FIFO status. The status is a count of the number of bytes in a FIFO. 00: FIFO empty 01: 1 bytes in the FIFO 10: 2 bytes in the FIFO 11: reserved	0x0	R
[5:4]	MTXFSTA	Master transmit FIFO status. The status is a count of the number of bytes in a FIFO. 00: FIFO empty 01: 1 bytes in the FIFO 10: 2 bytes in the FIFO 11: reserved	0x0	R
[3:2]	SRXFSTA	Slave receive FIFO status. The status is a count of the number of bytes in a FIFO. 00: FIFO empty 01: 1 bytes in the FIFO 10: 2 bytes in the FIFO 11: reserved	0x0	R
[1:0]	STXFSTA	Slave transmit FIFO status. The status is a count of the number of bytes in a FIFO. 00: FIFO empty 01: 1 bytes in the FIFO 10: 2 bytes in the FIFO 11: reserved	0x0	R

Master and Slave Shared Control Register

Address: 0x40003450, Reset: 0x0000, Name: I2C1SHCON

Table 176. Bit Descriptions for I2C1SHCON

Bits	Bit Name	Description	Reset	Access
[15:1]	RESERVED	Reserved.	0x0000	RW
0	RESET	Write a 1 to this bit to reset the I^2C start and stop detection circuits. Setting this bit resets the LINEBUSY status bit.	0x0	W

SERIAL PERIPHERAL INTERFACES

SPI FEATURES

Two complete hardware serial peripheral interfaces with the following standard SPI features:

- Serial clock phase mode and serial clock polarity mode
- LSB first transfer option
- Loopback mode
- Master or slave mode
- Transfer and interrupt mode
- Continuous transfer mode
- Tx/Rx FIFO
- Interrupt mode, interrupt after one, two, three, or four bytes
- Rx overflow mode and Tx underrun mode
- Open-circuit data output mode
- Full duplex communications supported (simultaneous transmit/receive)

SPI OVERVIEW

The ADuCM320 integrates two complete hardware serial peripheral interfaces (SPI). SPI is an industry-standard, synchronous serial interface that allows eight bits of data to be synchronously transmitted and simultaneously received, that is, full duplex. The two SPIs implemented on the ADuCM320 can operate to a maximum bit rate of 20 Mbps in both master and slave modes.

The SPI blocks have an additional DMA feature. Each SPI block has two DMA channels that interface with a μDMA controller of the ARM Cortex-M3 processor. One DMA channel is used for transmitting data and the other is used for receiving data.

SPI OPERATION

The SPI port can be configured for master or slave operation and consists of four pins: MISO, MOSI, SCLK, and $\overline{\text{CS}}$.

Note that the GPIOs used for SPI communication must be configured in SPI mode before enabling the SPI peripheral and that the internal pull-up resistors on the SPI pins should be disabled via the GPxPUL registers when using the SPI.

MISO (Master In, Slave Out) Pin

The MISO pin is configured as an input line in master mode and an output line in slave mode. The MISO line on the master (data in) should be connected to the MISO line in the slave device (data out). The data is transferred as byte-wide (8-bit) serial data, MSB first.

MOSI (Master Out, Slave In) Pin

The MOSI pin is configured as an output line in master mode and an input line in slave mode. The MOSI line on the master (data out) should be connected to the MOSI line in the slave device (data in). The data is transferred as byte-wide (8-bit) serial data, MSB first.

SCLK (Serial Clock I/O) Pin

The master serial clock (SCLK) synchronizes the data being transmitted and received through the MOSI SCLK period. Therefore, a byte is transmitted/received after eight SCLK periods. The SCLK pin is configured as an output in master mode and as an input in slave mode.

In master mode, the polarity and phase of the clock are controlled by the SPIxCON register, and the bit rate is defined in the SPIxDIV register as follows:

$$f_{SERIALCLOCK} = \frac{SPICLK}{2 \times (1 + SPIxDIV)}$$

where *SPICLK* is the 80 MHz system clock divided by the factor set in the CLKCON1[2:0] register.

It is possible to disable the clocks to SPI0 and SPI1 separately:

- CLKCON5[0] = 1 disables the clock to SPI0.
- CLKCON5[1] = 1 disables the clock to SPI1.

By reducing the clock rate to the SPI blocks, it is possible to reduce the power consumption of the SPI block.

The maximum data rate is 20 Mbps.

In slave mode, the SPIxCON register must be configured with the phase and polarity of the expected input clock. The slave accepts data from an external master up to 20 Mbps.

In both master and slave mode, data is transmitted on one edge of the SCLK signal and sampled on the other. Therefore, it is important that the polarity and phase be configured the same for the master and slave devices.

Chip Select ($\overline{\text{CS}}$ Input) Pin

In SPI slave mode, a transfer is initiated by the assertion of $\overline{\text{CS}}$, which is an active low input signal. The SPI port then transmits and receives 8-bit data until the transfer is concluded by deassertion of $\overline{\text{CS}}$. In slave mode, $\overline{\text{CS}}$ is always an input.

In SPI master mode, the $\overline{\text{CS}}$ is an active low output signal. It asserts itself automatically at the beginning of a transfer and deasserts itself upon completion.

$\overline{\text{CS}}$ should always be configured as an SPI pin in GPxCON when the SPI is used. If an ADuCM320 master wants to communicate with multiple SPI slaves, the $\overline{\text{CS}}$ should be left floating and the GPIOs can be connected to the $\overline{\text{CS}}$ lines of the slaves. The CSRSG and CSFLG bits (SPIxSTA[14] and SPIxSTA[13], respectively) can be used to determine when to pull the GPIOs low or high.

SPI TRANSFER INITIATION

In master mode, the transfer and interrupt mode bit, TIM (SPIxCON[6]), determines the manner in which an SPI serial transfer is initiated. If the TIM bit is set, a serial transfer is initiated after a write to the Tx FIFO. If the TIM bit is cleared, a serial transfer is initiated after a read of the Rx FIFO; the read must be done while the SPI interface is idle. A read done during an active transfer does not initiate another transfer.

For any setting of SPIxCON[1] and SPIxCON[6], the SPI simultaneously receives and transmits data. Therefore, during data transmission, the SPI is also receiving data and filling up the Rx FIFO. If the data is not read from the Rx FIFO, the overflow interrupt occurs when the FIFO starts to overflow. If the user does not want to read the Rx data or receive overflow interrupts, SPIxCON[12] can be set and the receive data is not saved to the Rx FIFO.

Similarly, when the user wants to only receive data and does not want to write data to the Tx FIFO, SPIxCON[13] can be set to avoid receiving underrun interrupts from the Tx FIFO.

Tx Initiated Transfer

For transfers initiated by a write to the Tx FIFO, the SPI starts transmitting as soon as the first byte is written to the FIFO, irrespective of the configuration in SPIxCON[15:14]. The first byte is immediately read from the FIFO, written to the Tx shift register, and the transfer commences.

If the continuous transfer enable bit, SPIxCON[11], is set, the transfer continues until no valid data is available in the Tx FIFO. There is no stall period between transfers where $\overline{\text{CS}}$ is deasserted; $\overline{\text{CS}}$ is asserted and remains asserted for the duration of the transfer until the Tx FIFO is empty. Determining when the transfer stops does not depend on SPIxCON[15:14]; the transfer stops when there is no valid data left in the FIFO. Conversely, the transfer continues while there is valid data in the FIFO.

If the continuous transfer enable bit, SPIxCON[11], is cleared, each transfer consists of a single 8-bit serial transfer. If valid data exists in the Tx FIFO, a new transfer is initiated after a stall period where $\overline{\text{CS}}$ is deasserted.

Rx Initiated Transfer

Transfers initiated by a read of the Rx FIFO depend on the number of bytes to be received in the FIFO. If SPIxCON[15:14] = 11 and a read to the Rx FIFO occurs, the SPI initiates a 4-byte transfer. If continuous mode is set, the four bytes occur continuously with no deassertion of $\overline{\text{CS}}$ between bytes. If continuous mode is not set, the four bytes occur with stall periods between transfers where the $\overline{\text{CS}}$ is deasserted. A read of the Rx FIFO while the SPI is receiving data does not initiate another transfer after the present transfer is complete.

In slave mode, a transfer is initiated by the assertion of $\overline{\text{CS}}$ ($\overline{\text{CS}}$ = 0).

The device as a slave transmits and receives 8-bit data until the transfer is concluded by the deassertion of $\overline{\text{CS}}$ ($\overline{\text{CS}}$ = 1).

The SPI transfer protocol diagrams (see Figure 23 and Figure 24) illustrate the data transfer protocol for the SPI and the effects of the CPHA and CPOL bits in the control register (SPIxCON) on that protocol.

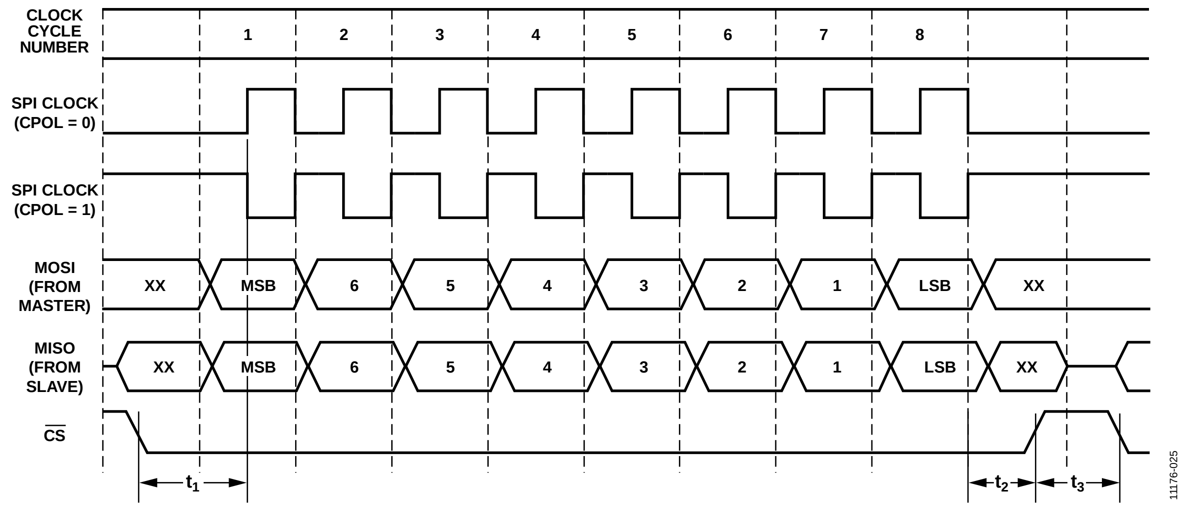


Figure 23. SPI Transfer Protocol CPHA = 0

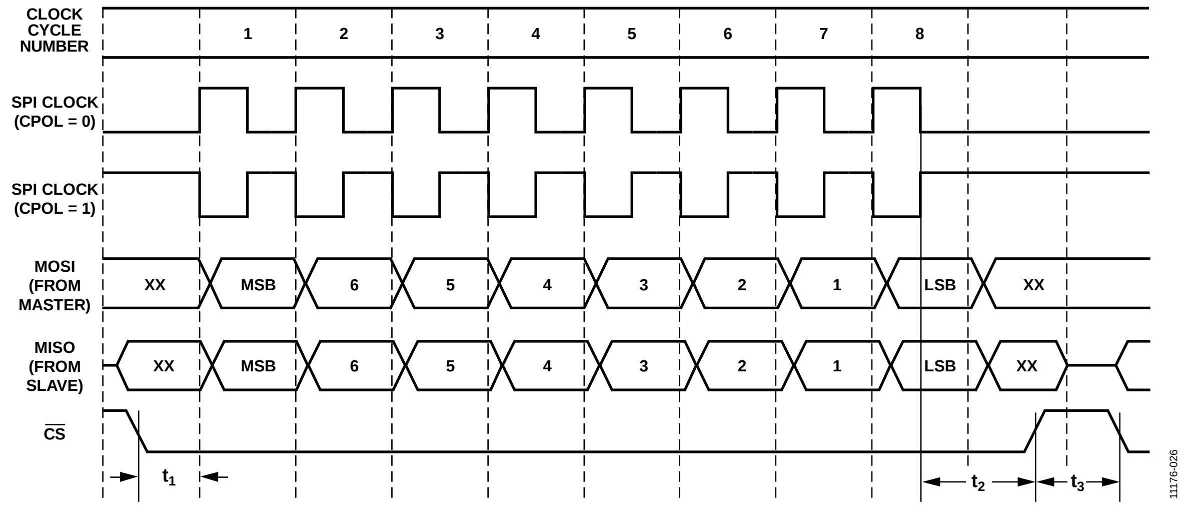


Figure 24. SPI Transfer Protocol CPHA = 1

SPI Data Underrun and Overflow

If the transmit zeros enable bit, ZEN (SPIxCON[7]), is cleared, the last byte from the previous transmission is shifted out when a transfer is initiated with no valid data in the FIFO. If ZEN is set to 1, 0s are transmitted when a transfer is initiated with no valid data in the FIFO.

If the Rx overflow overwrite enable bit, RXOF (SPIxCON[8]), is set, the valid data in the Rx FIFO is overwritten by the new serial byte received if there is no space left in the FIFO. If RXOF is cleared, the new serial byte received is discarded if there is no space left in the FIFO.

When the RXOF is set, the contents of the SPI Rx FIFO are undefined and its contents should be discarded by user code.

Full Duplex Operation

Simultaneous reads/writes are supported on the SPI.

When implementing full duplex transfers in master mode, use the following procedure:

1. Initiate a transfer sequence via a transmit on the MOSI pin. Set SPIxCON[6] = 1. If interrupts are enabled, interrupts are triggered when a transmit interrupt occurs but not when a byte is received.
2. If you are using interrupts, the SPI Tx interrupt indicated by SPIxSTA[5] or the Tx FIFO underrun interrupt (SPIxSTA[4]) is asserted approximately 3 × SPICLK to 4 × SPICLK periods into the transfer of the first byte. Reload a byte into the Tx FIFO, if necessary, by writing to SPIxTX.
3. The first byte received via the MISO pin does not update the Rx FIFO status bits (SPIxSTA[10:8]) until 12 × SPICLK periods after $\overline{\text{CS}}$ has gone low. Therefore, two transmit interrupts may occur before the first receive byte is ready to be handled.
4. After the last transmit interrupt has occurred, it may be necessary to read two more bytes. It is recommended that SPIxSTA[10:8] are polled outside of the SPI interrupt handler after the last transmit interrupt is handled.

SPI INTERRUPTS

There is one interrupt line per SPI and four sources of interrupts. SPIxSTA[0] reflects the state of the interrupt line, and SPIxSTA[7:4] reflects the state of the four sources.

The SPI generates either TIRQ or RIRQ. Both interrupts cannot be enabled at the same time. The appropriate interrupt is enabled using the TIM bit, SPIxCON[6]. If TIM = 1, TIRQ is enabled. If TIM = 0, RIRQ is enabled.

In addition, note that the SPI0 and SPI1 interrupt source must be enabled in the NVIC register as follows: ISER0[19] = SPI0, ISER0[20] = SPI1.

Tx Interrupt

If TIM (SPIxCON[6]) is set, the Tx FIFO status causes the interrupt. The SPIxCON[15:14] bits control when the interrupt occurs, as shown in Table 177.

Table 177. SPIxCON[15:14] IRQ Mode Bits

SPIxCON[15:14]	Interrupt Condition
00	An interrupt is generated after each byte that is transmitted. The interrupt occurs when the byte is read from the FIFO and written to the shift register.
01	An interrupt is generated after every two bytes that are transmitted.
10	An interrupt occurs after every third byte that is transmitted.
11	An interrupt occurs after every fourth byte that is transmitted.

The interrupts are generated depending on the number of bytes transmitted and not on the number of bytes in the FIFO. This is unlike the Rx interrupt, which depends on the number of bytes in the Rx FIFO and not the number of bytes received.

The transmit interrupt is cleared by a read to the status register. The status of this interrupt can be read by reading SPIxSTA[5]. The interrupt is disabled if SPIxCON[13] is left high.

A write to the control register, SPIxCON, resets the transmitted byte counter back to 0. For example, in a case where SPIxCON[15:14] is set to 0x3 and SPIxCON is written to after three bytes have been transmitted, the Tx interrupt does not occur until another four bytes have been transmitted.

Rx Interrupt

If TIM (SPIxCON[6]) is cleared, the Rx FIFO status causes the interrupt. The SPIxCON[15:14] bits control when the interrupt occurs. The interrupt is cleared by a read of SPIxSTA. The status of this interrupt can be read by reading SPIxSTA[6].

Interrupts are only generated when data is written to the FIFO. For example, if the SPIxCON[15:14] bits are set to 0x00, an interrupt is generated after the first byte is received. When the status register is read, the interrupt is deactivated. If the byte is not read from the FIFO, the interrupt is not regenerated. Another interrupt is not generated until another byte is received in the FIFO.

The interrupt depends on the number of valid bytes in FIFO and not the number of bytes received. For example, when the SPIxCON[15:14] bits are set to 0x1, an interrupt is generated after a byte is received if there are two or more bytes in the FIFO. The interrupt is not generated after every two bytes received.

The interrupt is disabled if SPIxCON[12] is left high.

Underrun/Overflow Interrupts

SPIxSTA[7] and SPIxSTA[4] generate SPI interrupts.

When a transfer starts with no data in the Tx FIFO, SPIxSTA[4] is set to indicate an underrun condition. This causes an interrupt. The interrupt (and status bit) are cleared upon a read of the status register. This interrupt occurs irrespective of SPIxCON[15:14]. This interrupt is disabled if SPIxCON[13] is set.

When data is received and the Rx FIFO is already full, SPIxSTA[7] is set to 1, indicating an overflow condition. This causes an interrupt. The interrupt (and status bit) are cleared upon a read of the status register. This interrupt occurs irrespective of SPIxCON[15:14]. This interrupt is disabled if SPIxCON[12] is set.

When the SPI Rx overflow bit (SPIxSTA[7]) is set to 1, the contents of the SPI Rx FIFO are undetermined and should not be used. The user should flush the Rx FIFO upon detecting this error condition.

All interrupts are cleared either by a read of the status register or when SPIxCON[0] is deasserted. The Rx and Tx interrupts are also cleared if the relevant flush bits are asserted. Otherwise, the interrupts stay active even if the SPI is reconfigured.

SPI WIRE-OR'ED MODE (WOM)

To prevent contention when the SPI is used in a multimaster or multislave system, the data output pins, MOSI and MISO, can be configured to behave as open-circuit drivers. An external pull-up resistor is required when this feature is selected. The WOM bit (SPIxCON[4]) controls the pad enable outputs for the data lines.

SPI CSERR CONDITION

The CSERR bit (SPIxSTA[12]) indicates if an erroneous deassertion of the $\overline{\text{CS}}$ signal has been detected before the completion of all eight SCLK cycles. This bit generates an interrupt and is available in all modes of operation: slave, master, and during DMA transfers.

If an interrupt generated by the CSERR bit (SPIxSTA[12]) occurs, the SPI ENABLE bit (SPIxCON[0]) should be disabled and restarted to enable a clean recovery. This ensures that subsequent transfers are error free. The BCRST bit (SPIxDIV[7]) should be set at all times in both slave mode and master mode except when a midbyte stall in SPI communication is required. In this case, the CSERR flag is set but can be ignored.

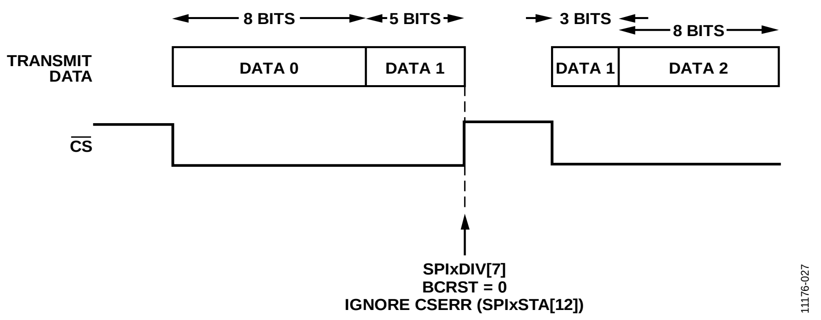


Figure 25. SPI Communication: Midbyte Stall

Note that the SPI should only be reenabled when the $\overline{\text{CS}}$ signal is high.

SPI DMA

DMA operation is provided on both SPI channels. Two DMA channels are dedicated to transmit and receive. The SPI DMA channels should be configured in the μDMA controller of the ARM Cortex-M3 processor.

It is possible to enable a DMA request on one or two channels at the same time by setting the DMA request bits for receive or transmit in the SPIxDMA register. If only the DMA transmit request (SPIxDMA[1]) is enabled, the Rx FIFO overflows during an SPI transfer, unless the received data is read by user code, and an overflow interrupt is generated. To avoid generating overflow interrupts, the Rx FIFO flush bit should be set or the SPI interrupt should be disabled in the NVIC. If only the DMA receive request (SPIxDMA[2]) is enabled, the Tx FIFO is underrun. To avoid an underrun interrupt, the SPI interrupt should be disabled.

The SPI Tx (SPIxSTA[5]) and SPI Rx (SPIxSTA[6]) interrupts are not generated when using DMA. The SPI TXUR (SPIxSTA[4]) and RXOF (SPIxSTA[7]) interrupts are generated when using DMA. The SPIxCON[15:14] bits are not used in transmit mode and should be set to 0x00 in receive mode.

The enable bit (SPIxDMA[0]) controls the start of a DMA transfer. DMA requests are only generated when enable = 1. At the end of a DMA transfer, that is, when receiving a DMA SPI transfer interrupt, this bit needs to be cleared to prevent extra DMA requests to the μDMA controller. The data still present in the Tx FIFO is transmitted if in Tx mode.

DMA Master Transmit Configuration

The DMA SPI Tx channel should be configured.

The NVIC should be configured to enable DMA Tx master interrupt (for example, enable DMA Tx master interrupt SPI0 Tx using ISER0[26]).

When all data present in the DMA buffer are transmitted, the DMA generates an interrupt. User code should disable the DMA request. Data is still in the Tx FIFO because the DMA request is generated each time there is free space in the Tx FIFO to always keep the FIFO full. User code can check how many bytes are still present in the FIFO in the FIFO status register.

DMA Master Receive Configuration

The SPIxCNT register is available in DMA receive master mode only. It sets the number of receive bytes required by the SPI master or the number of clocks that the master needs to generate. When the required number of bytes are received, no more transfers are initiated. To initiate a DMA master receive transfer, a dummy read should be completed by user code. This dummy read should be added to the SPIxCNT number.

The counter counting the bytes as they are received is reset either when SPI is disabled in SPIxCON[0] or if the SPIxCNT register is modified by user code.

Performing SPIx DMA Master Receive

The DMA SPI Rx channel should be configured.

The NVIC should be configured to enable DMA Rx master interrupt (for example, enable DMA Rx master interrupt SPI1 Rx using ISER0[29]).

The DMA transfer stops when the number of bytes have been transferred. Note that the DMA buffer must be of the same size as SPI1CNT to generate a DMA interrupt when the transfer is complete.

SPI AND POWER-DOWN MODES

In master mode, before entering power-down mode it is recommended to disable the SPI block in SPIxCON[0]. In slave mode, in either mode of operation, interrupt driven or DMA, the $\overline{\text{CS}}$ line level should be checked via the GPIO registers to ensure that the SPI is not communicating and that the SPI block is disabled while the $\overline{\text{CS}}$ line is high. At power-up, the SPI block can be reenabled.

REGISTER SUMMARY: SPI0

Table 178. SPI0 Register Summary

Address	Name	Description	Reset	RW
0x4002C000	SPI0STA	Status register	0x0000	R
0x4002C004	SPI0RX	Receive register	0x0000	R
0x4002C008	SPI0TX	Transmit register	0x0000	W
0x4002C00C	SPI0DIV	Baud rate selection register	0x0000	RW
0x4002C010	SPI0CON	SPI configuration register	0x0000	RW
0x4002C014	SPI0DMA	SPI DMA enable register	0x0000	RW
0x4002C018	SPI0CNT	Transfer byte count register	0x0000	RW

REGISTER DETAILS: SPI0

Status Register

Address: 0x4002C000, Reset: 0x0000, Name: SPI0STA

Table 179. Bit Descriptions for SPI0STA

Bits	Bit Name	Description	Reset	Access
15	RESERVED	Reserved.	0x0	R
14	CSRSG	Detected a rising edge on CS, in CONT mode. This bit causes an interrupt. This can be used to identify the end of an SPI data frame. 0: cleared to 0 when the status register is read. 1: set to 1 when there was a rising edge in CS line, when the device was in master mode, continuous transfer, high frequency mode and CSIRQ_EN was asserted.	0x0	RC
13	CSFLG	Detected a falling edge on CS, in CONT mode. This bit causes an interrupt. This can be used to identify the start of an SPI data frame. 0: cleared to 0 when the status register is read. 1: set to 1 when there was a falling edge in CS line, when the device was in master mode, continuous transfer, high frequency mode and CSIRQ_EN was asserted	0x0	RC
12	CSERR	Detected a CS error condition. 0: cleared to 0 when the status register is read. 1: set to 1 when the CS line was de-asserted abruptly, even before the full byte of data was transmitted completely. This bit causes an interrupt.	0x0	RC
11	RXS	SPI Rx FIFO excess bytes present. 0: this bit is cleared when the number of bytes in the FIFO is equal or less than the number in SPI0CON[15:14]. 1: this bit is set when there are more bytes in the Rx FIFO than indicated in the MOD bits in SPI0CON.	0x0	R

Bits	Bit Name	Description	Reset	Access
[10:8]	RXFSTA	SPI Rx FIFO status. 000: Rx FIFO empty 001: 1 valid byte in FIFO 010: 2 valid bytes in the FIFO 011: 3 valid bytes in the FIFO 100: 4 valid bytes in the FIFO	0x0	R
7	RXOF	SPI Rx FIFO overflow. 0: cleared when the SPISTA register is read. 1: set when the Rx FIFO was already full when new data was loaded to the FIFO. This bit generates an interrupt except when RFLUSH is set in SPI0CON.	0x0	RC
6	RX	SPI Rx IRQ. Not available in DMA mode. 0: cleared when the SPI0STA register is read. 1: set when a receive interrupt occurs. This bit is set when TIM in SPI0CON is cleared and the required number of bytes have been received.	0x0	RC
5	TX	SPI Tx IRQ. Status bit. Not available in DMA mode. 0: CLR. Cleared to 0 when the SPI0STA register is read. 1: SET. Set to 1 when a transmit interrupt occurs. This bit is set when TIM in SPI0CON is set and the required number of bytes have been transmitted.	0x0	RC
4	TXUR	SPI Tx FIFO underflow. 0: cleared to 0 when the SPI0STA register is read. 1: set to 1 when a transmit is initiated without any valid data in the Tx FIFO. This bit generates an interrupt except when TFLUSH is set in SPI0CON.	0x0	RC
[3:1]	TXFSTA	SPI Tx FIFO status. 000: Tx FIFO empty 001: 1 valid byte in FIFO 010: 2 valid bytes in FIFO 011: 3 valid bytes in FIFO 100: 4 valid bytes in FIFO	0x0	R
0	IRQ	SPI interrupt status. 0: cleared to 0 after reading SPI0STA. 1: set to 1 when an SPI based interrupt occurs.	0x0	RC

Receive Register

Address: 0x4002C004, Reset: 0x0000, Name: SPI0RX

Table 180. Bit Descriptions for SPI0RX

Bits	Bit Name	Description	Reset	Access
[15:8]	DMA_DATA_BYTE_2	8-bit receive buffer. These 8-bits are used only in the DMA mode, where all FIFO accesses happen as half-word access. They return zeros if DMA is disabled.	0x0	R
[7:0]	DATA_BYTE_1	8-bit receive buffer.	0x0	R

Transmit Register

Address: 0x4002C008, Reset: 0x0000, Name: SPI0TX

Table 181. Bit Descriptions for SPI0TX

Bits	Bit Name	Description	Reset	Access
[15:8]	DMA_DATA_BYTE_2	8-bit transmit buffer. These 8-bits are used only in the DMA mode, where all FIFO accesses happen as half-word access. They return zeros if DMA is disabled.	0x0	W
[7:0]	DATA_BYTE_1	8-bit transmit buffer.	0x0	W

Baud Rate Selection Register

Address: 0x4002C00C, Reset: 0x0000, Name: SPI0DIV

Table 182. Bit Descriptions for SPI0DIV

Bits	Bit Name	Description	Reset	Access
[15:9]	RESERVED	Reserved.	0x0	R
8	CSIRQ_EN	Enable interrupt on every CS edge in CONT and HFM mode. If this bit is set and the SPI module is in continuous mode, any edge on CS generates an interrupt and the corresponding status bits (CSRSG, CSFLG) are asserted. If this bit is clear, no interrupt is generated. This bit has no effect if the SPI is not in continuous mode and high speed mode.	0x0	RW
7	BCRST	Reset mode for CSERR. If this bit is set, the bit counter is reset after a CS error condition and the Cortex is expected to clear the SPI enable bit. If this bit is clear, the bit counter continues from where it stopped. SPI can receive the remaining bits when CS gets asserted and Cortex has to ignore the CSERR interrupt. However, it is strongly recommended to set this bit for a graceful recovery after a CS error.	0x0	RW
6	HFM	High frequency mode. This bit is used for applications using high frequency where the pad introduces a significant delay on the SCL. This can cause a significant enough difference between the serial clock and the data being received on the Rx shift register. In this mode, the Rx shift register is clocked by SCLIN instead of UCLK.	0x0	R/W
[5:0]	DIV	SPI clock divider. DIV is the factor used to divide UCLK to generate the serial clock.	0x0	RW

SPI Configuration Register

Address: 0x4002C010, Reset: 0x0000, Name: SPI0CON

Table 183. Bit Descriptions for SPI0CON

Bits	Bit Name	Description	Reset	Access
[15:14]	MOD	SPI IRQ mode bits. These bits configure when the Tx/Rx interrupts occur in a transfer. For DMA Rx transfer, these bits should be 00. 00: Tx interrupt occurs when 1 byte has been transferred. Rx interrupt occurs when 1 or more bytes have been received into the FIFO. 01: Tx interrupt occurs when 2 bytes has been transferred. Rx interrupt occurs when 2 or more bytes have been received into the FIFO. 10: Tx interrupt occurs when 3 bytes has been transferred. Rx interrupt occurs when 3 or more bytes have been received into the FIFO. 11: Tx interrupt occurs when 4 bytes has been transferred. Rx interrupt occurs when the Rx FIFO is full, or 4 bytes present.	0x0	RW
13	TFLUSH	SPI Tx FIFO flush enable. 0: clear this bit to disable Tx FIFO flushing. 1: set this bit to flush the Tx FIFO. This bit does not clear itself and should be toggled if a single flush is required. If this bit is left high, then either the last transmitted value or 0x00 is transmitted depending on the ZEN bit. Any writes to the Tx FIFO are ignored while this bit is set.	0x0	RW
12	RFLUSH	SPI Rx FIFO flush enable. 0: clear this bit to disable Rx FIFO flushing. 1: set this bit to flush the Rx FIFO. This bit does not clear itself and should be toggled if a single flush is required. If this bit is set, all incoming data is ignored and no interrupts are generated. If set and TIM = 0, a read of the Rx FIFO initiates a transfer.	0x0	RW
11	CON	Continuous transfer enable. 0: DIS. Cleared by user to disable continuous transfer. Each transfer consists of a single 8-bit serial transfer. If valid data exists in the SPI0TX register, a new transfer is initiated after a stall period of 1 serial clock cycle. 1: EN. Set by user to enable continuous transfer. In master mode, the transfer continues until no valid data is available in the Tx register. CS is asserted and remains asserted for the duration of each 8-bit serial transfer until Tx is empty.	0x0	RW
10	LOOPBACK	Loopback enable. 0: cleared by user to be in normal mode. 1: set by user to connect MISO to MOSI and test software.	0x0	RW

Bits	Bit Name	Description	Reset	Access
9	OEN	Slave MISO output enable. 0: clear this bit to disable the output driver on the MISO pin. The MISO pin is open-circuit when this bit is clear. 1: set this bit for MISO to operate as normal.	0x0	RW
8	RXOF	SPIRX overflow overwrite enable. 0: cleared by user, the new serial byte received is discarded. 1: set by user, the valid data in the Rx register is overwritten by the new serial byte received.	0x0	RW
7	ZEN	Transmit zeros enable. 0: clear this bit to transmit the last transmitted value when there is no valid data in the Tx FIFO. 1: set this bit to transmit "0x00" when there is no valid data in the Tx FIFO.	0x0	RW
6	TIM	SPI transfer and interrupt mode. 0: cleared by user to initiate transfer with a read of the SPI0RX register. Interrupt only occurs when Rx is full. 1: set by user to initiate transfer with a write to the SPI0TX register. Interrupt only occurs when Tx is empty.	0x0	RW
5	LSB	LSB first transfer enable. 0: MSB transmitted first 1: LSB transmitted first	0x0	RW
4	WOM	SPI wired Or mode. 1: enables open circuit data output enable. External pull-ups required on data out pins. 0: normal output levels.	0x0	RW
3	CPOL	Serial clock polarity. 0: serial clock idles low 1: serial clock idles high	0x0	RW
2	CPHA	Serial clock phase mode. 1: serial clock pulses at the beginning of each serial bit transfer 0: serial clock pulses at the end of each serial bit transfer	0x0	RW
1	MASEN	Master mode enable. 0: enable slave mode 1: enable master mode	0x0	RW
0	ENABLE	SPI enable. 0: disable the SPI 1: enable the SPI	0x0	RW

SPI DMA Enable Register

Address: 0x4002C014, Reset: 0x0000, Name: SPI0DMA

Table 184. Bit Descriptions for SPI0DMA

Bits	Bit Name	Description	Reset	Access
[15:3]	RESERVED	Reserved.	0x0	R
2	IENRXDMA	Enable receive DMA request. 0: disable RX DMA interrupt 1: enable RX DMA interrupt	0x0	RW
1	IENTXDMA	Enable transmit DMA request. 0: disable TX DMA interrupt 1: enable TX DMA interrupt	0x0	RW
0	ENABLE	Enable DMA for data transfer. Set by user code to start a DMA transfer. Cleared by user code at the end of DMA transfer. This bit needs to be cleared to prevent extra DMA request to the μDMA controller.	0x0	RW

Transfer Byte Count Register

Address: 0x4002C018, Reset: 0x0000, Name: SPI0CNT

Table 185. Bit Descriptions for SPI0CNT

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
[7:0]	COUNT	Transfer byte count. COUNT indicates the number of bytes to be transferred. Count is used in both receive and transmit transfer types. The COUNT value assures that a master mode transfer terminates at the proper time and that 16-bit DMA transfers are byte padded or discarded as required to match odd transfer counts. Reset by clearing SPI0CON[0] or if SPI0CNT is updated.	0x0	RW

REGISTER SUMMARY: SPI1

Table 186. SPI1 Register Summary

Address	Name	Description	Reset	RW
0x40030000	SPI1STA	Status register	0x0000	R
0x40030004	SPI1RX	Receive register	0x0000	R
0x40030008	SPI1TX	Transmit register	0x0000	W
0x4003000C	SPI1DIV	Baud rate selection register	0x0000	RW
0x40030010	SPI1CON	SPI configuration register	0x0000	RW
0x40030014	SPI1DMA	SPI DMA enable register	0x0000	RW
0x40030018	SPI1CNT	Transfer byte count register	0x0000	RW

REGISTER DETAILS: SPI1

Status Register

Address: 0x40030000, Reset: 0x0000, Name: SPI1STA

Table 187. Bit Descriptions for SPI1STA

Bits	Bit Name	Description	Reset	Access
15	RESERVED	Reserved.	0x0	R
14	CSRSG	Detected a rising edge on CS, in CONT mode. This bit causes an interrupt. This can be used to identify the end of an SPI data frame. 0: cleared to 0 when the status register is read 1: set to 1 when there was a rising edge in CS line, when the device was in master mode, continuous transfer, high frequency mode and CSIRQ_EN was asserted.	0x0	RC

Bits	Bit Name	Description	Reset	Access
13	CSFLG	Detected a falling edge on CS, in CONT mode. This bit causes an interrupt. This can be used to identify the start of an SPI data frame. 0: cleared to 0 when the status register is read. 1: set to 1 when there was a falling edge in CS line, when the device was in master mode, continuous transfer, high frequency mode and CSIRQ_EN was asserted.	0x0	RC
12	CSERR	Detected a CS error condition. 0: cleared to 0 when the status register is read. 1: set to 1 when the CS line was de-asserted abruptly, even before the full byte of data was transmitted completely. This bit causes an interrupt.	0x0	RC
11	RXS	SPI Rx FIFO excess bytes present. 0: cleared to 0 when the number of bytes in the FIFO is equal or less than the number in SPI1CON[15:14]. 1: set to 1 when there are more bytes in the Rx FIFO than indicated in the MOD bits in SPI1CON.	0x0	R
[10:8]	RXFSTA	SPI Rx FIFO status. 000: Rx FIFO empty 001: 1 valid byte in FIFO 010: 2 valid bytes in the FIFO 011: 3 valid bytes in the FIFO 100: 4 valid bytes in the FIFO	0x0	R
7	RXOF	SPI Rx FIFO overflow. 0: cleared to 0 when the SPI1STA register is read. 1: set to 1 when the Rx FIFO was already full when new data was loaded to the FIFO. This bit generates an interrupt except when RFLUSH is set in SPI1CON.	0x0	RC
6	RX	SPI Rx IRQ. Not available in DMA mode. Set when a receive interrupt occurs. 0: cleared to 0 when the SPI1STA register is read. 1: set to 1 when TIM in SPI1CON is cleared and the required number of bytes have been received.	0x0	RC
5	TX	SPI Tx IRQ. Status bit. Not available in DMA mode. 0: CLR. Cleared to 0 when the SPI1STA register is read. 1: SET. Set to 1 when a transmit interrupt occurs. This bit is set when TIM in SPI1CON is set and the required number of bytes have been transmitted.	0x0	RC
4	TXUR	SPI Tx FIFO underflow. 0: cleared to 0 when the SPI1STA register is read. 1: set to 1 when a transmit is initiated without any valid data in the Tx FIFO. This bit generates an interrupt except when TFLUSH is set in SPI1CON.	0x0	RC
[3:1]	TXFSTA	SPI Tx FIFO status. 000: Tx FIFO empty 001: 1 valid byte in FIFO 010: 2 valid bytes in FIFO 011: 3 valid bytes in FIFO 100: 4 valid bytes in FIFO	0x0	R
0	IRQ	SPI interrupt status. 0: cleared to 0 after reading SPI1STA. 1: set to 1 when an SPI based interrupt occurs.	0x0	RC

Receive Register

Address: 0x40030004, Reset: 0x0000, Name: SPI1RX

Table 188. Bit Descriptions for SPI1RX

Bits	Bit Name	Description	Reset	Access
[15:8]	DMA_DATA_BYTE_2	8-bit receive buffer. These 8-bits are used only in the DMA mode, where all FIFO accesses happen as half-word access. They return zeros if DMA is disabled.	0x0	R
[7:0]	DATA_BYTE_1	8-bit receive buffer.	0x0	R

Transmit Register

Address: 0x40030008, Reset: 0x0000, Name: SPI1TX

Table 189. Bit Descriptions for SPI1TX

Bits	Bit Name	Description	Reset	Access
[15:8]	DMA_DATA_BYTE_2	8-bit transmit buffer. These 8-bits are used only in the DMA mode, where all FIFO accesses happen as half-word access. They return zeros if DMA is disabled.	0x0	W
[7:0]	DATA_BYTE_1	8-bit transmit buffer.	0x0	W

Baud Rate Selection Register

Address: 0x4003000C, Reset: 0x0000, Name: SPI1DIV

Table 190. Bit Descriptions for SPI1DIV

Bits	Bit Name	Description	Reset	Access
[15:9]	RESERVED	Reserved.	0x0	R
8	CSIRQ_EN	Enable interrupt on every CS edge in CONT and HFM mode. If this bit is set and the SPI module is in continuous mode, any edge on CS generates an interrupt and the corresponding status bits (CSRSG, CSFLG) are asserted. If this bit is clear, no interrupt is generated. This bit has no effect if the SPI is not in continuous mode and high speed mode.	0x0	RW
7	BCRST	Reset mode for CSERR. If this bit is set, the bit counter is reset after a CS error condition and the Cortex is expected to clear the SPI enable bit. If this bit is clear, the bit counter continues from where it stopped. SPI can receive the remaining bits when CS gets asserted and Cortex has to ignore the CSERR interrupt. However, it is strongly recommended to set this bit for a graceful recovery after a CS error.	0x0	RW
6	HFM	High frequency mode. This bit is used for applications using high frequency where the pad introduces a significant delay on the SCL. This can cause a significant enough difference between the serial clock and the data being received on the Rx shift register. In this mode, the Rx shift register is clocked by SCLIN instead of UCLK.	0x0	RW
[5:0]	DIV	SPI clock divider. DIV is the factor used to divide UCLK to generate the serial clock.	0x0	RW

SPI Configuration Register

Address: 0x40030010, Reset: 0x0000, Name: SPI1CON

Table 191. Bit Descriptions for SPI1CON

Bits	Bit Name	Description	Reset	Access
[15:14]	MOD	SPI IRQ mode bits. These bits configure when the Tx/Rx interrupts occur in a transfer. For DMA Rx transfer, these bits should be 00. 00: Tx interrupt occurs when 1 byte has been transferred. Rx interrupt occurs when 1 or more bytes have been received into the FIFO. 01: Tx interrupt occurs when 2 bytes has been transferred. Rx interrupt occurs when 2 or more bytes have been received into the FIFO. 10: Tx interrupt occurs when 3 bytes has been transferred. Rx interrupt occurs when 3 or more bytes have been received into the FIFO. 11: Tx interrupt occurs when 4 bytes has been transferred. Rx interrupt occurs when the Rx FIFO is full, or 4 bytes present.	0x0	RW
13	TFLUSH	SPI Tx FIFO flush enable. 0: clear this bit to disable Tx FIFO flushing. 1: set this bit to flush the Tx FIFO. This bit does not clear itself and should be toggled if a single flush is required. If this bit is left high, then either the last transmitted value or "0x00" is transmitted depending on the ZEN bit. Any writes to the Tx FIFO are ignored while this bit is set.	0x0	RW
12	RFLUSH	SPI Rx FIFO flush enable. 0: clear this bit to disable Rx FIFO flushing. 1: set this bit to flush the Rx FIFO. This bit does not clear itself and should be toggled if a single flush is required. If this bit is set all incoming data is ignored and no interrupts are generated. If set and TIM = 0, a read of the Rx FIFO initiates a transfer.	0x0	RW
11	CON	Continuous transfer enable. 0: DIS. Cleared by user to disable continuous transfer. Each transfer consists of a single 8-bit serial transfer. If valid data exists in the SPI1TX register, a new transfer is initiated after a stall period of 1 serial clock cycle. 1: EN. Set by user to enable continuous transfer. In master mode, the transfer continues until no valid data is available in the Tx register. CS is asserted and remains asserted for the duration of each 8-bit serial transfer until Tx is empty.	0x0	RW
10	LOOPBACK	Loopback enable. 0: cleared by user to be in normal mode. 1: set by user to connect MISO to MOSI and test software.	0x0	RW
9	OEN	Slave MISO output enable. 0: clear this bit to disable the output driver on the MISO pin. The MISO pin is open-circuit when this bit is clear. 1: set this bit for MISO to operate as normal.	0x0	RW
8	RXOF	SPIRX overflow overwrite enable. 0: cleared by user, the new serial byte received is discarded. 1: set by user, the valid data in the Rx register is overwritten by the new serial byte received.	0x0	RW
7	ZEN	Transmit zeros enable. 0: clear this bit to transmit the last transmitted value when there is no valid data in the Tx FIFO. 1: set this bit to transmit 0x00 when there is no valid data in the Tx FIFO.	0x0	RW
6	TIM	SPI transfer and interrupt mode. 0: cleared by user to initiate transfer with a read of the SPIRX register. Interrupt only occurs when Rx is full. 1: set by user to initiate transfer with a write to the SPITX register. Interrupt only occurs when Tx is empty.	0x0	RW

Bits	Bit Name	Description	Reset	Access
5	LSB	LSB first transfer enable. 0: MSB transmitted first 1: LSB transmitted first	0x0	RW
4	WOM	SPI wired Or mode. 0: normal output levels 1: enables open circuit data output enable. External pull-ups required on data out pins	0x0	RW
3	CPOL	Serial clock polarity. 0: serial clock idles low 1: serial clock idles high	0x0	RW
2	CPHA	Serial clock phase mode. 0: serial clock pulses at the end of each serial bit transfer 1: serial clock pulses at the beginning of each serial bit transfer	0x0	RW
1	MASEN	Master mode enable. 0: enable slave mode 1: enable master mode	0x0	RW
0	ENABLE	SPI enable. 0: disable the SPI 1: enable the SPI	0x0	RW

SPI DMA Enable Register

Address: 0x40030014, Reset: 0x0000, Name: SPI1DMA

Table 192. Bit Descriptions for SPI1DMA

Bits	Bit Name	Description	Reset	Access
[15:3]	RESERVED	Reserved.	0x0	R
2	IENRXDMA	Enable receive DMA request. 0: disable RX DMA interrupt 1: enable RX DMA interrupt	0x0	RW
1	IENTXDMA	Enable transmit DMA request. 0: disable TX DMA interrupt 1: enable TX DMA interrupt	0x0	RW
0	ENABLE	Enable DMA for data transfer. Set by user code to start a DMA transfer. Cleared by user code at the end of DMA transfer. This bit needs to be cleared to prevent extra DMA request to the μDMA controller.	0x0	RW

Transfer Byte Count Register

Address: 0x40030018, Reset: 0x0000, Name: SPI1CNT

Table 193. Bit Descriptions for SPI1CNT

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
[7:0]	COUNT	Transfer byte count. COUNT indicates the number of bytes to be transferred. Count is used in both receive and transmit transfer types. The COUNT value assures that a master mode transfer terminates at the proper time and that 16-bit DMA transfers are byte padded or discarded as required to match odd transfer counts. Reset by clearing SPI1CON[0] or if SPI1CNT is updated.	0x0	RW

UART SERIAL INTERFACE

UART FEATURES

- Industry-standard 16,450 UART peripheral
- Support for DMA

UART OVERVIEW

The UART peripheral is a full-duplex universal asynchronous receiver/transmitter (UART), compatible with the industry-standard 16,450. The UART is responsible for converting data between serial and parallel formats. The serial communication follows an asynchronous protocol, supporting various word lengths, stop bits, and parity generation options.

This UART also contains interrupt handling hardware. The UART features a fractional divider that facilitates high accuracy baud rate generation.

Interrupts can be generated from several unique events, such as full/empty data buffer, transfer error detection, and break detection.

UART OPERATION

Serial Communications

An asynchronous serial communication protocol is followed with these options:

- 5 to 8 data bits
- 1, 2, or 1½ stop bits
- None, even, or odd parity
- The baud rate is as follows:

$$Baud\ Rate = UCLK/CDPCLK \div (2 \times 16 \times COMDIV) \div (M + N \div 2048)$$

where:
$COMDIV$ = 1 to 65,536.
M = 1 to 3.
N = 0 to 2047.
$UCLK/CDPCLK$ is the divided 80 MHz clock as configured via CLKCON1[10:8]

All data-words require a start bit and at least one stop bit. This creates a range from seven bits to twelve bits for each word. Transmit operation is initiated by writing to the transmit holding register (COMTX). After a synchronization delay, the data is moved to the internal transmit shift register (TSR), where it is shifted out at a baud (bit) rate equal to the following with start, stop, and parity bits appended as required:

$$UCLK/CDPCLK \div (2 \times 16 \times COMDIV) \div (M + N \div 2048)$$

All data-words begin with a low-going start bit. The transfer of COMTX to the TSR causes the transmit register empty status flag to be set.

The receive operation uses the same data format as the transmit configuration except for the number of stop bits, which is always one. After detection of the start bit, the received word is shifted into the internal receive shift register (RSR). After the appropriate number of bits (including stop bits) are received, the data and any status are updated, and the RSR is transferred to the receive buffer register (COMRX). The receive buffer register full status flag is updated upon the transfer of the received word to this buffer and the appropriate synchronization delay.

A sampling clock equal to 16 times the baud rate is used to sample the data as close to the midpoint of the bit as possible. A receive filter is also present that removes spurious pulses of less than two times the sampling clock period.

Note that data is transmitted and received least significant bit first. This is often not the assumed case by the user. However, it is standard for the protocol.

For power saving purposes, it is possible to disable the system clock to the UART via the CLKCON5[5] register. By default, the clock to the UART is disabled (CLKCON5[5] = 1).

Programmed I/O Mode

In this mode, the software is responsible for moving data to and from the UART. This is typically accomplished by interrupt service routines that respond to the transmit and receive interrupts by either reading or writing data as appropriate. This mode puts certain constraints on the software itself in that the software must respond within a certain time to prevent overflow errors from occurring in the receive channel.

Polling the status flag is processor intensive and not typically used unless the system can tolerate the overhead. Interrupts can be disabled using the COMIEN register.

Writing COMTX when it is not empty or reading COMRX when it is not full produces incorrect results and should not be done. In the former case, COMTX is overwritten by the new word, and the previous word is never transmitted. In the latter case, the previously received word is read again. Both of these errors must be avoided in software by correctly using either interrupts or status register polling. These errors are not detected in hardware.

Enable/Disable Bit

Before the ADuCM320 enters power-down mode, it is recommended to disable the serial interfaces. A bit is provided in the UART control register to disable the UART serial peripheral. This bit disables the clock to the peripheral. When setting this bit, care must be taken in software that no data is being transmitted or received. If this bit is set during communication, the data transfer does not complete; the receive or transmit register contains only part of the data.

Interrupts

The UART peripheral has one interrupt output to the interrupt controller for both Rx and Tx interrupts. The COMIIR register must be read by software to determine the cause of the interrupt. Note that in DMA mode the break interrupt is not available.

In I/O mode when receiving, the interrupt is generated for the following cases:

- COMRX full
- Receive overflow error
- Receive parity error
- Receive framing error
- Break interrupt (UART RxD held low)
- COMTX empty

Buffer Requirements

This UART is double buffered (holding register and shift register).

DMA Mode

In this mode, user code does not move data to and from the UART. DMA request signals going to the external DMA block indicate that the UART is ready to transmit or receive data. These DMA request signals can be disabled in the COMIEN register.

Example Code to Set Up UART Receive DMA Channel

```
void UARTRXDMAINIT(void)
{
NVIC_EnableIRQ(DMA_UART_RX_IRQn);                         // UArt Tx DMA interrupt enable
pADI_UART->COMLCR = COMLCR_WLS_EIGHTBITS | COMLCR_STOP;   // 8 data bits + 1 stop bit
pADI_UART->COMDIV = 0x41;                                 // Set UART baud rate
pADI_UART->COMFBR = COMFBR_FBEN_EN | 0x803;               // DIVM = 1, DIVN = 3
pADI_GP1->GPCON = 0x5;                                    // Configure P1.0/P1.1 for UART
Dma_Init();
pADI_DMA->DMACFG = 0x1;
UARTDMAREAD(uxUARTRXData, 4);                             // Enable DMA mode in DMA controller
pADI_DMA->DMAENSET = 0x20;                                // Enable UART_RX_DMA Channel
pADI_UART->COMIEN = 0x20;                                 // Enable DMA Rx transfers
}

void UARTDMAREAD(unsigned char *pucRX_DMA, unsigned int iNumVals)
{
 DmaDesc Desc;
 // Common configuration of all the descriptors used here
 Desc.ctrlCfg.bits.cycle_ctrl = DMA_BASIC;
 desc.ctrlcfg.bits.next_useburst = 0x0;
 desc.ctrlcfg.bits.r_power = 0;
 Desc.ctrlCfg.Bits.src_prot_ctrl = 0x0;
 Desc.ctrlCfg.Bits.dst_prot_ctrl = 0x0;
 Desc.ctrlCfg.Bits.src_size = DMA_SIZE_BYTE;
 Desc.ctrlCfg.Bits.dst_size = DMA_SIZE_BYTE;
 // Rx primary descriptor
 Desc.srcEndPtr = (unsigned int)(&pADI_UART->COMRX);
 Desc.destEndPtr = (unsigned int)(pucTX_DMA + (iNumVals - 0x1));
 Desc.ctrlCfg.Bits.n_minus_1 = iNumRX - 0x1;
 Desc.ctrlCfg.Bits.src_inc = DMA_SRCINC_NO;
 Desc.ctrlCfg.Bits.dst_inc = DMA_DSTINC_BYTE;
 *Dma_GetDescriptor(UARTRX_C) = Desc;
}
                                                            // UART DMA Rx IRQ handler
void DMA_UART_RX_Int_Handler()
{
 NVIC_DisableIRQ(DMA_UART_RX_IRQn);                         // Clear interrupt source
}
```

Example Code to Set Up UART Transmit DMA Channel

```
void UARTTXDMAINIT(void)
{
 NVIC_EnableIRQ(DMA_UART_TX_IRQn);                          // UART Tx DMA interrupt sources
 pADI_UART->COMLCR = COMLCR_WLS_8BITS + COMLCR_STOP;        // 8 data bits + 1 stop bit
 pADI_UART->COMDIV = 0x41;                                  // Set UART baud rate
 pADI_UART->COMFBR = COMFBR_FBEN_EN | 0x803;                // DIVM = 1, DIVN = 3
 pADI_GP1->GPCON = 0x5;                                     // Configure P1.0/P1.1 for UART
 Dma_Init();
 pADI_DMA->DMACFG = 0x1;                                    // Enable DMA mode in DMA controller
 UARTDMAWRITE(uxUARTTXData, 16);
 pADI_DMA->DMAENSET = 0x10;                                 // Enable UART_TX_DMA channel
 pADI_UART->COMIEN = 0x10;                                  // Enable DMA Tx transfers
}

void UARTDMAWRITE(unsigned char *pucTX_DMA, unsigned int iNumVals)
{
 DmaDesc Desc;

 // Common configuration of all the descriptors used here
 Desc.ctrlCfg.Bits.cycle_ctrl = DMA_BASIC;
 Desc.ctrlCfg.Bits.next_useburst = 0x0;
 Desc.ctrlCfg.Bits.r_power = 0;
 Desc.ctrlCfg.Bits.src_prot_ctrl = 0x0;
 Desc.ctrlCfg.Bits.dst_prot_ctrl = 0x0;
 Desc.ctrlCfg.Bits.src_size = DMA_SIZE_BYTE;
 Desc.ctrlCfg.Bits.dst_size = DMA_SIZE_BYTE;                // Tx primary descriptor

 Desc.srcEndPtr = (unsigned int)(pucTX_DMA + (iNumVals - 0x1));
 Desc.destEndPtr = (unsigned int)(&pADI_UART->COMTX);
 Desc.ctrlCfg.Bits.n_minus_1 = iNumRX - 0x1;
 Desc.ctrlCfg.Bits.src_inc = DMA_SRCINC_BYTE;
 Desc.ctrlCfg.Bits.dst_inc = DMA_DSTINC_NO;
 *Dma_GetDescriptor(UARTTX_C) = Desc;

}

                                                            // UART DMA Tx IRQ handler
void DMA_UART_TX_Int_Handler()
{
 NVIC_DisableIRQ(DMA_UART_TX_IRQn);                         // Clear interrupt source
}
```

REGISTER SUMMARY: UART

Table 194. UART Register Summary

Address	Name	Description	Reset	RW
0x40005000	COMTX	Transmit holding register	0x0000	W
0x40005000	COMRX	Receive buffer register	0x0000	R
0x40005004	COMIEN	Interrupt enable register	0x0000	RW
0x40005008	COMIIR	Interrupt identification register	0x0001	RC
0x4000500C	COMLCR	Line control register	0x0000	RW
0x40005010	COMMCR	Modem control register	0x0000	RW
0x40005014	COMLSR	Line status register	0x0060	RC
0x40005018	COMMSR	Modem status register	0x0000	RC
0x4000501C	COMSCR	Scratch buffer register	0x0000	RW
0x40005024	COMFBR	Fractional baud rate register	0x0000	RW
0x40005028	COMDIV	Baud rate divider register	0x0001	RW

REGISTER DETAILS: UART

Transmit Holding Register

Address: 0x40005000, Reset: 0x0000, Name: COMTX

COMRX and COMTX share the same address while they are implemented as different registers. If these registers are written to, the user accesses the transmit holding register (COMTX). If these registers are read from, the user accesses the receive buffer register (COMRX).

Table 195. Bit Descriptions for COMTX

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
[7:0]	THR	Transmit holding register. This is an 8-bit register to which the user can write the data to be sent. If the ETBEI bit is set in the COMIEN register, an interrupt is generated when COMTX is empty. If user code sets ETBEI while COMTX is already empty, an interrupt is generated immediately.	0x0	W

Receive Buffer Register

Address: 0x40005000, Reset: 0x0000, Name: COMRX

Table 196. Bit Descriptions for COMRX

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
[7:0]	RBR	Receive buffer register. This is an 8-bit register from which the user can read received data. If the ERBFI bit is set in the COMIEN register, an interrupt is generated when this register is fully loaded with the received data via serial input port. If user code sets the ERBFI bit while COMRX is already full, an interrupt is generated immediately.	0x0	R

Interrupt Enable Register

Address: 0x40005004, Reset: 0x0000, Name: COMIEN

COMIEN is the interrupt enable register that is used to configure which interrupt source generates the interrupt. Only the lowest four bits in this register enable interrupts. Bit 4 and Bit 5 enable UART DMA signals. The UART DMA channel and interrupt must be configured in the DMA block.

Table 197. Bit Descriptions for COMIEN

Bits	Bit Name	Description	Reset	Access
[15:6]	RESERVED	Reserved.	0x0	R
5	EDMAR	DMA requests in receive mode. 0: DMA requests disabled 1: DMA requests enabled	0x0	RW

Bits	Bit Name	Description	Reset	Access
4	EDMAT	DMA requests in transmit mode. 0: DMA requests are disabled 1: DMA requests are enabled	0x0	RW
3	EDSSI	Modem status interrupt. Interrupt is generated when any of COMMSR[3:0] are set. 0: interrupt disabled 1: interrupt enabled	0x0	RW
2	ELSI	Rx status interrupt. 0: interrupt disabled 1: interrupt enabled	0x0	RW
1	ETBEI	Transmit buffer empty interrupt. 0: interrupt disabled 1: interrupt enabled	0x0	RW
0	ERBFI	Receive buffer full interrupt. 0: interrupt disabled 1: interrupt enabled	0x0	RW

Interrupt Identification Register

Address: 0x40005008, Reset: 0x0001, Name: COMIIR

Table 198. Bit Descriptions for COMIIR

Bits	Bit Name	Description	Reset	Access
[15:3]	RESERVED	Reserved.	0x0	R
[2:1]	STA	Interrupt status. When NIRQ is low (active low), this indicates an interrupt and the STA bit decoding below is used. 00: modem status interrupt (read COMMSR to clear) 01: transmit buffer empty interrupt (write to COMTX or read COMIIR to clear) 10: receive buffer full interrupt (read COMRX to clear) 11: receive line status interrupt (read COMLSR to clear)	0x0	RC
0	NIRQ	Interrupt flag. 0: interrupt occurred. Source of interrupt indicated in the STA bits. 1: no interrupt occurred.	0x1	RC

Line Control Register

Address: 0x4000500C, Reset: 0x0000, Name: COMLCR

Table 199. Bit Descriptions for COMLCR

Bits	Bit Name	Description	Reset	Access
[15:7]	RESERVED	Reserved.	0x0	R
6	BRK	Set break. 0: force TxD to 0 1: normal TxD operation	0x0	RW
5	SP	Stick parity. Used to force parity to defined values. When set, the parity is based on the following bit settings : EPS = 1 and PEN = 1, parity is forced to 0 EPS = 0 and PEN = 1, parity is forced to 1 EPS = X and PEN = 0, no parity is transmitted 0: Parity is not forced based on EPS and PEN 1: Parity forced based on EPS and PEN	0x0	RW
4	EPS	Parity select. This bit only has meaning if parity is enabled (PEN set). 0: odd parity is transmitted and checked 1: even parity is transmitted and checked	0x0	RW

Bits	Bit Name	Description	Reset	Access
3	PEN	Parity enable. Used to control of the parity bit transmitted and checked. The value transmitted and the value checked are based on the settings of EPS and SP. 0: parity is not transmitted or checked 1: parity is transmitted and checked	0x0	RW
2	STOP	Stop bit. Used to control the number of stop bits transmitted. In all cases, only the first stop bit is evaluated on data received. 0: send 1 stop bit regardless of the word length (WLS). 1: send a number of stop bits based on the word length. Transmit 1.5 stop bits if the word length is 5 bits (WLS = 00), or 2 stop bits if the word length is 6 (WLS = 01), 7 (WLS = 10), or 8 bits (WLS = 11).	0x0	RW
[1:0]	WLS	Word length select. Selects the number of bits per transmission. 00: 5 bits 01: 6 bits 10: 7 bits 11: 8 bits	0x0	RW

Modem Control Register

Address: 0x40005010, Reset: 0x0000, Name: COMMCR

Table 200. Bit Descriptions for COMMCR

Bits	Bit Name	Description	Reset	Access
[15:5]	RESERVED	Reserved.	0x0	R
4	LOOPBACK	Loopback mode. In loop back mode, the SOUT is forced high. The modem signals are also directly connected to the status inputs (RTS to CTS, DTR to DSR, OUT1 to RI, and OUT2 to DCD). 0: normal operation, loopback disabled 1: loopback enabled	0x0	RW
3	OUT2	Output 2. 0: force OUT2 to a Logic 1 1: force OUT2 to a Logic 0	0x0	RW
2	OUT1	Output 1. 0: force OUT1 to a Logic 1 1: force OUT1 to a Logic 0	0x0	RW
1	RTS	Request to send. 0: force RTS to a Logic 1 1: force RTS to a Logic 0	0x0	RW
0	DTR	Data Terminal Ready. 0: force DTR to a Logic 1 1: force DTR to a Logic 0	0x0	RW

Line Status Register

Address: 0x40005014, Reset: 0x0060, Name: COMLSR

Table 201. Bit Descriptions for COMLSR

Bits	Bit Name	Description	Reset	Access
[15:7]	RESERVED		0x0	R
6	TEMT	COMTX and shift register empty status. 0: COMTX has been written to and contains data to be transmitted. Care should be taken not to overwrite its value. 1: COMTX and the transmit shift register are empty and it is safe to write new data to COMTX. Data has been transmitted.	0x1	R
5	THRE	COMTX empty. THRE is cleared when COMRX is read. 0: COMTX has been written to and contains data to be transmitted. Care should be taken not to overwrite its value. 1: COMTX is empty and it is safe to write new data to COMTX. The previous data may not have been transmitted yet and can still be present in the shift register.	0x1	R
4	BI	Break indicator. If set, this bit self clears after COMLSR is read. 0: SIN was not detected to be longer than the maximum word length. 1: SIN was held low for more than the maximum word length.	0x0	RC
3	FE	Framing error. If set, this bit self clears after COMLSR is read. 0: no invalid stop bit was detected. 1: an invalid stop bit was detected on a received word.	0x0	RC
2	PE	Parity error. If set, this bit self clears after COMLSR is read. 0: No parity error was detected 1: A parity error occurred on a received word.	0x0	RC
1	OE	Overrun error. If set, this bit self clears after COMLSR is read. 0: receive data has not been overwritten 1: receive data was overwritten by new data before COMRX was read.	0x0	RC
0	DR	Data ready. This bit is cleared only by reading COMRX. If set, this bit does not self clear. 0: COMRX does not contain new receive data. 1: COMRX contains receive data that should be read.	0x0	RC

Modem Status Register

Address: 0x40005018, Reset: 0x0000, Name: COMMSR

Table 202. Bit Descriptions for COMMSR

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
7	DCD	Data carrier detect. This bit reflects the direct status complement of the DCD pin. 0: DCD is currently logic high 1: DCD is currently logic low	0x0	R
6	RI	Ring indicator. This bit reflects the direct status complement of the DCD pin. 0: RI is currently logic high 1: RI is currently logic low	0x0	R
5	DSR	Data set ready. This bit reflects the direct status complement of the DCD pin. 0: DSR is currently logic high 1: DSR is currently logic low	0x0	R
4	CTS	Clear to send. This bit reflects the direct status complement of the DCD pin. 0: CTS is currently logic high 1: CTS is currently logic low	0x0	R

Bits	Bit Name	Description	Reset	Access
3	DDCD	Delta DCD. If set, this bit self clears after COMMSR is read. 0: DCD has not changed state since COMMSR was last read 1: DCD changed state since COMMSR last read	0x0	R
2	TERI	Trailing edge RI. If set, this bit self clears after COMMSR is read. 0: RI has not changed from 0 to 1 since COMMSR last read 1: RI changed from 0 to 1 since COMMSR last read	0x0	R
1	DDSR	Delta DSR. If set, this bit self clears after COMMSR is read. 0: DSR has not changed state since COMMSR was last read 1: DSR changed state since COMMSR last read	0x0	R
0	DCTS	Delta CTS. If set, this bit self clears after COMMSR is read. 0: CTS has not changed state since COMMSR was last read 1: CTS changed state since COMMSR last read	0x0	R

Scratch Buffer Register

Address: 0x4000501C, Reset: 0x0000, Name: COMSCR

Table 203. Bit Descriptions for COMSCR

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
[7:0]	SCR	Scratch. The scratch register is an 8-bit register used to store intermediate results. The value contained in the scratch register does not affect UART functionality or performance. Only 8 bits of this register are implemented. Bit 15 to Bit 8 are read only and always return 0x00 when read. Writable with any value from 0 to 255. A read returns the last value written.	0x0	RW

Fractional Baud Rate Register

Address: 0x40005024, Reset: 0x0000, Name: COMFBR

Table 204. Bit Descriptions for COMFBR

Bits	Bit Name	Description	Reset	Access
15	FBEN	Fractional baud rate generator enable. The generating of fractional baud rate can be described by the following formula, and the final baud rate of UART operation is calculated as Baud rate = ((UCLK/CDPCLK)/(2 × (M + N/2048)) 16 × COMDIV.	0x0	RW
[14:13]	RESERVED	Reserved.	0x0	R
[12:11]	DIVM	Fractional baud rate M divide bits 1 to 3. This bit should not be 0.	0x0	RW
[10:0]	DIVN	Fractional baud rate N divide bits 0 to 2047.	0x0	RW

Baud Rate Divider Register

Address: 0x40005028, Reset: 0x0001, Name: COMDIV

Table 205. Bit Descriptions for COMDIV

Bits	Bit Name	Description	Reset	Access
[15:0]	DIV	Baud rate divider. The COMDIV register should not be 0, which is not specified. The range of allowed DIV values is from 1 to 65535.	0x1	RW

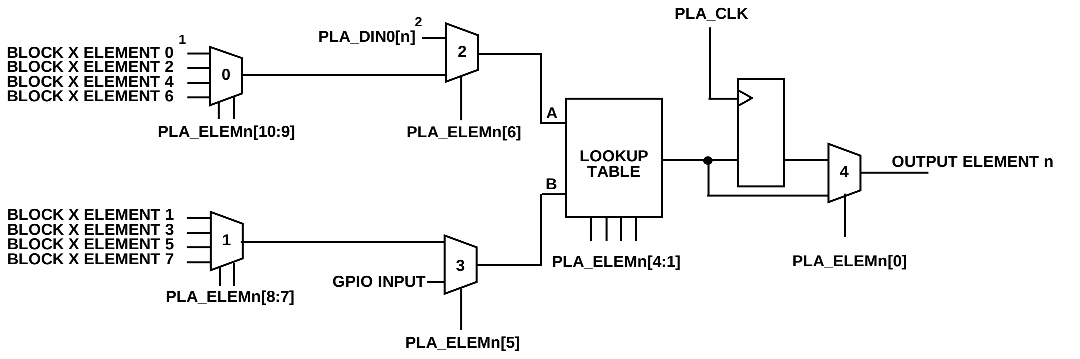
PLA

PLA FEATURES

The ADuCM320 integrates a fully programmable logic array (PLA) that consists of four independent but interconnected PLA blocks. Each block consists of eight PLA elements: Block X Element 0 to Block X Element 7, where X is the block number. Each ADuCM320 has four blocks, resulting in a total of 32 PLA elements: Element 0 to Element 31.

PLA OVERVIEW

Each PLA element contains a two-input lookup table that can be configured to generate any logic output function based on two inputs and a flip-flop.



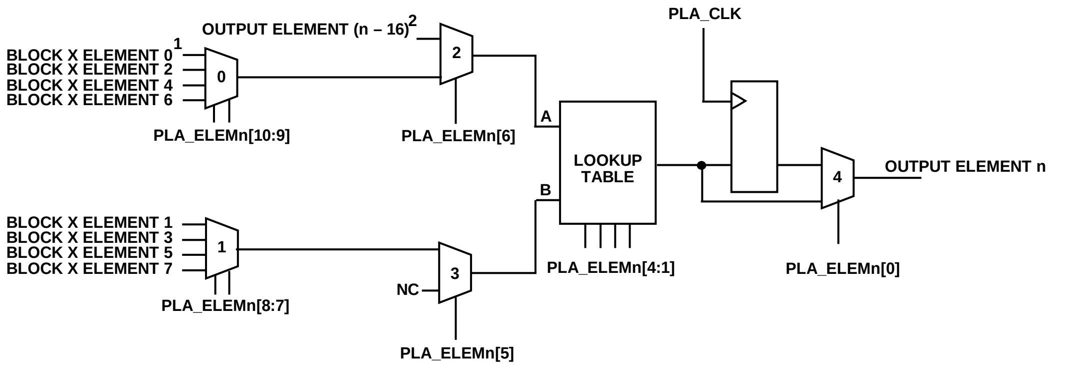
WHERE:
BLOCK X IS BLOCK 0 OR BLOCK 1.
PLA_ELEMn IS THE MMR CONTROLING ELEMENT n, n = 0 TO 15.
NC = NO CONNECTION.

[1]THE FIRST SELECTION OF MUX0 IS THE FEEDBACK FROM BLOCK X ELEMENT 0, WHERE X IS THE NUMBER OF THE CURRENT BLOCK. IF THE FIRST ELEMENT IN THE BLOCK IS BEING CONFIGURED, THEN THE FEEDBACK COMES FROM ANOTHER BLOCK. SEE THE INTERBLOCK CONNECTION DIAGRAM FOR MORE DETAILS.

[2]BLOCK 0 AND BLOCK 1 ARE SET IN THE CORRESPONDING BIT IN THE PLA_DIN0 MMR.

11176-124

Figure 26. PLA Element: Block 0 and Block 1



WHERE:
BLOCK X IS BLOCK 2 OR BLOCK 3.
PLA_ELEMn IS THE MMR CONTROLING ELEMENT n, n = 16 TO 31.
NC = NO CONNECTION.

[1]THE FIRST SELECTION OF MUX0 IS THE FEEDBACK FROM BLOCK X ELEMENT 0, WHERE X IS THE NUMBER OF THE CURRENT BLOCK. IF THE FIRST ELEMENT IN THE BLOCK IS BEING CONFIGURED, THEN THE FEEDBACK COMES FROM ANOTHER BLOCK. SEE THE INTERBLOCK CONNECTION DIAGRAM.

[2]FOR BLOCK 2 AND BLOCK 3 THE INPUT COMES FROM THE OUTPUT OF ELEMENT(n – 16), WHERE n IS THE NUMBER OF THE ELEMENT BEING CONFIGURED. FOR EXAMPLE, FOR ELEMENT 25 THE INPUT TO MUX 2 COMES FROM ELEMENT 9. THIS ALLOWS GPIO INPUTS TO BE INDIRECTLY CONNECTED TO ELEMENTS IN BLOCK 2 AND BLOCK 3.

11176-125

Figure 27. PLA Element: Block 2 and Block 3

In total, 28 GPIO pins are available on each ADuCM320 for the PLA. These include 14 input pins and 14 output pins, which must be configured in the GPxCON register as PLA pins before using the PLA.

PLA OPERATION

The PLA is configured via a set of user MMRs. The output(s) of the PLA can be routed to the internal interrupt system, to the PLA_DOUTx MMRs, or to any of the 14 PLA output pins.

The GPIO inputs to the PLA are always connected to their corresponding elements regardless of the setting in GPxCON. This means that a pin could be used as both an output and an input to the PLA at the same time.

A PLA block can have several clock sources for its output flip-flops, or the flip-flops can be individually bypassed. All output flip-flops in the same block that are not bypassed share the same clock source. The configuration of the clock sources can be found in the PLA clock select register (PLA_CLK).

Each PLA element in a block can be connected to other elements in the same block by configuring the output of Mux 0 and Mux 1. The configuration of these two multiplexers can be found in the PLA_ELEMn configuration register. A complete list of the possible connections is given in Table 207 and Table 208.

The four blocks can be interconnected as follows:

- Output of Element 7 (Block 0 Element 7) can be fed back to the Input 0 of Mux 0 of Element 8 (Block 1 Element 0).
- Output of Element 15 (Block 1 Element 7) can be fed back to Input 0 of Mux 0 of Element 16 (Block 2 Element 0).
- Output of Element 23 (Block 2 Element 7) can be fed back to the Input 0 of Mux 0 of Element 24 (Block 3 Element 0).
- Output of Element 31 (Block 3 Element 7) can be fed back to Input 0 of Mux 0 of Element 0 (Block 0 Element 0).

See Figure 28 for more information.

There are four interrupts available for the PLA. These can be configured to trigger on the output of any element using the PLA_IRQ0 and PLA_IRQ1 registers. The interrupts are active high; therefore, the interrupts continue to be triggered until the output of the element goes low or until the IRQ is disabled. If an active low interrupt is required, an extra element must be configured as an inverter and then the interrupt must be configured to monitor the output of this new element. If an edge triggered interrupt is required, two extra elements must be used and configured as an edge detector (($\overline{A}$) AND A).

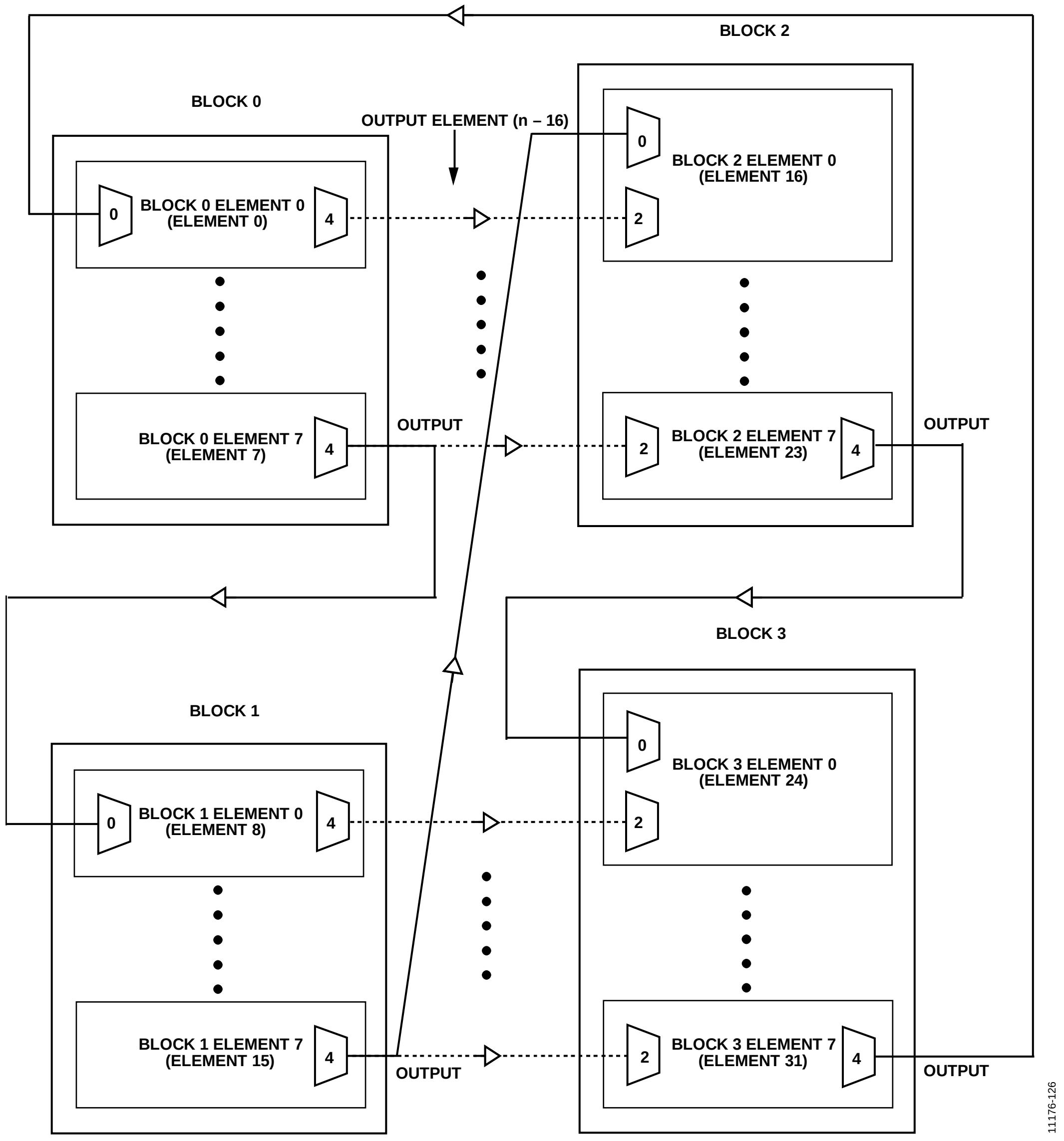


Figure 28. PLA Interblock Connections

Table 206. Element GPIO Input/Output

PLA Block 0			PLA Block 1			PLA Block 2		PLA Block 3	
Element	Input	Output	Element	Input	Output	Element	Output	Element	Output
0	P0.0		8	P2.0		16		24	
1	P0.1		9			17		25	
2	P0.2	P0.4	10	P2.2	P1.4	18	P2.4	26	P3.4
3	P0.3	P0.5	11		P1.5	19		27	P3.5
4	P1.0	P0.6	12	P3.0	P1.6	20	P2.6	28	
5	P1.1	P0.7	13	P3.1	P1.7	21	P2.7	29	P3.7
6	P1.2		14	P3.2		22		30	
7	P1.3		15	P3.3		23		31	

Table 207. Mux 0 Feedback Configuration

PLA_ELEMn [10:9]	PLA_ELEM0	PLA_ELEM1 to PLA_ELEM7	PLA_ELEM8	PLA_ELEM9 to PLA_ELEM15	PLA_ELEM16	PLA_ELEM17 to PLA_ELEM23	PLA_ELEM24	PLA_ELEM25 to PLA_ELEM31
00	Element 31	Element 0	Element 7	Element 8	Element 15	Element 16	Element 23	Element 24
01	Element 2	Element 2	Element 10	Element 10	Element 18	Element 18	Element 26	Element 26
10	Element 4	Element 4	Element 12	Element 12	Element 20	Element 20	Element 28	Element 28
11	Element 6	Element 6	Element 14	Element 14	Element 22	Element 22	Element 30	Element 30

Table 208. Mux 1 Feedback Configuration

PLA_ELEMn [8:7]	PLA_ELEM0	PLA_ELEM1 to PLA_ELEM7	PLA_ELEM8	PLA_ELEM9 to PLA_ELEM15	PLA_ELEM16	PLA_ELEM17 to PLA_ELEM23	PLA_ELEM24	PLA_ELEM25 to PLA_ELEM31
00	Element 1	Element 1	Element 9	Element 9	Element 17	Element 17	Element 25	Element 25
01	Element 3	Element 3	Element 11	Element 11	Element 19	Element 19	Element 27	Element 27
10	Element 5	Element 5	Element 13	Element 13	Element 21	Element 21	Element 29	Element 29
11	Element 7	Element 7	Element 15	Element 15	Element 23	Element 23	Element 31	Element 31

Table 209. Lookup Table Configuration

PLA_ELEMn[4:1]	Function
0000	0
0001	A NOR B
0010	$\overline{A}$ AND B
0011	$\overline{A}$
0100	A AND $\overline{B}$
0101	$\overline{B}$
0110	A XOR B
0111	A NAND B
1000	A AND B
1001	A EXNOR B
1010	B
1011	$\overline{A}$ OR B
1100	A
1101	A OR $\overline{B}$
1110	A OR B
1111	1

REGISTER SUMMARY: PLA

Table 210. PLA Register Summary

Address	Name	Description	Reset	RW
0x40005800	PLA_ELEMn	ELEMx configuration register.	0x0000	RW
0x40005880	PLA_CLK	PLA clock select.	0x0000	RW
0x40005884	PLA_IRQ0	Interrupt register for Block 0 and Block 1.	0x0000	RW
0x40005888	PLA_IRQ1	Interrupt register for Block 2 and Block 3.	0x0000	RW
0x4000588C	PLA_ADC	ADC configuration register.	0x0000	RW
0x40005890	PLA_DIN0	Data input for Block 0 and Block 1.	0x0000	RW
0x40005898	PLA_DOUT0	Data output for Block 0 and Block 1.	0x0000	R
0x4000589C	PLA_DOUT1	Data output for Block 2 and Block 3.	0x0000	R
0x400058A0	PLA_LCK	Write lock register. Can only be set once every reset.	0x0000	RW1S

REGISTER DETAILS: PLA

ELEMx Configuration Register

Address: 0x40005800 to 0x4000587C (Increments of 0x4), Reset: 0x0000, Name: PLA_ELEMn

Table 211. Bit Descriptions for PLA_ELEMn

Bits	Bit Name	Description	Reset	Access
[15:11]	RESERVED	Not used.	0x00	Reserved
[10:9]	MUX0	Even element feedback selection (in respective block). 00: feedback from Element 0 (all except Element 0)/input from other block (Element 0 only) 01: feedback from Element 2 10: feedback from Element 4 11: feedback from Element 6	0x0	RW
[8:7]	MUX1	Odd element feedback selection (in respective block). 00: feedback from Element 1 01: feedback from Element 3 10: feedback from Element 5 11: feedback from Element 7	0x0	RW
6	MUX2	Select between corresponding bit from PLA_DINx register or even feedback mux. 0: PLA_DINx input 1: even feedback mux	0x0	RW
5	MUX3	Select between GPIO Bus input and odd feedback input (for Element 16 to Element 31, odd feedback is always selected). 0: odd feedback mux 1: GPIO input	0x0	RW
[4:1]	TBL	Bit 4, Bit 3, Bit 2, Bit 1 configures output for {mux2_out, mux3_out} = 11, 10, 01, 00 respectively. 0000: 0. 0001: NOR. 0010: B and not A. 0011: NOT A. 0100: A and not B. 0101: Not B. 0110: EXOR. 0111: NAND. 1000: AND. 1001: EXNOR. 1010: B. 1011: B or not A. 1100: A. 1101: A or not B. 1110: OR 1111: 1.	0x0	RW

Bits	Bit Name	Description	Reset	Access
0	MUX4	Select or bypass flip-flop output. 0: FF output 1: bypass output	0x0	RW

PLA Clock Select Register

Address: 0x40005880, Reset: 0x0000, Name: PLA_CLK

Table 212. Bit Descriptions for PLA_CLK

Bits	Bit Name	Description	Reset	Access
15	RESERVED	Not used.	0x0	Reserved
[14:12]	BLOCK3	Clock select for Block 3. 000: GPIO clock on P0.3 001: GPIO clock on P1.1 010: GPIO clock on P2.0 011: HCLK 100: MOSC (16 MHz) 101: Timer 0 110: Timer 2 111: KOSC (32 kHz)	0x0	RW
11	RESERVED	Not used.	0x0	Reserved
[10:8]	BLOCK2	Clock select for Block 2. 000: GPIO clock on P0.3 001: GPIO clock on P1.1 010: GPIO clock on P2.0 011: HCLK 100: MOSC (16 MHz) 101: Timer 0 110: Timer 2 111: KOSC (32 kHz)	0x0	RW
7	RESERVED	Not used.	0x0	Reserved
[6:4]	BLOCK1	Clock select for Block 1. 000: GPIO clock on P0.3 001: GPIO clock on P1.1 010: GPIO clock on P2.0 011: HCLK 100: MOSC (16 MHz) 101: Timer 0 110: Timer 2 111: KOSC (32 kHz)	0x0	RW
3	RESERVED	Not used.	0x0	Reserved
[2:0]	BLOCK0	Clock select for Block 0. 000: GPIO clock on P0.3 001: GPIO clock on P1.1 010: GPIO clock on P2.0 011: HCLK 100: MOSC (16 MHz) 101: Timer 0 110: Timer 2 111: KOSC (32 kHz)	0x0	RW

Interrupt Register for Block 0 and Block 1

Address: 0x40005884, Reset: 0x0000, Name: PLA_IRQ0

Table 213. Bit Descriptions for PLA_IRQ0

Bits	Bit Name	Description	Reset	Access
[15:13]	RESERVED	Not used.	0x0	Reserved
12	IRQ1_EN	IRQ1 enable. 0: disable IRQ1 interrupt 1: enable IRQ1 interrupt	0x0	RW
[11:8]	IRQ1_SRC	IRQ1 source select (Elements 0 to Element 15). 4-bit value corresponds to element number (e.g. 1011 selects Element 11).	0x0	RW
[7:5]	RESERVED	Not used.	0x0	Reserved
4	IRQ0_EN	IRQ0 enable. 0: disable IRQ0 interrupt 1: enable IRQ0 interrupt	0x0	RW
[3:0]	IRQ0_SRC	IRQ0 source select (Element 0 to Element 15). 4-bit value corresponds to element number (for example, 1011 selects Element 11).	0x0	RW

Interrupt Register for Block 2 and Block 3

Address: 0x40005888, Reset: 0x0000, Name: PLA_IRQ1

Table 214. Bit Descriptions for PLA_IRQ1

Bits	Bit Name	Description	Reset	Access
[15:13]	RESERVED	Not used.	0x0	Reserved
12	IRQ3_EN	IRQ3 enable. 0: disable IRQ3 interrupt 1: enable IRQ3 interrupt	0x0	RW
[11:8]	IRQ3_SRC	IRQ3 source select (Elements 16 to Element 31) Element number corresponds to 4-bit value + 16 (for example, 1011 selects Element 27).	0x0	RW
[7:5]	RESERVED	Not used.	0x0	Reserved
4	IRQ2_EN	IRQ2 enable. 0: disable IRQ2 interrupt 1: enable IRQ2 interrupt	0x0	RW
[3:0]	IRQ2_SRC	IRQ2 source select (Elements 16 to Element 31). Element number corresponds to 4-bit value + 16 (for example, 1011 selects Element 27).	0x0	RW

ADC Configuration Register

Address: 0x4000588C, Reset: 0x0000, Name: PLA_ADC

Table 215. Bit Descriptions for PLA_ADC

Bits	Bit Name	Description	Reset	Access
[15:6]	RESERVED	Not used.	0x000	Reserved
5	CONVST_EN	Bit to enable ADC start convert from PLA. 0: disable 1: enable	0x0	RW
[4:0]	CONVST_SRC	Element for ADC start convert source. The binary value corresponds to the element number. For example, Element 23 is 10111.	0x00	RW

Data Input for Block 0 and Block 1 Register

Address: 0x40005890, Reset: 0x0000, Name: PLA_DIN0

Table 216. Bit Descriptions for PLA_DIN0

Bits	Bit Name	Description	Reset	Access
[15:0]	DIN	Input bit to Element 15 to Element 0.	0x0	RW

Data Output for Block 0 and Block 1 Register

Address: 0x40005898, Reset: 0x0000, Name: PLA_DOUT0

Table 217. Bit Descriptions for PLA_DOUT0

Bits	Bit Name	Description	Reset	Access
[15:0]	DOUT	Output bit from Element 15 to Element 0.	0x0	R

Data Output for Block 2 and Block 3 Register

Address: 0x4000589C, Reset: 0x0000, Name: PLA_DOUT1

Table 218. Bit Descriptions for PLA_DOUT1

Bits	Bit Name	Description	Reset	Access
[15:0]	DOUT	Output bit from Element 31 to Element 16.	0x0	R

Write Lock Register

Address: 0x400058A0, Reset: 0x0000, Name: PLA_LCK

This register can only be set once every reset.

Table 219. Bit Descriptions for PLA_LCK

Bits	Bit Name	Description	Reset	Access
[15:1]	RESERVED	Not used.	0x0000	Reserved
0	LOCK	Set to disable writing to registers. 0: writing to registers allowed 1: writing to registers disabled	0x0	RW1S

GENERAL-PURPOSE TIMERS

GENERAL-PURPOSE TIMERS FEATURES

- Three identical general-purpose, 16-bit count-up/count-down timers
  - Timer 0, Timer 1, and Timer 2
- Clocked from five different clock sources
  - Peripheral clock (PCLK)
  - 80 MHz system clock (HCLK)
  - 32 kHz internal oscillator (LFOSC)
  - 16 MHz external crystal (HFXTAL) or internal 16 MHz oscillator (HFOSC), depending on the value in CLKCON0[11].
- Clock sources can be scaled down using a prescaler 16, 256, or 32768. Additionally, two of the clocks can be scaled down using a prescaler of 4, and the other two clock sources can be used directly (prescaler of 1).
- Two modes
  - Free running
  - Periodic
- Capture events feature
  - Capability to capture 15 different events on each timer

GENERAL-PURPOSE TIMERS BLOCK DIAGRAM

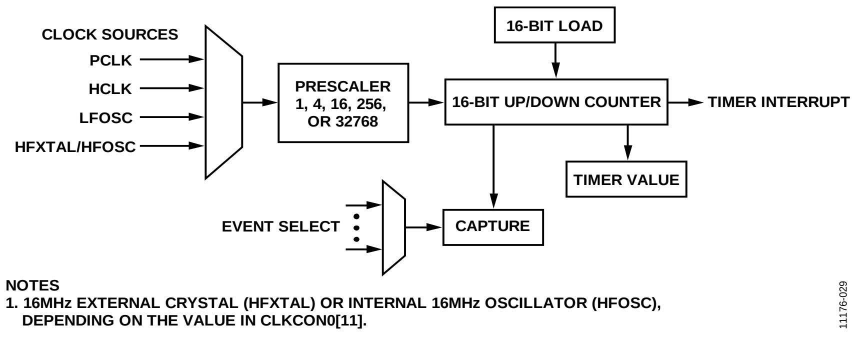

CLOCK SOURCES
PCLK
HCLK
LFOSC
HFXTAL/HFOSC
PRESCALER 1, 4, 16, 256, OR 32768
16-BIT LOAD
16-BIT UP/DOWN COUNTER
TIMER INTERRUPT
TIMER VALUE
EVENT SELECT
CAPTURE
NOTES
1. 16MHz EXTERNAL CRYSTAL (HFXTAL) OR INTERNAL 16MHz OSCILLATOR (HFOSC), DEPENDING ON THE VALUE IN CLKCON0[11].
11176-029

Figure 29. General-Purpose Timers Block Diagram

GENERAL-PURPOSE TIMERS OVERVIEW

Timer 0, Timer 1, and Timer 2 are three identical general-purpose, 16-bit count-up/count-down timers. They can be clocked from five different clock sources:

- PCLK
- HCLK
- 32 kHz internal oscillator (LFOSC)
- 16 MHz external crystal (HFXTAL) or internal 16 MHz oscillator (HFOSC), depending on the value in CLKCON0[11].

The clock sources can be scaled down using a prescaler of 1, 4, 16, 256, or 32768.

The timers can be either free running or periodic.

- In free running mode, the counter decrements from full scale to zero scale or increments from zero scale to full scale and then restarts.
- In periodic mode, the counter decrements or increments from the value in the load register (TxLD MMR, where x is 0 for Timer 0, 1 for Timer 1, and 2 for Timer 2) until zero scale or full scale is reached and then restarts at the value stored in the load register.

The value of a counter can be read at any time by accessing its value register (TxVAL).

The TxCON register selects the timer mode, configures the clock source, selects count-up/count-down, starts the counter, and controls the event capture function.

An interrupt signal is generated each time the value of the counter reaches 0 when counting down, or each time the counter value reaches the maximum value when counting up. An IRQ can be cleared by writing 1 to the time clear interrupt register of that particular timer (TxCLRI).

In addition, Timer 0, Timer 1, and Timer 2 have a capture register (TxCAP) that is triggered by a selected IRQ source initial assertion. When triggered, the current timer value is copied to TxCAP, and the timer continues to run. This feature can be used to determine the assertion of an event with increased accuracy.

GENERAL-PURPOSE TIMERS OPERATION

Free Running Mode

In free running mode, the timer is started by setting the enable bit (TxCON[4]) to 1 and the MOD bit (TxCON[3]) to 0. The timer increments from zero scale/full scale to full scale/zero scale if counting up/down. Full scale is $2^{16} - 1$ or 0xFFFF in binary format. Upon reaching full scale (or zero scale), a timeout interrupt occurs and TxSTA[0] is set. To clear the timer interrupt, user code must write 1 to TxCLRI[0]. If TxCON[7] is set, the timer keeps counting and reloads when the TxCLRI register is written.

Periodic Mode

In periodic mode, the initial TxLD value should be loaded before starting the timer by setting the enable bit (TxCON[4]) to 1. The timer value either increments from the value in TxLD to full scale or decrements from the value in TxLD to zero scale, depending on the TxCON[2] settings (count up/down). Upon reaching full scale or zero scale, the timer generates an interrupt. The TxLD is reloaded into TxVAL, and the timer continues counting up or down. The timer should be disabled prior to changing the TxCON or TxLD register. If the TxLD register is changed while the timer is being loaded, undefined results may occur. By default, the counter is reloaded automatically when generating the interrupt signal. If TxCON[7] is set to 1, the counter is also reloaded when user code writes TxCLRI. This allows user changes to the TxLD to take effect immediately instead of waiting until the next timeout.

The timer interval is calculated as follows:

If the timer is set to count down,

$$Interval = (TxLD \times Prescaler)/Source\ Clock$$

For example, if TxLD = 0x100, prescaler = 4, and clock source = UCLK, the interval is 12.8 µs (where UCLK = 80 MHz).

If the timer is set to count up,

$$Interval = ((Full\ Scale - TxLD) \times Prescaler)/Source\ Clock$$

Asynchronous Clock Source

Timers are started by setting the enable bit (TxCON[4]) to 1 in the control register of the corresponding timer.

However, when the timer clock source is HFXTAL or LFOSC, some precautions must be taken:

- The control register (TxCON) should not be written if TxSTA[6] is set. Therefore, TxSTA should be read prior to configuring the control register (TxCON). When TxSTA[6] is cleared, the register can be modified. This ensures that synchronizing the timer control between the processor and timer clock domains is complete. TxSTA[6] is the timer busy status bit.
- After clearing the interrupt in TxCLRI, it must be ensured that the register write has fully completed before returning from the interrupt handler. Use the data synchronization barrier (DSB) instruction if necessary and check that TxSTA[7] = 0.

  ```
  __asm void asmDSB()
  {
  nop
  DSB
  BX LR
  }
  ```
- The value of a counter can be read at any time by accessing its value register (TxVAL). In an asynchronous configuration, TxVAL should always be read twice. If the two readings are different, it should be read a third time to determine the correct value.

TxSTA should be read prior to writing to any timer register after setting or clearing the enable bit. When TxSTA[7] is cleared, registers can be modified. This ensures that the timer has completed synchronization between the processor and timer clock domains. The typical synchronization time is two timer clock periods.

The TxCON register enables the counter, selects the mode, selects the prescale value, and controls the event capture function.

Capture Event Function

There are several interrupt events that can be captured by the general-purpose timers. These events are shown in Table 220. Any one of the events associated with a general-purpose timer can cause a capture of the 16-bit TxVAL register into the 16-bit TxCAP register. TxCON has a 4-bit field that can be used to select which event to capture.

When the selected interrupt event occurs, the TxVAL register is copied into the TxCAP register. TxSTA[1] is set, indicating that a capture event is pending. The bit is cleared by writing 1 to TxCLRI[1]. The TxCAP register also holds its value and cannot be overwritten until a 1 is written to TxCLRI[1].

Table 220. Capture Event Function

Event Select Bits (TxCON.EVENT), TxCON[11:8]	T0 Capture Source	T1 Capture Source	T2 Capture Source
0000	Wake-up timer	External Interrupt 4	External Interrupt 7
0001	External Interrupt 0	External Interrupt 5	External Interrupt 8
0010	External Interrupt 1	Reserved	SPI1
0011	External Interrupt 2	Flash controller	I2C0 slave
0100	Reserved	UART	I2C0 master
0101	External Interrupt 4	SPI0	PLA 2
0110	External Interrupt 5	PLA 0	PLA 3
0111	Reserved	PLA 1	PWM trip
1000	External Interrupt 7	DMA error	PWM0
1001	External Interrupt 8	DMA done (any)	PWM1
1010	Watchdog timer	Reserved	PWM2
1011	Reserved	Reserved	PWM3
1100	Reserved	Reserved	Low Voltage Analog Die Interrupt 1
1101	Low Voltage Analog Die Interrupt 0	I2C1 slave	External Interrupt 0
1110	MDIO	I2C1 master	External Interrupt 1
1111	GP Timer 1	GP Timer 2	GP Timer 1

REGISTER SUMMARY: GENERAL-PURPOSE TIMER 0

Table 221. Timer 0 Register Summary

Address	Name	Description	Reset	RW
0x40000000	T0LD	16-bit load value register	0x0000	RW
0x40000004	T0VAL	16-bit timer value register	0x0000	R
0x40000008	T0CON	Control register	0x000A	RW
0x4000000C	T0CLRI	Clear interrupt register	0x0000	W
0x40000010	T0CAP	Capture register	0x0000	R
0x4000001C	T0STA	Status register	0x0000	R

REGISTER DETAILS: GENERAL-PURPOSE TIMER 0

16-Bit Load Value Register

Address: 0x40000000, Reset: 0x0000, Name: T0LD

Table 222. Bit Descriptions for T0LD

Bits	Bit Name	Description	Reset	Access
[15:0]	LOAD	Load value. The up/down counter is periodically loaded with this value if periodic mode is selected (T0CON[3]=1). LOAD writes during up/down counter timeout events are delayed until the event has passed.	0x0	RW

16-Bit Timer Value Register

Address: 0x40000004, Reset: 0x0000, Name: T0VAL

Table 223. Bit Descriptions for T0VAL

Bits	Bit Name	Description	Reset	Access
[15:0]	VAL	Current count. Reflects the current up/down counter value. Value delayed two PCLK cycles due to clock synchronizers.	0x0	R

Control Register

Address: 0x40000008, Reset: 0x000A, Name: T0CON

Table 224. Bit Descriptions for T0CON

Bits	Bit Name	Description	Reset	Access
[15:13]	RESERVED	Reserved.	0x0	R
12	EVENTEN	Event select. Used to enable and disabling the capture of events. Used in conjunction with the EVENT select range: when a selected event occurs the current value of the up/down counter is captured in T0CAP. 0: Events will not be captured. 1: Events will be captured.	0x0	RW
[11:8]	EVENT	Event select range. Timer event select range (0 to 15).	0x0	RW
7	RLD	Reload control. RLD is only used for periodic mode; this bit allows the user to select whether the Up/Down counter should be reset only on a timeout event or also when T0CLRI[0] is set. 1: resets the up/down counter when T0CLRI[0] is set 0: up/down counter is only reset on a timeout event	0x0	RW
[6:5]	CLK	Clock select. Used to select a timer clock from the four available clock sources. 00: PCLK. 01: HCLK. 10: LFOSC. 32 KHz OSC. 11: HFXTAL. 16 MHz OSC or XTAL, dependent on the value in CLKCON0[11].	0x0	RW
4	ENABLE	Timer enable. Used to enable and disable the timer. Clearing this bit resets the timer, including the T0VAL register. 0: DIS. Timer is disabled (default). 1: EN. Timer is enabled.	0x0	RW

Bits	Bit Name	Description	Reset	Access
3	MOD	Timer mode. This bit is used to control whether the timer runs in periodic or free running mode. In periodic mode the up/down counter starts at the defined LOAD value (T0LD); in free running mode, the up/down counter starts at 0x0000 or 0xFFFF depending on whether the timer is counting up or down. 0: FREERUN. Timer runs in free running mode. 1: PERIODIC. Timer runs in periodic mode (default).	0x1	RW
2	UP	Count up. Used to control whether the timer increments (counts up) or decrements (counts down) the up/down counter. 0: DIS. Timer is set to count down (default). 1: EN. Timer is set to count up.	0x0	RW
[1:0]	PRE	Prescaler. Controls the prescaler division factor applied to the timer's selected clock. If CLK Source 0 (PCLK) or CLK Source 1 (HCLK) is selected, then Prescaler Value 0 means divide by 4, else, it means divide by 1. 00: source clock/[1 or 4] 01: source clock/16 10: source clock/256 11: source clock/32,768	0x2	RW

Clear Interrupt Register

Address: 0x4000000C, Reset: 0x0000, Name: T0CLRI

Table 225. Bit Descriptions for T0CLRI

Bits	Bit Name	Description	Reset	Access
[15:2]	RESERVED	Reserved.	0x0	R
1	CAP	Clear captured event interrupt. This bit is used to clear a capture event interrupt. 0: no effect 1: clear the capture event interrupt	0x0	W1C
0	TMOUT	Clear timeout interrupt. This bit is used to clear a timeout interrupt. 0: no effect 1: clears the timeout interrupt	0x0	W1C

Capture Register

Address: 0x40000010, Reset: 0x0000, Name: T0CAP

Table 226. Bit Descriptions for T0CAP

Bits	Bit Name	Description	Reset	Access
[15:0]	CAP	16-bit captured value. T0CAP holds its value until T0CLRI[1] is set by user code. T0CAP is not overwritten even if another event occurs without writing to the T0CLRI[1].	0x0	R

Status Register

Address: 0x4000001C, Reset: 0x0000, Name: T0STA

Table 227. Bit Descriptions for T0STA

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
7	PDOK	T0CLRI synchronization. This bit is set automatically when the user sets T0CLRI[0] = 1. It is cleared automatically when the clear interrupt request has crossed clock domains and taken effect in the timer clock domain. 0: CLR. The interrupt is cleared in the timer clock domain. 1: SET. T0CLRI[0] is being updated in the timer clock domain.	0x0	R
6	BUSY	Timer busy. This bit informs the user that a write to T0CON is still crossing into the timer clock domain. This bit should be checked after writing T0CON and further writes should be suppressed until this bit is cleared. 0: CLR. Timer ready to receive commands to T0CON. 1: SET. Timer not ready to receive commands to T0CON.	0x0	R
[5:2]	RESERVED	Reserved.	0x0	R
1	CAP	Capture event pending. 0: CLR. No capture event is pending. 1: SET. A capture event is pending.	0x0	R
0	TMOUT	Timeout event occurred. This bit set automatically when the value of the counter reaches zero while counting down or reaches full scale when counting up. This bit is cleared when T0CLRI[0] is set by the user. 0: CLR. No timeout event has occurred. 1: SET. A timeout event has occurred.	0x0	R

REGISTER SUMMARY: GENERAL-PURPOSE TIMER 1

Table 228. Timer 1 Register Summary

Address	Name	Description	Reset	RW
0x40000400	T1LD	16-bit load value register	0x0000	RW
0x40000404	T1VAL	16-bit timer value register	0x0000	R
0x40000408	T1CON	Control register	0x000A	RW
0x4000040C	T1CLRI	Clear interrupt register	0x0000	W
0x40000410	T1CAP	Capture register	0x0000	R
0x4000041C	T1STA	Status register	0x0000	R

REGISTER DETAILS: GENERAL-PURPOSE TIMER 1

16-Bit Load Value Register

Address: 0x40000400, Reset: 0x0000, Name: T1LD

Table 229. Bit Descriptions for T1LD

Bits	Bit Name	Description	Reset	Access
[15:0]	LOAD	Load value. The up/down counter is periodically loaded with this value if periodic mode is selected (T1CON[3]=1). LOAD writes during up/down counter timeout events are delayed until the event has passed.	0x0	RW

16-Bit Timer Value Register

Address: 0x40000404, Reset: 0x0000, Name: T1VAL

Table 230. Bit Descriptions for T1VAL

Bits	Bit Name	Description	Reset	Access
[15:0]	VAL	Current count. Reflects the current up/down counter value. Value delayed two PCLK cycles due to clock synchronizers.	0x0	R

Control Register

Address: 0x40000408, Reset: 0x000A, Name: T1CON

Table 231. Bit Descriptions for T1CON

Bits	Bit Name	Description	Reset	Access
[15:13]	RESERVED	Reserved.	0x0	R
12	EVENTEN	Event select. Used to enable and disabling the capture of events. Used in conjunction with the EVENT select range: when a selected event occurs the current value of the up/down counter is captured in T1CAP. 0: Events are not captured 1: Events are captured	0x0	RW
[11:8]	EVENT	Event select range. Timer event select range (0 to 15).	0x0	RW
7	RLD	Reload control. RLD is only used for periodic mode; this bit allows the user to select whether the up/down counter should be reset only on a timeout event or also when T1CLRI[0] is set. 1: resets the up/down counter when T1CLRI[0] is set 0: up/down counter is only reset on a timeout event	0x0	RW
[6:5]	CLK	Clock select. Used to select a timer clock from the four available clock sources. 00: PCLK. 01: HCLK. 10: LFOSC. 32 KHz OSC 11: HFXTAL. 16 MHz OSC or XTAL, Dependent on the value in CLKCON0[11].	0x0	RW
4	ENABLE	Timer enable. Used to enable and disable the timer. Clearing this bit resets the timer, including the T1VAL register. 0: DIS. Timer is disabled (default). 1: EN. Timer is enabled.	0x0	RW

Bits	Bit Name	Description	Reset	Access
3	MOD	Timer mode. This bit is used to control whether the timer runs in periodic or free running mode. In periodic mode, the up/down counter starts at the defined LOAD value (T1LD); in free running mode, the up/down counter starts at 0x0000 or 0xFFFF depending on whether the timer is counting up or down. 0: FREERUN. Timer runs in free running mode. 1: PERIODIC. Timer runs in periodic mode (default).	0x1	RW
2	UP	Count up. Used to control whether the timer increments (counts up) or decrements (counts down) the up/down counter. 0: DIS. Timer is set to count down (default) 1: EN. Timer is set to count up	0x0	RW
[1:0]	PRE	Prescaler. Controls the prescaler division factor applied to the timer's selected clock. If CLK Source 0 (PCLK) or CLK Source 1 (HCLK) is selected, then Prescaler Value 0 means divide by 4, else it means divide by 1. 00: source clock/[1 or 4] 01: source clock/16 10: source clock/256 11: source clock/32,768	0x2	RW

Clear Interrupt Register

Address: 0x4000040C, Reset: 0x0000, Name: T1CLRI

Table 232. Bit Descriptions for T1CLRI

Bits	Bit Name	Description	Reset	Access
[15:2]	RESERVED	Reserved.	0x0	R
1	CAP	Clear captured event interrupt. This bit is used to clear a capture event interrupt. 0: no effect 1: clear the capture event interrupt	0x0	W1C
0	TMOUT	Clear timeout interrupt. This bit is used to clear a timeout interrupt. 0: no effect 1: clears the timeout interrupt	0x0	W1C

Capture Register

Address: 0x40000410, Reset: 0x0000, Name: T1CAP

Table 233. Bit Descriptions for T1CAP

Bits	Bit Name	Description	Reset	Access
[15:0]	CAP	16-bit captured value. T1CAP holds its value until T1CLRI[1] is set by user code. T1CAP is not overwritten even if another event occurs without writing to the T1CLRI[1].	0x0	R

Status Register

Address: 0x4000041C, Reset: 0x0000, Name: T1STA

Table 234. Bit Descriptions for T1STA

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
7	PDOK	T1CLRI synchronization. This bit is set automatically when the user sets T1CLRI[0] = 1. It is cleared automatically when the clear interrupt request has crossed clock domains and taken effect in the timer clock domain. 0: CLR. The interrupt is cleared in the timer clock domain. 1: SET. T1CLRI[0] is being updated in the timer clock domain.	0x0	R
6	BUSY	Timer busy. This bit informs the user that a write to T1CON is still crossing into the timer clock domain. This bit should be checked after writing T1CON and further writes should be suppressed until this bit is cleared. 0: CLR. Timer ready to receive commands to T1CON. 1: SET. Timer not ready to receive commands to T1CON.	0x0	R
[5:2]	RESERVED	Reserved.	0x0	R
1	CAP	Capture event pending. 0: CLR. No capture event is pending. 1: SET. A capture event is pending.	0x0	R
0	TMOUT	Timeout event occurred. This bit set automatically when the value of the counter reaches zero while counting down or reaches full scale when counting up. This bit is cleared when T1CLRI[0] is set by the user. 0: CLR. No timeout event has occurred. 1: SET. A timeout event has occurred.	0x0	R

REGISTER SUMMARY: GENERAL-PURPOSE TIMER 2

Table 235. Timer 2 Register Summary

Address	Name	Description	Reset	RW
0x40000800	T2LD	16-bit load value register	0x0000	RW
0x40000804	T2VAL	16-bit timer value register	0x0000	R
0x40000808	T2CON	Control register	0x000A	RW
0x4000080C	T2CLRI	Clear interrupt register	0x0000	W
0x40000810	T2CAP	Capture register	0x0000	R
0x4000081C	T2STA	Status register	0x0000	R

REGISTER DETAILS: GENERAL-PURPOSE TIMER 2

16-Bit Load Value Register

Address: 0x40000800, Reset: 0x0000, Name: T2LD

Table 236. Bit Descriptions for T2LD

Bits	Bit Name	Description	Reset	Access
[15:0]	LOAD	Load value. The up/down counter is periodically loaded with this value if periodic mode is selected (T2CON[3]=1). LOAD writes during up/down counter timeout events are delayed until the event has passed.	0x0	RW

16-Bit Timer Value Register

Address: 0x40000804, Reset: 0x0000, Name: T2VAL

Table 237. Bit Descriptions for T2VAL

Bits	Bit Name	Description	Reset	Access
[15:0]	VAL	Current count. Reflects the current up/down counter value. Value delayed two PCLK cycles due to clock synchronizers.	0x0	R

Control Register

Address: 0x40000808, Reset: 0x000A, Name: T2CON

Table 238. Bit Descriptions for T2CON

Bits	Bit Name	Description	Reset	Access
[15:13]	RESERVED	Reserved.	0x0	R
12	EVENTEN	Event select. Used to enable and disabling the capture of events. Used in conjunction with the EVENT select range: when a selected event occurs the current value of the Up/Down counter is captured in T2CAP. 0: events are not captured 1: events are captured	0x0	RW
[11:8]	EVENT	Event select range. Timer event select range (0 to 15).	0x0	RW
7	RLD	Reload control. RLD is only used for periodic mode; this bit allows the user to select whether the up/down counter should be reset only on a timeout event or also when T2CLRI[0] is set. 1: resets the up/down counter when T2CLRI[0] is set 0: up/down counter is only reset on a timeout event	0x0	RW
[6:5]	CLK	Clock select. Used to select a timer clock from the four available clock sources. 00: PCLK. 01: HCLK. 10: LFOSC. 32 KHz OSC. 11: HFXTAL. 16 MHz OSC or XTAL, dependent on the value in CLKCON0[11].	0x0	RW
4	ENABLE	Timer enable. Used to enable and disable the timer. Clearing this bit resets the timer, including the T2VAL register. 0: DIS. Timer is disabled (default). 1: EN. Timer is enabled.	0x0	RW

Bits	Bit Name	Description	Reset	Access
3	MOD	Timer mode. This bit is used to control whether the timer runs in periodic or free running mode. In periodic mode, the up/down counter starts at the defined LOAD value (T2LD); in free running mode, the up/down counter starts at 0x0000 or 0xFFFF depending on whether the timer is counting up or down. 0: FREERUN. Timer runs in free running mode. 1: PERIODIC. Timer runs in periodic mode (default).	0x1	RW
2	UP	Count up. Used to control whether the timer increments (counts up) or decrements (counts down) the up/down counter. 0: DIS. Timer is set to count down (default). 1: EN. Timer is set to count up.	0x0	RW
[1:0]	PRE	Prescaler. Controls the prescaler division factor applied to the timer's selected clock. If CLK Source 0 (PCLK) or CLK Source 1 (HCLK) is selected, then Prescaler Value 0 means divide by 4, else it means divide by 1. 00: source clock/[1 or 4] 01: source clock/16 10: source clock/256 11: source clock/32,768	0x2	RW

Clear Interrupt Register

Address: 0x4000080C, Reset: 0x0000, Name: T2CLRI

Table 239. Bit Descriptions for T2CLRI

Bits	Bit Name	Description	Reset	Access
[15:2]	RESERVED	Reserved.	0x0	R
1	CAP	Clear captured event interrupt. This bit is used to clear a capture event interrupt. 0: no effect 1: clear the capture event interrupt	0x0	W1C
0	TMOUT	Clear timeout interrupt. This bit is used to clear a timeout interrupt. 0: no effect 1: clears the timeout interrupt	0x0	W1C

Capture Register

Address: 0x40000810, Reset: 0x0000, Name: T2CAP

Table 240. Bit Descriptions for T2CAP

Bits	Bit Name	Description	Reset	Access
[15:0]	CAP	16-bit captured value. T2CAP holds its value until T2CLRI[1] is set by user code. T2CAP is not overwritten even if another event occurs without writing to the T2CLRI[1].	0x0	R

Status Register

Address: 0x4000081C, Reset: 0x0000, Name: T2STA

Table 241. Bit Descriptions for T2STA

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
7	PDOK	T2CLRI synchronization. This bit is set automatically when the user sets T2CLRI[0] = 1. It is cleared automatically when the clear interrupt request has crossed clock domains and taken effect in the timer clock domain. 0: CLR. The interrupt is cleared in the timer clock domain. 1: SET. T2CLRI[0] is being updated in the timer clock domain.	0x0	R
6	BUSY	Timer Busy. This bit informs the user that a write to T2CON is still crossing into the timer clock domain. This bit should be checked after writing T2CON and further writes should be suppressed until this bit is cleared. 0: CLR. Timer ready to receive commands to T2CON. 1: SET. Timer not ready to receive commands to T2CON.	0x0	R
[5:2]	RESERVED	Reserved.	0x0	R
1	CAP	Capture event pending. 0: CLR. No capture event is pending. 1: SET. A capture event is pending.	0x0	R
0	TMOUT	Timeout event occurred. This bit set automatically when the value of the counter reaches zero while counting down or reaches full scale when counting up. This bit is cleared when T2CLRI[0] is set by the user. 0: CLR. No timeout event has occurred. 1: SET. A timeout event has occurred.	0x0	R

WATCHDOG TIMER

WATCHDOG TIMER FEATURES

16-bit count-down timer, which can be used to recover from an invalid software state.

Clocked by the 32 kHz internal oscillator (LFOSC) with a programmable prescaler (1, 16, 256, or 4096).

WATCHDOG TIMER BLOCK DIAGRAM

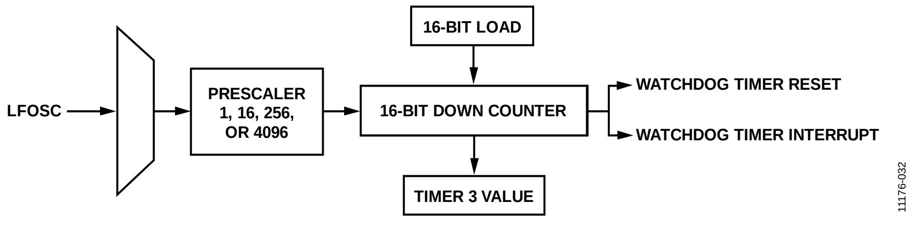


Figure 30. Watchdog Timer Block Diagram

WATCHDOG TIMER OVERVIEW

The watchdog timer (Timer 3) is used to recover from an invalid software state. When enabled, this timer requires periodic servicing to prevent it from forcing a reset of the device. For debug purposes, the timer can be configured to generate an interrupt instead of a reset.

The watchdog timer is clocked by the internal 32.768 kHz oscillator, LFOSC. It is clocked at all times except during a reset.

The watchdog timer is a 16-bit count-down timer with a programmable prescaler. The prescaler is selectable and can divide LFOSC by a factor of 1, 16, 256, or 4096.

WATCHDOG TIMER OPERATION

The watchdog timer is enabled by default after a reset.

User code should disable the watchdog timer at the start of user code when debugging or if the watchdog timer is not required.

```
T3CON = 0x00;                                      // Disable watchdog timer
```

Enabling the watchdog timer (set T3CON[5] = 1) also write protects T3CON and T3LD.

This means that after kernel execution, user code can disable the timer and then reconfigure it with T3CON[5] = 1 only once. Then T3CON and T3LD are write protected. T3STA[4] indicates if the timer configuration has been locked. Only a reset clears T3CON[5], unlocking T3CON and T3LD, and allows reconfiguration of the timer.

If T3CON is not modified, user code can change T3LD at any time. If T3CON[5] is cleared to 0, the timer is disabled. Settings can be modified, and the timer can be reenabled.

When the watchdog timer is used in interrupt mode, T3STA[0], the WDT interrupt bit, is set to 1 for only a very short period (2 × PCLK). Therefore, T3STA[0] should not be used for polling purposes.

If T3CLRI is cleared, allow enough CPU cycles to allow the write to happen again so that no lockup happens in the WDT. The T3STA_CLRI bit can be used to verify when the write to the CLRI registers has finished and therefore it is safe to write again.

REGISTER SUMMARY: WATCHDOG TIMER

Table 242. Watchdog Timer Register Summary

Address	Name	Description	Reset	RW
0x40002580	T3LD	Load value register	0x1000	RW
0x40002584	T3VAL	Current count value register	0x1000	R
0x40002588	T3CON	Control register	0x00E9	RW
0x4000258C	T3CLRI	Clear interrupt register	0x0000	W
0x40002598	T3STA	Status register	0x0000	R

REGISTER DETAILS: WATCHDOG TIMER

Load Value Register

Address: 0x40002580, Reset: 0x1000, Name: T3LD

Table 243. Bit Descriptions for T3LD

Bits	Bit Name	Description	Reset	Access
[15:0]	LOAD	Load value	0x1000	RW

Current Count Value Register

Address: 0x40002584, Reset: 0x1000, Name: T3VAL

Table 244. Bit Descriptions for T3VAL

Bits	Bit Name	Description	Reset	Access
[15:0]	CCOUNT	Current count value.	0x1000	R

Control Register

Address: 0x40002588, Reset: 0x00E9, Name: T3CON

Table 245. Bit Descriptions for T3CON

Bits	Bit Name	Description	Reset	Access
[15:7]	RESERVED	Reserved.	0x1	R
6	MOD	Timer mode. Note that in free running mode it wraps around at 0x1000. 0: FREERUN. Cleared by user to operate in free running mode. 1: PERIODIC. Set by user to operate in periodic mode (default).	0x1	RW
5	ENABLE	Timer enable. 0: DIS. Cleared by user to disable the timer. 1: EN. Set by user to enable the timer (default).	0x1	RW
4	RESERVED	Reserved.	0x0	R
[3:2]	PRE	Prescaler. 00: DIV1. Source clock/1. 01: DIV16. Source clock/16. 10: DIV256. Source clock/256 (default). 11: DIV4096. Source clock/4096	0x2	RW
1	IRQ	Timer interrupt. 0: DIS. Cleared by user to generate a reset on a time out (default). 1: EN. Set by user to generate an interrupt when the timer times out. This feature is provided for debug purposes and is only available in active mode.	0x0	RW
0	PMD	Power Mode Disable. PMD controls the behavior of the watchdog when in hibernate mode. If the application requires prolonged periods of time spent in hibernate mode and it is not desirable to periodically wake up to service the watchdog timer, the counter within the watchdog timer can be suspended when entering the hibernate power mode. Regardless of how the PMD bit is set, it is recommended that the watchdog timer be cleared before entering hibernate mode. 0: DIS. The watchdog timer continues its countdown while in hibernate mode. 1: EN. When hibernate mode is entered, the watchdog counter suspends its countdown. When hibernate mode is exited, the countdown resumes from its current count value (the count is not reset).	0x1	RW

Clear Interrupt Register

Address: 0x4000258C, Reset: 0x0000, Name: T3CLRI

Table 246. Bit Descriptions for T3CLRI

Bits	Bit Name	Description	Reset	Access
[15:0]	CLRWDG	Clear watchdog. User writes 0xCCCC to reset/reload/restart T3 or clear IRQ. A write of any other value causes a watchdog reset. Write only, reads 0. Do not write to this register if using the timer in IRQ mode.	0x0	W

Status Register

Address: 0x40002598, Reset: 0x0000, Name: T3STA

Table 247. Bit Descriptions for T3STA

Bits	Bit Name	Description	Reset	Access
[15:5]	RESERVED	Reserved.	0x0	R
4	LOCK	Lock status bit. Set automatically in hardware if T3CON[5] has been set by user code. Cleared by default and until user code sets T3CON[5].	0x0	R
3	CON	T3CON write sync in progress. 0: internal bus and T3 clock domains T3CON configuration values match 1: internal bus T3CON register values are being synchronized to T3 clock domain	0x0	R
2	LD	T3LD write sync in progress. 0: internal bus and T3 clock domains T3LD values match 1: internal bus T3LD value is being synchronized to T3 clock domain	0x0	R
1	CLRI	T3CLRI write sync in progress. 0: internal bus T3CLRI write sync not done 1: internal bus T3CLRI write is being synced to T3 clock domain. T3 is restarted (if 0xCCCC was written) when sync is complete	0x0	R
0	IRQ	WDT interrupt. 0: T3 interrupt not pending 1: T3 interrupt pending	0x0	R

WAKE-UP TIMER

WAKE-UP TIMER FEATURES

- 32-bit counter (count down or count up)
- Three clock sources with programmable prescaler (1, 16, 256, or 32768)
    - Peripheral clock (PCLK)
    - 32 kHz internal oscillator (LFOSC)
    - External clock applied on Pin P1.0 (ECLKIN)
- Four compare points, one automatic increment

WAKE-UP TIMER BLOCK DIAGRAM

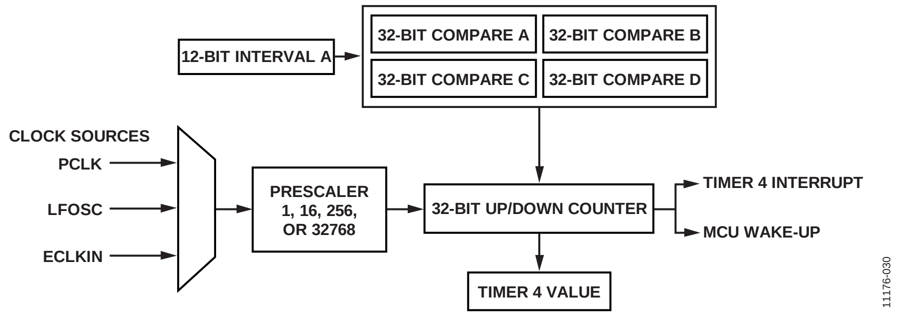


Figure 31. Wake-Up Timer Block Diagram

WAKE-UP TIMER OVERVIEW

The wake-up timer (Timer 4) block consists of a 32-bit counter clocked from one of three different sources: the system clock (PCLK), the internal oscillator (LFOSC), or an external clock applied on Pin P1.0 (ECLKIN). The selected clock source can be scaled down using a prescaler of 1, 16, 256, or 32768. The wake-up timer continues to run independent of the clock source used when the PCLK clock is disabled.

The timer can be used in free running or periodic mode. In free running mode, the timer counts from 0x00000000 to 0xFFFFFFFF and then restarts at 0x00000000. In periodic mode, the timer counts from 0x00000000 to T4WUFD (T4WUFD0 and T4WUFD1).

In addition, the wake-up timer has four specific time fields to compare with the wake-up counter: T4WUFA, T4WUFB, T4WUFC, and T4WUFD. All four wake-up compare points can generate interrupts or wake-up signals. When the timer is in free running mode, T4WUFA, T4WUFB, T4WUFC, andT4WUFD must be reconfigured in software to generate a periodic interrupt.

WAKE-UP TIMER OPERATION

The wake-up timer comparator registers must be configured before starting the timer. The timer is started by writing the control enable bit (T4CON[7]). The timer increments until the value reaches full scale in free running mode or when T4WUFD matches the wake-up value, T4VAL.

The wake-up timer is a 32-bit timer. Its current value is stored in two 16-bit registers: T4VAL1 stores the upper 16 bits, and T4VAL0 stores the lower 16 bits.

When T4VAL0 is read, T4VAL1 is frozen at its current value until it is subsequently read. The control bit FREEZE (T4CON[3]) must be set to freeze the T4VAL register between the lower and upper reads.

Clock Selection

Clock selection is made by setting T4CON[10:9].

If PCLK is selected (T4CON[10:9] = 00), configuring T4CON[1:0] = 00 results in a prescaler of 4.

Synchronization to the LFOSC clock domain is done automatically by hardware, and precautions concerning asynchronous clocks as described in Timer 0, Timer 1, and Timer 2 do not apply.

Compare Field Registers

Hardware Updated Field

T4INC is a 12-bit interval register that is used to update the compare value in T4WUFAx by hardware. When a new value is written in T4INC, Bits[16:5] of the internal 32-bit compare register (T4WUFAx) are loaded with the new T4INC value. If the new compare value is less than the T4WUFD value in periodic mode or less than 0xFFFFFFFF in free running mode, this 32-bit compare register is automatically incremented with the contents of T4INC (shifted by five) each time the wake-up counter reaches the value in this compare register. If the new compare value is greater than these limits, it is recalculated as follows.

In free running mode, the new value is

$$T4WUFA = Old\ T4WUFA + (32 \times T4INC) - \text{0xFFFFFFFF}.$$

In periodic mode, the new value is

$$T4WUFA = Old\ T4WUFA + (32 \times T4INC) - \text{T4WUFD}.$$

The maximum programmable interval is just above 4 seconds.

T4INC is compared with Bits[16:5] of the timer value. Because it is shifted left by five bits, its value must be multiplied by 32 to obtain the compare value.

With the default value of 0xC8 (where for calculation purposes 0xC8 = 200 in decimal), a prescaler = 1, and 32 kHz clock selected,

$$Interval = ((200 \times 32) + 1) \times 1/32{,}768 = 195.3155\ \text{ms}$$

To modify the interval value, the timer must be stopped so that the interval register can be loaded in the compare register if T4CON[11] = 0.

To modify the interval value, set STOPINC (T4CON[11] = 1) while the timer is running.

The new T4INC value takes effect after the next Wake-Up Field A interrupt. If the user is writing to this register while the timer is enabled, the STOPINC bit should be set before writing to it, and then STOPINC should be cleared after the update.

Software Updated Field

T4WUFB, T4WUFC, and T4WUFD are 32-bit values programmed by the user in the T4WUFx0 and T4WUFx1 registers (x = B, C, or D). T4WUFD contains the load value when the wake-up timer is configured in periodic mode.

The T4WUFBx and T4WUFCx registers can be written to at any time, but the corresponding interrupt enable (T4IEN[1] or T4IEN[2]) must be disabled. After the register is updated, the interrupt can be reenabled.

In periodic mode, the T4WUFDx registers can be written to only when the timer is disabled. In free running mode, the T4WUFDx registers can be written to while the timer is running. Before doing so, the corresponding interrupt enable (T4IEN[3]) must be disabled. After the register is updated, the interrupt can be reenabled.

In free running mode, T4WUFB, T4WUFC, and T4WUFD can be written to at any time, but the corresponding interrupt enable in the T4IEN register must be disabled. After the register is updated, the interrupt can be reenabled. In periodic mode, this is only applicable to T4WUFB and T4WUFC.

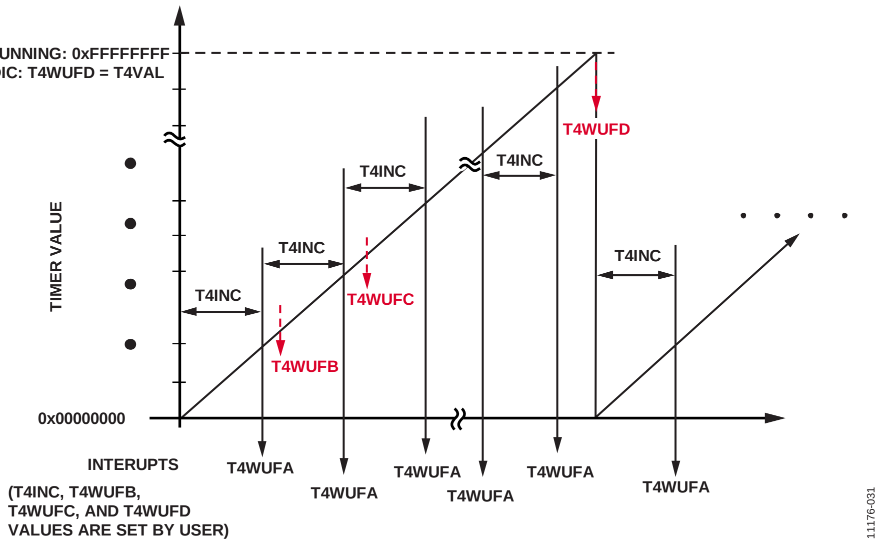


Figure 32. Wake-Up Timer Fields Action

Interrupts/Wake-Up Signals

An interrupt is generated when the counter value corresponds to any of the compare points or full scale in free running mode. The timer continues counting or is reset to 0.

The wake-up timer generates five maskable interrupts. They are enabled in the T4IEN register. Interrupts can be cleared by setting the corresponding bit in the T4CLRI register.

Note that it takes two 32 kHz clock cycles for the interrupt clear to take effect when the 32 kHz internal oscillator is used.

Ensure that the register write has fully completed before returning from the interrupt handler. Use the data synchronization barrier (DSB) instruction if necessary. The following is a code example showing how to implement the DSB ARM Cortex-M3 instruction in a C program.

```
void Ext_Int4_Handler ()
{
 EiClr(EXTINT4);
 __DSB();
}
```

During that time, the part should not be placed in any of the power-down modes. IRQCRY (T4STA[6]) indicates when the device can be placed in power-down mode.

The timer is stopped and reset when clearing the timer enable bit in the T4CON register (T4CON[7]).

REGISTER SUMMARY: WAKE-UP TIMER

Table 248. Wake-Up Timer Register Summary

Address	Name	Description	Reset	RW
0x40002500	T4VAL0	Current count value—least significant 16 bits	0x0000	R
0x40002504	T4VAL1	Current count value—most significant 16 bits	0x0000	R
0x40002508	T4CON	Control register	0x0040	RW
0x4000250C	T4INC	12-bit interval for Wake-Up Field A	0x00C8	RW
0x40002510	T4WUFB0	Wake-Up Field B—least significant 16 bits	0x1FFF	RW
0x40002514	T4WUFB1	Wake-Up Field B—most significant 16 bits	0x0000	RW
0x40002518	T4WUFC0	Wake-Up Field C—least significant 16 bits	0x2FFF	RW
0x4000251C	T4WUFC1	Wake-Up Field C—most significant 16 bits	0x0000	RW
0x40002520	T4WUFD0	Wake-Up Field D—least significant 16 bits	0x3FFF	RW
0x40002524	T4WUFD1	Wake-Up Field D—most significant 16 bits	0x0000	RW
0x40002528	T4IEN	Interrupt enable register	0x0000	RW
0x4000252C	T4STA	Status register	0x0000	R
0x40002530	T4CLRI	Clear interrupt register	0x0000	W
0x4000253C	T4WUFA0	Wake-Up Field A—least significant 16 bits	0x1900	R
0x40002540	T4WUFA1	Wake-Up Field A—most significant 16 bits	0x0000	R

REGISTER DETAILS: WAKE-UP TIMER

Current Count Value—Least Significant 16 Bits Register

Address: 0x40002500, Reset: 0x0000, Name: T4VAL0

Table 249. Bit Descriptions for T4VAL0

Bits	Bit Name	Description	Reset	Access
[15:0]	T4VALL	Current count low. Least significant 16 bits of current count value.	0x0	R

Current Count Value—Most Significant 16 Bits Register

Address: 0x40002504, Reset: 0x0000, Name: T4VAL1

Table 250. Bit Descriptions for T4VAL1

Bits	Bit Name	Description	Reset	Access
[15:0]	T4VALH	Current count high. Most significant 16 bits of current count value.	0x0	R

Control Register

Address: 0x40002508, Reset: 0x0040, Name: T4CON

Table 251. Bit Descriptions for T4CON

Bits	Bit Name	Description	Reset	Access
[15:12]	RESERVED	Reserved.	0x0	R
11	STOP_WUFA	Disables updating Field A register T4WUFA. This bit when set stops the Wake-Up Field A register T4WUFA getting updated with the interval register I2INC value. This allows the user to update the interval T4INC or T4WUFA registers safely.	0x0	RW
[10:9]	CLK	Clock select. 00: PCLK: PCLK (default) 01: LFOSC: 32 kHz internal oscillator 10: LFOSC: 32 kHz internal oscillator 11: ECLKIN: external clock from P1.0	0x0	RW
8	WUEN	Wakeup enable. 0: DIS: cleared by user to disable the wake up timer when the core clock is off 1: EN: set by user to enable the wake up timer even when the core clock is off	0x0	RW

Bits	Bit Name	Description	Reset	Access
7	ENABLE	Timer enable. 0: DIS: disable the timer (default) 1: EN: enable the timer	0x0	RW
6	MOD	Timer mode. 0: PERIODIC: cleared by user to operate in periodic mode. In this mode, the timer counts up to T4WUFD. 1: FREERUN: set by user to operate in free running mode (default).	0x1	RW
[5:4]	RESERVED	Reserved. These bits should be written 0.	0x0	RW
3	FREEZE	Freeze enable. 0: DIS: Cleared by user to disable this feature (default). 1: EN: Set by user to enable the freeze of the high 16-bits after the lower bits have been read from T4VAL0. This ensures that the software reads an atomic shot of the timer. T4VAL1 unfreezes after it has been read.	0x0	RW
2	RESERVED	Reserved.	0x0	RW
[1:0]	PRE	Prescaler. 00: PREDIV1: source clock/1 (default). If the selected clock source is PCLK, this setting results in a prescaler of 4. 01: PREDIV16: source clock/16 10: PREDIV256: source clock/256 11: PREDIV32768: source clock/32,768	0x0	RW

12-Bit Interval for Wake-Up Field A Register

Address: 0x4000250C, Reset: 0x00C8, Name: T4INC

Table 252. Bit Descriptions for T4INC

Bits	Bit Name	Description	Reset	Access
[15:12]	RESERVED	Reserved.	0x0	R
[11:0]	INTERVAL	Interval for Wake-Up Field A.	0x0C8	RW

Wake-Up Field B—Least Significant 16 Bits Register

Address: 0x40002510, Reset: 0x1FFF, Name: T4WUFB0

Table 253. Bit Descriptions for T4WUFB0

Bits	Bit Name	Description	Reset	Access
[15:0]	T4WUFBL	Wake-Up Field B low. Least significant 16 bits of Wake-Up Field B.	0x1FFF	RW

Wake-Up Field B—Most Significant 16 Bits Register

Address: 0x40002514, Reset: 0x0000, Name: T4WUFB1

Table 254. Bit Descriptions for T4WUFB1

Bits	Bit Name	Description	Reset	Access
[15:0]	T4WUFBH	Wake-Up Field B High. Most significant 16 bits of Wake-Up Field B.	0x0	RW

Wake-Up Field C—Least Significant 16 Bits Register

Address: 0x40002518, Reset: 0x2FFF, Name: T4WUFC0

Table 255. Bit Descriptions for T4WUFC0

Bits	Bit Name	Description	Reset	Access
[15:0]	T4WUFCL	Wake-Up Field C Low. Least significant 16 bits of Wake-Up Field C.	0x2FFF	RW

Wake-Up Field C—Most Significant 16 Bits Register

Address: 0x4000251C, Reset: 0x0000, Name: T4WUFC1

Table 256. Bit Descriptions for T4WUFC1

Bits	Bit Name	Description	Reset	Access
[15:0]	T4WUFCH	Wake-Up Field C High. Most significant 16 bits of Wake-Up Field C.	0x0	RW

Wake-Up Field D—Least Significant 16 Bits Register

Address: 0x40002520, Reset: 0x3FFF, Name: T4WUFD0

Table 257. Bit Descriptions for T4WUFD0

Bits	Bit Name	Description	Reset	Access
[15:0]	T4WUFD0	Wake-Up Field D Low. Least significant 16 bits of Wake-Up Field C.	0x3FFF	RW

Wake-Up Field D—Most Significant 16 Bits Register

Address: 0x40002524, Reset: 0x0000, Name: T4WUFD1

Table 258. Bit Descriptions for T4WUFD1

Bits	Bit Name	Description	Reset	Access
[15:0]	T4WUFDH	Wake-Up Field D high. Most significant 16 bits of Wake-Up Field D.	0x0	RW

Interrupt Enable Register

Address: 0x40002528, Reset: 0x0000, Name: T4IEN

Table 259. Bit Descriptions for T4IEN

Bits	Bit Name	Description	Reset	Access
[15:5]	RESERVED	Reserved.	0x0	R
4	ROLL	Rollover interrupt enable. Used only in free running mode. Set by user to generate an interrupt when Timer2 rolls over. Cleared by user to disable the roll over interrupt (default).	0x0	RW
3	WUFD	T4WUFD interrupt enable. Set by user code to generate an interrupt when T4VAL reaches T4WUFD. Cleared by user code to disable T4WUFD interrupt (default).	0x0	RW
2	WUFC	T4WUFC interrupt enable. Set by user code to generate an interrupt when T4VAL reaches T4WUFC. Cleared by user code to disable T4WUFC interrupt (default).	0x0	RW
1	WUFB	T4WUFB interrupt enable. Set by user code to generate an interrupt when T4VAL reaches T4WUFB. Cleared by user code to disable T4WUFB interrupt (default).	0x0	RW
0	WUFA	T4WUFA interrupt enable. Set by user code to generate an interrupt when T4VAL reaches T4WUFA. Cleared by user code to disable T4WUFA interrupt (default).	0x0	RW

Status Register

Address: 0x4000252C, Reset: 0x0000, Name: T4STA

Table 260. Bit Descriptions for T4STA

Bits	Bit Name	Description	Reset	Access
[15:9]	RESERVED	Reserved.	0x0	R
8	PDOK	Enable bit synchronized. Indicates when a change in the enable bit is synchronized to the 32 kHz clock domain. It is set high when the enable bit (Bit 5) in the control register is set or cleared. It returns low when the change in the enable bit has been synchronized to the 32 kHz clock domain.	0x0	R
7	FREEZE	Timer value freeze. Set automatically to indicate that the value in T4VAL1 is frozen. Cleared by automatically when T4VAL1 is read.	0x0	R

Bits	Bit Name	Description	Reset	Access
6	IRQCRY	Wake-up status to power-down. Set automatically when any of the interrupts are still set in the external crystal clock domain. Cleared automatically when the interrupts are cleared, allowing power down mode. User code should wait for this bit to be cleared before entering power-down mode.	0x0	R
5	RESERVED	Reserved.	0x0	R
4	ROLL	Rollover interrupt flag. Used only in free running mode. Set automatically to indicate a roll over interrupt has occurred. Cleared automatically after a write to T4CLRI.	0x0	R
3	WUFD	T4WUFD interrupt flag. Set automatically to indicate a comparator interrupt has occurred. Cleared automatically after a write to the corresponding bit in T4CLRI.	0x0	R
2	WUFC	T4WUFC interrupt flag. Set automatically to indicate a comparator interrupt has occurred. Cleared automatically after a write to the corresponding bit in T4CLRI.	0x0	R
1	WUFB	T4WUFB interrupt flag. Set automatically to indicate a comparator interrupt has occurred. Cleared automatically after a write to the corresponding bit in T4CLRI.	0x0	R
0	WUFA	T4WUFA interrupt flag. Set automatically to indicate a comparator interrupt has occurred. Cleared automatically after a write to the corresponding bit in T4CLRI.	0x0	R

Clear Interrupt Register

Address: 0x40002530, Reset: 0x0000, Name: T4CLRI

Table 261. Bit Descriptions for T4CLRI

Bits	Bit Name	Description	Reset	Access
[15:5]	RESERVED	Reserved.	0x0	R
4	ROLL	Rollover interrupt clear. Used only in free running mode. Set by user code to clear a roll over interrupt flag. Cleared automatically after synchronization.	0x0	RW
3	WUFD	T4WUFD interrupt clear.	0x0	RW
2	WUFC	T4WUFC interrupt clear. Set by user code to clear a T4WUFC interrupt flag. Cleared automatically after synchronization.	0x0	RW
1	WUFB	T4WUFB interrupt clear. Set by user code to clear a T4WUFB interrupt flag. Cleared automatically after synchronization.	0x0	RW
0	WUFA	T4WUFA interrupt clear. Set by user code to clear a T4WUFA interrupt flag. Cleared automatically after synchronization.	0x0	RW

Wake-Up Field A—Least Significant 16 Bits Register

Address: 0x4000253C, Reset: 0x1900, Name: T4WUFA0

Table 262. Bit Descriptions for T4WUFA0

Bits	Bit Name	Description	Reset	Access
[15:0]	T4WUFAL	Wake-Up Field A low. Least significant 16 bits of Wake-Up Field A.	0x1900	RW

Wake-Up Field A—Most Significant 16 Bits Register

Address: 0x40002540, Reset: 0x0000, Name: T4WUFA1

Table 263. Bit Descriptions for T4WUFA1

Bits	Bit Name	Description	Reset	Access
[15:0]	T4WUFAH	Wake-Up Field A high. Most significant 16 bits of Wake-Up Field A.	0x0	RW

PWM

PWM FEATURES

- 8-channel PWM interface
- H-bridge mode supported on 2 pairs

PWM OVERVIEW

The ADuCM320 integrates an 8-channel PWM interface. Eight channels are grouped as three pairs (0 to 3). The first two pairs of PWM outputs (PWM0, PWM1, PWM2, and PWM3) can be configured in standard mode or to drive an H-bridge. Pair 2 and Pair 3 can be configured in standard mode only. The PWM pairs and modes are summarized in Table 264.

Table 264. PWM Channel Grouping

Port Name	Description	PWM Mode Available
PWM0	High-side PWM output for Pair 0	H-bridge and standard
PWM1	Low-side PWM output for Pair 0	H-bridge and standard
PWM2	High-side PWM output for Pair 1	H-bridge and standard
PWM3	Low-side PWM output for Pair 1	H-bridge and standard
PWM4	High-side PWM output for Pair 2	Standard
PWM5	Low-side PWM output for Pair 2	Standard
PWM6	High-side PWM output for Pair 3	Standard
PWM7	Low-side PWM output for Pair 3	Standard

On power-up, the PWM outputs default to H-bridge mode for Pair 0 and Pair 1. In the standard mode, the user has control over the period of each pair of outputs and over the duty cycle of each individual output.

In the event of external fault conditions, a falling edge on the PWM_{TRIP} pin provides an instantaneous shutdown of the PWM controller. All PWM outputs are placed in the off state, that is, in low state for the low side and high state for the high side, and a PWM_{TRIP} interrupt can be generated.

PWM OPERATION

The PWM clock is selectable via PWMCON0 with one of the following values: HCLK divided by 2, 4, 8, 16, 32, 64, 128, or 256.

In all modes, the PWMxCOMx MMRs control the point at which the PWM output changes state. An example is shown in Figure 33.

Each pair has an associated counter. The length of the PWM period is defined by PWMxLEN.

The PWM waveforms are set by the count value of the 16-bit timer and the compare register contents.

An example for PWM Pair 0 (Port PWM0 and Port PWM1) is

- The low-side waveform, PWM1, goes high when the timer count reaches PWM0LEN, and it goes low when the timer count reaches the value held in PWM0COM2 or when the high-side waveform PWM0 goes low.
- The high-side waveform, PWM0, goes high when the timer count reaches the value held in PWM0COM0, and it goes low when the timer count reaches the value held in PWM0COM1.

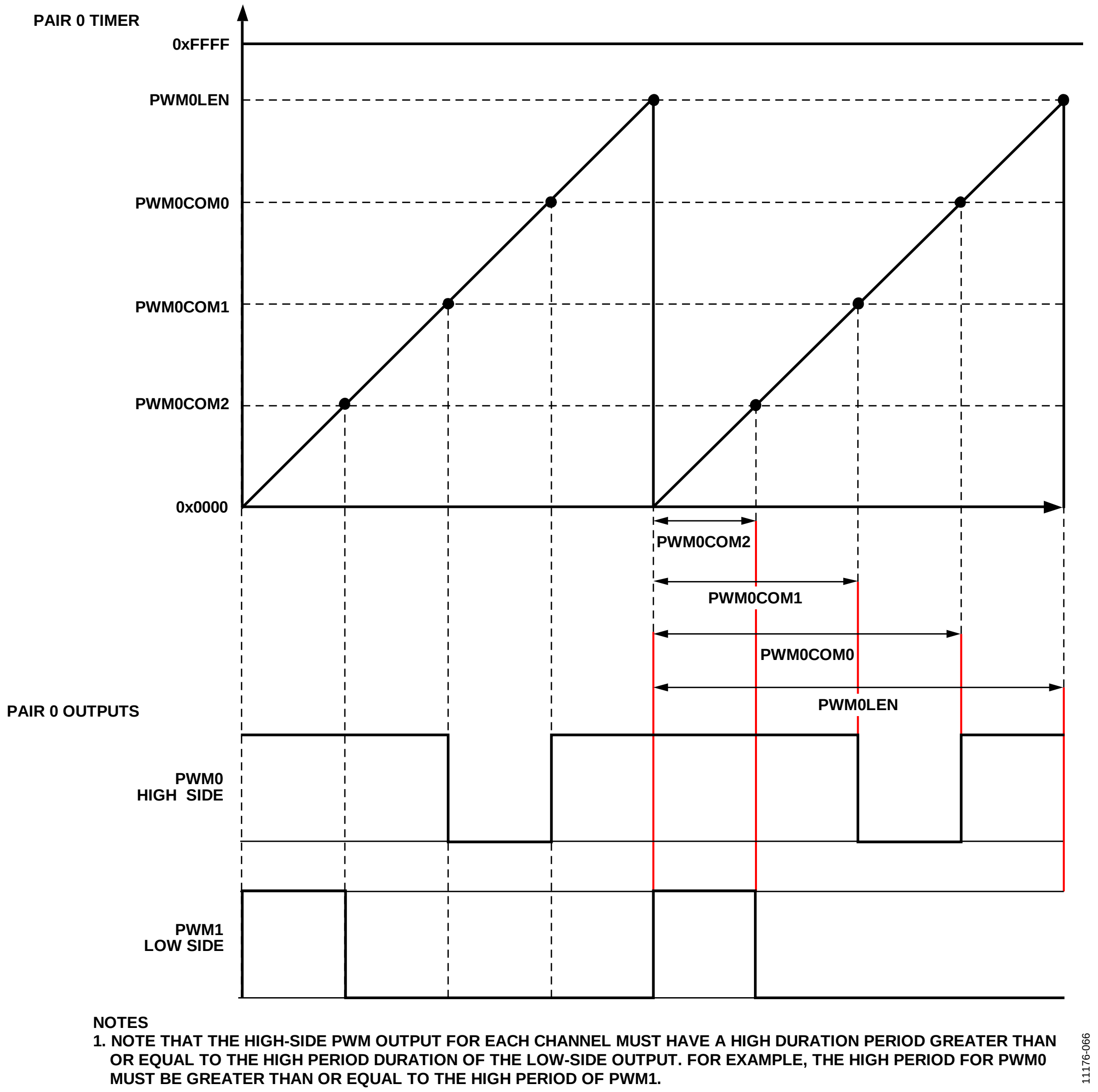

PAIR 0 TIMER
0xFFFF
PWM0LEN
PWM0COM0
PWM0COM1
PWM0COM2
0x0000
PWM0COM2
PWM0COM1
PWM0COM0
PWM0LEN
PAIR 0 OUTPUTS
PWM0
HIGH SIDE
PWM1
LOW SIDE
NOTES
1. NOTE THAT THE HIGH-SIDE PWM OUTPUT FOR EACH CHANNEL MUST HAVE A HIGH DURATION PERIOD GREATER THAN OR EQUAL TO THE HIGH PERIOD DURATION OF THE LOW-SIDE OUTPUT. FOR EXAMPLE, THE HIGH PERIOD FOR PWM0 MUST BE GREATER THAN OR EQUAL TO THE HIGH PERIOD OF PWM1.
11176-066

Figure 33. Waveform of PWM Channel Pair in Standard Mode

Table 265 lists equations for the period and duration for both the outputs of a PWM channel.

Table 265. PWM Equations

PWM	Period	Duration
Low Side (PWM1)	$t_{UCLK/DIV} \times (PWM0LEN + 1) \times N_{PRESCALE}$	High duration If PWMCOM2 < PWMCOM1: $t_{UCLK/DIV} \times (PWM0LEN - PWM0COM2) \times N_{PRESCALE}$ Otherwise: $t_{UCLK} \times (PWM0LEN - PWM0COM1) \times N_{PRESCALE}$
High Side (PWM0)	$t_{UCLK/DIV} \times (PWM0LEN + 1) \times N_{PRESCALE}$	Low duration $t_{UCLK/DIV} \times (PWM0COM0 - PWM0COM1) \times N_{PRESCALE}$

Note that:

- $t_{UCLK/DIV}$ is the PWM clock frequency selected by CLKCON1[2:0].
- $N_{PRESCALE}$ is the prescaler value as determined by PWMCON0[8:6].

Standard Mode

In standard mode, each pair is individually controlled by a selection of registers, as shown in Table 266.

Table 266. Compare Register Descriptions in Standard Mode (Base Address: 0x40024000)

Pair	Name	Description
0	PWM0COM0	PWM0 output goes high when the PWM timer reaches the count value stored in this register.
	PWM0COM1	PWM0 output goes low when the PWM timer reaches the count value stored in this register.
	PWM0COM2	PWM1 output goes low when the PWM timer reaches the count value stored in this register.
	PWM0LEN	PWM1 output goes high when the PWM timer reaches the count value stored in this register.
1	PWM1COM0	PWM2 output goes high when the PWM timer reaches the count value stored in this register.
	PWM1COM1	PWM2 output goes low when the PWM timer reaches the count value stored in this register.
	PWM1COM2	PWM3 output goes low when the PWM timer reaches the count value stored in this register.
	PWM1LEN	PWM3 output goes high when the PWM timer reaches the count value stored in this register.
2	PWM2COM0	PWM4 output goes high when the PWM timer reaches the count value stored in this register.
	PWM2COM1	PWM4 output goes low when the PWM timer reaches the count value stored in this register.
	PWM2COM2	PWM5 output goes low when the PWM timer reaches the count value stored in this register.
	PWM2LEN	PWM5 output goes high when the PWM timer reaches the count value stored in this register.
3	PWM3COM0	PWM6 output goes high when the PWM timer reaches the count value stored in this register.
	PWM3COM1	PWM6 output goes low when the PWM timer reaches the count value stored in this register.
	PWM3COM2	PWM7 output goes low when the PWM timer reaches the count value stored in this register.
	PWM3LEN	PWM7 output goes high when the PWM timer reaches the count value stored in this register.

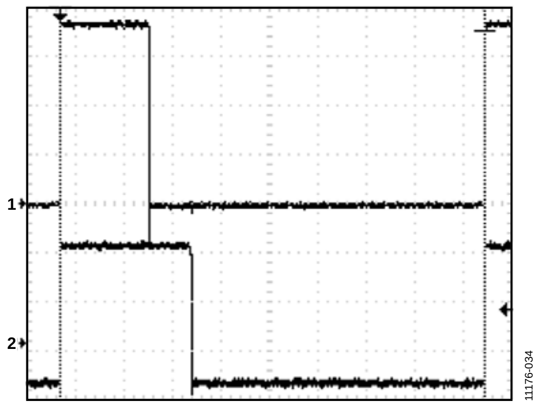


Figure 34. PWM Output on PWM0 and PWM1 Pins (PWM0 is Channel 2)

H-Bridge Mode

In H-bridge mode, the period and duty cycle of the four outputs are controlled using the Pair 0 registers: PWM0COM0, PWM0COM1, PWM0COM2, and PWM0LEN. In addition, the state of the output is controlled by PWMCON0 Bit 9, Bit 5, Bit 4, and Bit 2, as summarized in Table 267.

An example of H-bridge configuration is shown in Figure 35. Note that only PWM0 to PWM3 participate in H-bridge mode; other outputs (PWM4 to PWM7) do not and continue to generate standard mode output.

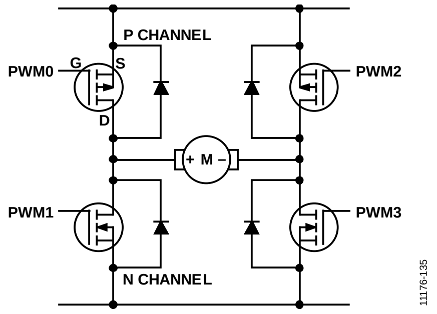


Figure 35. Example H-Bridge Configuration

Table 267. PWM Output in H-Bridge Mode

PWM Control Bits				PWM Outputs[1]				
ENA PWMCON0[9]	POINV PWMCON0[5]	HOFF PWMCON0[4]	DIR PWMCON0[2]	PWM0	PWM1	PWM2	PWM3	State of Motor
0	X	0	X	1 (Disable)	1 (Enable)	1 (Disable)	1 (Enable)	Brake
X	X	1	X	1 (Disable)	0 (Disable)	1 (Disable)	0 (Disable)	Free run
1	0	0	0	0 (Enable)	0 (Disable)	HS	LS	Move controlled by LS on PWM3
1	0	0	1	HS	LS	0 (Enable)	0 (Disable)	Move controlled by LS on PWM1
1	1	0	0	$\overline{LS}$	$\overline{HS}$	1 (Disable)	1 (Enable)	Move controlled by $\overline{LS}$ on PWM0
1	1	0	1	1 (Disable)	1 (Enable)	$\overline{LS}$	$\overline{HS}$	Move controlled by $\overline{LS}$ on PWM2

[1] HS = high side, LS = low side, $\overline{HS}$ = inverse of high side, $\overline{LS}$ = inverse of low side, as programmed in PWM0 registers.

PWM INTERRUPT GENERATION

PWM Trip Function Interrupt

When the PWM trip function is enabled (TRIPEN, PWMCON1[6]) and the PWM trip input signal goes low (falling edge), the PWM peripheral disables itself (PWMCON0[0] = 0). It also generates the PWM trip interrupt. The interrupt is cleared by setting PWMCLRI[4].

When using the PWM trip interrupt, clear the PWM interrupt before exiting the ISR. This prevents the generation of multiple interrupts.

PWM Output Pairs Interrupts

In standard mode, each PWM pair has a dedicated interrupt: IRQPWM0, IRQPWM1, IRQPWM2, IRQPWM3.

When the interrupt generation is enabled (PWMCON0[10]) and the counter value for Pair 0 changes from PWM0LEN to 0, it also generates the IRQPWM0 interrupt. The interrupt is cleared by setting PWMCLRI[0].

When the interrupt generation is enabled (PWMCON0[10]) and the counter value for Pair 1 changes from PWM1LEN to 0, it also generates the IRQPWM1 interrupt. The interrupt is cleared by setting PWMCLRI[1].

When the interrupt generation is enabled (PWMCON0[10]) and the counter value for Pair 2 changes from PWM2LEN to 0, it also generates the IRQPWM2 interrupt. The interrupt is cleared by setting PWMCLRI[2].

When the interrupt generation is enabled (PWMCON0[10]) and the counter value for Pair 3 changes from PWM3LEN to 0, it also generates the IRQPWM3 interrupt. The interrupt is cleared by setting PWMCLRI[3].

In H-bridge mode, Pair 0 and Pair 1 are used in the bridge configuration and generate on interrupt only, IRQPWM0. While Pair 0 and Pair 1 are in H-bridge mode, Pair 2 and Pair 3 can be used in standard mode and they can generate the IRQPWM2 and IRQPWM3 interrupts.

REGISTER SUMMARY: PWM

Table 268. PWM Register Summary

Address	Name	Description	Reset	RW
0x40024000	PWMCON0	PWM control register	0x0012	RW
0x40024004	PWMCON1	ADC conversion start and trip control register	0x0000	RW
0x40024008	PWMICLR	Hardware trip configuration register	0x0000	RW1C
0x40024010	PWM0COM0	Compare Register 0 for PWM0 and PWM1	0x0000	RW
0x40024014	PWM0COM1	Compare Register 1 for PWM0 and PWM1	0x0000	RW
0x40024018	PWM0COM2	Compare Register 2 for PWM0 and PWM1	0x0000	RW
0x4002401C	PWM0LEN	Period value register for PWM0 and PWM1	0x0000	RW
0x40024020	PWM1COM0	Compare Register 0 for PWM2 and PWM3	0x0000	RW
0x40024024	PWM1COM1	Compare Register 1 for PWM2 and PWM3	0x0000	RW
0x40024028	PWM1COM2	Compare Register 2 for PWM2 and PWM3	0x0000	RW
0x4002402C	PWM1LEN	Period value register for PWM2 and PWM3	0x0000	RW
0x40024030	PWM2COM0	Compare Register 0 for PWM4 and PWM5	0x0000	RW
0x40024034	PWM2COM1	Compare Register 1 for PWM4 and PWM5	0x0000	RW
0x40024038	PWM2COM2	Compare Register 2 for PWM4 and PWM5	0x0000	RW
0x4002403C	PWM2LEN	Period value register for PWM4 and PWM5	0x0000	RW
0x40024040	PWM3COM0	Compare Register 0 for PWM6 and PWM7	0x0000	RW
0x40024044	PWM3COM1	Compare Register 1 for PWM6 and PWM7	0x0000	RW
0x40024048	PWM3COM2	Compare Register 2 for PWM6 and PWM7	0x0000	RW
0x4002404C	PWM3LEN	Period value register for PWM6 and PWM7	0x0000	RW

REGISTER DETAILS: PWM

PWM Control Register

Address: 0x40024000, Reset: 0x0012, Name: PWMCON0

Table 269. Bit Descriptions for PWMCON0

Bits	Bit Name	Description	Reset	Access
15	SYNC	Set to enable PWM synchronization from the SYNC pin of the PWM. 0: Ignore transition from the SYNC pin 1: All PWM counters are reset on the next clock cycle after detection of a falling edge from SYNC pin	0x0	RW
14	PWM7INV	Set to invert PWM7 output.	0x0	RW
13	PWM5INV	Set to invert PWM5 output.	0x0	RW
12	PWM3INV	Set to invert PWM3 output.	0x0	RW
11	PWM1INV	Set to invert PWM1 output.	0x0	RW
10	PWMIEN	Set to enable interrupts for PWM.	0x0	RW
9	ENA	When HOFF=0 and HMODE=1, this serves as enable for Pair 0 and Pair 1. 0: disable Pair 0 and Pair 1 1: enable Pair 0 and Pair 1	0x0	RW
[8:6]	PWMCMP	PWM clock prescaler. Sets HCLK divider. 000: HCLK/2 001: HCLK/4 010: HCLK/8 011: HCLK/16 100: HCLK/32 101: HCLK/64 110: HCLK/128 111: HCLK/256	0x0	RW
5	POINV	Set to invert PWM outputs for Pair 0 and Pair 1 when PWM is in H-bridge mode.	0x0	RW
4	HOFF	Set to turn off the high-side for Pair 0 and Pair 1 when PWM is in H-bridge mode.	0x1	RW

Bits	Bit Name	Description	Reset	Access
3	LCOMP	Signal to load a new set of compare register values. In standard mode, this bit is cleared when the new values are loaded in the compare registers for all the channels. In H-bridge mode, this bit is not cleared; however, the user must write a value of 1 to this bit for the compare registers to be loaded. 0: use the values previously store in the compare and length registers 1: load the internal compare registers with values stored in the PWMxCOMx and PWMxLEN registers	0x0	RW
2	DIR	Direction control when PWM is in H-bridge mode. 0: PWM2 and PWM3 act as output signals while PWM0 and PWM1 are held low 1: PWM0 and PWM1 act as output signals while PWM2 and PWM3 are held low	0x0	RW
1	HMODE	Set to enable H-bridge mode.	0x1	RW
0	PWMEN	Master enable for PWM. 0: disable all PWM outputs 1: enable all PWM outputs	0x0	RW

ADC Conversion Start And Trip Control Register

Address: 0x40024004, Reset: 0x0000, Name: PWMCON1

Table 270. Bit Descriptions for PWMCON1

Bits	Bit Name	Description	Reset	Access
[15:7]	RESERVED	Reserved. Return 0 on reads.	0x00	Reserved
6	TRIP_EN	Set to enable PWM trip functionality.	0x0	RW
[5:0]	RESERVED	Reserved.	0x0	Reserved

Hardware Trip Configuration Register

Address: 0x40024008, Reset: 0x0000, Name: PWMICLR

Table 271. Bit Descriptions for PWMICLR

Bits	Bit Name	Description	Reset	Access
[15:5]	RESERVED	Reserved. Return 0 on reads.	0x000	Reserved
4	TRIP	Write a 1 to clear latched IRQPWMTrip interrupt. Returns 0 on reads.	0x0	RW1C
3	PWM3	Write a 1 to clear latched IRQPWM3 interrupt. Returns 0 on reads.	0x0	RW1C
2	PWM2	Write a 1 to clear latched IRQPWM2 interrupt. Returns 0 on reads.	0x0	RW1C
1	PWM1	Write a 1 to clear latched IRQPWM1 interrupt. Returns 0 on reads.	0x0	RW1C
0	PWM0	Write a 1 to clear latched IRQPWM0 interrupt. Returns 0 on reads.	0x0	RW1C

Compare Register 0 for PWM0 and PWM1

Address: 0x40024010, Reset: 0x0000, Name: PWM0COM0

Table 272. Bit Descriptions for PWM0COM0

Bits	Bit Name	Description	Reset	Access
[15:0]	COM0	Compare Register 0 data.	0x0	RW

Compare Register 1 for PWM0 and PWM1

Address: 0x40024014, Reset: 0x0000, Name: PWM0COM1

Table 273. Bit Descriptions for PWM0COM1

Bits	Bit Name	Description	Reset	Access
[15:0]	COM1	Compare Register 1 data.	0x0	RW

Compare Register 2 for PWM0 and PWM1

Address: 0x40024018, Reset: 0x0000, Name: PWM0COM2

Table 274. Bit Descriptions for PWM0COM2

Bits	Bit Name	Description	Reset	Access
[15:0]	COM2	Compare Register 2 data.	0x0	RW

Period Value Register for PWM0 and PWM1

Address: 0x4002401C, Reset: 0x0000, Name: PWM0LEN

Table 275. Bit Descriptions for PWM0LEN

Bits	Bit Name	Description	Reset	Access
[15:0]	LEN	Period value.	0x0	RW

Compare Register 0 for PWM2 and PWM3

Address: 0x40024020, Reset: 0x0000, Name: PWM1COM0

Table 276. Bit Descriptions for PWM1COM0

Bits	Bit Name	Description	Reset	Access
[15:0]	COM0	Compare Register 0 data.	0x0	RW

Compare Register 1 for PWM2 and PWM3

Address: 0x40024024, Reset: 0x0000, Name: PWM1COM1

Table 277. Bit Descriptions for PWM1COM1

Bits	Bit Name	Description	Reset	Access
[15:0]	COM1	Compare Register 1 data.	0x0	RW

Compare Register 2 for PWM2 and PWM3

Address: 0x40024028, Reset: 0x0000, Name: PWM1COM2

Table 278. Bit Descriptions for PWM1COM2

Bits	Bit Name	Description	Reset	Access
[15:0]	COM2	Compare Register 2 data.	0x0	RW

Period Value Register for PWM2 and PWM3

Address: 0x4002402C, Reset: 0x0000, Name: PWM1LEN

Table 279. Bit Descriptions for PWM1LEN

Bits	Bit Name	Description	Reset	Access
[15:0]	LEN	Period value.	0x0	RW

Compare Register 0 for PWM4 and PWM5

Address: 0x40024030, Reset: 0x0000, Name: PWM2COM0

Table 280. Bit Descriptions for PWM2COM0

Bits	Bit Name	Description	Reset	Access
[15:0]	COM0	Compare Register 0 data.	0x0	RW

Compare Register 1 for PWM4 and PWM5

Address: 0x40024034, Reset: 0x0000, Name: PWM2COM1

Table 281. Bit Descriptions for PWM2COM1

Bits	Bit Name	Description	Reset	Access
[15:0]	COM1	Compare Register 1 data.	0x0	RW

Compare Register 2 for PWM4 and PWM5

Address: 0x40024038, Reset: 0x0000, Name: PWM2COM2

Table 282. Bit Descriptions for PWM2COM2

Bits	Bit Name	Description	Reset	Access
[15:0]	COM2	Compare Register 2 data.	0x0	RW

Period Value Register for PWM4 and PWM5

Address: 0x4002403C, Reset: 0x0000, Name: PWM2LEN

Table 283. Bit Descriptions for PWM2LEN

Bits	Bit Name	Description	Reset	Access
[15:0]	LEN	Period value.	0x0	RW

Compare Register 0 for PWM6 and PWM7

Address: 0x40024040, Reset: 0x0000, Name: PWM3COM0

Table 284. Bit Descriptions for PWM3COM0

Bits	Bit Name	Description	Reset	Access
[15:0]	COM0	Compare Register 0 data.	0x0	RW

Compare Register 1 for PWM6 and PWM7

Address: 0x40024044, Reset: 0x0000, Name: PWM3COM1

Table 285. Bit Descriptions for PWM3COM1

Bits	Bit Name	Description	Reset	Access
[15:0]	COM1	Compare Register 1 data.	0x0	RW

Compare Register 2 for PWM6 and PWM7

Address: 0x40024048, Reset: 0x0000, Name: PWM3COM2

Table 286. Bit Descriptions for PWM3COM2

Bits	Bit Name	Description	Reset	Access
[15:0]	COM2	Compare Register 2 data.	0x0	RW

Period Value Register for PWM6 and PWM7

Address: 0x4002404C, Reset: 0x0000, Name: PWM3LEN

Table 287. Bit Descriptions for PWM3LEN

Bits	Bit Name	Description	Reset	Access
[15:0]	LEN	Period value.	0x0	RW

MDIO

MDIO FEATURES

The MDIO interface hardware can receive complete MDIO frames without software intervention. The MDIO interface hardware can also transmit complete MDIO frames without software intervention as long as the data to be sent is provided before receiving the turnaround bits (TA) of the read or post read increment address frame. To assist in using and supplying the relevant data, interrupts are generated at the end of every complete frame. If the PHYADR or DEVADD received does not match the expected values, the frame is not acted upon. Interrupts can also be generated after every valid PHYADR and DEVADD to permit more sophisticated control within frames.

MDIO OVERVIEW

This MDIO interface is designed for compliance with CFP management interface architecture (as per Draft CFP MSA Management Interface Specification Version 2.0 r07, June 30, 2011), as shown in Figure 36. This architecture includes an MDIO hardware interface to handle the serial communications. The transfer of data between this CFP MDIO interface and the MDIO defined memory blocks is done via software.

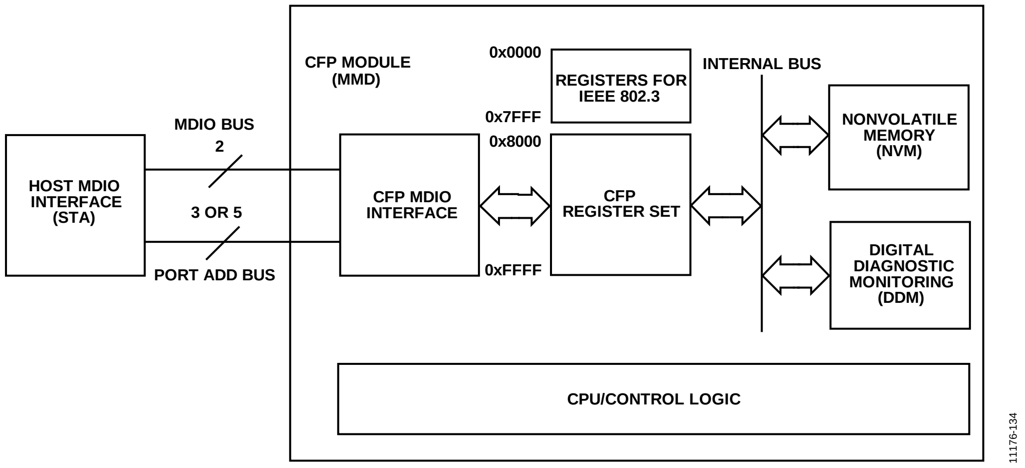
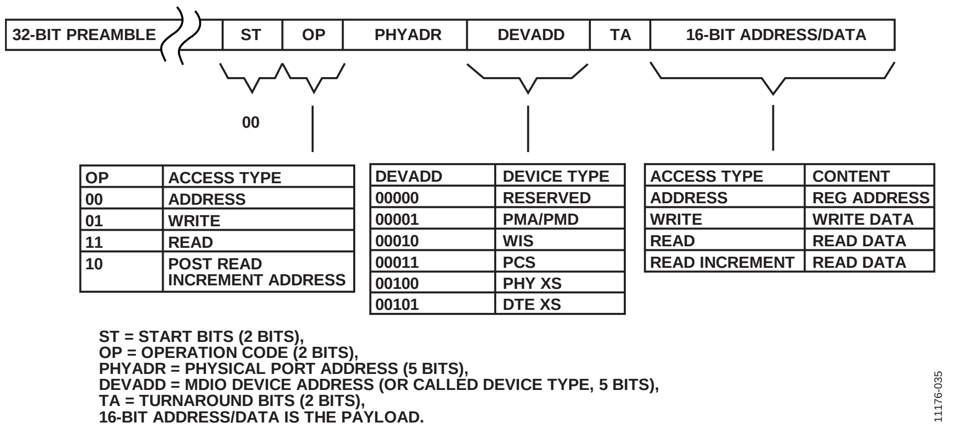


Figure 36. CFP Management Interface Architecture

MDIO OPERATION

MDIO Frame Structure

The MDIO interface uses the communication data frame structure defined in IEEE 802.3 Clause 45. The frame structure is shown in Figure 37. Each frame can be either an address frame or a data frame. The total bit length of each frame is 64, consisting of 32 bits preamble, and the frame command body. The command body consists of six portions, as illustrated in Figure 37. More information about the various frame types is provided in Table 288. All values are transmitted MSB first.



OP	ACCESS TYPE
00	ADDRESS
01	WRITE
11	READ
10	POST READ INCREMENT ADDRESS

DEVADD	DEVICE TYPE
00000	RESERVED
00001	PMA/PMD
00010	WIS
00011	PCS
00100	PHY XS
00101	DTE XS

ACCESS TYPE	CONTENT
ADDRESS	REG ADDRESS
WRITE	WRITE DATA
READ	READ DATA
READ INCREMENT	READ DATA

ST = START BITS (2 BITS),
OP = OPERATION CODE (2 BITS),
PHYADR = PHYSICAL PORT ADDRESS (5 BITS),
DEVADD = MDIO DEVICE ADDRESS (OR CALLED DEVICE TYPE, 5 BITS),
TA = TURNAROUND BITS (2 BITS),
16-BIT ADDRESS/DATA IS THE PAYLOAD.

Figure 37. MDIO Frame Structure

Table 288. Frame Details for Different Frame Types[1]

		Management Frame Fields							
Frame	**Idle**	**PRE**	**ST**	**OP**	**PHYADR**	**DEVADD**	**TA**	**Address/Data**	**Idle**
Write Address	Z	1…1	00	00	aaaaa	aaaaa	10	aaaaaaaaaaaaaaaa	Z
Write Data	Z	1…1	00	01	aaaaa	aaaaa	10	dddddddddddddddd	Z
Read Data	Z	1…1	00	11	aaaaa	aaaaa	z0	dddddddddddddddd	Z
Post Read Increment Address	Z	1…1	00	10	aaaaa	aaaaa	z0	dddddddddddddddd	Z

[1] During the idle condition, MDC and MDIO are not actively driven. During the second bit of TA and during the 16-bit data of the read and post read increment address add frames, MDIO is driven by the MMD. At all other times, ADC and MDIO are driven by the STA bits.

IDLE (Idle Condition)

The idle condition for the MDIO is a high-impedance state.

PRE (Preamble)

At the beginning of each transaction, the STA (host) sends a sequence of at least 32 contiguous bits sent one bit at a time to the MDIO, with 32 corresponding clock cycles on MDC, to establish the start of a frame.

ST (Start of Frame)

After the PRE the ST (consisting of two zero bits) indicates the start of the frame information.

OP (Operation Code)

The OP specifies the action to be taken, as described in Table 289.

Table 289. Operation Code

OP	Descriptions
00	Set the address for a subsequent write or read frame.
01	Write to the previously set address.
11	Read from the previously set address.
10	Read from the previously set address. Then increment the address. Note that user code must increment the address in the MDADR register.

PHYADR (Physical Address)

This address is five bits, allowing 32 unique addresses. The PHYADR is set either by 5 pins or by software.

DEVAD (Device Address)

This address is five bits and selects the device type. In the CFP standard, only MDIO Device Address 1 is supported.

TA (Turnaround)

This time is used to change from being driven by the STA to being driven by the MMD as per Figure 37.

Address/Data

The address/data field is 16 bits.

Typical Usage Sequence

Most of the MDIO interface is implemented in hardware, thus requiring minimal software effort.

1. Enable the MDIO onto the physical pins by writing 0x0555 to GP3CON.
2. Set the frame parameters by means of MDPHY, MDCON, and MDPIN.
3. Set the interrupts with MDIEN plus the required system interrupt settings.
4. At this stage, the address and write frames can be received in MDRXD and MDADR, respectively.
5. Data must be placed in MDTXD in advance of the read or post read increment address frame so that it can be automatically inserted for the frame.

No software intervention is required during any of the transmissions, although frame progress can be monitored with MDFRM during or upon completion of each frame. MDSTA should not be used to check frame progress because this MMR is automatically cleared and bits could be lost if read at an inappropriate time. If it is required to monitor frame progress, then the appropriate time to read MDSTA should be based on interrupts or by polling the MDIO bit in INTSETP0 in the interrupt system. MDSTA should be read only once per frame. MDIO must have the highest interrupt priority of all peripherals; otherwise, MDIO events are likely to be lost.

MDIO Interrupt Power-Up Register Write Sequence

To avoid false MDIO interrupts on startup, the order of register writes is important. The following is a code example showing how to correctly configure the MDIO interrupt on startup.

```
pADI_MDIO->MDCON = 0x0006;
pADI_MDIO->MDPHY = 0x0700;
sta = pADI_MDIO->MDSTA;              //read the MDSTA register to clear any interrupts
pADI_MDIO->MDIEN = 0x000F;
NVIC_ClearPendingIRQ(MDIO_IRQn);     //clear any pending interrupts in the Cortex
```

BLOCK SWITCHING

For MDIO applications, the system memory is separated into two flash blocks, as shown in Figure 38.

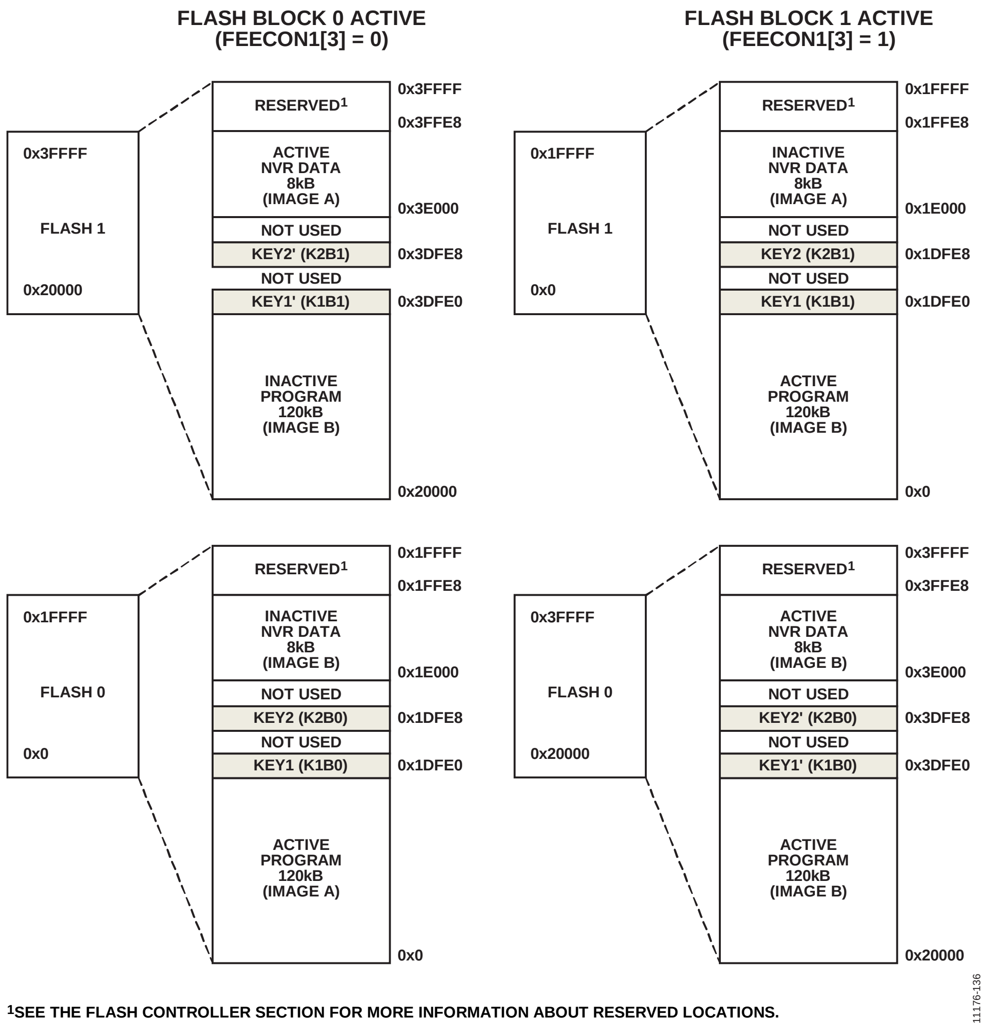


Figure 38. Memory Maps for MDIO Block Switching

When using the kernel flash block swapping feature, the stack pointer may not have the correct value at the start of user code; therefore, it is important that the first instructions in the firmware loads the stack pointer with the value stored at Address 0x0.

The following code is an example of these first instructions in µVision.

```
Reset_Handler    PROC
                 EXPORT  Reset_Handler             [WEAK]
              IMPORT  SystemInit
              IMPORT  __main
       MOVS  R0, #0x0 // Store value 0x0 in R0
       LDR R13, [R0]   //Load the stack pointer (R13) with the value at address 0x0
                 LDR     R0, =SystemInit
                 BLX     R0
                 LDR     R0, =__main
                 BX      R0
                 ENDP
```

Flash Block Partitioning

In the MDIO dual program image configuration, the Program Image A in Flash 0 and the NVR Data Block A in Flash 1 should be used together or, alternatively, the Program Image B in Flash 1 and the NVR Data Block B in Flash 0 should be used together. Because the data and code are in different flash blocks, the code can continue executing in the active program image while flash operations are performed on the associated NVR data flash block. Only one combination should be used at a time, which is known as the active combination. The other combination can be updated with new code if required, and by means of the block switching described, code can be made to execute from the new code. In the unswitched mode, Flash Block 0 is mapped from 0 to 0x1FFFF and Flash Block 1 is mapped from 0x20000 to 0x3FFFF. In the switched mode, Flash Block 1 is mapped from 0 to 0x1FFFF and Flash Block 0 is mapped from 0x20000 to 0x3FFFF. Code should be built to run in the address range 0 to 0x1FFFF and should only be run in this range. A mechanism is provided in the kernel to run from the appropriate flash block after any reset. This is described in the subsequent subsections of the MDIO section.

For a complete understanding, see Table 291, Figure 39, and Table 290 for additional information.

Program Image

The choice of which blocks are used is determined by the kernel and based on keys placed at the top of the two 120 kB program image blocks. There are six modes of operation:

- Debug mode
- Downloader mode (no valid code)
- Normal running from Program Image A (Flash 0)
- Trial run from Program Image A (Flash 0)
- Normal running from Program Image B (Flash 1)
- Trial run from Program Image B (Flash 1)

Each of these modes can be entered only via a reset. Every reset causes the kernel to run, and the kernel chooses the appropriate mode according to keys in the program images.

Each program image contains two keys.

For the active program image, Key1 at Address 0x1DFE0 has a numeric value that indicates the update number. The higher the Key1 value, the more recent the update.

Key2 at Address 0x1DFE8 is used to manage trial runs. A value of 0xFFFFFFFF (erased) indicates a new download. When trial run has passed, this must be indicated by changing the value to 0.

Key1’ of the other program image is at 0x3DFE0. Key2’ of the other program image is at 0x3DFE8.

The user program space CRCs can be stored at 0x1DFFC for Flash 0 and at 0x3DFFC for Flash 1. The CRC is not required as part of the block selection mechanism but should be included for increased robustness. The user code can check this CRC periodically.

Note that the keys are placed just below the 120 kB boundary, which is assumed to be the top of the program space. There is no technical reason why some code cannot be placed above this boundary or why some data cannot be placed below this boundary.

Debug Mode

If after a reset the kernel determines that the download pin (P2.3) is high, the kernel enters user code regardless of the keys. This mode is intended for debugging only

Choosing the Active Block

After any reset, the kernel chooses the active program image.

Figure 39 is the flowchart for choosing the active program image.

Initially, the kernel assumes that the program image with the larger Key1 is to be made active. If the associated Key2 is not 0, this code has not passed the trial run and should not yet be used. Instead, the kernel investigates using the other program image. If the Key2’ of the other program image is 0, that program image is chosen. Based on these decisions, the kernel then sets the active program image and exits to user code. If neither program image has a valid Key2, the kernel enters its own download mode.

Trial Run Mode

After user code is entered, the code checks whether a trial run or a normal run should be performed. A trial run is indicated if the active Key1 is less than the other Key1'. In a trial run, the old code first checks that the new program image is functioning correctly. The trial run starts in the old program image and performs initial checks, such as CRCs and other checks that the user deems necessary, on the new program image. The trial run can then continue by switching to the new image using Bit 3 of MMR FEECON1. It is recommended that the code that performs the switching be at a fixed location in Flash Page0 and be the same in all revisions. The code following the switching point should include sufficient identical code so that the CPU pipeline plus the flash lookahead buffers contain the expected code after switching. The user must also clear the memory cache to prevent old code from executing after the switch.

The trial run should also copy all necessary data from the old NVR to the new NVR. After the new flash blocks are shown to be correct, the user code must write 0 to Key2' of the new flash block to mark the block as good. The user code can then initiate normal operation. Alternatively, a software reset can be issued, and then the device enters normal mode in the new program image.

A reset may occur during a trial run, for instance, due to power loss or during a watchdog event due to program hanging or a deliberate software reset. In this case, a trial run restarts in the old code, and then the trial run code decides how to proceed.

Normal Mode

The user code must check whether a trial run or a normal run should be performed. A normal run is indicated if the active Key1 is larger than the other Key1. During normal operation, the MDIO master can send download information to the active user code so that new code is written to the other program image. Such a download must also write the new Key1' with a value of one more than the active Key1. The new Key2' must be left erased as 0xFFs. After the download, the part must be reset to allow a trial run to occur.

Typical Sequence

A typical sequence is shown in Table 290.

On a new device, the initial code can be downloaded via SW JTAG if P2.3 is held high during a reset; otherwise, the kernel enters its own downloader because there is no valid key. At the end of the download to Flash 0, Key1 is set to 1 and Key2 is set to 0.

After a reset, normal code is run from Flash 0 because its Key1 is greater than Key1' (0xFFs = −1) and its Key2 is 0. User code can receive MDIO frames instructing it to download code to Flash 1, which results in the new Key2' being erased and 2 being written to the new Key1'.

After a reset, the kernel activates Flash 0 for a trial run on the new code because Key2' of Flash 1 is 0xFFs. If the trial run passes, then the user code sets Key2 to 0 and issues a software reset.

After a reset, the kernel selects Flash 1 because its Key1 is still larger than the other Key1 and the active Key2 is 0. User code can receive MDIO frames instructing it to download code, including copying the NVR data block, to Flash 0, which results in Key2 being erased and 3 being written to Key1.

Changeover back to Flash 0 is then similar to the change to Flash 1.

Table 290. Example Block Switching Sequence

Number of Software Download	Key2 of Flash 0	Key1 of Flash 0	Key2 of Flash 1	Key1 of Flash 1	Status	Reset Required?
Not applicable	0xFFFFFFFF	0xFFFFFFFF	0xFFFFFFFF	0xFFFFFFFF	Initial startup	
1	0	1	0xFFFFFFFF	0xFFFFFFFF	Kernel has downloaded Code1 to Flash 0	Yes
1	0	1	0xFFFFFFFF	0xFFFFFFFF	Code1 normal execution in Flash 0	No
2	0	1	0xFFFFFFFF	2	Code1 has downloaded Code2 to Flash 1	Yes
2	0	1	0xFFFFFFFF	2	Code1 starts a trial run on Code2 in Flash 1	No
2	0	1	0	2	Code2 trial run complete	Yes
2	0	1	0	2	Code 2 normal execution in Flash 1	No
3	0xFFFFFFFF	3	0	2	Code2 has downloaded Code3 to Flash 0	Yes
3	0xFFFFFFFF	3	0	2	Code2 starts trial run on Code3 in Flash 0	No
3	0	3	0	2	Code3 trial mode complete	Yes
3	0	3	0	2	Code3 normal execution in Flash 0	No
4	0	3	0xFFFFFFFF	4	Code3 has downloaded Code4 to Flash 1	Yes
4	0	3	0xFFFFFFFF	4	Code3 starts a trial run on Code4 in Flash 1	No
4	0	3	0	4	Code 4 trial mode complete	Yes
4	0	3	0	4	Code4 normal execution in Flash 1	No
…		…		…	…	

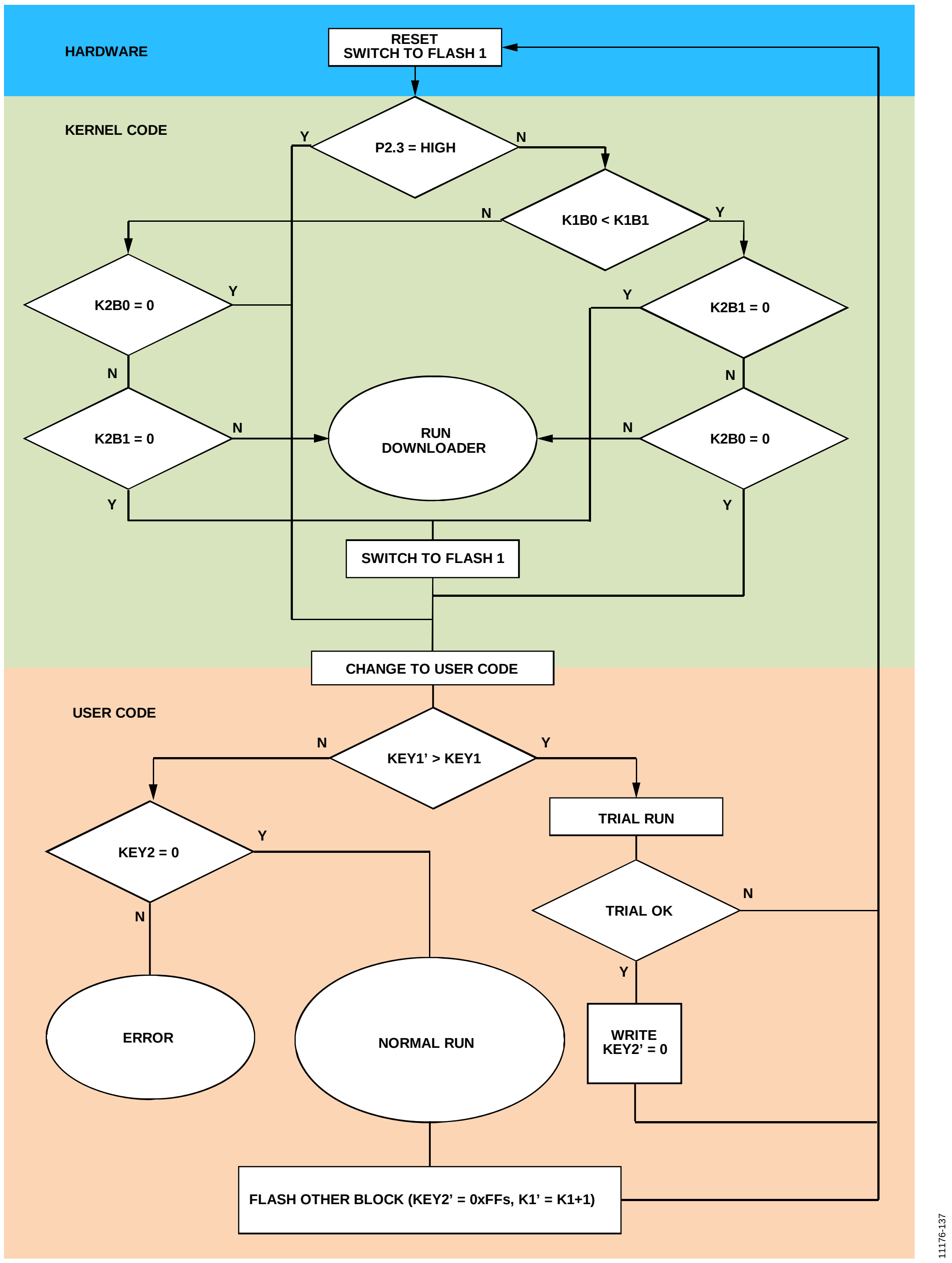


Figure 39. Flowchart for MDIO Memory Block Switching

Table 291. Definition of Keys

Key[1, 2]	Description
K1B0	Key1 in Flash Block 0 at 0x1DFE0.
K1B1	Key1 in Flash Block 1 at 0x3DFE0.
K2B0	Key2 in Flash Block 0 at 0x1DFE8.
K2B1	Key2 in Flash Block 1 at 0x3DFE8.
Key1	Key used to identify latest revision in active flash block at 0x1DFE0.
Key2	Key used for trial runs in active flash block at 0x1DFE8.
Key1′	Key1 for the other flash block at 0x3DFE0.
Key2′	Key2 for the other flash block at 0x3DFE8.
0xFFs	0xFFFFFFFFFFFFFFFF

[1] Key1, Key2, Key1′, and Key2′ refer to the keys as seen by the user.
[2] K1B0, K1B1, K2B0, and K2B1 refer to the keys as seen by the kernel before block switching occurs.

REGISTER SUMMARY: MDIO INTERFACE (MDIO)

Names and short descriptions of bits refer to the active state represented by a high (1) level unless explicitly enumerated.

Table 292. MDIO Register Summary

Address	Name	Description	Reset	Access
0x40005C00	MDCON	MDIO block control	0x0000	RW
0x40005C04	MDFRM	MDIO received frame control information	0x0000	R
0x40005C08	MDRXD	MDIO received data	0x000X	R
0x40005C0C	MDADR	MDIO received address	0x000X	R
0x40005C10	MDTXD	MDIO data for transmission	0x0000	RW
0x40005C14	MDPHY	MDIO PHYADDR software values and selection and DEVADD	0x0400	RW
0x40005C18	MDSTA	MDIO progress signaling through frame	0x0000	RW
0x40005C1C	MDIEN	MDIO interrupt enables	0x0000	RW
0x40005C20	MDPIN	MDIO read PHYADDR pins	0x0000	RW

REGISTER DETAILS: MDIO

MDIO Block Control Register

Address: 0x40005C00, Reset: 0x0000, Name: MDCON

Control for MDIO block.

Table 293. Bit Descriptions for MDCON

Bits	Bit Name	Description	Reset	Access
[15:3]	RESERVED	Reserved.	0x0	R
2	MD_DRV	0: MDIO drive open-drain. 1: MDIO drive push-pull.	0x0	RW
1	MD_PHY	0: MDIO PHY uses 5 bits. 1: MDIO PHY uses 3 bits. Unused PHY bits are ignored.	0x0	RW
0	MD_RST	Write 1 to reset MDIO block. Hardware immediately clears MD_RST again.	0x0	W

MDIO Received Frame Control Information Register

Address: 0x40005C04, Reset: 0x0000, Name: MDFRM

Contains control information of last frame received.

Table 294. Bit Descriptions for MDFRM

Bits	Bit Name	Description	Reset	Access
[15:12]	RESERVED	Reserved	0x0	R
[11:7]	MD_DEV	Received DEVADD	0x0	R
[6:2]	MD_PHY	Received PHYADR	0x0	R
[1:0]	MD_OP	Received OP 00: address frame 01: write frame 10: post read increment address frame 11: read frame	0x0	R

MDIO Received Data Register

Address: 0x40005C08, Reset: 0x000X, Name: MDRXD

Data received from last write frame.

Table 295. Bit Descriptions for MDRXD

Bits	Bit Name	Description	Reset	Access
[15:0]	MD_RXD	Received data	0xx	R

MDIO Received Address Register

Address: 0x40005C0C, Reset: 0x000X, Name: MDADR

Data received from last address frame.

Table 296. Bit Descriptions for MDADR

Bits	Bit Name	Description	Reset	Access
[15:0]	MD_ADR	Received address.	0xx	R

MDIO Data for Transmission Register

Address: 0x40005C10, Reset: 0x0000, Name: MDTXD

Data to be transmitted by next Data frame.

Table 297. Bit Descriptions for MDTXD

Bits	Bit Name	Description	Reset	Access
[15:0]	MD_TXD	Data that is to be transmitted by the next read or post read increment address frame. Before a read frame, the master sends an address frame to specify which data is to be read. After this address frame, the user software must place this requested data into MD_TXD before it is required by the read frame. The time available is at least 45 MDIO clock cycles being a minimum of the read frame preamble and up to 3 cycles before TA. This is equivalent to 900 CPU clock cycles.	0x0000	RW

MDIO PHYADDR Software Values and Selection and DEVADD Register

Address: 0x40005C14, Reset: 0x0400, Name: MDPHY

Sets expected values for control part of frame.

Table 298. Bit Descriptions for MDPHY

Bits	Bit Name	Description	Reset	Access
15	RESERVED	Reserved.	0x0	R
[14:10]	MD_DEVADD	Expected DEVADD. Normally 01.	0x1	RW
[9:5]	MD_PHYSEL	Selects expected PHYADR bits. For each of the 5 bits: 0: sets expected PHYADR.x = PRTADRx pin. 1: sets expected PHYADR.x = MD_PHYSW.x.	0x0	RW
[4:0]	MD_PHYSW	Software provided PHYADR bits. Chosen according to corresponding MD_PHYSEL bits.	0x0	RW

MDIO Progress Signaling Through Frame Register

Address: 0x40005C18, Reset: 0x0000, Name: MDSTA

Indicates progress through frame.

Table 299. Bit Descriptions for MDSTA

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
7	MD_PHYN	Set at end of PHYADR if PHYADR nonmatching. Cleared by reading MDSTA.	0x0	RC
6	MD_PHYM	Set at end of PHYADR if PHYADR matching. Cleared by reading MDSTA.	0x0	RC
5	MD_DEVN	Set at end of DEVADD if DEVADD nonmatching. Cleared by reading MDSTA.	0x0	RC
4	MD_DEVM	Set at end of DEVADD if DEVADD matching. Cleared by reading MDSTA.	0x0	RC
3	MD_RDF	Set at end of Read frame if DEVADD and PHYADR are matching. Cleared by reading MDSTA.	0x0	RC
2	MD_INCF	Set at end of post read increment address frame if DEVADD and PHYADR are matching. Cleared by reading MDSTA.	0x0	RC
1	MD_ADRF	Set at end of Address frame if DEVADD and PHYADR are matching. Cleared by reading MDSTA.	0x0	RC
0	MD_WRF	Set at end of Write frame if DEVADD and PHYADR are matching. Cleared by reading MDSTA.	0x0	RC

MDIO Interrupt Enables Register

Address: 0x40005C1C, Reset: 0x0000, Name: MDIEN

Enables interrupts on specified events.

Table 300. Bit Descriptions for MDIEN

Bits	Bit Name	Description	Reset	Access
[15:8]	RESERVED	Reserved.	0x0	R
7	MD_PHYNI	If set, interrupt is requested when MD_PHYN becomes active.	0x0	RW
6	MD_PHYMI	If set, interrupt is requested when MD_PHYM becomes active.	0x0	RW
5	MD_DEVNI	If set, interrupt is requested when MD_DEVN becomes active.	0x0	RW
4	MD_DEVMI	If set, interrupt is requested when MD_DEVM becomes active.	0x0	RW
3	MD_RDFI	If set, interrupt is requested when MD_RDF becomes active.	0x0	RW
2	MD_INCFI	If set, interrupt is requested when MD_INCF becomes active.	0x0	RW
1	MD_ADRI	If set, interrupt is requested when MD_ADRF becomes active.	0x0	RW
0	MD_WRFI	If set, interrupt is requested when MD_WRF becomes active.	0x0	RW

MDIO Read PHYADDR Pins Register

Address: 0x40005C20, Reset: 0x0000, Name: MDPIN

Reads the MDIO address pins.

Table 301. Bit Descriptions for MDPIN

Bits	Bit Name	Description	Reset	Access
[15:5]	RESERVED	Reserved	0x0	R
[4:0]	MD_PIN	Reads PRTADR pins	0x0	R

HARDWARE DESIGN CONSIDERATIONS

TYPICAL SYSTEM CONFIGURATION

Figure 40 shows a typical ADuCM320 configuration. The figure illustrates some of the hardware considerations. The four 0.47 µF capacitors on DVDD_REG1, DVDD_REG2, AVDD_REG1, and AVDD_REG2 should be placed as close as possible to the pins. VDD1 should either have a separate power supply or should be filtered from the other digital supply using an inductor bead and a resistor. The same applies to the AV_{DD} supply.

Decoupling capacitors are required between each power and associated ground as indicated in the ADuCM320 data sheet. These capacitors must be placed as close as possible to the pins and in such a way that the current paths do not interfere with one another. All GNDs need to be connected together in as close to a star connection as the layout allows.

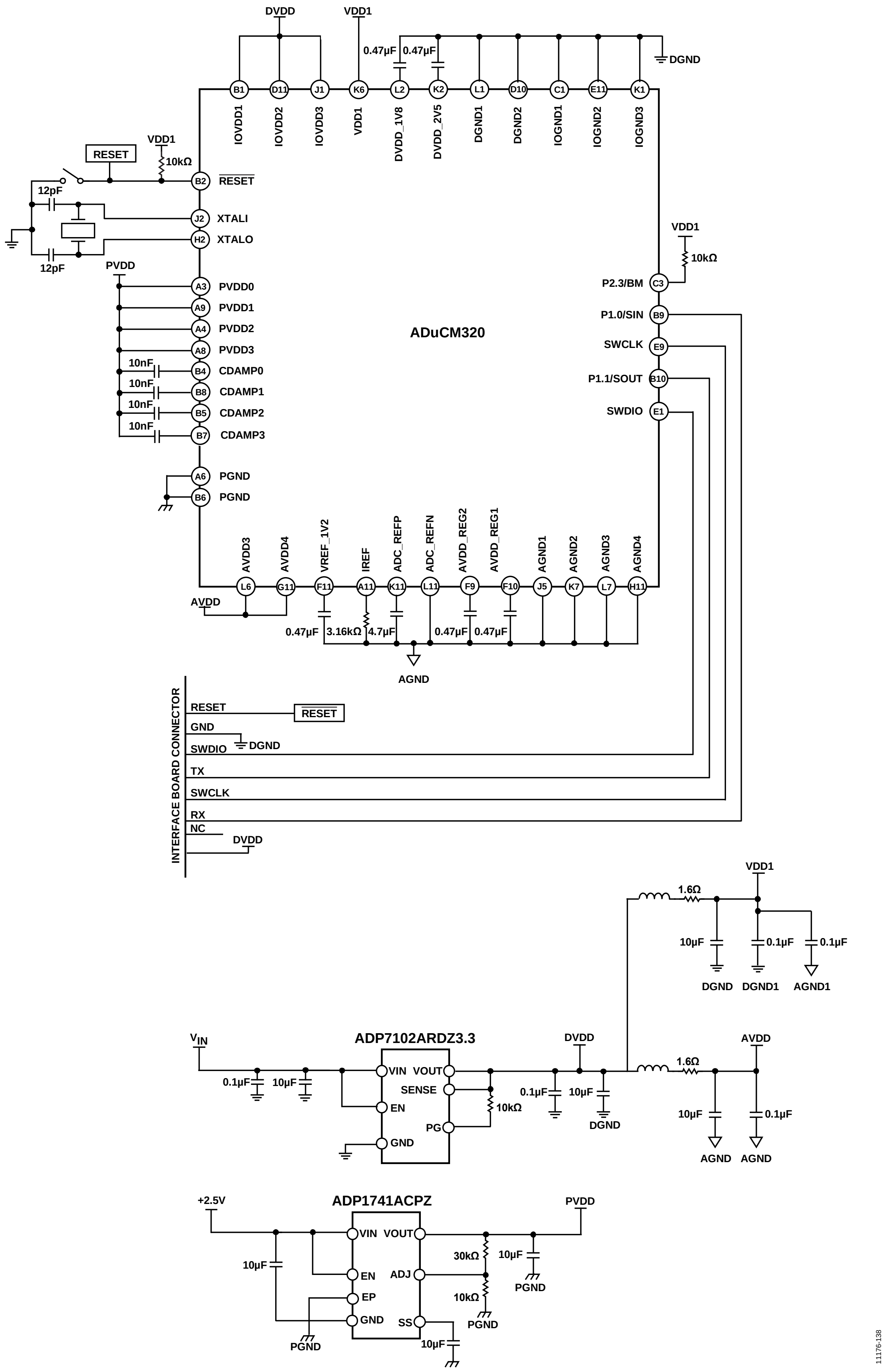


Figure 40. Typical System Configuration

SERIAL WIRE DEBUG INTERFACE

Serial wire debug (SWD) provides a debug port for pin limited packages. SWD replaces the 5-pin JTAG port with a clock (SWDCLK) and a single bidirectional data pin (SWDIO), providing all the normal JTAG debug and test functionality. SWDIO and SWCLK are overlaid on the TMS and TCK pins on the ARM 20-pin JTAG interface.

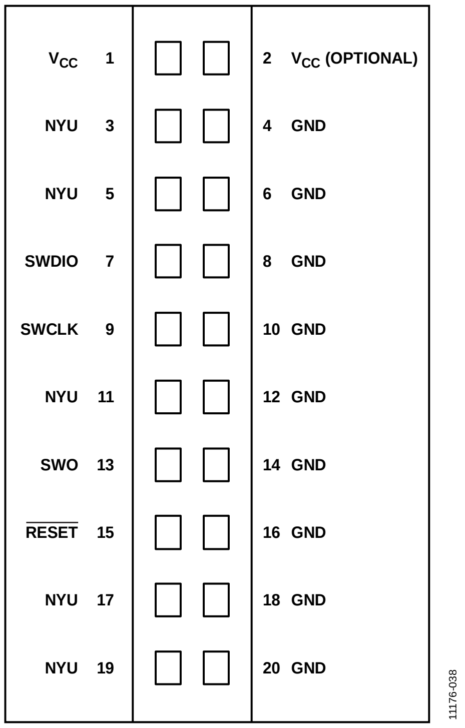


Figure 41. SWD 20-Pin Connector Pinout

Table 302. SWD Connections

Signal	Connect To
SWDIO	Data I/O pin. Use a 100 kΩ pull-up resistor to VCC.
SWO	No connect.
SWCLK	Clock pin. Use a 100 kΩ pull-up resistor to VCC.
VCC	Positive supply voltage—power supply for JTAG interface drivers.
GND	Digital ground.
RESET	No connect.