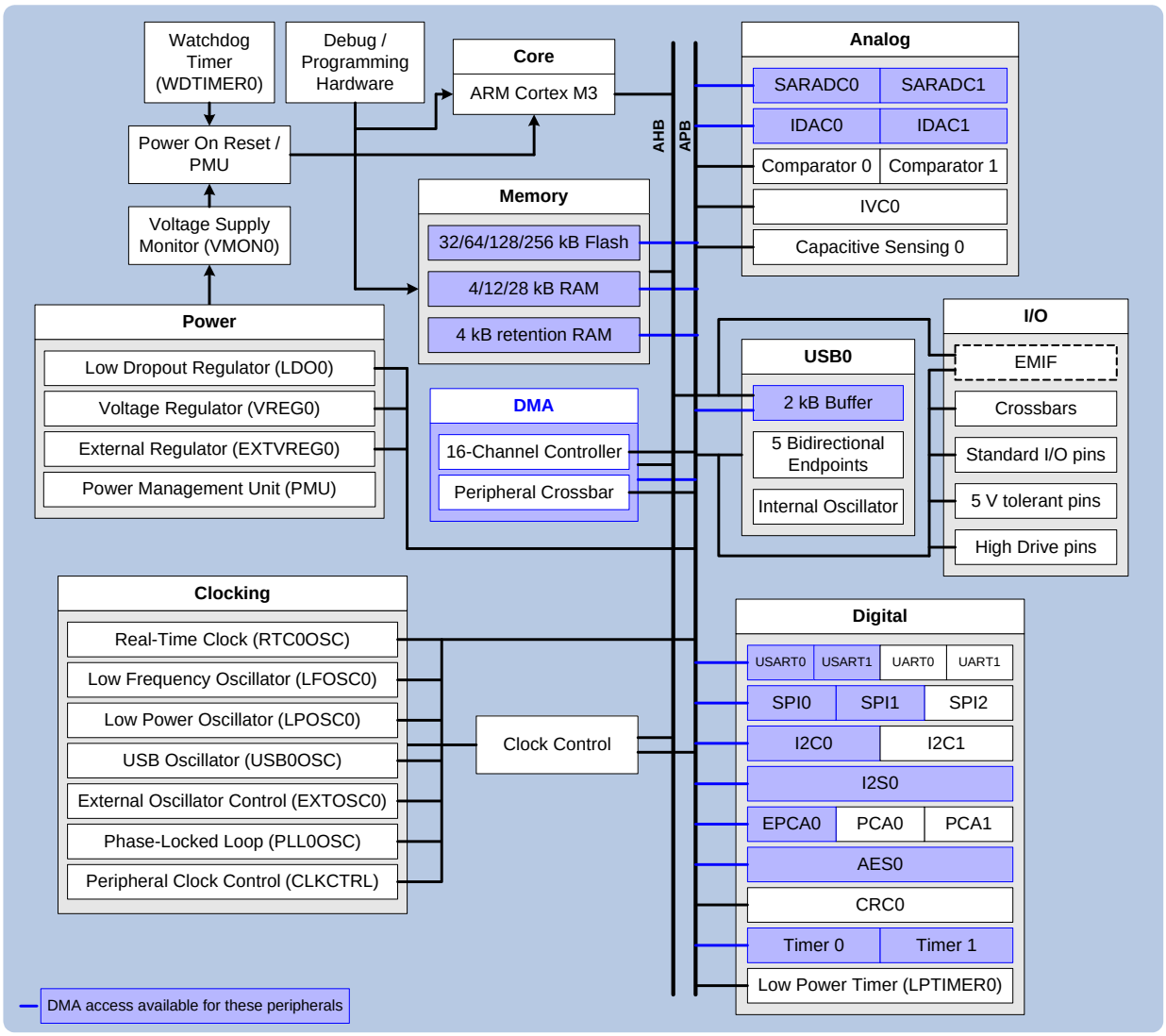
SiM3U1xx/SiM3C1xx Reference Manual

This reference manual accompanies several documents to provide the complete description of SiM3U1xx/SiM3C1xx devices, part of the Silicon Laboratories 32-bit ARM Cortex-M3 family of microcontrollers.

This document provides the detailed description for all peripherals available on all SiM3U1xx/SiM3C1xx devices. The peripheral mix varies across different members of the device families. Refer to the device data sheet for details on the specific peripherals available for each member of the device family. In the event that the device data sheet and this document contain conflicting information, the device data sheet should be considered the authoritative source.



Copyright © 2012 by Silicon Laboratories

Table of Contents

1. Related Documents and Conventions 11
1.1. Related Documents 11
1.2. Conventions 11
2. Memory Organization 12
2.1. Flash Region 13
2.2. RAM Region 14
2.3. Peripheral Region 15
2.4. External Memory 16
2.5. Cortex-M3 Internal Peripherals 16
3. SiM3U1xx/SiM3C1xx Register Memory Map 17
4. Interrupts 34
4.1. System Exceptions 34
4.2. Interrupt Vector Table 35
4.3. Priorities 40
5. Clock Control (CLKCTRL0) 43
5.1. Clock Control Features 43
5.2. CLKCTRL0 Registers 45
5.3. CLKCTRL0 Register Memory Map 53
6. Reset Sources (RSTSRC0) 55
6.1. Reset Sources Features 55
6.2. RSTSRC0 Registers 59
6.3. RSTSRC0 Register Memory Map 64
7. Register Security (LOCK0) 65
7.1. Security Features 65
7.2. LOCK0 Registers 66
7.3. LOCK0 Register Memory Map 72
8. Port I/O Configuration 73
8.1. Port Bank Description 73
8.2. Crossbars 74
8.3. Port Bank Standard (PBSTD) Features 85
8.4. Standard Modes of Operation 86
8.5. Assigning Standard Port Bank Pins to Analog and Digital Functions 86
8.6. Standard Port Match and Capacitive Sensing (CAPSENSE0) Activity Monitoring 87
8.7. Standard Port Bank Pulse Generator 87
8.8. High Drive (PBHD) Features 88
8.9. High Drive Modes of Operation 89
8.10. High Drive Configuration Procedure 90
8.11. High Drive Function Selection 90
8.12. Port Bank Security 92
8.13. Ports and Power Mode 9 92
8.14. Debugging Interfaces 93
8.15. External Memory Interface (EMIF) 94
8.16. External Interrupts 97
8.17. PBCFG0 Registers 99

8.18. PBCFG0 Register Memory Map 112
8.19. PBSTD0, PBSTD1, PBSTD2, and PBSTD3 Registers 114
8.20. PBSTDn Register Memory Map 125
8.21. PBHD4 Registers 128
8.22. PBHD4 Register Memory Map 140
9. Power 142
9.1. Power Modes 142
9.2. Power Management Unit (PMU0) 144
9.3. PMU0 Registers 147
9.4. PMU0 Register Memory Map 156
10. Core Voltage Regulator (LDO0) 158
10.1. Core Voltage Regulator Features 158
10.2. Functional Description 159
10.3. LDO0 Registers 160
10.4. LDO0 Register Memory Map 161
11. Device Identification (DEVICEID0) and Universally Unique Identifier 162
11.1. Device ID Features 162
11.2. Universally Unique Identifier (UUID) 162
11.3. DEVICEID0 Registers 163
11.4. DEVICEID0 Register Memory Map 167
12. Advanced Encryption Standard (AES0) 169
12.1. AES Features 169
12.2. Overview 170
12.3. Interrupts 170
12.4. Debug Mode 170
12.5. DMA Configuration and Usage 171
12.6. Using the AES0 Module for Electronic Codebook (ECB) 173
12.7. Using the AES0 Module for Cipher Block Chaining (CBC) 176
12.8. Using the AES0 Module for Counter (CTR) 183
12.9. Performing “In-Place” Ciphers 186
12.10. Using the AES0 Module in Software Mode 187
12.11. AES0 Registers 188
12.12. AES0 Register Memory Map 208
13. Capacitive Sensing (CAPSENSE0) 212
13.1. Capacitive Sensing Features 212
13.2. Overview 213
13.3. Measurement Overview 214
13.4. Conversion and Input Modes 216
13.5. Conversion Rate 217
13.6. Accumulation Modes 217
13.7. Measuring Multiple Channels in a Single Measurement 217
13.8. Pin Monitoring 217
13.9. Compare Threshold 218
13.10. Interrupts 218
13.11. Additional Options 218
13.12. Taking a Measurement 219

13.13.CAPSENSE0 Registers 220
13.14.CAPSENSE0 Register Memory Map 230
14. Comparator (CMP0 and CMP1) 232
14.1.Comparator Features 232
14.2.Overview 233
14.3.Inputs 233
14.4.Outputs 239
14.5.Response Time 240
14.6.Hysteresis 240
14.7.Interrupts and Flags 240
14.8.CMP0 and CMP1 Registers 241
14.9.CMPn Register Memory Map 246
15. Cyclic Redundancy Check (CRC0) 247
15.1.CRC Features 247
15.2.Overview 248
15.3.Interrupts 248
15.4.DMA Configuration and Usage 248
15.5.Byte-Level Bit Reversal and Byte Reordering 249
15.6.CRC0 Registers 252
15.7.CRC0 Register Memory Map 256
16. DMA Controller (DMACTRL0) 257
16.1.DMA Controller Features 257
16.2.Overview 259
16.3.Interrupts 259
16.4.Configuring a DMA Channel 259
16.5.DMA Channel Transfer Structures 260
16.6.Transfer Types 265
16.7.Data Requests 272
16.8.Masking Channels 273
16.9.Errors 273
16.10.Arbitration 274
16.11.DMACTRL0 Registers 275
16.12.DMACTRL0 Register Memory Map 309
17. DMA Crossbar (DMAXBAR0) 314
17.1.DMA Crossbar Features 314
17.2.Channel Priority 315
17.3.DMAXBAR0 Registers 316
17.4.DMAXBAR0 Register Memory Map 323
18. External Memory Interface (EMIF0) 324
18.1.EMIF Features 324
18.2.Overview 325
18.3.Signal Descriptions 326
18.4.Memory Interface 328
18.5.Non-Multiplexed Output Mode 330
18.6.Multiplexed Output Mode 330
18.7.Mixing Configurations 331

18.8.Transaction Timing 332
18.9.Idle and Off States 334
18.10.Additional Features 334
18.11.Configuring the External Memory Interface 334
18.12.EMIF0 Registers 335
18.13.EMIF0 Register Memory Map 337
18.14.EMIF0_IFx Registers 338
18.15.EMIFn_IFx Register Memory Map 348
19. External Oscillator (EXTOSC0) 350
19.1.External Oscillator Features 350
19.2.Introduction 351
19.3.External Crystal Oscillator 351
19.4.External CMOS Oscillator 352
19.5.External RC Oscillator 353
19.6.External C Oscillator 355
19.7.EXTOSC0 Registers 357
19.8.EXTOSC0 Register Memory Map 359
20. External Regulator (EXTVREG0) 360
20.1.External Regulator Features 360
20.2.Overview 361
20.3.Operating Modes 361
20.4.Current Sensing 363
20.5.Current Limiting 365
20.6.Foldback Limiting 366
20.7.Regulator Stability 367
20.8.Configuring the External Regulator 368
20.9.EXTVREG0 Registers 369
20.10.EXTVREG0 Register Memory Map 376
21. Flash Controller (FLASHCTRL0) 378
21.1.Flash Controller Features 378
21.2.Overview 379
21.3.Flash Read Control 379
21.4.Flash Write and Erase Control 380
21.5.FLASHCTRL0 Registers 383
21.6.FLASHCTRL0 Register Memory Map 389
22. Inter-Integrated Circuit Bus (I2C0 and I2C1) 391
22.1.I2C Features 391
22.2.I2C Protocol 392
22.3.Clocking 396
22.4.Operational Modes 396
22.5.Error Handling 407
22.6.Additional Features 408
22.7.Debug Mode 409
22.8.DMA Configuration and Usage 410
22.9.I2C0 and I2C1 Registers 415
22.10.I2Cn Register Memory Map 431

23. Integrated Interchip Sound (I2S0) 433
23.1. I2S Features 433
23.2. Signal Descriptions and Protocol Overview 434
23.3. Clocking 434
23.4. Clock (SCK) Signal Formatting 435
23.5. Word Select or Frame Sync (WS) Signal Formatting 435
23.6. Basic Data (SD) Signal Formatting 436
23.7. Left-Justified and Longer-Delay Formats 436
23.8. Time-Division Multiplexing (TDM) 437
23.9. Transmitter and Receiver 438
23.10. Interrupts and Flags 439
23.11. Debug Mode 439
23.12. Module Reset 439
23.13. DMA Usage and Configuration 440
23.14. I2S0 Registers 441
23.15. I2S0 Register Memory Map 460
24. Current Mode Digital-to-Analog Converter (IDAC0 and IDAC1) 463
24.1. IDAC Features 463
24.2. IDAC Setup 464
24.3. Using the IDAC in On-Demand Mode 468
24.4. Using the IDAC in Periodic FIFO-Only Mode 468
24.5. Using the IDAC in Periodic FIFO Wrap Mode 468
24.6. Using the IDAC in Periodic DMA Mode (on select IDAC peripherals only) 469
24.7. Adjusting the IDAC Output Current 469
24.8. Debug Mode 469
24.9. IDAC0 and IDAC1 Registers 470
24.10. IDACn Register Memory Map 479
25. Current-to-Voltage Converter (IVC0) 481
25.1. IVC Features 481
25.2. Functional Description 482
25.3. Configuration 482
25.4. IVC0 Registers 483
25.5. IVC0 Register Memory Map 485
26. Low Power Oscillator (LPOSC0) 486
26.1. Low Power Oscillator Features 486
26.2. Operation 487
26.3. LPOSC0 Registers 488
26.4. LPOSC0 Register Memory Map 489
27. Low Power Timer (LPTIMER0) 490
27.1. Low Power Timer Features 490
27.2. Clocking 491
27.3. Configuring the Timer 492
27.4. Interrupts 492
27.5. Output 493
27.6. Automatic Reset 493
27.7. Debug Mode 493

27.8.LPTIMER0 Registers 494
27.9.LPTIMER0 Register Memory Map 499
28. Enhanced Programmable Counter Array (EPCA0) 500
28.1.Enhanced Programmable Counter Array Features 500
28.2.Module Overview 502
28.3.Clocking 503
28.4.Interrupts 504
28.5.Outputs 504
28.6.Triggers 507
28.7.Operational Modes 508
28.8.DMA Configuration and Usage 519
28.9.EPCA0 Registers 521
28.10.EPCA0 Register Memory Map 534
28.11.EPCA0_CH0-5 Registers 536
28.12.EPCAn_CHx Register Memory Map 542
29. Programmable Counter Array (PCA0 and PCA1) 544
29.1.Programmable Counter Array Features 544
29.2.Module Overview 545
29.3.Clocking 546
29.4.Interrupts 547
29.5.Outputs 547
29.6.Operational Modes 548
29.7.PCA0 and PCA1 Registers 559
29.8.PCAn Register Memory Map 565
29.9.PCA0_CH0-1 and PCA1_CH0-1 Registers 566
29.10.PCAn_CHx Register Memory Map 572
30. Phase-Locked Loop (PLL0) 574
30.1.PLL Features 574
30.2.Overview 575
30.3.Interrupts 575
30.4.Output Modes 575
30.5.Additional Features 580
30.6.Advanced Setup Examples 582
30.7.PLL0 Registers 583
30.8.PLL0 Register Memory Map 592
31. Real Time Clock and Low Frequency Oscillator (RTC0) 593
31.1.RTC Features 593
31.2.Overview 594
31.3.Clocking 594
31.4.Accessing the Timer 600
31.5.Alarms 600
31.6.Interrupts 602
31.7.Usage Models 602
31.8.RTC0 Registers 603
31.9.RTC0 Register Memory Map 613
32. SAR Analog-to-Digital Converter (SARADC0 and SARADC1) 615
32.1.SARADC Features 615

32.2.Tracking and Conversion Time 617
32.3.Burst Mode 621
32.4.Channel Sequencer 622
32.5.Sample Sync Generator 626
32.6.Voltage Reference Configuration 627
32.7.Power Configuration 628
32.8. Data Output 629
32.9.Interrupts 632
32.10.DMA Configuration and Usage 634
32.11.SARADC0 and SARADC1 Registers 635
32.12.SARADCn Register Memory Map 656
33. Serial Peripheral Interface (SPI0, SPI1 and SPI2) 659
33.1.SPI Features 659
33.2.Signal Descriptions 660
33.3.Clocking 661
33.4.Signal Format 661
33.5.Master Mode Configurations and Data Transfer 664
33.6.Slave Mode Configurations and Data Transfer 666
33.7.Interrupts 668
33.8.Debug Mode 668
33.9.Module Reset 668
33.10.DMA Configuration and Usage 669
33.11.SPI0, SPI1 and SPI2 Registers 670
33.12.SPIn Register Memory Map 679
34. Sample Sync Generator (SSG0) 680
34.1.SSG Features 680
34.2.Phase Generator 681
34.3.Pulse Generator 681
34.4.SSG0 Registers 685
34.5.SSG0 Register Memory Map 689
35. Timers (TIMER0 and TIMER1) 690
35.1.Timer Features 690
35.2.Clocking 691
35.3.Configuring Timer Interrupts 692
35.4.Timer Synchronization 693
35.5.Timer Modes 694
35.6.TIMER0 and TIMER1 Registers 704
35.7.TIMERn Register Memory Map 711
36. Universal Synchronous/Asynchronous Receiver/Transmitter (USART0 and USART1) . 712
36.1.USART Features 712
36.2.Basic Data Format 714
36.3.Baud Rate 714
36.4.Interrupts 715
36.5.Flow Control 716
36.6.Debug Mode 716

36.7.Sending Data 717
36.8.Receiving Data 719
36.9.Synchronous Communications 720
36.10.Additional Communication Support 722
36.11.DMA Configuration and Usage 726
36.12.USART0 and USART1 Registers 727
36.13.USARTn Register Memory Map 745
37. Universal Asynchronous Receiver/Transmitter (UART0 and UART1) 747
37.1.UART Features 747
37.2.Basic Data Format 749
37.3.Baud Rate 749
37.4.Interrupts 750
37.5.Flow Control 751
37.6.Debug Mode 751
37.7.Sending Data 752
37.8.Receiving Data 754
37.9.Additional Communication Support 755
37.10.UART0 and UART1 Registers 759
37.11.UARTn Register Memory Map 775
38. Universal Serial Bus Controller (USB0) 777
38.1.USB Features 777
38.2.Overview 778
38.3.Clocking 778
38.4.Endpoints 779
38.5.USB Transceiver 779
38.6.FIFO Management 779
38.7.Function Addressing 780
38.8.Function Configuration and Control 781
38.9.Interrupts 782
38.10.The Serial Interface Engine 782
38.11.Endpoint 0 782
38.12.Configuring Endpoints 1-4 783
38.13.Controlling Endpoints 1-4 IN 784
38.14.Controlling Endpoints 1-4 OUT 785
38.15.DMA Configuration and Usage 787
38.16.USB0 Registers 791
38.17.USB0 Register Memory Map 818
38.18.USB0_EP1-4 Registers 822
38.19.USBn_EPx Register Memory Map 830
39. Voltage Supply Monitor (VMON0) 832
39.1.Voltage Supply Monitor Features 832
39.2.VDD Supply Monitoring 833
39.3.VREGIN Pin Monitoring 834
39.4.VMON0 Registers 835
39.5.VMON0 Register Memory Map 837
40. Voltage Reference and Temperature Sensor (VREF0) 838

40.1.Voltage Reference Features 838
40.2.Functional Description 839
40.3.VREF0 and Temperature Sensor Registers 840
40.4.VREF0 Register Memory Map 841
41. Voltage Regulator (VREG0) 842
41.1.Voltage Regulator Features 842
41.2.Operational Modes 843
41.3.Interrupts and Flags 844
41.4.VREG0 Registers 845
41.5.VREG0 Register Memory Map 847
42. Watchdog Timer (WDTIMER0) 848
42.1.Watchdog Timer Features 848
42.2.Overview 849
42.3.Lock and Key Interface 849
42.4.Setting the Early Warning and Reset Thresholds 850
42.5.Interrupts and Flags 851
42.6.Debug Mode 851
42.7.WDTIMER0 Registers 852
42.8.WDTIMER0 Register Memory Map 857
Document Change List 859
Contact Information 860

1. Related Documents and Conventions

1.1. Related Documents

1.1.1. SiM3U1xx and SiM3C1xx Data Sheets

The Silicon Laboratories SiM3U1xx and SiM3C1xx Data Sheets provide specific information for each device family, including electrical characteristics, mechanical characteristics, and ordering information.

1.1.2. Hardware Access Layer (HAL) API Description

The Silicon Laboratories Hardware Access Layer (HAL) API provides functions to modify and read each bit in the SiM3U1xx and SiM3C1xx devices. This description can be found in the SiM3xxxx HAL API Reference Manual.

1.1.3. ARM Cortex-M3 Reference Manual

The ARM-specific features like the Nested Vector Interrupt Controller are described in the ARM Cortex-M3 reference documentation. The online reference manual can be found online at the following link: http://infocenter.arm.com/help/topic/com.arm.doc.subset.cortexm.m3/index.html#cortexm3.

1.2. Conventions

The block diagrams in this document use the following formatting conventions:

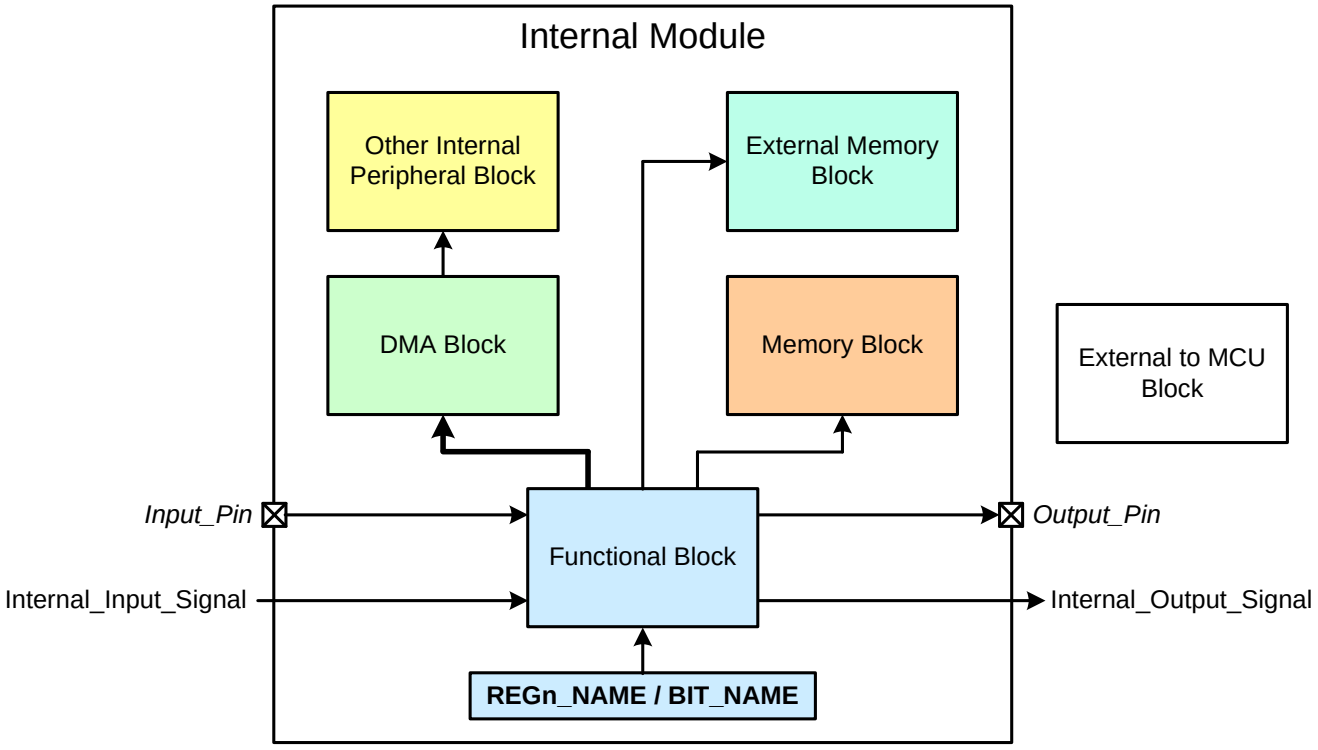


Figure 1.1. Block Diagram Conventions

2. Memory Organization

The memory organization of the SiM3U1xx/SiM3C1xx devices follows the standard ARM Cortex-M3 structure, shown in Figure 2.1. There is one 32-bit memory space shared amongst the flash, RAM, SiM3U1xx/SiM3C1xx Peripherals, External Memory, and M3 Peripherals. The unused memory addresses are reserved and should not be accessed.

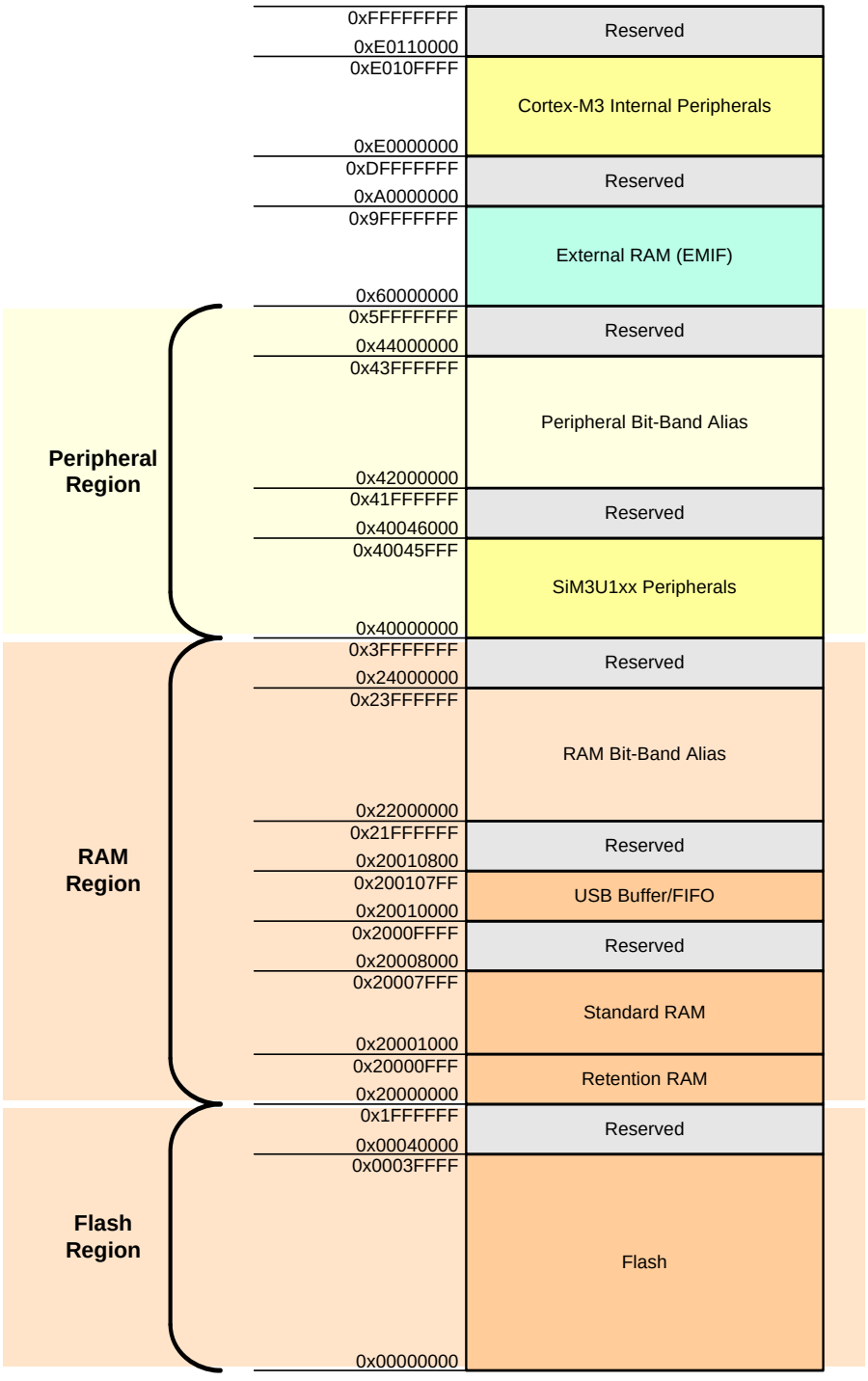


Figure 2.1. SiM3U1xx/SiM3C1xx Memory Map

2.1. Flash Region

The SiM3U1xx/SiM3C1xx devices implement 256, 128, 64, or 32 kB of flash which is accessible starting at 0x00000000. The flash can be read using standard ARM instructions. The FLASHCTRL0 module should be used to write and erase flash from firmware.

The flash block can be locked by writing to the lock word located at 0x0003FFFC. A value of 0xFFFFFFFF or 0x00000000 at this location will unlock the flash. Any other value written to this location will lock the entire flash from external (debugger) or firmware writes or reads until:

- An erase operation is initiated from firmware.
- An erase operation is initiated through the debug port (SWD/JTAG).
- Firmware writes 0x00000000 to the lock word.

The DMA can access all of flash.

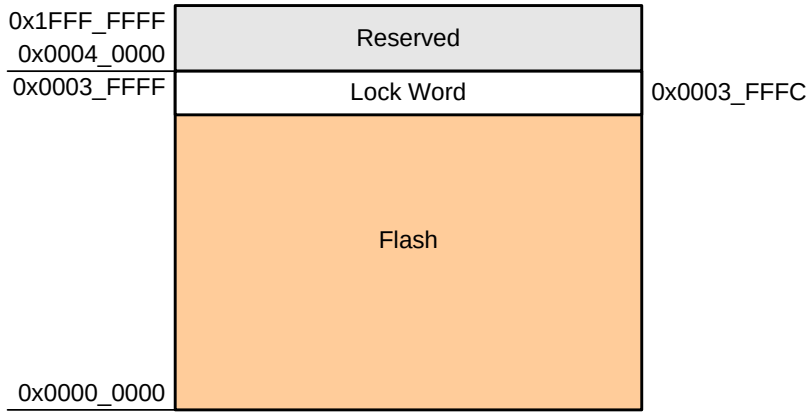


Figure 2.2. SiM3U16x Flash Memory Map (256 kB)

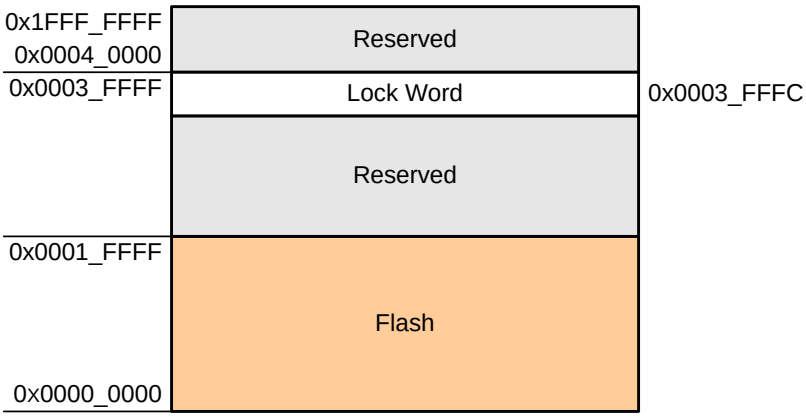


Figure 2.3. SiM3U15x Flash Memory Map (128 kB)

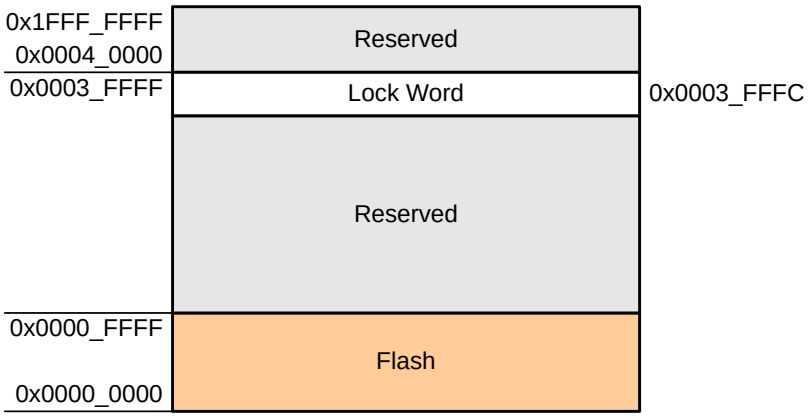


Figure 2.4. SiM3U14x Flash Memory Map (64 kB)

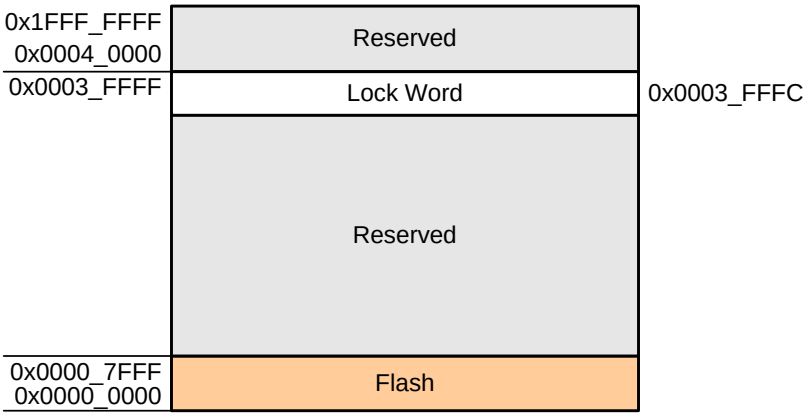


Figure 2.5. SiM3U13x Flash Memory Map (32 kB)

2.2. RAM Region

The RAM Region of SiM3U1xx/SiM3C1xx devices has the following areas: Standard RAM, Retention RAM, USB Buffer/FIFO RAM, and the RAM Bit-Banded Alias.

The Standard RAM region implements 28 kB (SiM3U16x and SiM3U15x), 12 kB (SiM3U14x), or 4 kB (SiM3U13x) of RAM and starts at location 0x20001000.

The SiM3U1xx/SiM3C1xx devices implement 4 kB of Retention RAM located at address 0x20000000. This RAM will retain it's value in Power Mode 9 as long as the VDD Monitor has not caused a reset.

The USB Buffer/FIFO RAM should be used by the USB0 Module and not accessed directly. If the USB0 Module is not in use, the clocks to this memory can be disabled to save power.

The RAM Bit-Band Alias region can be used to perform sets or clears of individual bits in the RAM. Each bit in the RAM region is represented by the least-significant bit at the word-aligned bit-band alias address.

2.3. Peripheral Region

The SiM3U1xx/SiM3C1xx peripheral registers are located starting at address 0x4000_0000. Registers for a specific module are typically located together in the peripheral region of memory to facilitate structure access from firmware. Each register may have up to four access methods, implemented as four separate locations in memory. The four possible access methods are named ALL, SET, CLR, and MSK.

The register's ALL access address is the primary access point for any register. Individual bits may be Read/Write (RW), Read-Only (RO), or Write-Only (WO). The ALL access address is implemented for all registers, and where absolute memory addresses are given in the documentation, they refer to the ALL address. For registers with write access, the ALL address will directly write all bits of the register. A read of the ALL address will read the current value in the register.

The SET and CLR addresses provide bit-wise, atomic write access to set and clear bits in the register without colliding with hardware. Writing a 1 to a bit in the SET address will set the corresponding bit, and writing a 1 to a bit in the CLR address will clear the corresponding bit. A write of 0 to either SET or CLR will have no effect on the corresponding bit. For registers implementing SET and CLR access methods, the SET address is at offset 0x4, and the CLR address is at offset 0x8 from the register's ALL access address. SET and CLR access are not implemented on every register.

The MSK address allows a write to a specific range of bits in the register. The upper 16 bits act as a mask for writing a value in the lower 16 bits of the register. For example, a write of 0x0F000400 to the MASK address would write a value of 4 to bits [11:8] of the register, while none of the rest of the bits are modified. For registers implementing the MSK access method, the MSK address is at offset 0xC from the registers ALL access address. MSK access is implemented for only a small set of registers which may require atomic, simultaneous writes of both 1s and 0s (such as port output registers).

Many control and status registers are supported by the SET and CLR access methods. The Peripheral Bit-Band Alias region can also be used to perform sets or clears of individual bits in the peripheral registers, which results in a read-modify-write operation on the bus. Each bit in the registers region is represented by the least-significant bit at the word-aligned bit-band alias address. When supported, it is recommended to use the SET and CLR registers instead of the Bit-Band Alias region to change individual bits in a register.

Each peripheral is discussed in detail in the corresponding chapter. The register map for the SiM3U1xx/SiM3C1xx devices can be found in "3. SiM3U1xx/SiM3C1xx Register Memory Map" . Detailed descriptions of each register and the bit fields within can be found in the specific peripheral section for that register.

2.4. External Memory

The EMIF Module accesses the External Memory Region. The EMIF module on the SiM3U1xx/SiM3C1xx devices supports two interfaces accessed at addresses 0x60000000 and 0x68000000, as shown in Figure 2.6.

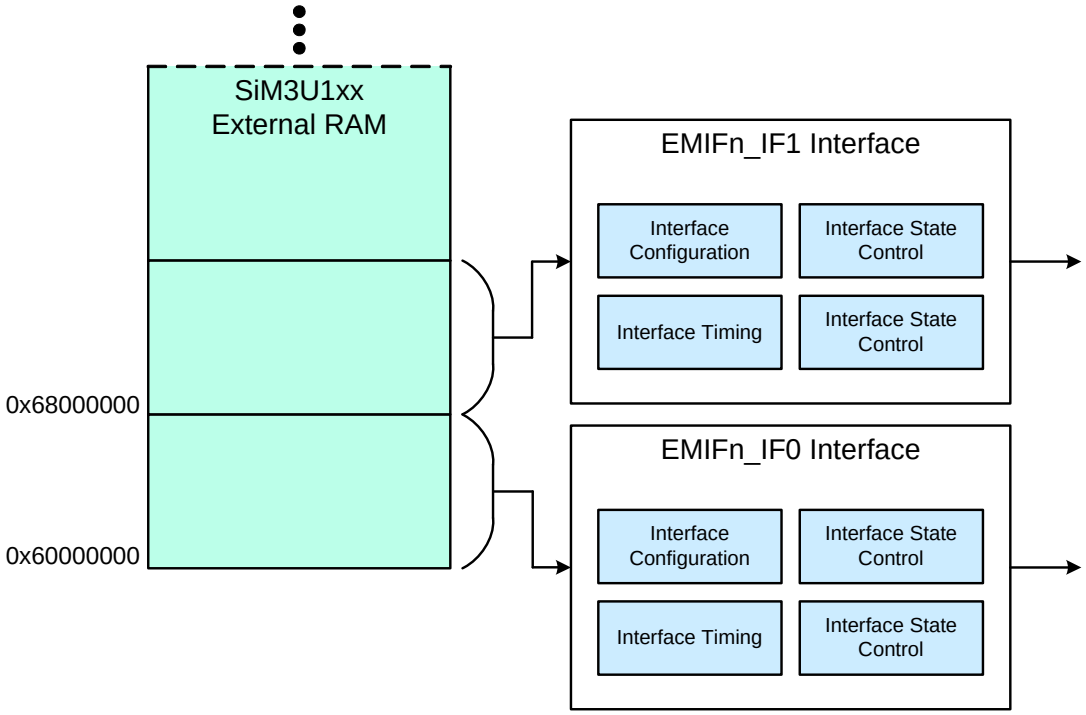


Figure 2.6. SiM3U1xx/SiM3C1xx External Memory Map

More information on the timing and configuration of this module can be found in the EMIF Module documentation.

2.5. Cortex-M3 Internal Peripherals

The Cortex-M3 Internal Peripherals space includes standard M3 functions, such as the NVIC and ETM. For more information on these functions of the ARM core, consult the ARM Cortex-M3 Reference Manual.

3. SiM3U1xx/SiM3C1xx Register Memory Map

This section details the register memory map for the SiM3U1xx/SiM3C1xx devices. Registers are listed in address order, beginning with 0x4000_0000

Table 3.1. Register Memory Map

Register Name	Title	Address (ALL Access)	SET (+0x4)	CLR(+0x8)	MSK (+0xC)
USART0 Registers					
USART0_CONFIG	Module Configuration	0x4000_0000	Y	Y	
USART0_MODE	Module Mode Select	0x4000_0010	Y	Y	
USART0_FLOWCN	Flow Control	0x4000_0020	Y	Y	
USART0_CONTROL	Module Control	0x4000_0030	Y	Y	
USART0_IPDELAY	Inter-Packet Delay	0x4000_0040			
USART0_BAUDRATE	Transmit and Receive Baud Rate	0x4000_0050			
USART0_FIFOCN	FIFO Control	0x4000_0060	Y	Y	
USART0_DATA	FIFO Input/Output Data	0x4000_0070			
USART1 Registers					
USART1_CONFIG	Module Configuration	0x4000_1000	Y	Y	
USART1_MODE	Module Mode Select	0x4000_1010	Y	Y	
USART1_FLOWCN	Flow Control	0x4000_1020	Y	Y	
USART1_CONTROL	Module Control	0x4000_1030	Y	Y	
USART1_IPDELAY	Inter-Packet Delay	0x4000_1040			
USART1_BAUDRATE	Transmit and Receive Baud Rate	0x4000_1050			
USART1_FIFOCN	FIFO Control	0x4000_1060	Y	Y	
USART1_DATA	FIFO Input/Output Data	0x4000_1070			
UART0 Registers					
UART0_CONFIG	Module Configuration	0x4000_2000	Y	Y	
UART0_MODE	Module Mode Select	0x4000_2010	Y	Y	
UART0_FLOWCN	Flow Control	0x4000_2020	Y	Y	
UART0_CONTROL	Module Control	0x4000_2030	Y	Y	
UART0_IPDELAY	Inter-Packet Delay	0x4000_2040			
UART0_BAUDRATE	Transmit and Receive Baud Rate	0x4000_2050			

Table 3.1. Register Memory Map

Register Name	Title	Address (ALL Access)	SET (+0x4)	CLR(+0x8)	MSK (+0xC)
UART0_FIFOCN	FIFO Control	0x4000_2060	Y	Y	
UART0_DATA	FIFO Input/Output Data	0x4000_2070			
UART1 Registers					
UART1_CONFIG	Module Configuration	0x4000_3000	Y	Y	
UART1_MODE	Module Mode Select	0x4000_3010	Y	Y	
UART1_FLOWCN	Flow Control	0x4000_3020	Y	Y	
UART1_CONTROL	Module Control	0x4000_3030	Y	Y	
UART1_IPDELAY	Inter-Packet Delay	0x4000_3040			
UART1_BAUDRATE	Transmit and Receive Baud Rate	0x4000_3050			
UART1_FIFOCN	FIFO Control	0x4000_3060	Y	Y	
UART1_DATA	FIFO Input/Output Data	0x4000_3070			
SPI0 Registers					
SPI0_DATA	Input/Output Data	0x4000_4000			
SPI0_CONTROL	Module Control	0x4000_4010	Y	Y	
SPI0_CONFIG	Module Configuration	0x4000_4020	Y	Y	
SPI0_CLKRATE	Module Clock Rate Control	0x4000_4030			
SPI0_FSTATUS	FIFO Status	0x4000_4040			
SPI1 Registers					
SPI1_DATA	Input/Output Data	0x4000_5000			
SPI1_CONTROL	Module Control	0x4000_5010	Y	Y	
SPI1_CONFIG	Module Configuration	0x4000_5020	Y	Y	
SPI1_CLKRATE	Module Clock Rate Control	0x4000_5030			
SPI1_FSTATUS	FIFO Status	0x4000_5040			

Table 3.1. Register Memory Map

Register Name	Title	Address (ALL Access)	SET (+0x4)	CLR(+0x8)	MSK (+0xC)
SPI2 Registers					
SPI2_DATA	Input/Output Data	0x4000_6000			
SPI2_CONTROL	Module Control	0x4000_6010	Y	Y	
SPI2_CONFIG	Module Configuration	0x4000_6020	Y	Y	
SPI2_CLKRATE	Module Clock Rate Control	0x4000_6030			
SPI2_FSTATUS	FIFO Status	0x4000_6040			
I2C0 Registers					
I2C0_CONTROL	Module Control	0x4000_9000	Y	Y	
I2C0_CONFIG	Module Configuration	0x4000_9010	Y	Y	
I2C0_SADDRESS	Slave Address	0x4000_9020			
I2C0_SMASK	Slave Address Mask	0x4000_9030			
I2C0_DATA	Data Buffer Access	0x4000_9040			
I2C0_TIMER	Timer Data	0x4000_9050			
I2C0_TIMERRL	Timer Reload Values	0x4000_9060			
I2C0_SCONFIG	SCL Signal Configuration	0x4000_9070			
I2C0_I2CDMA	DMA Configuration	0x4000_9080			
I2C1 Registers					
I2C1_CONTROL	Module Control	0x4000_A000	Y	Y	
I2C1_CONFIG	Module Configuration	0x4000_A010	Y	Y	
I2C1_SADDRESS	Slave Address	0x4000_A020			
I2C1_SMASK	Slave Address Mask	0x4000_A030			
I2C1_DATA	Data Buffer Access	0x4000_A040			
I2C1_TIMER	Timer Data	0x4000_A050			
I2C1_TIMERRL	Timer Reload Values	0x4000_A060			
I2C1_SCONFIG	SCL Signal Configuration	0x4000_A070			

Table 3.1. Register Memory Map

Register Name	Title	Address (ALL Access)	SET (+0x4)	CLR(+0x8)	MSK (+0xC)
EPCA0 Registers					
EPCA0_CH0_MODE	Channel Capture/Compare Mode	0x4000_E000			
EPCA0_CH0_CONTROL	Channel Capture/Compare Control	0x4000_E010	Y	Y	
EPCA0_CH0_CCAPV	Channel Compare Value	0x4000_E020			
EPCA0_CH0_CCAPVUPD	Channel Compare Update Value	0x4000_E030			
EPCA0_CH1_MODE	Channel Capture/Compare Mode	0x4000_E040			
EPCA0_CH1_CONTROL	Channel Capture/Compare Control	0x4000_E050	Y	Y	
EPCA0_CH1_CCAPV	Channel Compare Value	0x4000_E060			
EPCA0_CH1_CCAPVUPD	Channel Compare Update Value	0x4000_E070			
EPCA0_CH2_MODE	Channel Capture/Compare Mode	0x4000_E080			
EPCA0_CH2_CONTROL	Channel Capture/Compare Control	0x4000_E090	Y	Y	
EPCA0_CH2_CCAPV	Channel Compare Value	0x4000_E0A0			
EPCA0_CH2_CCAPVUPD	Channel Compare Update Value	0x4000_E0B0			
EPCA0_CH3_MODE	Channel Capture/Compare Mode	0x4000_E0C0			
EPCA0_CH3_CONTROL	Channel Capture/Compare Control	0x4000_E0D0	Y	Y	
EPCA0_CH3_CCAPV	Channel Compare Value	0x4000_E0E0			
EPCA0_CH3_CCAPVUPD	Channel Compare Update Value	0x4000_E0F0			
EPCA0_CH4_MODE	Channel Capture/Compare Mode	0x4000_E100			
EPCA0_CH4_CONTROL	Channel Capture/Compare Control	0x4000_E110	Y	Y	
EPCA0_CH4_CCAPV	Channel Compare Value	0x4000_E120			
EPCA0_CH4_CCAPVUPD	Channel Compare Update Value	0x4000_E130			
EPCA0_CH5_MODE	Channel Capture/Compare Mode	0x4000_E140			
EPCA0_CH5_CONTROL	Channel Capture/Compare Control	0x4000_E150	Y	Y	
EPCA0_CH5_CCAPV	Channel Compare Value	0x4000_E160			
EPCA0_CH5_CCAPVUPD	Channel Compare Update Value	0x4000_E170			
EPCA0_MODE	Module Operating Mode	0x4000_E180			
EPCA0_CONTROL	Module Control	0x4000_E190	Y	Y	

Table 3.1. Register Memory Map

Register Name	Title	Address (ALL Access)	SET (+0x4)	CLR(+0x8)	MSK (+0xC)
EPCA0_STATUS	Module Status	0x4000_E1A0	Y	Y	
EPCA0_COUNTER	Module Counter/Timer	0x4000_E1B0			
EPCA0_LIMIT	Module Upper Limit	0x4000_E1C0			
EPCA0_LIMITUPD	Module Upper Limit Update Value	0x4000_E1D0			
EPCA0_DTIME	Phase Delay Time	0x4000_E1E0			
EPCA0_DTARGET	DMA Transfer Target	0x4000_E200			
PCA0 Registers					
PCA0_CH0_MODE	Channel Capture/Compare Mode	0x4000_F000			
PCA0_CH0_CONTROL	Channel Capture/Compare Control	0x4000_F010	Y	Y	
PCA0_CH0_CCAPV	Channel Compare Value	0x4000_F020			
PCA0_CH0_CCAPVUPD	Channel Compare Update Value	0x4000_F030			
PCA0_CH1_MODE	Channel Capture/Compare Mode	0x4000_F040			
PCA0_CH1_CONTROL	Channel Capture/Compare Control	0x4000_F050	Y	Y	
PCA0_CH1_CCAPV	Channel Compare Value	0x4000_F060			
PCA0_CH1_CCAPVUPD	Channel Compare Update Value	0x4000_F070			
PCA0_MODE	Module Operating Mode	0x4000_F180			
PCA0_CONTROL	Module Control	0x4000_F190	Y	Y	
PCA0_STATUS	Module Status	0x4000_F1A0	Y	Y	
PCA0_COUNTER	Module Counter/Timer	0x4000_F1B0			
PCA0_LIMIT	Module Counter/Timer Upper Limit	0x4000_F1C0			
PCA1 Registers					
PCA1_CH0_MODE	Channel Capture/Compare Mode	0x4001_0000			
PCA1_CH0_CONTROL	Channel Capture/Compare Control	0x4001_0010	Y	Y	
PCA1_CH0_CCAPV	Channel Compare Value	0x4001_0020			
PCA1_CH0_CCAPVUPD	Channel Compare Update Value	0x4001_0030			
PCA1_CH1_MODE	Channel Capture/Compare Mode	0x4001_0040			
PCA1_CH1_CONTROL	Channel Capture/Compare Control	0x4001_0050	Y	Y	

Table 3.1. Register Memory Map

Register Name	Title	Address (ALL Access)	SET (+0x4)	CLR(+0x8)	MSK (+0xC)
PCA1_CH1_CCAPV	Channel Compare Value	0x4001_0060			
PCA1_CH1_CCAPVUPD	Channel Compare Update Value	0x4001_0070			
PCA1_MODE	Module Operating Mode	0x4001_0180			
PCA1_CONTROL	Module Control	0x4001_0190	Y	Y	
PCA1_STATUS	Module Status	0x4001_01A0	Y	Y	
PCA1_COUNTER	Module Counter/Timer	0x4001_01B0			
PCA1_LIMIT	Module Counter/Timer Upper Limit	0x4001_01C0			
TIMER0 Registers					
TIMER0_CONFIG	High and Low Timer Configuration	0x4001_4000	Y	Y	
TIMER0_CLKDIV	Module Clock Divider Control	0x4001_4010			
TIMER0_COUNT	Timer Value	0x4001_4020			
TIMER0_CAPTURE	Timer Capture/Reload Value	0x4001_4030			
TIMER1 Registers					
TIMER1_CONFIG	High and Low Timer Configuration	0x4001_5000	Y	Y	
TIMER1_CLKDIV	Module Clock Divider Control	0x4001_5010			
TIMER1_COUNT	Timer Value	0x4001_5020			
TIMER1_CAPTURE	Timer Capture/Reload Value	0x4001_5030			
USB0 Registers					
USB0_FADDR	Function Address	0x4001_8000			
USB0_POWER	Power Control	0x4001_8010			
USB0_IOINT	IN/OUT Endpoint Interrupt Flags	0x4001_8020		Y	
USB0_CMINT	Common Interrupt Flags	0x4001_8030		Y	
USB0_IOINTE	IN/OUT Endpoint Interrupt Control	0x4001_8040			
USB0_CMINTEPE	Common Interrupt and Endpoint Control	0x4001_8050			
USB0_CRCONTROL	Clock Recovery Control	0x4001_8060			
USB0_FRAME	Frame Number	0x4001_8070			
USB0_TCONTROL	Transceiver Control	0x4001_8200			

Table 3.1. Register Memory Map

Register Name	Title	Address (ALL Access)	SET (+0x4)	CLR(+0x8)	MSK (+0xC)
USB0_CLKSEL	Module Clock Select	0x4001_8300			
USB0_OSCCONTROL	Oscillator Control	0x4001_8310	Y	Y	
USB0_AFADJUST	Oscillator Additional Frequency Adjust	0x4001_8320	Y	Y	
USB0_FADJUST	Oscillator Frequency Adjust	0x4001_8330			
USB0_DMAFIFO	DMA Data FIFO Access	0x4001_8400			
USB0_DMACONTROL	DMA Control	0x4001_8410			
USB0_EP0CONTROL	Endpoint 0 Control	0x4001_8810			
USB0_EP0COUNT	Endpoint 0 Data Count	0x4001_8820			
USB0_EP0FIFO	Endpoint 0 Data FIFO Access	0x4001_8830			
USB0_EP1_EPMPSIZE	Endpoint Maximum Packet Size	0x4001_8880			
USB0_EP1_EPCONTROL	Endpoint Control	0x4001_8890			
USB0_EP1_EPCOUNT	Endpoint Data Count	0x4001_88A0			
USB0_EP1_EPFIFO	Endpoint Data FIFO Access	0x4001_88B0			
USB0_EP2_EPMPSIZE	Endpoint Maximum Packet Size	0x4001_8900			
USB0_EP2_EPCONTROL	Endpoint Control	0x4001_8910			
USB0_EP2_EPCOUNT	Endpoint Data Count	0x4001_8920			
USB0_EP2_EPFIFO	Endpoint Data FIFO Access	0x4001_8930			
USB0_EP3_EPMPSIZE	Endpoint Maximum Packet Size	0x4001_8980			
USB0_EP3_EPCONTROL	Endpoint Control	0x4001_8990			
USB0_EP3_EPCOUNT	Endpoint Data Count	0x4001_89A0			
USB0_EP3_EPFIFO	Endpoint Data FIFO Access	0x4001_89B0			
USB0_EP4_EPMPSIZE	Endpoint Maximum Packet Size	0x4001_8A00			
USB0_EP4_EPCONTROL	Endpoint Control	0x4001_8A10			
USB0_EP4_EPCOUNT	Endpoint Data Count	0x4001_8A20			
USB0_EP4_EPFIFO	Endpoint Data FIFO Access	0x4001_8A30			

Table 3.1. Register Memory Map

Register Name	Title	Address (ALL Access)	SET (+0x4)	CLR(+0x8)	MSK (+0xC)
SARADC0 Registers					
SARADC0_CONFIG	Module Configuration	0x4001_A000	Y	Y	
SARADC0_CONTROL	Measurement Control	0x4001_A010	Y	Y	
SARADC0_SQ7654	Channel Sequencer Time Slots 4-7 Setup	0x4001_A020			
SARADC0_SQ3210	Channel Sequencer Time Slots 0-3 Setup	0x4001_A030			
SARADC0_CHAR32	Conversion Characteristic 2 and 3 Setup	0x4001_A040	Y	Y	
SARADC0_CHAR10	Conversion Characteristic 0 and 1 Setup	0x4001_A050	Y	Y	
SARADC0_DATA	Output Data Word	0x4001_A060			
SARADC0_WCLIMITS	Window Comparator Limits	0x4001_A070			
SARADC0_ACC	Accumulator Initial Value	0x4001_A080			
SARADC0_STATUS	Module Status	0x4001_A090	Y	Y	
SARADC0_FIFOSTATUS	FIFO Status	0x4001_A0A0			
SARADC1 Registers					
SARADC1_CONFIG	Module Configuration	0x4001_B000	Y	Y	
SARADC1_CONTROL	Measurement Control	0x4001_B010	Y	Y	
SARADC1_SQ7654	Channel Sequencer Time Slots 4-7 Setup	0x4001_B020			
SARADC1_SQ3210	Channel Sequencer Time Slots 0-3 Setup	0x4001_B030			
SARADC1_CHAR32	Conversion Characteristic 2 and 3 Setup	0x4001_B040	Y	Y	
SARADC1_CHAR10	Conversion Characteristic 0 and 1 Setup	0x4001_B050	Y	Y	
SARADC1_DATA	Output Data Word	0x4001_B060			
SARADC1_WCLIMITS	Window Comparator Limits	0x4001_B070			
SARADC1_ACC	Accumulator Initial Value	0x4001_B080			
SARADC1_STATUS	Module Status	0x4001_B090	Y	Y	
SARADC1_FIFOSTATUS	FIFO Status	0x4001_B0A0			
SSG0 Registers					
SSG0_CONFIG	Module Configuration	0x4001_E000			
SSG0_CONTROL	Module Control	0x4001_E010	Y	Y	

Table 3.1. Register Memory Map

Register Name	Title	Address (ALL Access)	SET (+0x4)	CLR(+0x8)	MSK (+0xC)
CMP0 Registers					
CMP0_CONTROL	Module Control	0x4001_F000	Y	Y	
CMP0_MODE	Input and Module Mode	0x4001_F010	Y	Y	
CMP1 Registers					
CMP1_CONTROL	Module Control	0x4002_0000	Y	Y	
CMP1_MODE	Input and Module Mode	0x4002_0010	Y	Y	
CAPSENSE0 Registers					
CAPSENSE0_CONTROL	Module Control	0x4002_3000	Y	Y	
CAPSENSE0_MODE	Measurement Mode	0x4002_3010	Y	Y	
CAPSENSE0_DATA	Measurement Data	0x4002_3020			
CAPSENSE0_SCAN	Channel Scan Enable	0x4002_3030			
CAPSENSE0_CSTH	Compare Threshold	0x4002_3040			
CAPSENSE0_MUX	Mux Channel Select	0x4002_3050			
EMIF0 Registers					
EMIF0_CONTROL	Module Control	0x4002_6000	Y	Y	
EMIF0_STATUS	Module Status	0x4002_6020			
EMIF0_IF0_CONFIG	Interface Configuration	0x4002_6080	Y	Y	
EMIF0_IF0_IFRT	Interface Read Timing	0x4002_6090			
EMIF0_IF0_IFWT	Interface Write Timing	0x4002_60A0			
EMIF0_IF0_IFRCST	Interface Read Control States	0x4002_60B0			
EMIF0_IF0_IFWCST	Interface Write Control States	0x4002_60C0			
EMIF0_IF1_CONFIG	Interface Configuration	0x4002_6100	Y	Y	
EMIF0_IF1_IFRT	Interface Read Timing	0x4002_6110			
EMIF0_IF1_IFWT	Interface Write Timing	0x4002_6120			
EMIF0_IF1_IFRCST	Interface Read Control States	0x4002_6130			
EMIF0_IF1_IFWCST	Interface Write Control States	0x4002_6140			

Table 3.1. Register Memory Map

Register Name	Title	Address (ALL Access)	SET (+0x4)	CLR(+0x8)	MSK (+0xC)
AES0 Registers					
AES0_CONTROL	Module Control	0x4002_7000	Y	Y	
AES0_XFRSIZE	Number of Blocks	0x4002_7010			
AES0_DATAFIFO	Input/Output Data FIFO Access	0x4002_7020			
AES0_XORFIFO	XOR Data FIFO Access	0x4002_7030			
AES0_HWKEY0	Hardware Key Word 0	0x4002_7040			
AES0_HWKEY1	Hardware Key Word 1	0x4002_7050			
AES0_HWKEY2	Hardware Key Word 2	0x4002_7060			
AES0_HWKEY3	Hardware Key Word 3	0x4002_7070			
AES0_HWKEY4	Hardware Key Word 4	0x4002_7080			
AES0_HWKEY5	Hardware Key Word 5	0x4002_7090			
AES0_HWKEY6	Hardware Key Word 6	0x4002_70A0			
AES0_HWKEY7	Hardware Key Word 7	0x4002_70B0			
AES0_HWCTR0	Hardware Counter Word 0	0x4002_70C0			
AES0_HWCTR1	Hardware Counter Word 1	0x4002_70D0			
AES0_HWCTR2	Hardware Counter Word 2	0x4002_70E0			
AES0_HWCTR3	Hardware Counter Word 3	0x4002_70F0			
AES0_STATUS	Module Status	0x4002_7100	Y	Y	
CRC0 Registers					
CRC0_CONTROL	Module Control	0x4002_8000	Y	Y	
CRC0_DATA	Input/Result Data	0x4002_8010			
CRC0_RDATA	Bit-Reversed Output Data	0x4002_8020			

Table 3.1. Register Memory Map

Register Name	Title	Address (ALL Access)	SET (+0x4)	CLR(+0x8)	MSK (+0xC)
RTC0 Registers					
RTC0_CONFIG	RTC Configuration	0x4002_9000	Y	Y	
RTC0_CONTROL	RTC Control	0x4002_9010	Y	Y	
RTC0_ALARM0	RTC Alarm 0	0x4002_9020			
RTC0_ALARM1	RTC Alarm 1	0x4002_9030			
RTC0_ALARM2	RTC Alarm 2	0x4002_9040			
RTC0_SETCAP	RTC Timer Set/Capture Value	0x4002_9050			
RTC0_LFOCONTROL	LFOSC Control	0x4002_9060			
PBCFG0 Registers					
PBCFG0_CONTROL0	Global Port Control 0	0x4002_A000	Y	Y	
PBCFG0_CONTROL1	Global Port Control 1	0x4002_A010	Y	Y	
PBCFG0_XBAR0L	Crossbar 0 Control (Low)	0x4002_A020	Y	Y	
PBCFG0_XBAR0H	Crossbar 0 Control (High)	0x4002_A030	Y	Y	
PBCFG0_XBAR1	Crossbar 1 Control	0x4002_A040	Y	Y	
PBCFG0_PBKEY	Global Port Key	0x4002_A050			
PBSTD0 Registers					
PBSTD0_PB	Output Latch	0x4002_A0A0	Y	Y	Y
PBSTD0_PBPIN	Pin Value	0x4002_A0B0			
PBSTD0_PBMDSEL	Mode Select	0x4002_A0C0	Y	Y	
PBSTD0_PBSKIPEN	Crossbar Pin Skip Enable	0x4002_A0D0	Y	Y	
PBSTD0_PBOUTMD	Output Mode	0x4002_A0E0	Y	Y	
PBSTD0_PBDRV	Drive Strength	0x4002_A0F0	Y	Y	
PBSTD0_PM	Port Match Value	0x4002_A100	Y	Y	
PBSTD0_PMEN	Port Match Enable	0x4002_A110	Y	Y	

Table 3.1. Register Memory Map

Register Name	Title	Address (ALL Access)	SET (+0x4)	CLR(+0x8)	MSK (+0xC)
PBSTD1 Registers					
PBSTD1_PB	Output Latch	0x4002_A140	Y	Y	Y
PBSTD1_PBPIN	Pin Value	0x4002_A150			
PBSTD1_PBMDSEL	Mode Select	0x4002_A160	Y	Y	
PBSTD1_PBSKIPEN	Crossbar Pin Skip Enable	0x4002_A170	Y	Y	
PBSTD1_PBOUTMD	Output Mode	0x4002_A180	Y	Y	
PBSTD1_PBDRV	Drive Strength	0x4002_A190	Y	Y	
PBSTD1_PM	Port Match Value	0x4002_A1A0	Y	Y	
PBSTD1_PMEN	Port Match Enable	0x4002_A1B0	Y	Y	
PBSTD2 Registers					
PBSTD2_PB	Output Latch	0x4002_A1E0	Y	Y	Y
PBSTD2_PBPIN	Pin Value	0x4002_A1F0			
PBSTD2_PBMDSEL	Mode Select	0x4002_A200	Y	Y	
PBSTD2_PBSKIPEN	Crossbar Pin Skip Enable	0x4002_A210	Y	Y	
PBSTD2_PBOUTMD	Output Mode	0x4002_A220	Y	Y	
PBSTD2_PBDRV	Drive Strength	0x4002_A230	Y	Y	
PBSTD2_PM	Port Match Value	0x4002_A240	Y	Y	
PBSTD2_PMEN	Port Match Enable	0x4002_A250	Y	Y	
PBSTD2_PBLOCK	Lock Control	0x4002_A260			
PBSTD2_PBPGEN	Pulse Generator Pin Enable	0x4002_A270			
PBSTD2_PBPGPHASE	Pulse Generator Phase	0x4002_A280			
PBSTD3 Registers					
PBSTD3_PB	Output Latch	0x4002_A320	Y	Y	Y
PBSTD3_PBPIN	Pin Value	0x4002_A330			
PBSTD3_PBMDSEL	Mode Select	0x4002_A340	Y	Y	
PBSTD3_PBSKIPEN	Crossbar Pin Skip Enable	0x4002_A350	Y	Y	
PBSTD3_PBOUTMD	Output Mode	0x4002_A360	Y	Y	

Table 3.1. Register Memory Map

Register Name	Title	Address (ALL Access)	SET (+0x4)	CLR(+0x8)	MSK (+0xC)
PBSTD3_PBDRV	Drive Strength	0x4002_A370	Y	Y	
PBSTD3_PM	Port Match Value	0x4002_A380	Y	Y	
PBSTD3_PMEN	Port Match Enable	0x4002_A390	Y	Y	
PBSTD3_PBLOCK	Lock Control	0x4002_A3A0			
PBHD4 Registers					
PBHD4_PB	Output Latch	0x4002_A3C0	Y	Y	Y
PBHD4_PBPIN	Pin Value	0x4002_A3D0			
PBHD4_PBMDSEL	Mode Select	0x4002_A3E0	Y	Y	
PBHD4_PBDEN	Driver Enable	0x4002_A3F0	Y	Y	
PBHD4_PBDRV	Drive Strength	0x4002_A400	Y	Y	
PBHD4_PBILIMIT	Current Limit	0x4002_A410	Y	Y	
PBHD4_PBFSEL	Function Select	0x4002_A430			
PBHD4_PBSS	Safe State Control	0x4002_A440	Y	Y	
PBHD4_PBLOCK	Lock Control	0x4002_A450			
CLKCTRL0 Registers					
CLKCTRL0_CONTROL	Module Control	0x4002_D000			
CLKCTRL0_AHBCLKG	AHB Clock Gate	0x4002_D010	Y	Y	
CLKCTRL0_APBCLKG0	APB Clock Gate 0	0x4002_D020	Y	Y	
CLKCTRL0_APBCLKG1	APB Clock Gate 1	0x4002_D030	Y	Y	
CLKCTRL0_PM3CN	Power Mode 3 Clock Control	0x4002_D040			
RSTSRC0 Registers					
RSTSRC0_RESETEN	System Reset Source Enable	0x4002_D060	Y	Y	
RSTSRC0_RESETFLAG	System Reset Flags	0x4002_D070			
RSTSRC0_CONFIG	Configuration Options	0x4002_D080	Y	Y	

Table 3.1. Register Memory Map

Register Name	Title	Address (ALL Access)	SET (+0x4)	CLR(+0x8)	MSK (+0xC)
FLASHCTRL0 Registers					
FLASHCTRL0_CONFIG	Controller Configuration	0x4002_E000	Y	Y	
FLASHCTRL0_WRADDR	Flash Write Address	0x4002_E0A0			
FLASHCTRL0_WRDATA	Flash Write Data	0x4002_E0B0			
FLASHCTRL0_KEY	Flash Modification Key	0x4002_E0C0			
FLASHCTRL0_TCONTROL	Flash Timing Control	0x4002_E0D0			
VMON0 Registers					
VMON0_CONTROL	Module Control	0x4002_F000	Y	Y	
WDTIMER0 Registers					
WDTIMER0_CONTROL	Module Control	0x4003_0000	Y	Y	
WDTIMER0_STATUS	Module Status	0x4003_0010	Y	Y	
WDTIMER0_THRESHOLD	Threshold Values	0x4003_0020			
WDTIMER0_WDTKEY	Module Key	0x4003_0030			
IDAC0 Registers					
IDAC0_CONTROL	Module Control	0x4003_1000	Y	Y	
IDAC0_DATA	Output Data	0x4003_1010			
IDAC0_BUFSTATUS	FIFO Buffer Status	0x4003_1020	Y	Y	
IDAC0_BUFFER10	FIFO Buffer Entries 0 and 1	0x4003_1030			
IDAC0_BUFFER32	FIFO Buffer Entries 2 and 3	0x4003_1040			
IDAC0_GAINADJ	Output Current Gain Adjust	0x4003_1050			
IDAC1 Registers					
IDAC1_CONTROL	Module Control	0x4003_2000	Y	Y	
IDAC1_DATA	Output Data	0x4003_2010			
IDAC1_BUFSTATUS	FIFO Buffer Status	0x4003_2020	Y	Y	
IDAC1_BUFFER10	FIFO Buffer Entries 0 and 1	0x4003_2030			
IDAC1_BUFFER32	FIFO Buffer Entries 2 and 3	0x4003_2040			
IDAC1_GAINADJ	Output Current Gain Adjust	0x4003_2050			

Table 3.1. Register Memory Map

Register Name	Title	Address (ALL Access)	SET (+0x4)	CLR(+0x8)	MSK (+0xC)
DMACTRL0 Registers					
DMACTRL0_STATUS	Controller Status	0x4003_6000			
DMACTRL0_CONFIG	Controller Configuration	0x4003_6004			
DMACTRL0_BASEPTR	Base Pointer	0x4003_6008			
DMACTRL0_ABASEPTR	Alternate Base Pointer	0x4003_600C			
DMACTRL0_CHSTATUS	Channel Status	0x4003_6010			
DMACTRL0_CHSWRCN	Channel Software Request Control	0x4003_6014			
DMACTRL0_CHREQMSET	Channel Request Mask Set	0x4003_6020			
DMACTRL0_CHREQMCLR	Channel Request Mask Clear	0x4003_6024			
DMACTRL0_CHENSET	Channel Enable Set	0x4003_6028			
DMACTRL0_CHENCLR	Channel Enable Clear	0x4003_602C			
DMACTRL0_CHALTSET	Channel Alternate Select Set	0x4003_6030			
DMACTRL0_CHALTCLR	Channel Alternate Select Clear	0x4003_6034			
DMACTRL0_CHHPSET	Channel High Priority Set	0x4003_6038			
DMACTRL0_CHHPCLR	Channel High Priority Clear	0x4003_603C			
DMACTRL0_BERRCLR	Bus Error Clear	0x4003_604C			
DMAXBAR0 Registers					
DMAXBAR0_DMAXBAR0	Channel 0-7 Trigger Select	0x4003_7000	Y	Y	
DMAXBAR0_DMAXBAR1	Channel 8-15 Trigger Select	0x4003_7010	Y	Y	
LPTIMER0 Registers					
LPTIMER0_CONTROL	Module Control	0x4003_8000	Y	Y	
LPTIMER0_DATA	Timer and Comparator Data	0x4003_8010			
LPTIMER0_STATUS	Module Status	0x4003_8020	Y	Y	
LDO0 Registers					
LDO0_CONTROL	Control	0x4003_9000	Y	Y	
VREF0 Registers					
VREF0_CONTROL	Voltage Reference Control	0x4003_9010	Y	Y	

Table 3.1. Register Memory Map

Register Name	Title	Address (ALL Access)	SET (+0x4)	CLR(+0x8)	MSK (+0xC)
I2S0 Registers					
I2S0_TXCONTROL	Transmit Control	0x4003_A000	Y	Y	
I2S0_TXMODE	Transmit Mode	0x4003_A010	Y	Y	
I2S0_FSDUTY	Frame Sync Duty Cycle	0x4003_A020			
I2S0_RXCONTROL	Receive Control	0x4003_A030	Y	Y	
I2S0_RXMODE	Receive Mode	0x4003_A040	Y	Y	
I2S0_CLKCONTROL	Clock Control	0x4003_A050	Y	Y	
I2S0_TXFIFO	Transmit Data FIFO	0x4003_A060			
I2S0_RXFIFO	Receive Data FIFO	0x4003_A070			
I2S0_FIFOSTATUS	FIFO Status	0x4003_A080			
I2S0_FIFOCONTROL	FIFO Control	0x4003_A090	Y	Y	
I2S0_INTCONTROL	Interrupt Control	0x4003_A0A0	Y	Y	
I2S0_STATUS	Module Status	0x4003_A0B0	Y	Y	
I2S0_DMACONTROL	DMA Control	0x4003_A0C0	Y	Y	
I2S0_DBGCONTROL	Debug Control	0x4003_A0D0	Y	Y	
PLL0 Registers					
PLL0_DIVIDER	Reference Divider Setting	0x4003_B000			
PLL0_CONTROL	Module Control	0x4003_B010	Y	Y	
PLL0_SSPR	Spectrum Spreading Control	0x4003_B020			
PLL0_CALCONFIG	Calibration Configuration	0x4003_B030			
EXTOSC0 Registers					
EXTOSC0_CONTROL	Oscillator Control	0x4003_C000	Y	Y	
VREG0 Registers					
VREG0_CONTROL	Module Control	0x4004_0000	Y	Y	
LPOSC0 Registers					
LPOSC0_OSCVAL	Low Power Oscillator Output Value	0x4004_1000	Y	Y	

Table 3.1. Register Memory Map

Register Name	Title	Address (ALL Access)	SET (+0x4)	CLR(+0x8)	MSK (+0xC)
EXTVREG0 Registers					
EXTVREG0_CONTROL	Module Control	0x4004_2000	Y	Y	
EXTVREG0_CONFIG	Module Configuration	0x4004_2010			
EXTVREG0_STATUS	Module Status	0x4004_2020			
EXTVREG0_CSCONTROL	Current Sense Control	0x4004_2040	Y	Y	
EXTVREG0_CSCONFIG	Current Sense Configuration	0x4004_2050			
IVC0 Registers					
IVC0_CONTROL	Module Control	0x4004_4000	Y	Y	
PMU0 Registers					
PMU0_CONTROL	Module Control	0x4004_8000	Y	Y	
PMU0_CONFIG	Module Configuration	0x4004_8010	Y	Y	
PMU0_STATUS	Module Status	0x4004_8020	Y	Y	
PMU0_WAKEEN	Wake Source Enable	0x4004_8030	Y	Y	
PMU0_WAKESTATUS	Wake Source Status	0x4004_8040			
PMU0_PWEN	Pin Wake Pin Enable	0x4004_8050	Y	Y	
PMU0_PWPOL	Pin Wake Pin Polarity Select	0x4004_8060	Y	Y	
LOCK0 Registers					
LOCK0_KEY	Security Key	0x4004_9000			
LOCK0_PERIPHLOCK0	Peripheral Lock Control 0	0x4004_9020	Y	Y	
LOCK0_PERIPHLOCK1	Peripheral Lock Control 1	0x4004_9040	Y	Y	
SCONFIG0 Registers					
SCONFIG0_CONFIG	System Configuration	0x4004_90B0	Y	Y	
DEVICEID0 Registers					
DEVICEID0_DEVICEID0	Device ID Word 0	0x4004_90C0			
DEVICEID0_DEVICEID1	Device ID Word 1	0x4004_90D0			
DEVICEID0_DEVICEID2	Device ID Word 2	0x4004_90E0			
DEVICEID0_DEVICEID3	Device ID Word 3	0x4004_90F0			

4. Interrupts

SiM3U1xx/SiM3C1xx devices implement the standard nested-vectored interrupt controller (NVIC) available in the ARM Cortex-M3 core. The specific system exceptions, interrupt vectors, and priority implementation are described in the following sections.

4.1. System Exceptions

The system-level exceptions on SiM3U1xx/SiM3C1xx devices are shown in Table 4.1.

Table 4.1. System Exceptions

Exception Number	Type	Priority	Description
1	Reset	–3	System Reset.
2	NMI	–2	External NMI Input.
3	Hard Fault	–1	All fault conditions if the corresponding fault handler is disabled.
4	MemManage Fault	Programmable	MMU/MPU fault (not supported).
5	Bus Fault	Programmable	AHB error received from slave (either prefetch abort or data abort).
6	Usage Fault	Programmable	Exception due to program error.
7	Reserved		
8	Reserved		
9	Reserved		
10	Reserved		
11	SVcall	Programmable	System Service Call using SWI or SVC instruction.
12	Debug Monitor	Programmable	Breakpoint, Watchpoint, or external debug request.
13	Reserved		
14	PendSV	Programmable	Pendable reset for system device.
15	SYSTICK	Programmable	System tick timer.

4.2. Interrupt Vector Table

The interrupt vector table for SiM3U1xx/SiM3C1xx is shown in Table 4.2.

Table 4.2. Interrupt Vector Table

Position	Default Priority	Name/ Description	Sources	Default Address
	–3	Reset	System Reset	0x00000004
	–2	NMI	NMI	0x00000008
	–1	Hard Fault	All fault conditions if the corresponding fault handler is disabled	0x0000000C
	0	MemManage	MMU/MPU fault (not supported)	0x00000010
	1	Bus Fault	AHB error received from slave (either prefetch abort or data abort)	0x00000014
	2	Usage Fault	Exception due to program error	0x00000018
		Reserved		0x0000001C
		Reserved		0x00000020
		Reserved		0x00000024
		Reserved		0x00000028
	3	SVcall	System Service Call using SWI or SVC instruction	0x0000002C
	4	Debug Monitor	Breakpoint Watchpoint External debug request	0x00000030
		Reserved		0x00000034
	5	PendSV	Pendable reset for system device	0x00000038
	6	SYSTICK	System tick timer	0x0000003C
0	7	WDTIMER0	First threshold crossed	0x00000040
1	8	PBEXT0	External pin (INT0.x) rising edge External pin (INT0.x) falling edge	0x00000044
2	9	PBEXT1	External pin (INT1.x) rising edge External pin (INT1.x) falling edge	0x00000048
3	10	RTC0ALRM	Alarm 0 Alarm 1 Alarm 2	0x0000004C
4	11	DMACH0	DMA Channel 0 done	0x00000050
5	12	DMACH1	DMA Channel 1 done	0x00000054
6	13	DMACH2	DMA Channel 2 done	0x00000058
7	14	DMACH3	DMA Channel 3 done	0x0000005C
8	15	DMACH4	DMA Channel 4 done	0x00000060

Table 4.2. Interrupt Vector Table (Continued)

Position	Default Priority	Name/ Description	Sources	Default Address
9	16	DMACH5	DMA Channel 5 done	0x00000064
10	17	DMACH6	DMA Channel 6 done	0x00000068
11	18	DMACH7	DMA Channel 7 done	0x0000006C
12	19	DMACH8	DMA Channel 8 done	0x00000070
13	20	DMACH9	DMA Channel 9 done	0x00000074
14	21	DMACH10	DMA Channel 10 done	0x00000078
15	22	DMACH11	DMA Channel 11 done	0x0000007C
16	23	DMACH12	DMA Channel 12 done	0x00000080
17	24	DMACH13	DMA Channel 13 done	0x00000084
18	25	DMACH14	DMA Channel 14 done	0x00000088
19	26	DMACH15	DMA Channel 15 done	0x0000008C
20	27	TIMER0L	TIMER0 Low overflow	0x00000090
21	28	TIMER0H	TIMER0 High overflow	0x00000094
22	29	TIMER1L	TIMER1 Low overflow	0x00000098
23	30	TIMER1H	TIMER1 High overflow	0x0000009C
24	31	EPCA0	Counter overflow Halt external signal is high Channel compare or match Channel intermediate overflow	0x000000A0
25	32	PCA0	Counter overflow Channel compare or match Channel intermediate overflow	0x000000A4
26	33	PCA1	Counter overflow Channel compare or match Channel intermediate overflow	0x000000A8
27	34	USART0	Receive frame error Receive parity error Receive overrun Receive data request Transmit SmartCard parity error Transmit underrun Transmit data request Transmit complete	0x000000AC

Table 4.2. Interrupt Vector Table (Continued)

Position	Default Priority	Name/ Description	Sources	Default Address
28	35	USART1	Receive frame error Receive parity error Receive overrun Receive data request Transmit SmartCard parity error Transmit underrun Transmit data request Transmit complete	0x000000B0
29	36	SPI0	Shift Register empty FIFO underrun Mode fault Slave Select pin Illegal receive FIFO access Receive FIFO Overrun Receive FIFO read request Illegal transmit FIFO access Transmit FIFO overrun Transmit FIFO write request	0x000000B4
30	37	SPI1	Shift Register empty FIFO underrun Mode fault Slave Select pin Illegal receive FIFO access Receive FIFO Overrun Receive FIFO read request Illegal transmit FIFO access Transmit FIFO overrun Transmit FIFO write request	0x000000B8
31	38	SPI2	Shift Register empty FIFO underrun Mode fault Slave Select pin Illegal receive FIFO access Receive FIFO Overrun Receive FIFO read request Illegal transmit FIFO access Transmit FIFO overrun Transmit FIFO write request	0x000000BC

Table 4.2. Interrupt Vector Table (Continued)

Position	Default Priority	Name/ Description	Sources	Default Address
32	39	I2C0	Start Transmit complete Receive complete Acknowledge Stop Timer byte 0 overflow Timer byte 1 overflow Timer byte 2 overflow Timer byte 3 overflow Arbitration lost	0x000000C0
33	40	I2C1	Start Transmit complete Receive complete Acknowledge Stop Timer byte 0 overflow Timer byte 1 overflow Timer byte 2 overflow Timer byte 3 overflow Arbitration lost	0x000000C4
34	41	USB0	Endpoint 0 data Endpoint 1-4 IN or OUT data Suspend Resume Start-of-Frame (SOF) USB Reset	0x000000C8
35	42	SARADC0	Conversion complete Scan complete FIFO underrun FIFO overrun Window comparator threshold crossed	0x000000CC
36	43	SARADC1	Conversion complete Scan complete FIFO underrun FIFO overrun Window comparator threshold crossed	0x000000D0
37	44	CMP0	Rising edge occurred Falling edge occurred	0x000000D4
38	45	CMP1	Rising edge occurred Falling edge occurred	0x000000D8
39	46	CAPSENSE0	Conversion complete Compare threshold crossed End-of-Scan	0x000000DC

Table 4.2. Interrupt Vector Table (Continued)

Position	Default Priority	Name/ Description	Sources	Default Address
40	47	I2S0RX	Receive FIFO overflow Receive FIFO high watermark crossed	0x000000E0
41	48	I2S0TX	Transmit FIFO underflow Transmit FIFO low watermark crossed	0x000000E4
42	49	AES0	Operation complete Error occurred	0x000000E8
43	50	VDDLOW	VDD falls below the early warning threshold	0x000000EC
44	51	RTC0FAIL	RTC0 Oscillator failed	0x000000F0
45	52	PMATCH0	Port Match event	0x000000F4
46	53	UART0	Receive frame error Receive parity error Receive overrun Receive data request Transmit SmartCard parity error Transmit underrun Transmit data request Transmit complete	0x000000F8
47	54	UART1	Receive frame error Receive parity error Receive overrun Receive data request Transmit SmartCard parity error Transmit underrun Transmit data request Transmit complete	0x000000FC
48	55	IDAC0	Data buffer overrun Data buffer underrun Data buffer went empty	0x00000100
49	56	IDAC1	Data buffer overrun Data buffer underrun Data buffer went empty	0x00000104
50	57	LPTIMER0	Timer overflow Compare threshold crossed	0x00000108
51	58	PLL0	Lock saturation high or low Oscillator lock	0x0000010C
52	59	VBUS Invalid	The VBUS input is below the valid threshold	0x00000110
53	60	VREGLOW	VREGIN / 4 falls below the early warning threshold	0x00000114

4.3. Priorities

The SiM3U1xx/SiM3C1xx devices implement 4 bits of interrupt priority, as shown in Figure 4.1.

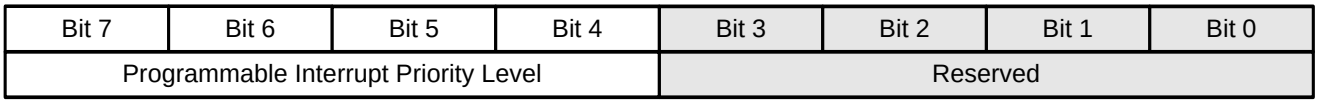


Figure 4.1. SiM3U1xx/SiM3C1xx Interrupt Priorities

In addition to the different priority levels, the NVIC allows for different priority groups, as shown in Table 4.3. These groups determine the number of bits used to determine the Preempt Priority and Subpriority settings for each interrupt. A higher priority interrupt can preempt or interrupt a lower priority interrupt. If two interrupts of the same priority occur at the same time, the interrupts cannot preempt each other and the interrupt with the higher Subpriority (lowest value) will be taken first. The Reset, NMI, and Hard Fault exceptions have fixed negative priorities to always take precedence over other interrupts in the system.

Table 4.3. Priority Groups

Priority Group	Preempt Priority Field Size	Subpriority Field Size
0–3	[7:4]	None
4	[7:5]	[4]
5	[7:6]	[5:4]
6	[7]	[6:4]
7	None	[7:4]

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Preempt Priority Level			Subpriority	Reserved			

Figure 4.2. Priority Group 4 Fields

Table 4.4. Priority Levels with Priority Group 4

Preempt Priority	Subpriority	Priority Value
1 (Highest)	1 (Highest)	0x00
1	2	0x10
2	1	0x20
2	2	0x30
3	1	0x40
3	2	0x50
4	1	0x60
4	2	0x70
5	1	0x80
5	2	0x90
6	1	0xA0
6	2	0xB0
7	1	0xC0
7	2	0xD0
8	1	0xE0
8 (Lowest)	2 (Lowest)	0xF0

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Subpriority Level				Reserved			

Figure 4.3. Priority Group 7 Fields

Table 4.5. Priority Levels with Priority Group 7 (Interrupts Cannot Preempt)

Preempt Priority	Subpriority	Priority Value
	1 (Highest)	0x00
	2	0x10
	3	0x20
	4	0x30
	5	0x40
	6	0x50
	7	0x60
	8	0x70
	9	0x80
	10	0x90
	11	0xA0
	12	0xB0
	13	0xC0
	14	0xD0
	15	0xE0
	16 (Lowest)	0xF0

The Priority Group and Interrupt Priority settings are in the NVIC.

5. Clock Control (CLKCTRL0)

This section describes the Clock Control (CLKCTRL) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

Note that features related to the USB peripheral are only available on the SiM3U1xx device family.

5.1. Clock Control Features

Clock Control includes the following features:

- Support for multiple oscillator sources for system clock with smooth and glitch-less transition.
- Individual clock gating controls for most peripherals and modules.
- Multiple options for AHB clock settings and a divider for the APB clock.
- Synchronization between AHB and APB clocks.

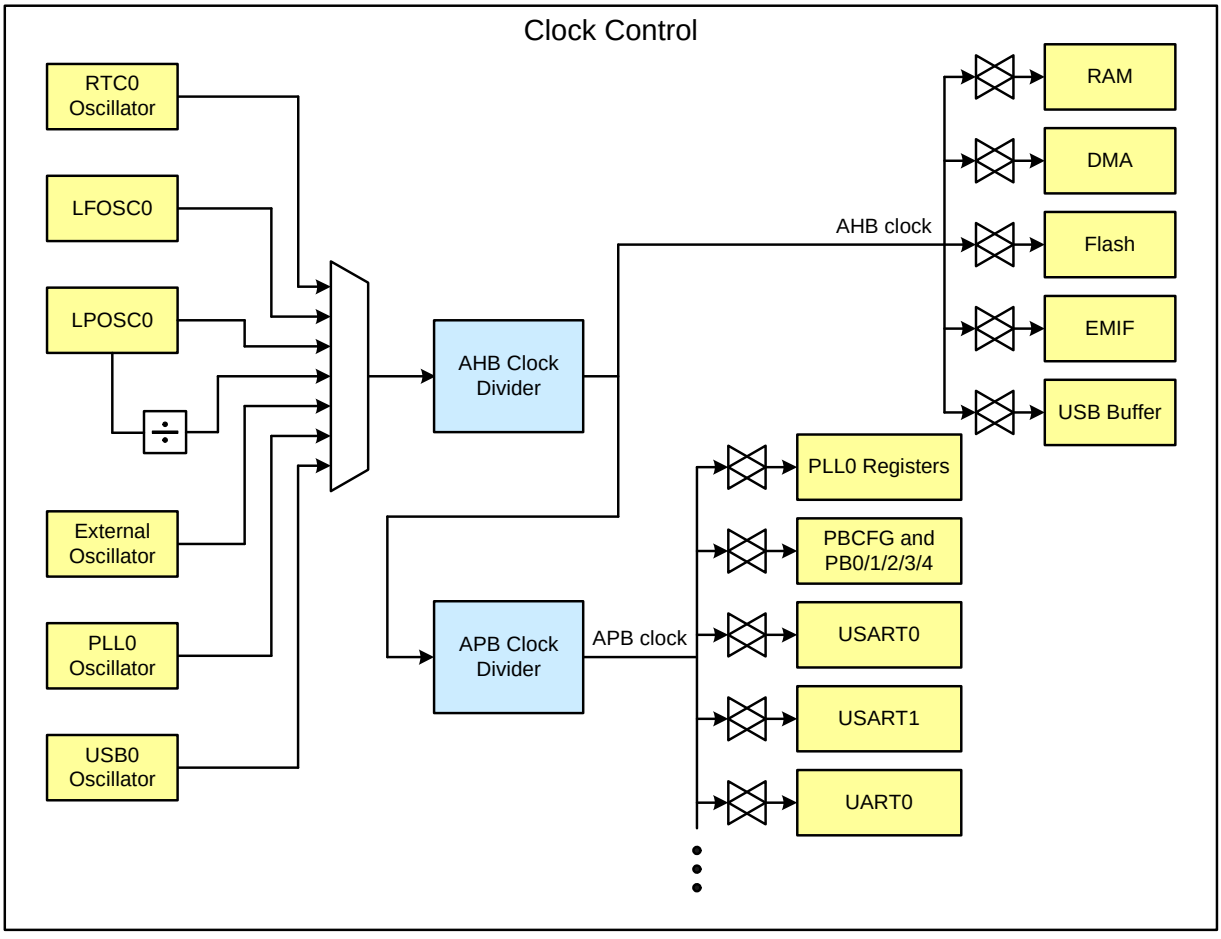


Figure 5.1. Clock Control Block Diagram

Clock Control generates the two system clocks: AHB and APB. The AHB clock services memory peripherals and can be derived from one of seven sources: the RTC0 Oscillator, the Low Frequency Oscillator, the Low Power Oscillator, the divided Low Power Oscillator, the External Oscillator, the PLL0 Oscillator, and the USB0 Oscillator. In addition, a divider for the AHB clock provides flexible clock options for the device. The APB clock services data peripherals and is synchronized with the AHB clock. The APB clock can be equal to the AHB clock (if AHB is less than or equal to 50 MHz) or set to the AHB clock divided by two.

Clock Control allows the AHB and APB clocks to be turned off to unused peripherals to save system power. Any registers in a peripheral with disabled clocks will be unable to be accessed (read or write) until the clocks are enabled. Most peripherals have clocks off by default after a power-on reset.

5.2. CLKCTRL0 Registers

This section contains the detailed register descriptions for CLKCTRL0 registers.

Register 5.1. CLKCTRL0_CONTROL: Module Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved		OBUSYF	EXTESEL	Reserved											APBDIV
Type	R		R	RW	R											RW
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved					AHBDIV			Reserved					AHBSEL		
Type	R					RW			R					RW		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

CLKCTRL0_CONTROL = 0x4002_D000

Table 5.1. CLKCTRL0_CONTROL Register Bit Descriptions

Bit	Name	Function
31:30	Reserved	Must write reset value.
29	OBUSYF	**Oscillators Busy Flag.** When set, this status bit indicates that a requested clock switch-over is in progress. Firmware should wait until the flag is clear to reconfigure the AHBSEL, AHBDIV, and APBDIV fields.
28	EXTESEL	**External Clock Edge Select.** Select the edge mode used by the external clock for the TIMER, PCA, and EPCA modules. This external clock is synchronized with the APB clock. 0: External clock generated by both rising and falling edges of the external oscillator. 1: External clock generated by only rising edges of the external oscillator.
27:17	Reserved	Must write reset value.
16	APBDIV	**APB Clock Divider.** Divides the APB clock from the AHB clock. This field should not be modified when OBUSYF is set. 0: APB clock is the same as the AHB clock (divided by 1). 1: APB clock is the AHB clock divided by 2.
15:11	Reserved	Must write reset value.

Table 5.1. CLKCTRL0_CONTROL Register Bit Descriptions

Bit	Name	Function
10:8	AHBDIV	**AHB Clock Divider.** Divides the AHB clock. This field should not be modified when OBUSYF is set. 000: AHB clock divided by 1. 001: AHB clock divided by 2. 010: AHB clock divided by 4. 011: AHB clock divided by 8. 100: AHB clock divided by 16. 101: AHB clock divided by 32. 110: AHB clock divided by 64. 111: AHB clock divided by 128.
7:3	Reserved	Must write reset value.
2:0	AHBSEL	**AHB Clock Source Select.** This field should not be modified when OBUSYF is set. 000: AHB clock source is the Low-Power Oscillator. 001: AHB clock source is the Low-Frequency Oscillator. 010: AHB clock source is the RTC Oscillator. 011: AHB clock source is the External Oscillator. 100: AHB clock source is the USB Oscillator. 101: AHB clock source is the PLL. 110: AHB clock source is a divided version of the Low-Power Oscillator. 111: Reserved.

Register 5.2. CLKCTRL0_AHBCLKG: AHB Clock Gate

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved											USB0BCEN	EMIF0CEN	FLASHCEN	DMACEN	RAMCEN
Type	R											RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

Register ALL Access Address

CLKCTRL0_AHBCLKG = 0x4002_D010

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 5.2. CLKCTRL0_AHBCLKG Register Bit Descriptions

Bit	Name	Function
31:5	Reserved	Must write reset value.
4	USB0BCEN	**USB0 Buffer Clock Enable.** 0: Disable the AHB clock to the USB0 Buffer (default). 1: Enable the AHB clock to the USB0 Buffer.
3	EMIF0CEN	**EMIF Clock Enable.** 0: Disable the AHB clock to the External Memory Interface (EMIF) (default). 1: Enable the AHB clock to the External Memory Interface (EMIF).
2	FLASHCEN	**Flash Clock Enable.** 0: Disable the AHB clock to the Flash. 1: Enable the AHB clock to the Flash (default).
1	DMACEN	**DMA Controller Clock Enable.** 0: Disable the AHB clock to the DMA Controller (default). 1: Enable the AHB clock to the DMA Controller.
0	RAMCEN	**RAM Clock Enable.** 0: Disable the AHB clock to the RAM. 1: Enable the AHB clock to the RAM (default).

Register 5.3. CLKCTRL0_APBCLKG0: APB Clock Gate 0

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved	FLCTRLCEN	EVREGCEN	USB0CEN	I2S0CEN	LPT0CEN	IDAC1CEN	IDAC0CEN	CRC0CEN	AES0CEN	CS0CEN	CMP1CEN	CMP0CEN	ADC1CEN	ADC0CEN	TIMER1CEN
Type	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	TIMER0CEN	SSG0CEN	PCA1CEN	PCA0CEN	EPCA0CEN	I2C1CEN	I2C0CEN	SPI2CEN	SPI1CEN	SPI0CEN	UART1CEN	UART0CEN	USART1CEN	USART0CEN	PB0CEN	PLL0CEN
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

CLKCTRL0_APBCLKG0 = 0x4002_D020

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 5.3. CLKCTRL0_APBCLKG0 Register Bit Descriptions

Bit	Name	Function
31	Reserved	Must write reset value.
30	FLCTRLCEN	**Flash Controller Clock Enable.** 0: Disable the APB clock to the Flash Controller Module (FLASHCTRL0) (default). 1: Enable the APB clock to the Flash Controller Module (FLASHCTRL0).
29	EVREGCEN	**External Regulator Clock Enable.** 0: Disable the APB clock to the External Regulator Module (EXTVREG0) (default). 1: Enable the APB clock to the External Regulator Module (EXTVREG0).
28	USB0CEN	**USB0 Module Clock Enable.** 0: Disable the APB clock to the USB0 Module (default). 1: Enable the APB clock to the USB0 Module.
27	I2S0CEN	**I2S0 Module Clock Enable.** 0: Disable the APB clock to the I2S0 Module (default). 1: Enable the APB clock to the I2S0 Module.
26	LPT0CEN	**Low Power Timer (LPTIMER0) Module Clock Enable.** 0: Disable the APB clock to the LPTIMER0 Module (default). 1: Enable the APB clock to the LPTIMER0 Module.

Table 5.3. CLKCTRL0_APBCLKG0 Register Bit Descriptions

Bit	Name	Function
25	IDAC1CEN	**IDAC1 Module Clock Enable.** 0: Disable the APB clock to the IDAC1 Module (default). 1: Enable the APB clock to the IDAC1 Module.
24	IDAC0CEN	**IDAC0 Module Clock Enable.** 0: Disable the APB clock to the IDAC0 Module (default). 1: Enable the APB clock to the IDAC0 Module.
23	CRC0CEN	**CRC0 Module Clock Enable.** 0: Disable the APB clock to the CRC0 Module (default). 1: Enable the APB clock to the CRC0 Module.
22	AES0CEN	**AES0 Module Clock Enable.** 0: Disable the APB clock to the AES0 Module (default). 1: Enable the APB clock to the AES0 Module.
21	CS0CEN	**Capacitive Sensing (CAPSENSE0) Module Clock Enable.** 0: Disable the APB clock to the CAPSENSE0 Module (default). 1: Enable the APB clock to the CAPSENSE0 Module.
20	CMP1CEN	**Comparator 1 Module Clock Enable.** 0: Disable the APB clock to the Comparator 1 Module (default). 1: Enable the APB clock to the Comparator 1 Module.
19	CMP0CEN	**Comparator 0 Module Clock Enable.** 0: Disable the APB clock to the Comparator 0 Module (default). 1: Enable the APB clock to the Comparator 0 Module.
18	ADC1CEN	**SARADC1 Module Clock Enable.** 0: Disable the APB clock to the SARADC1 Module (default). 1: Enable the APB clock to the SARADC1 Module.
17	ADC0CEN	**SARADC0 Module Clock Enable.** 0: Disable the APB clock to the SARADC0 Module (default). 1: Enable the APB clock to the SARADC0 Module.
16	TIMER1CEN	**TIMER1 Module Clock Enable.** 0: Disable the APB clock to the TIMER1 Module (default). 1: Enable the APB clock to the TIMER1 Module.
15	TIMER0CEN	**TIMER0 Module Clock Enable.** 0: Disable the APB clock to the TIMER0 Module (default). 1: Enable the APB clock to the TIMER0 Module.
14	SSG0CEN	**SSG0 Module Clock Enable.** 0: Disable the APB clock to the SSG0 Module (default). 1: Enable the APB clock to the SSG0 Module.
13	PCA1CEN	**PCA1 Module Clock Enable.** 0: Disable the APB clock to the PCA1 Module (default). 1: Enable the APB clock to the PCA1 Module.

Table 5.3. CLKCTRL0_APBCLKG0 Register Bit Descriptions

Bit	Name	Function
12	PCA0CEN	**PCA0 Module Clock Enable.** 0: Disable the APB clock to the PCA0 Module (default). 1: Enable the APB clock to the PCA0 Module.
11	EPCA0CEN	**EPCA0 Module Clock Enable.** 0: Disable the APB clock to the EPCA0 Module (default). 1: Enable the APB clock to the EPCA0 Module.
10	I2C1CEN	**I2C1 Module Clock Enable.** 0: Disable the APB clock to the I2C1 Module (default). 1: Enable the APB clock to the I2C1 Module.
9	I2C0CEN	**I2C0 Module Clock Enable.** 0: Disable the APB clock to the I2C0 Module (default). 1: Enable the APB clock to the I2C0 Module.
8	SPI2CEN	**SPI2 Module Clock Enable.** 0: Disable the APB clock to the SPI2 Module (default). 1: Enable the APB clock to the SPI2 Module.
7	SPI1CEN	**SPI1 Module Clock Enable.** 0: Disable the APB clock to the SPI1 Module (default). 1: Enable the APB clock to the SPI1 Module.
6	SPI0CEN	**SPI0 Module Clock Enable.** 0: Disable the APB clock to the SPI0 Module (default). 1: Enable the APB clock to the SPI0 Module.
5	UART1CEN	**UART1 Module Clock Enable.** 0: Disable the APB clock to the UART1 Module (default). 1: Enable the APB clock to the UART1 Module.
4	UART0CEN	**UART0 Module Clock Enable.** 0: Disable the APB clock to the UART0 Module (default). 1: Enable the APB clock to the UART0 Module.
3	USART1CEN	**USART1 Module Clock Enable.** 0: Disable the APB clock to the USART1 Module (default). 1: Enable the APB clock to the USART1 Module.
2	USART0CEN	**USART0 Module Clock Enable.** 0: Disable the APB clock to the USART0 Module (default). 1: Enable the APB clock to the USART0 Module.
1	PB0CEN	**Port Bank Module Clock Enable.** 0: Disable the APB clock to the Port Bank Modules (default). 1: Enable the APB clock to the Port Bank Modules.
0	PLL0CEN	**PLL Module Clock Enable.** 0: Disable the APB clock to the PLL0 registers (default). 1: Enable the APB clock to the PLL0 registers.

Register 5.4. CLKCTRL0_APBCLKG1: APB Clock Gate 1

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved													MISC2CEN	MISC1CEN	MISC0CEN
Type	R													RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Register ALL Access Address

CLKCTRL0_APBCLKG1 = 0x4002_D030

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 5.4. CLKCTRL0_APBCLKG1 Register Bit Descriptions

Bit	Name	Function
31:3	Reserved	Must write reset value.
2	MISC2CEN	**Miscellaneous 2 Clock Enable.** 0: Disable the APB clock to the OSCVLDF flag in the EXTOSC module (default). 1: Enable the APB clock to the OSCVLDF flag in the EXTOSC module.
1	MISC1CEN	**Miscellaneous 1 Clock Enable.** 0: Disable the APB clock to the Watchdog Timer (WDTIMER0), EMIF0, and DMA Crossbar (DMAXBAR0) modules. 1: Enable the APB clock to the Watchdog Timer (WDTIMER0), EMIF0, and DMA Crossbar (DMAXBAR0) modules (default).
0	MISC0CEN	**Miscellaneous 0 Clock Enable.** 0: Disable the APB clock to the RSTSRC0, LOCK0, VMON0, VREG0, LDO0, VREF0, EXTOSC0, LPOSC0, EXTVREG0, IVC0 and RTC0 modules (default). 1: Enable the APB clock to the RSTSRC0, LOCK0, VMON0, VREG0, LDO0, VREF0, EXTOSC0, LPOSC0, EXTVREG0, IVC0 and RTC0 modules.

Register 5.5. CLKCTRL0_PM3CN: Power Mode 3 Clock Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															PM3CEN
Type	R															RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved													PM3CSEL		
Type	R													RW		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

CLKCTRL0_PM3CN = 0x4002_D040

Table 5.5. CLKCTRL0_PM3CN Register Bit Descriptions

Bit	Name	Function
31:17	Reserved	Must write reset value.
16	PM3CEN	**Power Mode 3 Fast-Wake Clock Enable.** When set to 1, the core will automatically switch to the clock source defined by PM3CSEL during Power Mode 3, which speeds up the wakeup time. 0: Disable the core clock when in Power Mode 3. 1: The core clock is enabled and runs off the clock selected by PM3CSEL in Power Mode 3.
15:3	Reserved	Must write reset value.
2:0	PM3CSEL	**Power Mode 3 Fast-Wake Clock Source.** If PM3CEN is set to 1, this clock selection should be the LFOSC0 or RTC0OSC clock to save power while in Power Mode 3. Additionally, the AHB and APB source must be set to the Low-Power Oscillator or divided version of the Low-Power Oscillator. 000: Power Mode 3 clock source is the Low-Power Oscillator. 001: Power Mode 3 clock source is the Low-Frequency Oscillator. 010: Power Mode 3 clock source is the RTC Oscillator. 011: Power Mode 3 clock source is the External Oscillator. 100: Power Mode 3 clock source is the USB Oscillator. 101: Power Mode 3 clock source is the PLL. 110: Power Mode 3 clock source is a divided version of the Low-Power Oscillator. 111: Reserved.

5.3. CLKCTRL0 Register Memory Map

Table 5.6. CLKCTRL0 Memory Map

Register Name	CLKCTRL0_APBCLKG0	CLKCTRL0_AHBCLKG	CLKCTRL0_CONTROL
ALL Address	0x4002_D020	0x4002_D010	0x4002_D000
Access Methods	ALL | SET | CLR	ALL | SET | CLR	ALL
Bit 31	Reserved	Reserved	Reserved
Bit 30	FLCTRLCEN		
Bit 29	EVREGCEN		OBUSYF
Bit 28	USB0CEN		EXTESEL
Bit 27	I2S0CEN		Reserved
Bit 26	LPT0CEN		
Bit 25	IDAC1CEN		
Bit 24	IDAC0CEN		
Bit 23	CRC0CEN		
Bit 22	AES0CEN		
Bit 21	CS0CEN		
Bit 20	CMP1CEN		
Bit 19	CMP0CEN		
Bit 18	ADC1CEN		
Bit 17	ADC0CEN		
Bit 16	TIMER1CEN		APBDIV
Bit 15	TIMER0CEN		Reserved
Bit 14	SSG0CEN		
Bit 13	PCA1CEN		
Bit 12	PCA0CEN		
Bit 11	EPCA0CEN		
Bit 10	I2C1CEN		AHBDIV
Bit 9	I2C0CEN		
Bit 8	SPI2CEN		
Bit 7	SPI1CEN		Reserved
Bit 6	SPI0CEN		
Bit 5	UART1CEN		
Bit 4	UART0CEN	USB0BCEN	AHBSEL
Bit 3	USART1CEN	EMIF0CEN	
Bit 2	USART0CEN	FLASHCEN	
Bit 1	PB0CEN	DMACEN	
Bit 0	PLL0CEN	RAMCEN	

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

Table 5.6. CLKCTRL0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CLKCTRL0_APBCLKG1	0x4002_D030	ALL | SET | CLR	Reserved																													MISC2CEN	MISC1CEN	MISC0CEN
CLKCTRL0_PM3CN	0x4002_D040	ALL	Reserved															PM3CEN	Reserved													PM3CSEL		

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

6. Reset Sources (RSTSRC0)

This section describes the Reset Sources (RSTSRC) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

Note that features related to the USB peripheral are only available on the SiM3U1xx device family.

6.1. Reset Sources Features

The Reset Source block includes the following features:

- Separate enable mask bits for each reset source other than the $\overline{\text{RESET}}$ pin.
- Separate flags for each reset source indicating the cause of the last reset.

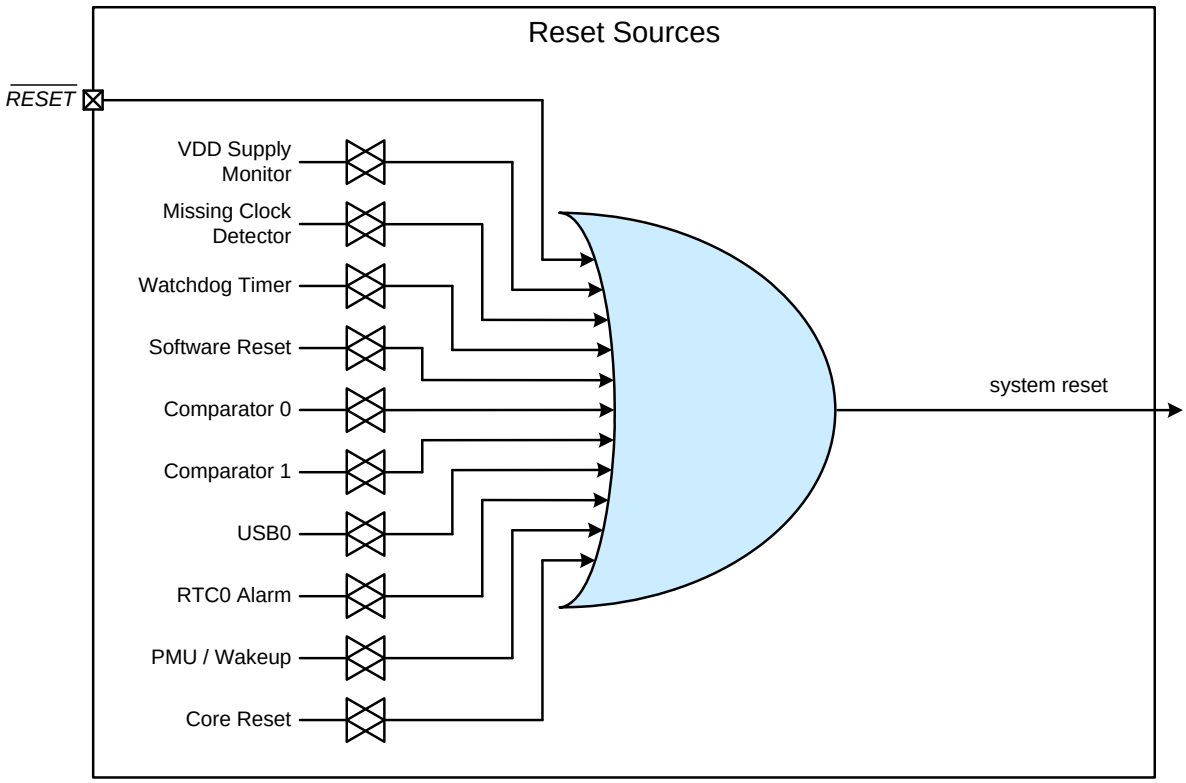


Figure 6.1. Reset Sources Block Diagram

Reset circuitry allows the controller to be easily placed in a predefined default condition. On entry to this reset state, the following occur:

- The core halts program execution.
- Module registers are initialized to their defined reset values unless the bits reset only with a power-on reset.
- External port pins are forced to a known state.
- Interrupts and timers are disabled.
- Clocks to all AHB peripherals other than the USB0 buffers are enabled.
- Clocks to all APB peripherals other than Watchdog Timer, EMIF0, and DMAXBAR are disabled.

All registers are reset to the predefined values noted in the register descriptions unless the bits only reset with a power-on reset. The contents of RAM are unaffected during a reset; any previously stored data is preserved as long as power is not lost.

The Port I/O latches are reset to 1 in open-drain mode. Weak pullups are enabled during and after the reset. For VDD Supply Monitor and power-on resets, the RESET pin is driven low until the device exits the reset state.

On exit from the reset state, the program counter (PC) is reset, and the system clock defaults to the internal low-power oscillator. The Watchdog Timer is enabled with the low frequency oscillator as its clock source. Program execution begins at location 0x00000000.

6.1.1. Power-On Reset

During power-up, the device is held in a reset state and the RESET pin voltage tracks VDD (through a weak pull-up) until the device is released from reset. After VDD settles above V_{POR}, a delay occurs before the device is released from reset; the delay decreases as the VDD ramp time increases (VDD ramp time is defined as how fast VDD ramps from 0 V to V_{POR}). Figure 6.2 plots the power-on and VDD monitor reset timing. For valid ramp times, the power-on reset delay (t_{POR}) is given in the electrical specifications chapter of the device data sheet.

Note: VDD ramp times slower than the maximum may cause the device to be released from reset before VDD reaches the V_{POR} level.

On exit from a power-on reset, the PORRF flag is set by hardware. When PORRF or VMONRF is set, all of the other reset flags in the RESETFLAG Register are indeterminate. Since all resets cause program execution to begin at the same location, firmware can read the PORF flag in the PMU STATUS register to determine if a power-up was the cause of reset. The contents of internal data memory, including retention RAM should be assumed to be undefined after a power-on reset.

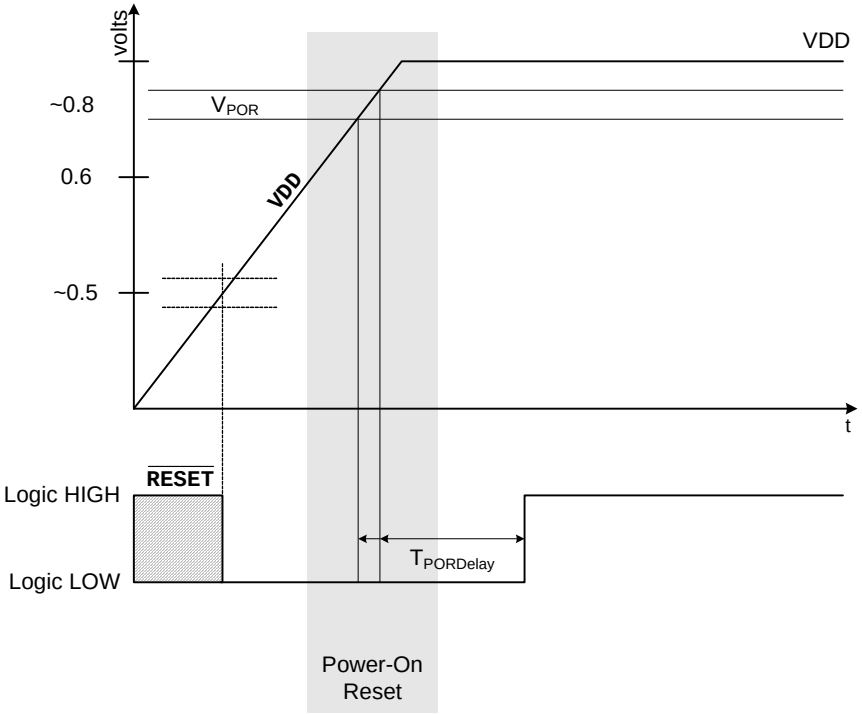


Figure 6.2. Power-On Reset Timing Diagram

6.1.2. VDD Monitor Power-Fail Reset

SiM3U1xx/SiM3C1xx devices have a VDD Supply Monitor that is enabled and selected as a reset source after each power-on. When enabled and selected as a reset source, any power down transition or power irregularity that causes VDD to drop below VRST will cause the RESET pin to be driven low and the core will be held in a reset state. When VDD returns to a level above VRST, the core will be released from the reset state.

After a power-fail reset, the VDDMRF flag reads 1, all of the other reset flags in the RESETFLAG Register are indeterminate, the contents of RAM are invalid, and the VDD Monitor is enabled and selected as a reset source. The enable state of the VDD Monitor and its selection as a reset source is only altered by power-on and power-fail resets. For example, if the VDD supply monitor is de-selected as a reset source and disabled by firmware, then a software reset is performed, the VDD Monitor will remain disabled and de-selected after the reset.

To allow firmware early notification that a power failure is about to occur, the VDDHI bit is cleared when the VDD supply falls below the VDD High threshold. The VDDHI bit can be configured to generate an interrupt.

Note: To protect the integrity of flash contents, the VDD Monitor must be enabled and selected as a reset source if firmware contains routines which erase or write flash memory. If the VDD Monitor is not enabled, any erase or write performed on flash memory will be ignored.

Important Notes:

- The Power-on Reset (POR) delay is not incurred after a VDD supply monitor reset.
- Firmware should take care not to inadvertently disable the VDD Monitor as a reset source when writing to RESETEN to enable other reset sources or to trigger a software reset. All writes to RESETEN should explicitly set VDDMREN to 1 to keep the VDD Monitor enabled as a reset source.
- The VDD Monitor must be enabled before selecting it as a reset source. Selecting the VDD Monitor as a reset source before it has stabilized may generate a system reset. In systems where this reset would be undesirable, a delay should be introduced between enabling the VDD Monitor and selecting it as a reset source. The procedure for enabling the VDD Monitor and selecting it as a reset source from a disabled state is:

1. Enable the VDD Monitor.
2. Wait for the VDD Monitor to stabilize (optional).
3. Select the VDD Monitor as a reset source (VDDMREN bit).

6.1.3. External Reset

The external RESET pin provides a means for external circuitry to force the device into a reset state. Asserting an active-low signal on the RESET pin generates a reset; an external pull-up and/or decoupling of the RESET pin may be necessary to avoid erroneous noise-induced resets. The external reset remains functional even when the device is in the low power modes. The PINRF flag is set on exit from an external reset.

6.1.4. Missing Clock Detector Reset

The missing clock detector (MCD) is a one-shot circuit that is triggered by the APB clock. The APB clock is derived from the AHB clock, so monitoring the APB clock will detect a failure in either clock tree. If the APB clock remains high or low for longer than the Missing Clock Detector Timeout, the one-shot will time out and generate a reset. After a MCD reset, the MCDRF flag will read 1, signifying the MCD as the reset source; otherwise, this bit reads 0. Writing a 1 to the MCDREN bit enables the Missing Clock Detector; writing a 0 disables it. The missing clock detector reset is automatically disabled when the device is in the low power modes. Upon exit from either low power state, the enabled/disabled state of this reset source is restored to its previous value. The state of the RESET pin is unaffected by this reset. Missing clock detector timeout values are given in the device data sheet electrical specifications tables.

6.1.5. Comparator Reset

Comparator 0 (CMP0) or Comparator 1 (CMP1) can be configured as a reset source by writing a 1 to the CMP0REN or CMP1REN bit. The Comparator should be enabled and allowed to settle prior to writing to the enable bits to prevent any turn-on chatter on the output from generating an unwanted reset. The Comparator reset is active-low: if the non-inverting input voltage (on CP0+) is less than the inverting input voltage (on CP0–), the device

is put into the reset state. After a Comparator reset, the CMP0RF or CMP1RF flag will read 1 signifying Comparator 0 or Comparator 1 as the reset source; otherwise, these bits read 0. The Comparator reset source remains functional even when the device is in the low power modes as long as Comparator is also enabled as a wake-up source. The state of the RESET pin is unaffected by this reset.

6.1.6. Watchdog Timer Reset

The Watchdog Timer (WDTIMER0) can be used to recover from certain types of system malfunctions. If a system malfunction prevents user firmware from updating the Watchdog Timer, a reset is generated and the WDTRF bit is set to 1. The Watchdog Timer can be enabled or disabled as a reset source using the WDTREN bit. Note that WDTREN will always read back 1 if the Watchdog Timer is ever disabled. The Watchdog Timer is automatically disabled as a reset source when the device is in the low power modes. Upon exit from either low power state, the enabled/disabled state of this reset source is restored to its previous value.The state of the RESET pin is unaffected by this reset.

6.1.7. RTC Reset

The RTC0 Module can generate a system reset on two events: RTC0 Oscillator Fail or RTC0 Alarm 0. The RTC0 Oscillator Fail event occurs when the RTC0 Missing Clock Detector is enabled and the RTC0 clock is below the Missing Clock Detector Trigger Frequency. An RTC0 Alarm event occurs when the RTC0 Alarm 0 is enabled and the RTC0 timer value matches the Alarm 0 threshold value. The RTC0 can be configured as a reset source by writing a 1 to the RTC0REN bit. The RTC0 reset remains functional even when the device is in a low power mode. The state of the RESET pin is unaffected by this reset.

6.1.8. Software Reset

Firmware may force a reset by writing a 1 to the SWREN bit. The SWRF bit will read 1 following a firmware forced reset. The state of the RESET pin is unaffected by this reset.

6.1.9. Core Reset

The Core Reset is a firmware reset generated in the NVIC by setting the SYSRESETREQ bit.

6.1.10. USB Reset

Writing 1 to the USB0REN bit selects USB0 as a reset source. With USB0 selected as a reset source, a system reset will be generated when either of the following occur:

1. Reset signaling is detected on the USB network. The USB Function Controller (USB0) must be enabled for Reset signaling to be detected.
2. A rising or falling edge on the VBUS pin.

The USB0RF bit will read 1 following a USB0 reset. The state of the RESET pin is unaffected by this reset.

6.1.11. PMU or Wake Reset

The PMU will issue a system reset whenever one of the wakeup events occurs. In power mode PM9, this is the only way to wake the system from this mode other than a RESET pin reset. The WAKERF flag indicates when a PMU Wake reset occurs.

All RSTSRC0 registers may be locked against writes by setting the CLKRSTL bit in the LOCK0_PERIPHLOCK0 register to 1.

6.2. RSTSRC0 Registers

This section contains the detailed register descriptions for RSTSRC0 registers.

Register 6.1. RSTSRC0_RESETEN: System Reset Source Enable

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved				WAKEREN	RTC0REN	USB0REN	CMP1REN	CMP0REN	SWREN	WDTREN	MCDREN	Reserved	VMONREN	Reserved	
Type	R				RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	
Reset	0	0	0	0	1	0	0	0	0	0	1	0	1	1	1	1

Register ALL Access Address

RSTSRC0_RESETEN = 0x4002_D060

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 6.1. RSTSRC0_RESETEN Register Bit Descriptions

Bit	Name	Function
31:12	Reserved	Must write reset value.
11	WAKEREN	**PMU Wakeup Reset Enable.** This bit always reads back as 1. 0: Reserved. 1: Enable the PMU Wakeup event as a reset source.
10	RTC0REN	**RTC0 Reset Enable.** 0: Disable the RTC0 event as a reset source. 1: Enable the RTC0 event as a reset source.
9	USB0REN	**USB0 Reset Enable.** 0: Disable the USB0 reset event as a reset source. 1: Enable the USB0 reset event as a reset source.
8	CMP1REN	**Comparator 1 Reset Enable.** 0: Disable the Comparator 1 event as a reset source. 1: Enable the Comparator 1 event as a reset source.
7	CMP0REN	**Comparator 0 Reset Enable.** 0: Disable the Comparator 0 event as a reset source. 1: Enable the Comparator 0 event as a reset source.

Table 6.1. RSTSRC0_RESETEN Register Bit Descriptions

Bit	Name	Function
6	SWREN	**Software Reset.** Writing a 1 to this bit generates a Software Reset.
5	WDTREN	**Watchdog Timer Reset Enable.** 0: Disable the Watchdog Timer event as a reset source. 1: Enable the Watchdog Timer event as a reset source.
4	MCDREN	**Missing Clock Detector Reset Enable.** 0: Disable the Missing Clock Detector event as a reset source. 1: Enable the Missing Clock Detector event as a reset source.
3	Reserved	Must write reset value.
2	VMONREN	**Voltage Supply Monitor VDD Reset Enable.** 0: Disable the Voltage Supply Monitor VDD event as a reset source. 1: Enable the Voltage Supply Monitor VDD event as a reset source.
1:0	Reserved	Must write reset value.

Register 6.2. RSTSRC0_RESETFLAG: System Reset Flags

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved				WAKERF	RTC0RF	USB0RF	CMP1RF	CMP0RF	SWRF	WDTRF	MCDRF	CORERF	VMONRF	PORRF	PINRF
Type	R				R	R	R	R	R	R	R	R	R	R	R	R
Reset	0	0	0	0	X	X	X	X	X	X	X	X	X	X	X	X

Register ALL Access Address
RSTSRC0_RESETFLAG = 0x4002_D070

Table 6.2. RSTSRC0_RESETFLAG Register Bit Descriptions

Bit	Name	Function
31:12	Reserved	Must write reset value.
11	WAKERF	**PMU Wakeup Reset Flag.** This flag is an indeterminate value when VMONRF or PORRF is set. 0: A PMU Wakeup event did not cause the last system reset. 1: A PMU Wakeup event caused the last system reset.
10	RTC0RF	**RTC0 Reset Flag.** This flag is an indeterminate value when VMONRF or PORRF is set. 0: An RTC0 event did not cause the last system reset. 1: An RTC0 event caused the last system reset.
9	USB0RF	**USB0 Reset Flag.** This flag is an indeterminate value when VMONRF or PORRF is set. 0: A USB0 Reset event did not cause the last system reset. 1: A USB0 Reset event caused the last system reset.
8	CMP1RF	**Comparator 1 Reset Flag.** This flag is an indeterminate value when VMONRF or PORRF is set. 0: A Comparator 1 event did not cause the last system reset. 1: A Comparator 1 event caused the last system reset.
7	CMP0RF	**Comparator 0 Reset Flag.** This flag is an indeterminate value when VMONRF or PORRF is set. 0: A Comparator 0 event did not cause the last system reset. 1: A Comparator 0 event caused the last system reset.

Table 6.2. RSTSRC0_RESETFLAG Register Bit Descriptions

Bit	Name	Function
6	SWRF	**Software Reset Flag.** This flag is an indeterminate value when VMONRF or PORRF is set. 0: A Software Reset event did not cause the last system reset. 1: A Software Reset event caused the last system reset.
5	WDTRF	**Watchdog Timer Reset Flag.** This flag is an indeterminate value when VMONRF or PORRF is set. 0: A Watchdog Timer event did not cause the last system reset. 1: A Watchdog Timer event caused the last system reset.
4	MCDRF	**Missing Clock Detector Reset Flag.** This flag is an indeterminate value when VMONRF or PORRF is set. 0: A Missing Clock Detector event did not cause the last system reset. 1: A Missing Clock Detector event caused the last system reset.
3	CORERF	**Core Reset Flag.** This flag is an indeterminate value when VMONRF or PORRF is set. 0: A Core Reset event did not cause the last system reset. 1: A Core Reset event caused the last system reset.
2	VMONRF	**Voltage Supply Monitor VDD Reset Flag.** After checking PORRF, firmware should then check VMONRF for the last reset source, as all other flags are indeterminate if VMONRF is set. This flag is an indeterminate value when PORRF is set. 0: A Voltage Supply Monitor VDD Reset event did not cause the last system reset. 1: A Voltage Supply Monitor VDD Reset event caused the last system reset.
1	PORRF	**Power-On Reset Flag.** This flag should be checked first by firmware as all other flags are indeterminate if PORRF is set. This flag is an indeterminate value when VMONRF is set. Firmware can check the PORF bit in the PMU STATUS register to determine if the last reset was caused by a power-on reset. 0: A Power-On Reset event did not cause the last system reset. 1: A Power-On Reset event caused the last system reset.
0	PINRF	**Pin Reset Flag.** This flag is an indeterminate value when VMONRF or PORRF is set. 0: A $\overline{\text{RESET}}$ pin event did not cause the last system reset. 1: A $\overline{\text{RESET}}$ pin event caused the last system reset.

Register 6.3. RSTSRC0_CONFIG: Configuration Options

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved															PMSEL
Type	R															RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

RSTSRC0_CONFIG = 0x4002_D080

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 6.3. RSTSRC0_CONFIG Register Bit Descriptions

Bit	Name	Function
31:1	Reserved	Must write reset value.
0	PMSEL	**Power Mode Select.** This bit is used to enable the reset circuitry to wake the device from power mode 9 (PM9). It should be set to 1 before entering PM9. This bit should be cleared to 0 for all other low power modes.

6.3. RSTSRC0 Register Memory Map

Table 6.4. RSTSRC0 Memory Map

Register Name	RSTSRC0_RESETEN	RSTSRC0_RESETFLAG	RSTSRC0_CONFIG
ALL Address	0x4002_D060	0x4002_D070	0x4002_D080
Access Methods	ALL | SET | CLR	ALL	ALL | SET | CLR
Bit 31	Reserved	Reserved	Reserved
Bit 30			
Bit 29			
Bit 28			
Bit 27			
Bit 26			
Bit 25			
Bit 24			
Bit 23			
Bit 22			
Bit 21			
Bit 20			
Bit 19			
Bit 18			
Bit 17			
Bit 16			
Bit 15			
Bit 14			
Bit 13			
Bit 12			
Bit 11	WAKEREN	WAKERF	
Bit 10	RTC0REN	RTC0RF	
Bit 9	USB0REN	USB0RF	
Bit 8	CMP1REN	CMP1RF	
Bit 7	CMP0REN	CMP0RF	
Bit 6	SWREN	SWRF	
Bit 5	WDTREN	WDTRF	
Bit 4	MCDREN	MCDRF	
Bit 3	Reserved	CORERF	
Bit 2	VMONREN	VMONRF	
Bit 1	Reserved	PORRF	
Bit 0		PINRF	PMSEL

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

7. Register Security (LOCK0)

This section describes the Register Security (LOCK) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

Note that features related to the USB peripheral are only available on the SiM3U1xx device family.

7.1. Security Features

The Security module includes the following features:

- Centralized global key protection implementation with mask bits for each peripheral group.

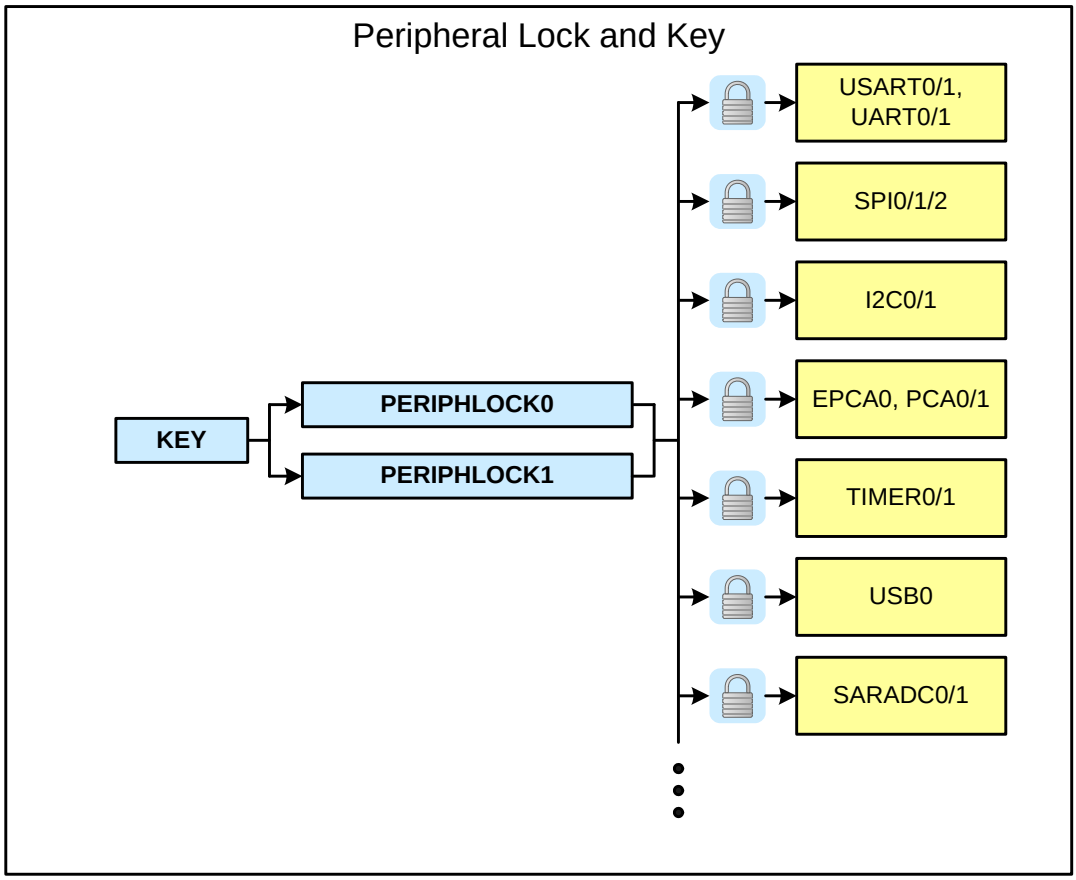


Figure 7.1. Security Block Diagram

The peripherals on the SiM3U1xx/SiM3C1xx devices have a register lock and key mechanism that prevents any undesired accesses of the peripherals from firmware. Each bit in the PERIPHLOCKx registers controls a set of peripherals. A key sequence must be written in order to the KEY register to modify any of the bits in PERIPHLOCK0 or PERIPOHLOCK1. Any subsequent write to KEY will then inhibit any accesses of the PERIPHLOCK registers until they are unlocked again through KEY. Reading the KEY register indicates the current status of the PERIPHLOCK registers lock state.

If a peripheral's registers are locked, all writes will be ignored. The registers can always be read, regardless of the peripheral's lock state.

7.2. LOCK0 Registers

This section contains the detailed register descriptions for LOCK0 registers.

Register 7.1. LOCK0_KEY: Security Key

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved								KEY							
Type	R								RW							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

LOCK0_KEY = 0x4004_9000

Table 7.1. LOCK0_KEY Register Bit Descriptions

Bit	Name	Function
31:8	Reserved	Must write reset value.
7:0	KEY	**Peripheral Lock Mask Key.** This field prevents accidental writes to the PERIPHLOCK0 and PERIPHLOCK1 registers. Writing 0xA5 and 0xF1 will unlock the PERIPHLOCK registers. Any value written to KEY while the PERIPHLOCK registers are unlocked will lock the register. Reading this register returns the current lock state (0 = Registers locked and no keys written, 1 = Registers locked and first key written, 2 = Registers unlocked)

Register 7.2. LOCK0_PERIPHLOCK0: Peripheral Lock Control 0

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved			IVCL	Reserved	EVREGL	LPOSCL	VREGL	EXTOSCL	PLLL	I2SL	VREFL	LPTL	DMAXBARL	DMACTRLL	IDACL
Type	RW			RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	VMONL	CLKRSTL	RTCL	CRCL	AESL	EMIFL	CSL	CMPL	SSGL	SARADCL	USBL	TIMERL	PCAL	I2CL	SPIL	USARTL
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

LOCK0_PERIPHLOCK0 = 0x4004_9020

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 7.2. LOCK0_PERIPHLOCK0 Register Bit Descriptions

Bit	Name	Function
31:29	Reserved	Must write reset value.
28	IVCL	**IVC Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the IVC0 Module registers. 1: Lock the IVC0 Module registers (bits can still be read).
27	Reserved	Must write reset value.
26	EVREGL	**External Regulator Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the External Regulator (EXTVREG0) Module registers. 1: Lock the External Regulator (EXTVREG0) Module registers (bits can still be read).
25	LPOSCL	**Low Power Oscillator Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the Low Power Oscillator (LPOSC0) Module registers. 1: Lock the Low Power Oscillator (LPOSC0) Module registers (bits can still be read).

Table 7.2. LOCK0_PERIPHLOCK0 Register Bit Descriptions

Bit	Name	Function
24	VREGL	**Voltage Regulator Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the Voltage Regulator (VREG0) Module registers. 1: Lock the Voltage Regulator (VREG0) Module registers (bits can still be read).
23	EXTOSCL	**External Oscillator Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the External Oscillator (EXTOSC0) Module registers. 1: Lock the External Oscillator (EXTOSC0) Module registers (bits can still be read).
22	PLLL	**PLL Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the PLL0 Module registers. 1: Lock the PLL0 Module registers (bits can still be read).
21	I2SL	**I2S Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the I2S0 Module registers. 1: Lock the I2S0 Module registers (bits can still be read).
20	VREFL	**Voltage Reference Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the Voltage Reference (VREF0) Module registers. 1: Lock the Voltage Reference (VREF0) Module registers (bits can still be read).
19	LPTL	**Low Power Timer Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the Low Power Timer (LPTIMER0) Module registers. 1: Lock the Low Power Timer (LPTIMER0) Module registers (bits can still be read).
18	DMAXBARL	**DMA Crossbar Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the DMA Crossbar (DMAXBAR0) Module registers. 1: Lock the DMA Crossbar (DMAXBAR0) Module registers (bits can still be read).
17	DMACTRLL	**DMA Controller Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the DMA Controller (DMACTRL0) Module registers. 1: Lock the DMA Controller (DMACTRL0) Module registers (bits can still be read).
16	IDACL	**IDAC Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the IDAC0 and IDAC1 Module registers. 1: Lock the IDAC0 and IDAC1 Module registers (bits can still be read).
15	VMONL	**Voltage Supply Monitor Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the Voltage Supply Monitor (VMON0) Module registers. 1: Lock the Voltage Supply Monitor (VMON0) Module registers (bits can still be read).

Table 7.2. LOCK0_PERIPHLOCK0 Register Bit Descriptions

Bit	Name	Function
14	CLKRSTL	**Clock Control and Reset Sources Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the Clock Control (CLKCTRL) and Reset Sources (RSTSRC) Module registers. 1: Lock the Clock Control (CLKCTRL) and Reset Sources (RSTSRC) Module registers (bits can still be read).
13	RTCL	**RTC Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the RTC0 Module registers. 1: Lock the RTC0 Module registers (bits can still be read).
12	CRCL	**CRC Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the CRC0 Module registers. 1: Lock the CRC0 Module registers (bits can still be read).
11	AESL	**AES Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the AES0 Module registers. 1: Lock the AES0 Module registers (bits can still be read).
10	EMIFL	**EMIF Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the External Memory Interface (EMIF0) Module registers. 1: Lock the External Memory Interface (EMIF0) Module registers (bits can still be read).
9	CSL	**Capacitive Sensing Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the Capacitive Sensing (CAPSENSE0) Module registers. 1: Lock the Capacitive Sensing (CAPSENSE0) Module registers (bits can still be read).
8	CMPL	**Comparator Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the Comparator 0 and Comparator 1 Module registers. 1: Lock the Comparator 0 and Comparator 1 Module registers (bits can still be read).
7	SSGL	**SSG Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the SSG0 Module registers. 1: Lock the SSG0 Module registers (bits can still be read).
6	SARADCL	**SARADC Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the SARADC0 and SARADC1 Module registers. 1: Lock the SARADC0 and SARADC1 Module registers (bits can still be read).

Table 7.2. LOCK0_PERIPHLOCK0 Register Bit Descriptions

Bit	Name	Function
5	USBL	**USB Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the USB0 Module registers. 1: Lock the USB0 Module registers (bits can still be read).
4	TIMERL	**Timer Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the TIMER0 and TIMER1 Module registers. 1: Lock the TIMER0 and TIMER1 Module registers (bits can still be read).
3	PCAL	**PCA Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the EPCA0, PCA0, and PCA1 Module registers. 1: Lock the EPCA0, PCA0, and PCA1 Module registers (bits can still be read).
2	I2CL	**I2C Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the I2C0 and I2C1 Module registers. 1: Lock the I2C0 and I2C1 Module registers (bits can still be read).
1	SPIL	**SPI Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the SPI0, SPI1, and SPI2 Module registers. 1: Lock the SPI0, SPI1, and SPI2 Module registers (bits can still be read).
0	USARTL	**USART/UART Module Lock Enable.** This bit cannot be written until the PERIPHLOCK0 register is unlocked using KEY. 0: Unlock the USART0, USART1, UART0, and UART1 Module registers. 1: Lock the USART0, USART1, UART0, and UART1 Module registers (bits can still be read).

Register 7.3. LOCK0_PERIPHLOCK1: Peripheral Lock Control 1

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved															PMUL
Type	R															RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

LOCK0_PERIPHLOCK1 = 0x4004_9040

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 7.3. LOCK0_PERIPHLOCK1 Register Bit Descriptions

Bit	Name	Function
31:1	Reserved	Must write reset value.
0	PMUL	**PMU Module Lock Enable.** This bit cannot be written until the PERIPHLOCK1 register is unlocked using KEY. 0: Unlock the PMU Module registers. 1: Lock the PMU Module registers (bits can still be read).

7.3. LOCK0 Register Memory Map

Table 7.4. LOCK0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
LOCK0_KEY	0x4004_9000	ALL	Reserved																								KEY							
LOCK0_PERIPHLOCK0	0x4004_9020	ALL | SET | CLR	Reserved		IVCL	Reserved	EVREGL	LPOSCL	VREGL	EXTOSCL	PLLL	I2SL	VREFL	LPTL	DMAXBARL	DMACTRLL	IDACL	VMONL	CLKRSTL	RTCL	CRCL	AESL	EMIFL	CSL	CMPL	SSGL	SARADCL	USBL	TIMERL	PCAL	I2CL	SPIL	USARTL
LOCK0_PERIPHLOCK1	0x4004_9040	ALL | SET | CLR	Reserved																														PMUL

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

8. Port I/O Configuration

This section describes the Port I/O Crossbars and general Port Bank configuration and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

8.1. Port Bank Description

The features of the SiM3U1xx/SiM3C1xx port banks are shown in Table 8.1.

Table 8.1. Port Bank Features

Port Bank	Type	Crossbar 0	Crossbar 1	Pulse Generator	Port Mapped Level Shifter	5 V Tolerant
PB0	Standard (PBSTD)	✓				
PB1	Standard (PBSTD)	✓				
PB2	Standard (PBSTD)		✓	✓	(PB2.0-5 only)	
PB3	Standard (PBSTD)		✓			✓
PB4	High Drive (PBHD)					✓

Note: I/O pins in Port Banks 0, 1, and 2 operate between 0 V and V_{IO}. Port Bank 3 I/O pins can operate at voltages up to V_{IO} + 2 V when configured as digital inputs (or as open-drain digital outputs with an external pull-up resistor), which allows them to interface to 5 V signals when V_{IO} = 3.3 V. Port Bank 4 I/O pins operate between 0 V and V_{IOHD}, which allows them to function as digital inputs or as open-drain or push-pull digital outputs up to the maximum value for V_{IOHD}.

8.2. Crossbars

8.2.1. Crossbar Features

There are two port I/O crossbars included on the SiM3U1xx/SiM3C1xx, with the following features:

- Flexible assignment of many digital peripherals to port pins.
- Pin skip capabilities to reserve specific I/O for other purposes (analog signals, GPIO, layout considerations)

The port I/O crossbars on the SiM3U1xx/SiM3C1xx are used to route many of the digital peripherals to the devices I/O pins. They allow the user to select the specific mix of peripherals that are needed for the application, and route them out of the device, leaving the unused pins available for general-purpose I/O. Crossbar 0 controls peripherals to PB0 and PB1, while Crossbar 1 controls peripherals to PB2 and PB3.

Table 8.2. Crossbar Peripheral Availability

Peripheral Name	Functional Group	Available on Crossbar 0 (PB0 and PB1)	Available on Crossbar 1 (PB2 and PB3)
AHB Clock / 16		✓	
Comparator 0	Synchronous Output	✓	✓
	Asynchronous Output	✓	
Comparator 1	Synchronous Output	✓	✓
	Asynchronous Output	✓	
EPCA0		✓	
I2C0		✓	✓
I2C1			✓
I2S0	Transmitter	✓	✓
	Receiver		✓
LPTIMER0			✓
PB0	High Drive Kill		✓
PCA0		✓	
PCA1		✓	
RTC0			✓

Table 8.2. Crossbar Peripheral Availability (Continued)

Peripheral Name	Functional Group	Available on Crossbar 0 (PB0 and PB1)	Available on Crossbar 1 (PB2 and PB3)
SPI0	Clock/Data	✓	
	Slave Select		
SPI1	Clock/Data	✓	✓
	Slave Select		
SPI2	Clock/Data	✓	✓
	Slave Select		
SSG0			✓
Timer 0	Count	✓	
	External Output	✓	
Timer 1	Count	✓	
	External Output	✓	
UART0	Data	✓	✓
	Flow Control		
UART1	Data	✓	✓
	Flow Control		
USART0	Data	✓	
	Flow Control		
	Clock Out		
USART1	Data	✓	✓
	Flow Control		
	Clock Out		

8.2.2. Crossbar Configuration

The peripherals which are routed through each crossbar have a specific priority order in which they are assigned to pins and a specific range of pins where they can be routed. The crossbar assigns peripherals in priority order, starting with the highest-priority peripherals and the lowest-order port pins available to the peripherals. When a peripheral is enabled, all of the pins associated with that peripheral are routed in sequence. Some peripherals are split into multiple functional groups, to route only the necessary pins. Additionally, pin skip registers can be used to prevent the crossbar from assigning peripherals to those pins.

When configuring the crossbar, all settings should be made to the crossbar and Port Bank registers before enabling the crossbar. This ensures that peripherals will not shift around while each one is being enabled and Port I/O pins will remain stable. The settings in PBOUTMD, PBMDSEL, or PBSKIPEN will not take effect until the crossbars are enabled.

If any pins are used for special functions not associated with the crossbars, these pins should be skipped using the corresponding Port Bank PBSKIPEN register. This applies to External Interrupts, SARADC0/1 inputs, Capacitive Sensing inputs, Comparator inputs, IDAC0/1 outputs, PB2 pins used with the pulse generator, and other analog and non-crossbar signals. The only EMIF pins that must be skipped by the crossbar are CS0 and CS1, but the EMIF will take precedence over the crossbar function during any read or write operation.

Registers XBAR0H, XBAR0L, and XBAR1 are used to enable peripherals on the different crossbars. Some peripherals use multiple signals that are all enabled when the corresponding bit is set to 1. For example, enabling I2C0 on the crossbar enables both data and clock signals (SDA and SCL). Several peripherals have individual groups of pins that can be enabled or disabled as well. In the crossbar priority tables (Table 8.3 and Table 8.4), such pins are listed with the same numerical priority and a letter (A, B, or C) indicating the priority of the functional group. In all cases, the primary functional group (A) must be enabled on the crossbar for the other groups (B or C) to be routed out. For example, on crossbar 0, the USART0 Data group (priority 1A) must be enabled in order to use either the USART0 Flow Control group (priority 1B) or the Clock Output group (priority 1C).

When a peripheral is selected in more than one place (crossbars or high-drive I/O), the output pins will map to all places where that peripheral is enabled. For input pins, only one of the selections will be valid. Priority is given first to the high-drive I/O, then to crossbar 1, and finally to crossbar 0.

8.2.2.1. Crossbar 0

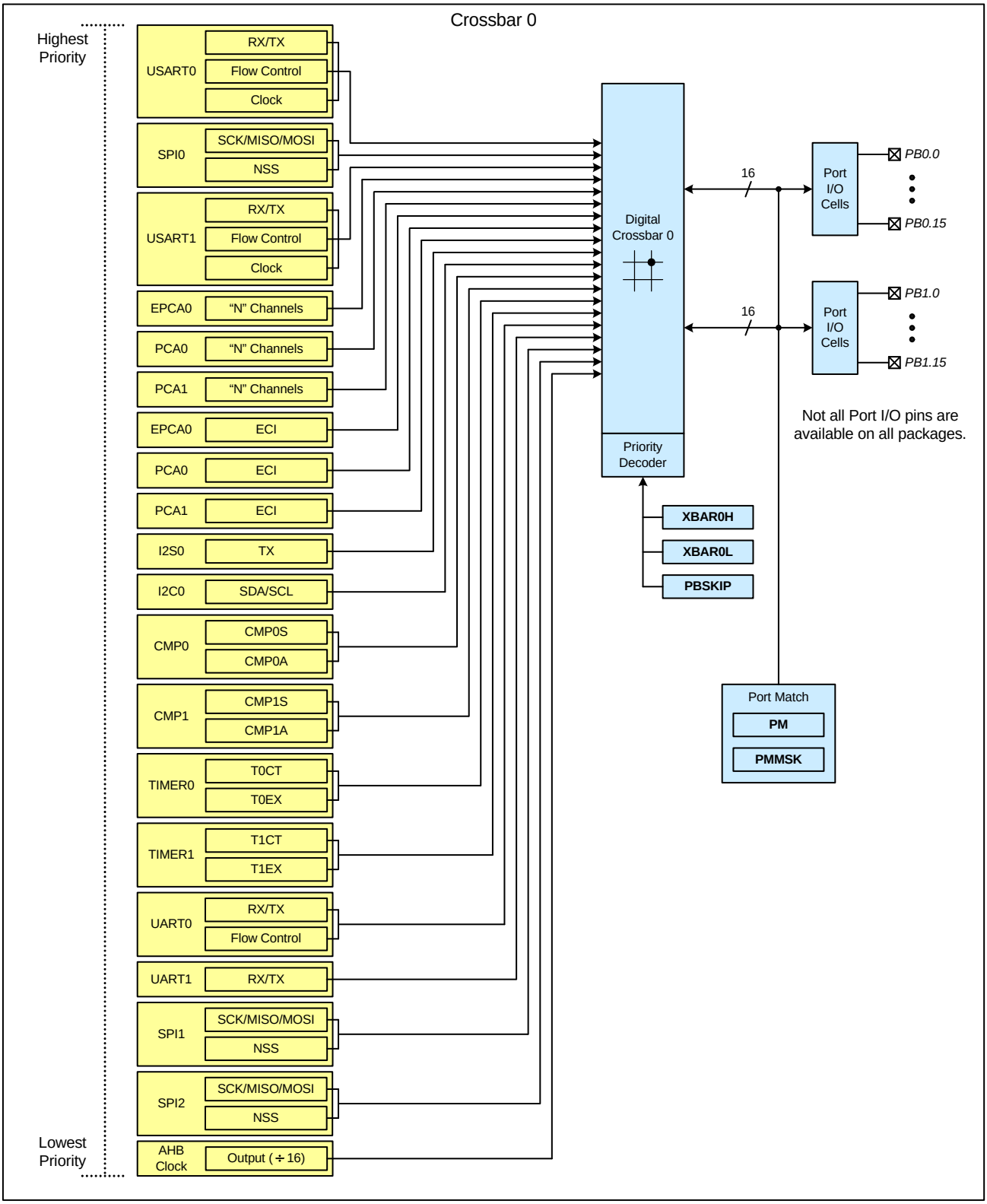


Figure 8.1. Crossbar 0 Block Diagram

Table 8.3. Crossbar 0 Peripherals and Priority

Peripheral Name	Functional Group	Priority	Enable Bits	Signal Names (in order)	First Available Port Pin	Last Available Port Pin
USART0	Data	1-A	USART0EN	USART0_TX USART0_RX	PB0.0	PB1.15
	Flow Control	1-B	USART0FCEN	USART0_RTS USART0_CTS		
	Clock Out	1-C	USART0CEN	USART0_UCLK		
SPI0	Clock/Data	2-A	SPI0EN	SPI0_SCK SPI0_MISO SPI0_MOSI	PB0.0	PB1.15
	Slave Select	2-B	SPI0NSSEN	SPI0_NSS		
USART1	Data	3-A	USART1EN	USART1_TX USART1_RX	PB0.0	PB1.15
	Flow Control	3-B	USART1FCEN	USART1_RTS USART1_CTS		
	Clock Out	3-C	USART1CEN	USART1_UCLK		
EPCA0		4	EPCA0EN[2:0]	EPCA0_STD_CEX0 EPCA0_STD_CEX1 EPCA0_STD_CEX2 EPCA0_STD_CEX3 EPCA0_STD_CEX4 EPCA0_STD_CEX5	PB0.0	PB1.15
PCA0		5	PCA0EN[1:0]	PCA0_CEX0 PCA0_CEX1	PB0.0	PB1.15
PCA1		6	PCA1EN[1:0]	PCA1_CEX0 PCA1_CEX1	PB0.0	PB1.15
EECI0		7	EECI0EN	EPCA0_ECI	PB0.0	PB1.15
ECI0		8	ECI0EN	PCA0_ECI	PB0.0	PB1.15
ECI1		9	ECI1EN	PCA1_ECI	PB0.0	PB1.15
I2S0 Transmitter		10	I2S0TXEN	I2S0_TX_WS I2S0_TX_SCK I2S0_TX_SD	PB0.0	PB1.15
I2C0		11	I2C0EN	I2C0_SDA I2C0_SCL	PB0.0	PB1.15

Table 8.3. Crossbar 0 Peripherals and Priority (Continued)

Peripheral Name	Functional Group	Priority	Enable Bits	Signal Names (in order)	First Available Port Pin	Last Available Port Pin
Comparator 0	Synchronous Output	12	CMP0SEN	CMP0_S	PB0.0	PB1.15
	Asynchronous Output	13	CMP0AEN	CMP0_A		
Comparator 1	Synchronous Output	14	CMP1SEN	CMP1_S	PB0.0	PB1.15
	Asynchronous Output	15	CMP1AEN	CMP1_A		
Timer 0	Count	16	TMR0CTEN	TIMER0_CT	PB0.0	PB1.15
	Input / Output	17	TMR0EXEN	TIMER0_EX	PB0.0	PB1.15
Timer 1	Count	18	TMR1CTEN	TIMER1_CT	PB0.0	PB1.15
	Input / Output	19	TMR1EXEN	TIMER1_EX	PB0.0	PB1.15
UART0	Data	20-A	UART0EN	UART0_TX UART0_RX	PB0.0	PB1.15
	Flow Control	20-B	UART0FCEN	UART0_RTS UART0_CTS		
UART1	Data	21-A	UART1EN	UART1_TX UART1_RX	PB0.0	PB1.15
SPI1	Clock/Data	22-A	SPI1EN	SPI1_SCK SPI1_MISO SPI1_MOSI	PB0.0	PB1.15
	Slave Select	22-B	SPI1NSSEN	SPI1_NSS		
SPI2	Clock/Data	23-A	SPI2EN	SPI2_SCK SPI2_MISO SPI2_MOSI	PB0.0	PB1.15
	Slave Select	23-B	SPI2NSSEN	SPI2_NSS		
AHB Clock / 16		24	AHBEN	AHB_OUT	PB0.0	PB1.15

8.2.2.2. Crossbar 1

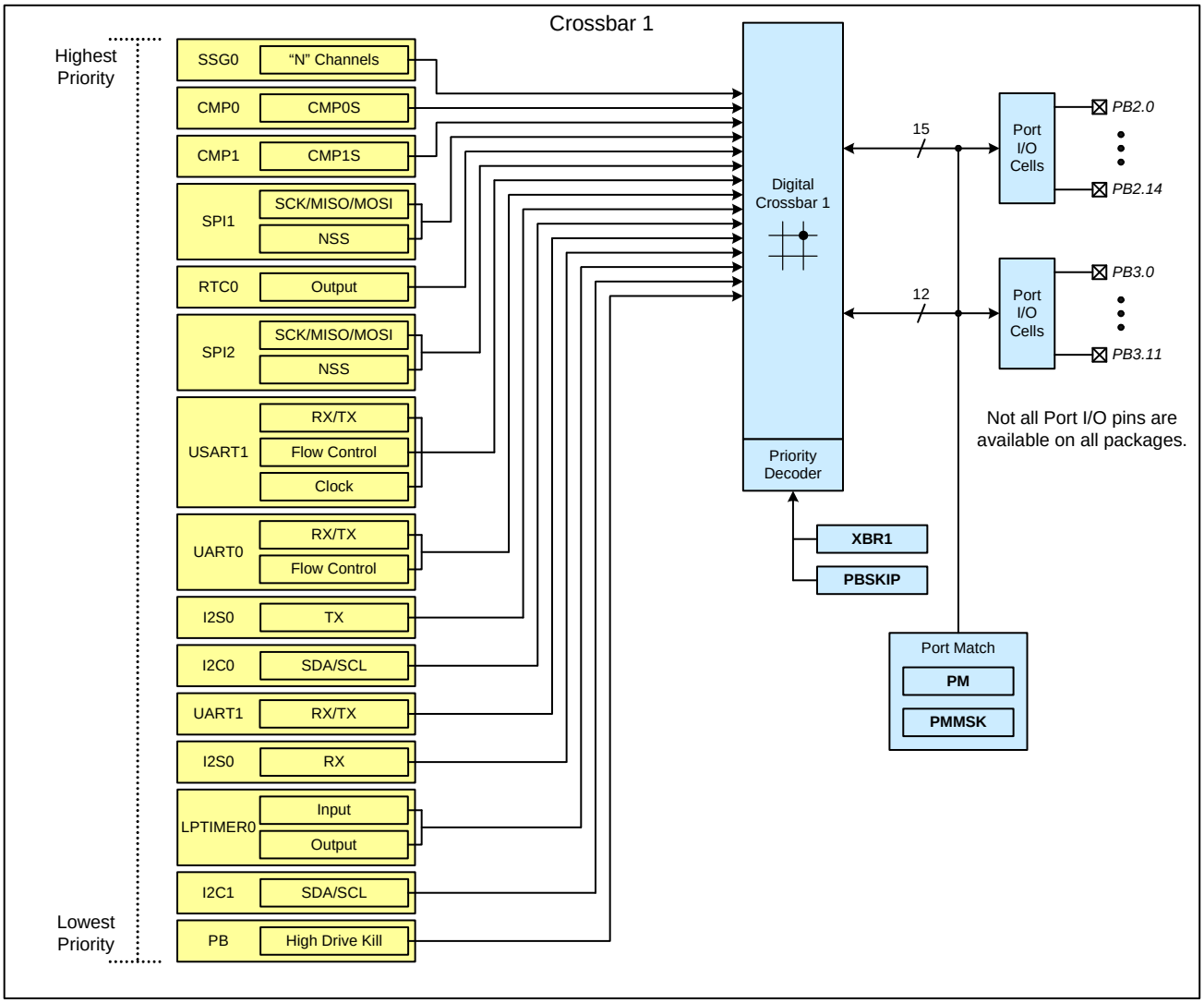


Figure 8.2. Crossbar 1 Block Diagram

Table 8.4. Crossbar 1 Peripherals and Priority

Peripheral Name	Functional Group	Priority	Enable Bits	Signal Names (in order)	First Available Port Pin	Last Available Port Pin
SSG0		1	SSG0EN[1:0]	SSG0_EX0 SSG0_EX1 SSG0_EX2 SSG0_EX3	PB2.0	PB3.11
Comparator 0	Synchronous Output	2	CMP0SEN	CMP0_S	PB2.0	PB3.11
Comparator 1	Synchronous Output	3	CMP1SEN	CMP1_S	PB2.0	PB3.11
SPI1	Clock/Data	4-A	SPI1EN	SPI1_SCK SPI1_MISO SPI1_MOSI	PB2.2	PB3.11
	Slave Select	4-B	SPI1NSSEN	SPI1_NSS		
RTC0		5	RTC0EN	RTC0_OUT	PB2.5	PB3.11
SPI2	Clock / Data	6-A	SPI2EN	SPI2_SCK SPI2_MISO SPI2_MOSI	PB2.6	PB3.11
	Slave Select	6-B	SPI2NSSEN	SPI2_NSS		
USART1	Data	7-A	USART1EN	USART1_TX USART1_RX	PB2.6	PB3.11
	Flow Control	7-B	USART1FCEN	USART1_RTS USART1_CTS		
	Clock Out	7-C	USART1CEN	USART1_UCLK		
UART0	Data	8-A	UART0EN	UART0_TX UART0_RX	PB2.11	PB3.11
	Flow Control	8-B	UART0FCEN	UART0_RTS UART0_CTS		
I2S0 Transmitter		9	I2S0TXEN	I2S0_TX_WS I2S0_TX_SCK I2S0_TX_SD	PB2.11	PB3.11

Table 8.4. Crossbar 1 Peripherals and Priority (Continued)

Peripheral Name	Functional Group	Priority	Enable Bits	Signal Names (in order)	First Available Port Pin	Last Available Port Pin
I2C0		10	I2C0EN	I2C0_SDA I2C0_SCL	PB3.0	PB3.11
UART1	Data	11	UART1EN	UART1_TX UART1_RX	PB3.3	PB3.11
I2S0 Receiver		12	I2S0RXEN	I2S0_RX_WS I2S0_RX_SCK I2S0_RX_SD	PB3.0	PB3.11
LPTIMER0		13	LPT0OEN	LPTIMER0_OUT	PB3.1	PB3.11
I2C1		14	I2C1EN	I2C1_SDA I2C1_SCL	PB3.2	PB3.11
PB	High Drive Kill	15	KILLHDEN	$\overline{\text{PB_HDKill}}$	PB3.3	PB3.11

8.2.2.3. Crossbar Configuration Examples

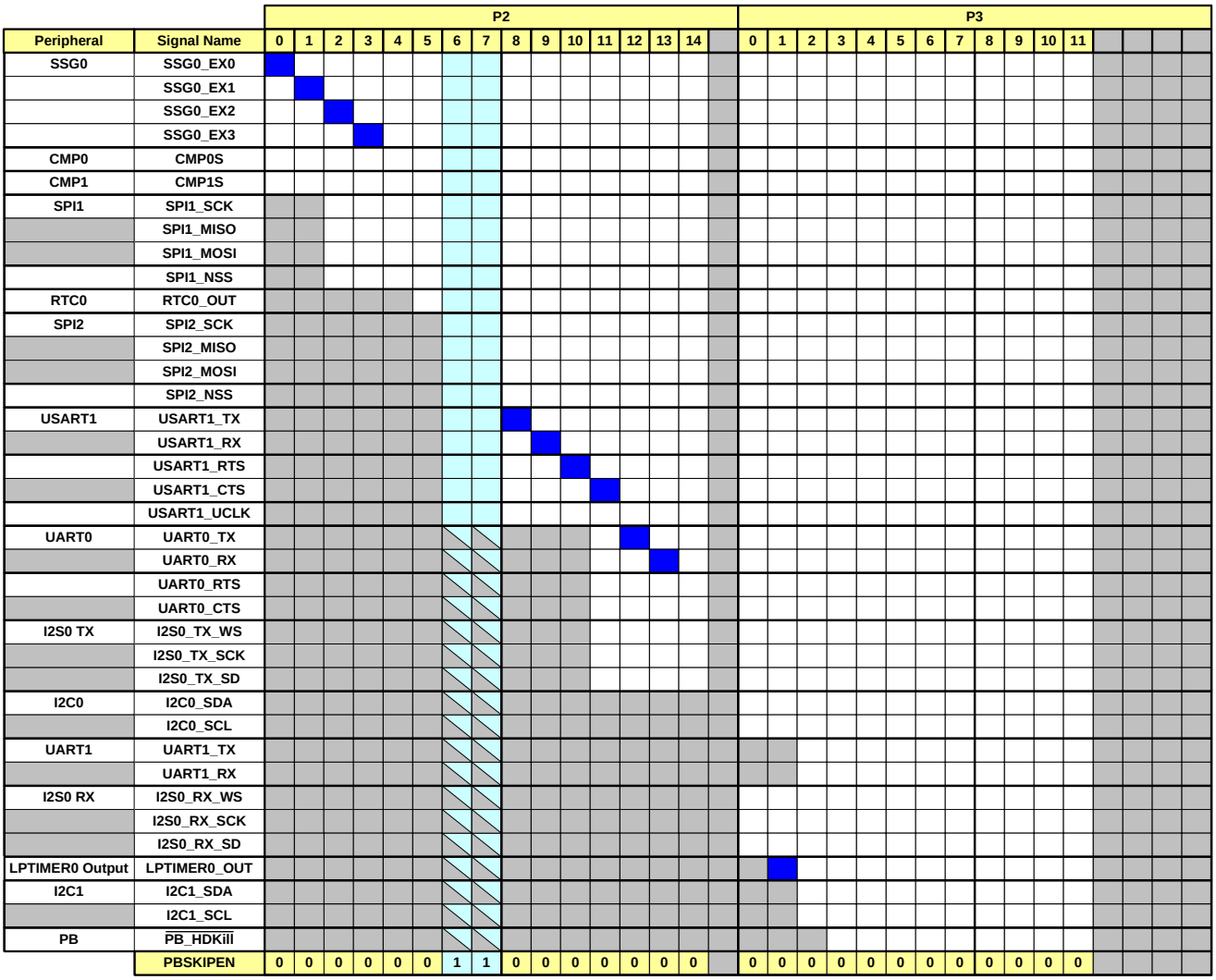
Configuring Crossbar 0 to enable the SPI0 SCK, MISO, and MOSI pins, all EPCA0 channels, UART0 data, and UART1 data would look like:

		P0																P1													
Peripheral	Signal Name	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
USART0	USART0_TX																																
	USART0_RX																																
	USART0_RTS																																
	USART0_CTS																																
	USART0_UCLK																																
SPI0	SPI0_SCK									■																							
	SPI0_MISO										■																						
	SPI0_MOSI											■																					
	SPI0_NSS																																
USART1	USART1_TX																																
	USART1_RX																																
	USART1_RTS																																
	USART1_CTS																																
	USART1_UCLK																																
EPCA0	EPCA0_CEX0												■																				
	EPCA0_CEX1													■																			
	EPCA0_CEX2																■																
	EPCA0_CEX3																	■															
	EPCA0_CEX4																		■														
	EPCA0_CEX5																			■													
PCA0	PCA0_CEX0																																
	PCA0_CEX1																																
PCA1	PCA1_CEX0																																
	PCA1_CEX1																																
EPCA0 ECI	EPCA0_ECI																																
PCA0 ECI	PCA0_ECI																																
PCA1 ECI	PCA1_ECI																																
I2S0 TX	I2S0_TX_WS																																
	I2S0_TX_SCK																																
	I2S0_TX_SD																																
I2C0	I2C0_SDA																																
	I2C0_SCL																																
CMP0	CMP0S																																
	CMP0A																																
CMP1	CMP1S																																
	CMP1A																																
TIMER0	TIMER0_CT																																
	TIMER0_EX																																
TIMER1	TIMER1_CT																																
	TIMER1_EX																																
UART0	UART0_TX																				■												
	UART0_RX																					■											
	UART0_RTS																																
	UART0_CTS																																
UART1	UART1_TX																						■										
	UART1_RX																							■									
SPI1	SPI1_SCK																																
	SPI1_MISO																																
	SPI1_MOSI																																
	SPI1_NSS																																
SPI2	SPI2_SCK																																
	SPI2_MISO																																
	SPI2_MOSI																																
	SPI2_NSS																																
AHB Clock / 16	AHB_OUT																																
	PBSKIPEN	1	1	1	1	1	1	1	1	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 8.3. Crossbar 0 Example Configuration

In this particular example, PB0.0-PB0.7, PB0.13, and PB0.14 are skipped. As a result, the SPI0 pins are placed on PB0.8-PB0.10, since this is the highest priority peripheral. The EPCA0 signals are next, and they're placed on PB0.11-PB1.2, since PB0.13 and PB0.14 are skipped. The UART0 RX and TX pins are then allocated to PB1.3 and PB1.4, since this is the next peripheral in the priority list. Finally, the UART1 pins are placed on PB1.5 and PB1.6.

Similarly, configuring Crossbar 1 to enable all four channels of SSG0, USART1 data and flow control, UART0 data, and the LPTIMER0 output would look like:



		P2															P3														
Peripheral	Signal Name	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	0	1	2	3	4	5	6	7	8	9	10	11				
SSG0	SSG0_EX0	■																														
	SSG0_EX1		■																													
	SSG0_EX2			■																												
	SSG0_EX3				■																											
CMP0	CMP0S																															
CMP1	CMP1S																															
SPI1	SPI1_SCK																															
	SPI1_MISO																															
	SPI1_MOSI																															
	SPI1_NSS																															
RTC0	RTC0_OUT																															
SPI2	SPI2_SCK																															
	SPI2_MISO																															
	SPI2_MOSI																															
	SPI2_NSS																															
USART1	USART1_TX									■																						
	USART1_RX										■																					
	USART1_RTS											■																				
	USART1_CTS												■																			
	USART1_UCLK																															
UART0	UART0_TX													■																		
	UART0_RX														■																	
	UART0_RTS																															
	UART0_CTS																															
I2S0 TX	I2S0_TX_WS																															
	I2S0_TX_SCK																															
	I2S0_TX_SD																															
I2C0	I2C0_SDA																															
	I2C0_SCL																															
UART1	UART1_TX																															
	UART1_RX																															
I2S0 RX	I2S0_RX_WS																															
	I2S0_RX_SCK																															
	I2S0_RX_SD																															
LPTIMER0 Output	LPTIMER0_OUT																	■														
I2C1	I2C1_SDA																															
	I2C1_SCL																															
PB	PB_HDKill																															
	PBSKIPEN	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				

Figure 8.4. Crossbar 1 Example Configuration

Because the SSG0 peripheral has the highest priority on Crossbar 1, this peripheral is allocated first. The first pin this peripheral can appear on is PB2.0, so the SSG0 channels appear on PB2.0-PB2.3.

The USART1 peripheral has the next highest priority. The first pin this peripheral can appear on is PB2.6, but both PB2.6 and PB2.7 are skipped in this particular crossbar configuration. The USART1 data signals then appear on PB2.8 and PB2.9, and the flow control signals appear on PB2.10 and PB2.11.

The next peripheral in the priority list is UART0. In this instance, only the data lines of UART0 are enabled, so the flow control signals do not appear on the port pins. The first pin this peripheral can appear on is PB2.11, so this peripheral is shifted to PB2.12 and PB2.13 because of the higher-priority USART1 peripheral already allocated to PB2.11.

The final peripheral enabled is the LPTIMER0 output. This peripheral has the lowest priority of those enabled, so it is allocated last. The first pin this signal can appear on is PB3.1, so this pin is allocated to the LPTIMER0 output. This leaves PB2.14 and PB3.0 unallocated.

8.3. Port Bank Standard (PBSTD) Features

The Port Bank Standard module includes the following features:

- Push-pull or open-drain output modes and analog or digital input modes.
- Option for high or low output drive strength.
- Port Match allows any pin or combination of pins to generate an interrupt.
- Internal pull-up resistors are enabled or disabled on a port-by-port basis.
- Pulse generator logic which can produce fast pulses on one or more output pins (PB2 only).
- PB2.0-PB2.5 can also serve as inputs to the Port Mapped Level Shifters available on PB4.0-PB4.5.

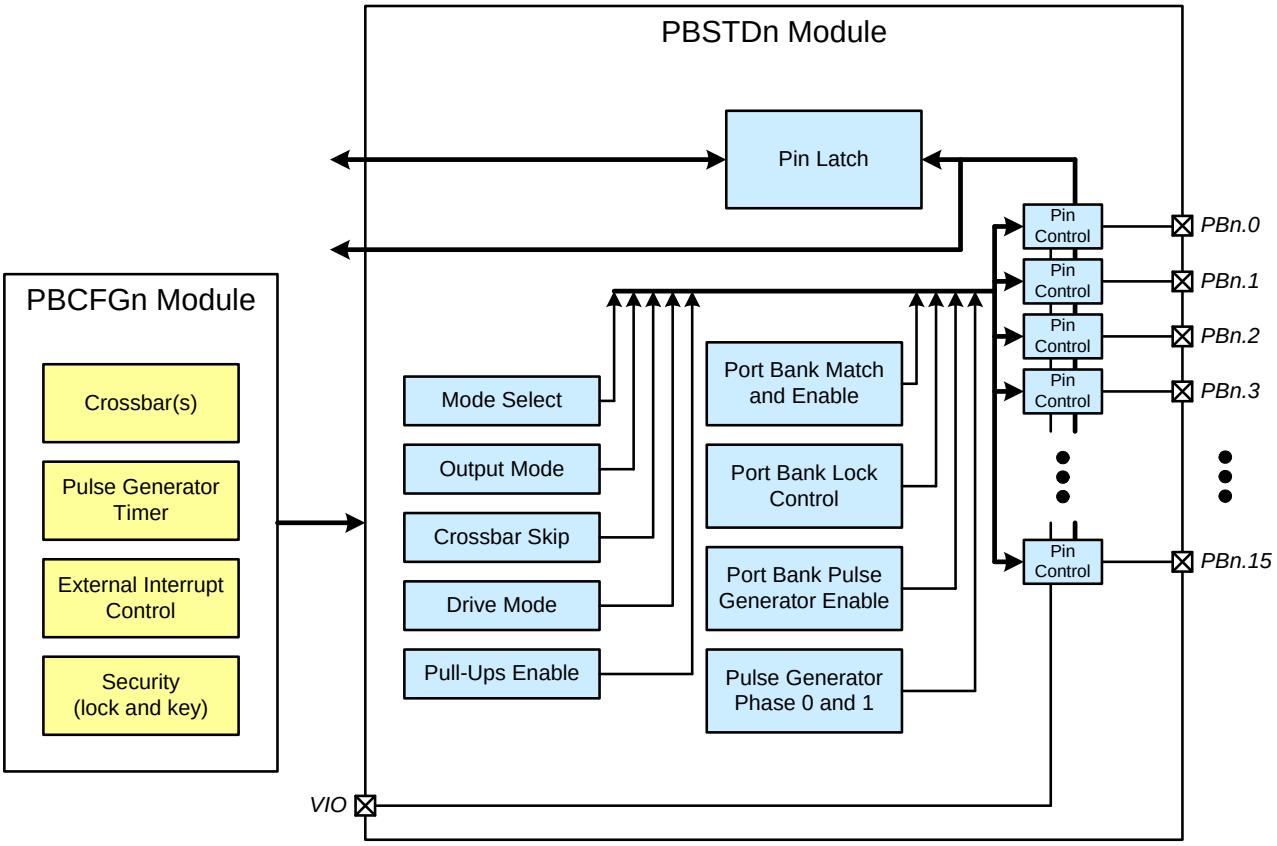


Figure 8.5. PBSTD Block Diagram

8.4. Standard Modes of Operation

Each Port Bank pin can be configured by firmware for analog I/O or digital I/O using the PBMDSEL register. On reset, all Port Bank cells default to a digital high impedance state with weak pull-ups enabled.

8.4.1. Port Bank Pins Configured for Analog I/O

Any pins used as a Comparator, SARADC, IVC, CAPSENSE, EXTOSC0, IDAC, or VREF input or output should be configured for analog I/O (PBMDSEL.x = 0). When a pin is configured for analog I/O, its weak pullup and digital receiver are disabled. In most cases, firmware should also disable the digital output drivers. Firmware will always read back a value of 0 from the PBPIN register for port pins configured for analog I/O regardless of the actual voltage on the pin.

Configuring pins as analog I/O saves power and isolates the port pin from digital interference. Port pins configured as digital inputs may still be used by analog peripherals; however, this practice is not recommended and may result in measurement errors.

8.4.2. Port Pins Configured For Digital I/O

Any pins used by digital peripherals (USART, SPI, I2C, etc.), external digital event capture functions, or as GPIO should be configured as digital I/O (PBMDSEL.x = 1). For digital I/O pins, one of two output modes (push-pull or open-drain) must be selected using the PBOUTMD register.

Push-pull outputs (PBOUTMD.x = 1) drive the port pad to the VIO or VSS supply rails based on the output logic value of the port pin. Open-drain outputs have the high side driver disabled; therefore, they only drive the port pad to VSS when the output logic value is 0 and become high impedance (both high and low drivers turned off) when the output logic value is 1.

When a digital I/O cell is placed in the high impedance state, a weak pull-up transistor pulls the port pad to the VIO supply voltage to ensure the digital input is at a defined logic state. Weak pull-ups are disabled when the I/O cell is driven to VSS to minimize power consumption and may be disabled on a port-by-port basis by clearing PBPUEN to 0. The user should ensure that digital I/O are always internally or externally pulled or driven to a valid logic state to prevent extra supply current caused by intermediate values. From the PB register, port pins configured for digital I/O always read back the logic state last written to the latch, regardless of the output logic value of the port pin. The output logic value of the pins (high or low) can be read using the PBPIN register.

8.4.3. Increasing Port I/O Drive Strength

Port Bank output drivers support a high and low drive strength; the default is low drive strength. The drive strength of a pin is configurable using the PBDRV register. See the electrical specifications chapter for the difference in output drive strength between the two modes.

8.4.4. Interfacing Port I/O to 5 V Logic

Port Bank PB3 pins configured for digital, open-drain operation are capable of interfacing to digital logic operating above the supply pin. To provide logic high “output” to external systems above the rail, a pullup resistor is required.

Important Note: In a multi-voltage interface, the external pull-up resistor should be sized to allow a current of at least 150 μA to flow into the port pin when the pin voltage is between V_{IO} + 0.4 V and V_{IO} + 1.0 V. When the pin voltage increases beyond this range, the current flowing into the port pin is minimal.

8.5. Assigning Standard Port Bank Pins to Analog and Digital Functions

Port Bank pins can be assigned to various analog, digital, and external interrupt functions. The port pins assigned to analog functions should be configured for analog I/O and port pins assigned to digital or external interrupt functions should be configured for digital I/O.

8.5.1. Assigning Port Bank Pins to Analog Functions

Port pins selected for analog functions should have their digital drivers disabled (PBOUTMD.x = 0 and PB.x = 1) and their corresponding bit in PBSKIPEN set to 1. This reserves the pin for use by the analog function and does not allow it to be claimed by the crossbars.

8.5.2. Assigning Port Bank Pins to Digital Functions

Any Port Bank pins not assigned to analog functions may be assigned to digital functions or used as GPIO. Most digital functions rely on the crossbars for pin assignment; however, some digital functions bypass the crossbars. Port pins used by these digital functions and any port pins selected for use as GPIO should have their corresponding bit in PBSKIPEN set to 1.

8.5.3. Assigning Port Bank Pins to External Digital Event Capture Functions

External digital event capture functions can be used to trigger an interrupt or wake the device from a low power mode when a transition occurs on a digital I/O pin. The digital event capture functions do not require dedicated pins and will function on both GPIO pins (PBSKIPEN.x = 1) and pins in use by the Crossbars (PBSKIPEN.x = 0). External digital event capture functions cannot be used on pins configured for analog I/O.

8.6. Standard Port Match and Capacitive Sensing (CAPSENSE0) Activity Monitoring

When MATMD is 0 in the CONTROL1 register, Port Match functionality allows system events to be triggered by a logic value change on PB0-PB3. A port match event occurs if the logic levels of any of the selected input pins match the firmware controlled value in the PM register. This allows firmware to be notified if a certain change occurs on PB0-PB3 input pins regardless of the crossbar settings.

The PMEN registers can be used to individually select which Port Bank pins should be compared against the PM registers.

A port match event may be used to generate an interrupt. If multiple pins are used to generate a port match event, the individual pins can be checked inside the Port Match ISR to determine the cause of the current interrupt. In this case, the event that causes the interrupt must be present long enough to enter the ISR and check the current state of the pins.

When MATMD is 1 or 2, the PM and PMEN registers are used for CAPSENSE activity monitoring. The PMEN register sets which bits can cause retries from the Capacitive Sensing (CAPSENSE0) module. In this mode, the PM register is read-only and indicates the current state of the pin (high or low). More information on activity monitoring can be found in the SiM3xxxx reference manual.

8.7. Standard Port Bank Pulse Generator

PB2 on SiM3U1xx/SiM3C1xx devices has an additional Pulse Generator feature. This Pulse Generator provides the capability to toggle the PB2 port from a single 32-bit word write with a preset 5-bit delay between the setting and clearing of the port. This 5-bit delay is implemented as a timer in the PGTIMER field where the actual count time is PGTIMER + 1. This timer is clocked from the APB clock, and the PGDONEF bit asserts when the timer expires.

Writing to the PBPGPHASE register will update the PB2 value with the phase 0 (PBPGPH0) value, start the counter, and clear the PGDONEF bit. When the timer expires, the PB2 register updates with the phase 1 (PBPGPH1) value, and PGDONEF is set. The PBPGEN register selects which pins are controlled by the pulse generator.

For example:

1. Set PB2's mask register PBPGEN to 0x000000FF.
2. Write the 5-bit pulse generator timer value, PGTIMER, to 0x0F for a 16 cycle delay.
3. Write the PBPGPHASE register with 0x000500FF to set PBPGPH0 and PBPGPH1 at the same time.
4. Poll for PGDONEF.

In this example, the Pulse Generator writes PB2[7:0] to 0xFF, waits 16 cycles, and then updates PB2[7:0] to 0x05. PB2[14:8] are unaffected by this operation since the PBPGEN register only enables bits [7:0] for a pulse generation update.

While PGDONEF is zero, the PB2 and PBPGPHASE bits that have been enabled for pulse generation with the PBPGEN register cannot be updated with a register write. In the previous example, any writes to PB2 during the 16 cycles will take according to the restrictions imposed by the PBPGEN register. For example, if PB2[14:8] was written to 0x5A, this write will take effect since PBPGEN[14:8] was set to 0x00. Conversely, if PB2[7:0] was written during this period, it will not take effect since PBPGEN[7:0] was set to 0xFF.

The P2 bits can be locked even if they are used with the Pulse Generator.

8.8. High Drive (PBHD) Features

The High Drive Port Bank module includes the following features:

- Push-pull or open-drain output modes and analog or digital input modes.
- Option for high or low output drive strength.
- Internal pull-up resistors are enabled or disabled for the port.
- Programmable safe state: high, low, or high impedance.
- Programmable drive strength and slew rates.
- Programmable current limiting.
- Powered from a separate supply (VIOHD, which can be up to 6 V) capable of delivering higher voltage and current.
- Internal VIOHD divider to provide other circuits with VIOHD/4.
- Supports various functions, including GPIO, UART1 pins, EPCA0 pins, or Port Mapped Level Shifting.

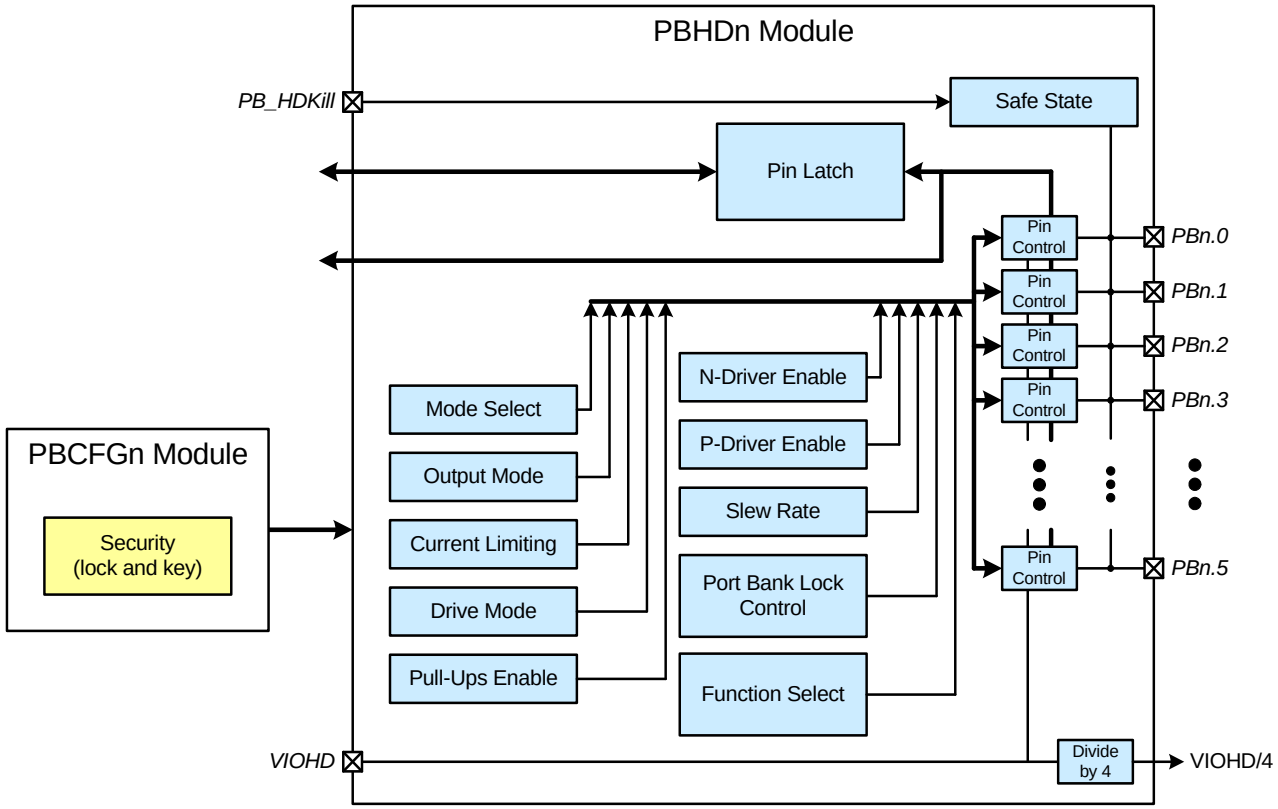


Figure 8.6. PBHD Block Diagram

8.9. High Drive Modes of Operation

In addition to the analog, digital, and drive strength modes described in Section 8.4, the high drive pins have individual N-channel (low) and P-channel (high) drivers that can be enabled. The different output modes are shown in Table 8.5.

Table 8.5. High Drive Output Modes

Output Mode	PBPDEN.x	PBNDEN.x
Tristate	0	0
NMOS Open-Drain	0	1
PMOS Open-Drain	1	0
Push-Pull	1	1

The pins are in tristate output mode after a reset.

The high drive pins also have slew rate control on a port basis using the PBSLEW field.

8.9.1. Safe States

All PBHD pins have a "safe state" which overrides the output of the pin and can be triggered by firmware or hardware. The safe state of the high drive pins can be specified in the PBSS register. Each pin can be low, high, high impedance, or unchanged when entering safe state. Firmware can place the pins in the safe state by setting the SSMDEN bit to 1. If the PB_HDKill signal is routed to pins using Crossbar 1, this input signal sets SSMDEN to 1 when asserted (active low). The PBSSSMD bit allows this PB_HDKill signal to take immediate effect or be deglitched and require two APB clocks to be recognized.

Once set by firmware or the PB_HDKill signal, the SSMDEN bit can only be cleared by writing 0 from firmware or when a device reset occurs. Entering the safe state does not modify the settings in the PB4 registers.

8.9.2. Current Limiting

All PBHD pins have current limiting circuitry built in. This can be useful to prevent short circuits from damaging external components or causing undesired system operation. Separate current limits can be set for the high and low-side drivers to limit the source and sink current respectively, to one of 16 different levels. These limits are configured using the PILIMIT and NILIMIT fields in the PBILIMIT register, and apply to all of the PBHD pins. Each pin can then have current limiting enabled or disabled individually, using the PBILEN field.

8.10. High Drive Configuration Procedure

The High Drive Port Bank pins are locked by default. Before configuring the High Drive Port Bank pins, all High Drive Port Bank pins must be unlocked using the following procedure:

1. Unlock the High Drive Port Bank pins by writing 0xA5 followed by 0xF1 to the PBCFG.KEY bitfield
2. Unlock all bits in the High Drive Port Bank by writing 0x00 to PBHD4.PBLOCK bitfield. Unless all bits in the High Drive Port Bank are unlocked, the bank-wide configuration bits (e.g., PBBIASEN, PBDRVEN, PBLVMD) cannot be modified.

After unlocking the High Drive Port Bank pins, firmware should use the following procedure to configure the high drive pins:

1. Set the safe state for each pin using PBSS.
2. Set PBBIASEN in PBDRV to 1 to enable current biasing.
3. In a separate write to PBDRV, set the PBLVMD bit according to the supply level. Low power mode (PBLVMD = 1) must be used if VIOHD is less than 2.7 V and is recommended if VIOHD is less than 3.0 V. PBLVMD must be cleared to 0 if VIOHD is greater than 3.6 V.
4. In a third write to PBDRV, set PBDRVEN to 1 to release the pins.
5. In another separate write, configure the appropriate drive strength for each pin and the global port controls for pull-ups (PBPUEN), slew rate (PBSLEW), and VIOHD tracking (PBVTRKEN) in the PBDRV register.
6. Enable current limiting per pin as needed by setting (PBILEN.x), setting the drive strength to low drive (PBDRV = 0), and setting the N-channel (NILIMIT) and P-channel (PILIMIT) modes. These N-channel and P-channel modes are global to all pins that have current limiting enabled.
7. Configure the remaining control registers PBDEN, PBMDSEL, and PBFSEL.

8.11. High Drive Function Selection

The port crossbars do not route to the high drive port bank pins, but certain hardware peripherals can be mapped directly to these pins. The pin mode select (PBMDSEL), output driver enable (PBDEN), drive (PBDRV), current limiting (PBILIMIT), and safe state (PBSS) settings can be used to modify the pin behavior in all of these modes. The auxiliary functions are as follows:

Table 8.6. High Drive Function Selections

Pin (PB4.x)	PBxSEL Value in PBFSEL				
	0	1	2	3	4
PB4.0	GPIO	PB2.0 PMLS	EPCA0_HD_CEX0		
PB4.1	GPIO	PB2.1 PMLS	EPCA0_HD_CEX1		
PB4.2	GPIO	PB2.2 PMLS	EPCA0_HD_CEX2	UART1_TX	
PB4.3	GPIO	PB2.3 PMLS	EPCA0_HD_CEX3	UART1_RX	
PB4.4	GPIO	PB2.4 PMLS	EPCA0_HD_CEX4	UART1_RTS	
PB4.5	GPIO	PB2.5 PMLS	EPCA0_HD_CEX5	UART1_CTS	LPTIMER0_OUT

8.11.1. GPIO

In this mode, a high drive pin behaves as a standard GPIO using the VIOHD and VSSHD logic levels.

In digital mode, the pin will assume the state of the port latch value in the PB register. If the N-Channel drivers are enabled, low values will be driven. If the P-Channel drivers are enabled for the pin, high values will be driven. The current logic level of the pin can be read using the PBPIN register. The analog or digital mode of the pin can be controlled using PBMDSEL. The drive modes and current limiting can be controlled using PBDRV and PBILIMIT. The safe state of the pin is set using PBSS.

8.11.2. Port Mapped Level Shift (PMLS)

Each PB4 high drive pin can serve as a buffered or level-shifted version of the corresponding PB2 pins. In this mode, the logic level seen on the PB2.x pin will be driven out on the corresponding PB4.x pin.

The primary purpose of the PMLS feature is to route an input on a PB2 pin as an output to the corresponding PB4 pin. It's also possible to use crossbar 1 to output a signal on a PB2 pin and use the PMLS feature to map it to a PB4 pin. This crossbar mapping feature is only available with crossbar outputs.

8.11.3. EPCA

When set to an EPCA0 output, the high drive pin is controlled by the EPCA0 module. The EPCA0 channel that controls the EPCA0_HD_CEXx output is set in the EPCA0 module with the HDOSEL field.

8.11.4. UART1

The high drive pins in UART1 mode serve as inputs or outputs for the UART1 module. The flow control signals RTS and CTS must be also be enabled in the UART1 module if they are selected as PBHD functions.

8.11.5. Low Power Timer

When set to the Low Power Timer output, the pin will toggle as defined in the LPTIMER0 register settings.

8.11.6. VIOHD Divider

The PBVTRKEN bit in the PBDRV register enables a divide-by-4 circuit on the VIOHD power supply. The VIOHD/4 signal can be used as an input to the SARDADCs or Comparator blocks for easy measurement of the VIOHD supply.

8.12. Port Bank Security

PB2, PB3, PB4, and the control registers have one shared lock which is unlocked by a series of writes to the KEY register. The Port Bank lock mechanism is split into two types: control registers (controlled by the LOCK bit) and Port Bank data registers (controlled by the PBLOCK register for that port bank). Both the control registers and Port Bank data registers for PB2 and PB3 are unlocked after a reset. The Port Bank data registers for PB4 and the lock state machine itself is in a locked state after a reset.

If any Port Bank pins or control registers are locked (LOCK or any bit in PBLOCK set to 1), the sequence of 0xA5 followed by 0xF1 must be written to the KEY register. Once unlocked, writing to a lockable register or writing an invalid value to KEY will relock the registers. All of the registers can be read at any time, regardless of the lock state. Additionally, writes to non-lockable registers (like CONTROL0) do not affect the state of the lock. The KEY register can be read at any time to return the status of the lock.

8.12.1. Control Registers

Control and Configuration registers are locked if the LOCK bit is set and the lock is in a locked state. This lock ensures the fields that control the peripheral output mapping cannot change inadvertently. Setting LOCK will lock CONTROL1, XBAR0L, XBAR0H, XBAR1, and all PBSKIPEN registers.

8.12.2. Port Bank Data Registers

PB2, PB3, and PB4 have lock mask registers that allow individual pins to be locked. Each bit in the PBLOCK register controls whether the respective pin fields in the other pin configuration registers (PBMDSEL, PBOUTMD, etc.) are locked for write access. Common control fields that apply to the entire port bank, like weak pull-ups for PB2 and PB3 and slew and weak pull-ups for PB4, are locked if any single lock mask bit is set for that bank.

The lock mask registers require the state machine to be unlocked for write access. After a reset, firmware must first unlock the Port Bank lock by writing the correct sequence to KEY before writing to the PBLOCK registers.

8.13. Ports and Power Mode 9

When entering PM9, the High Drive pins must be put into a low power mode. To do this:

1. Clear PBBIASEN to 0.
2. Clear PBILEN to 0.

PBLVMD should remain set if VIOHD is less than 2.7 V and is also recommended to remain set if VIOHD is less than 3.0 V. PBLVMD must be cleared to 0 if VIOHD will be greater than 3.6 V while the device is in Power Mode 9.

When the device enters PM9, the pins retain their state but are no longer powered. In addition, the settings are removed from the port control registers. Firmware must reconfigure the port registers (including safe states) after exiting PM9 prior to re-enabling the ports by writing a 0 to PINLPEN in the PMU.

8.14. Debugging Interfaces

After a reset, SiM3U1xx/SiM3C1xx devices are configured with JTAG enabled to allow external JTAG modules to connect. Firmware must disable this if not needed, freeing the JTAG pins for use with other functions. If the core is configured for Serial Wire (SW) mode and not JTAG, then the Serial Wire Viewer is enabled to come out of the TDO pin and the TRST and TDI pins are available for other Crossbar or GPIO functions. The Serial Wire Viewer (SWV) provides a single pin to send out TPIU messages.

Enabling the JTAG or ETM interfaces overrides all other Crossbar and GPIO functionality, so these pins should be skipped in the PBSKIPEN registers, and all writes to these bits in PB will be ignored. ETM must also be enabled in the core.

Table 8.7. Debug Interface Pin Information

Debug Interface	Debug Signal Name	Conditions	SiM3U1x7/C1x7 Pin Name	SiM3U1x6/C1x6 Pin Name	SiM3U1x4/C1x4 Pin Name
JTAG	TRST	JTAGEN = 1	PB1.2		
	TCK		SWCLK/TCK	SWCLK/TCK	
	TDO		PB1.3	PB0.14	
	TDI		PB1.4	PB0.15	
	TMS		SWDIO/TMS	SWDIO/TMS	
ETM	ETM0	ETMEN = 1	PB1.5		
	ETM1		PB1.6		
	ETM2		PB1.7		
	ETM3		PB1.8		
	TRACECLK		PB1.9		
Serial Wire Debug	SWCLK	Clock sequence on SWCLK	SWCLK/TCK	SWCLK	SWCLK
	SWDIO		SWDIO/TMS	SWDIO	SWDIO
Serial Wire Viewer	SWV	JTAGEN = 1, ARM debugger in SW mode	PB1.3	PB0.14	

8.15. External Memory Interface (EMIF)

8.15.1. EMIF Features

The External Memory Interface (EMIF0) allows external parallel asynchronous devices, like SRAMs and LCD controllers, to appear as part of the system memory map. The EMIF0 module includes the following features:

- Provides a memory mapped view of multiple external devices.
- Support for byte, half-word and word accesses regardless of external device data-width.
- Error indicator for certain invalid transfers.
- Minimum external timing allows for 3 clocks per write or 4 clocks per read.
- Output bus can be shared between non-muxed and muxed devices.
- Available extended address output allows for up to 24-bit address with 8-bit parallel devices.
- Support for 8-bit and 16-bit (muxed-mode only) devices with up to two chip-select signals.
- Support for internally muxed devices with dynamic address shifting.
- Fully programmable control signal waveforms.

The bits controlling the additional EMIF0 pins, including an extra chip select (CS1) and byte enable ($\overline{BE0}$), and the EMIF enable are located in the Port Bank CONTROL1 register. All other settings for the EMIF are configured with the EMIF peripheral registers discussed in the SiM3xxxx reference manual.

When the additional EMIF bits are selected, the resulting number of pins in a muxed configuration is EMIFWIDTH + 8. The EMIFWIDTH field should not be used in non-muxed mode.

Table 8.8. EMIF Muxed Configuration Pin Information

EMIF Signal	Description	Conditions	SiM3U1x7/C1x7 Pin Name	SiM3U1x6/C1x6 Pin Name
AD0m	Address/Data Bit 0		PB3.2	PB3.2
AD1m	Address/Data Bit 1		PB3.1	PB3.1
AD2m	Address/Data Bit 2		PB3.0	PB3.0
AD3m	Address/Data Bit 3		PB2.14	PB2.3
AD4m	Address/Data Bit 4		PB2.13	PB2.2
AD5m	Address/Data Bit 5		PB2.12	PB2.1
AD6m	Address/Data Bit 6		PB2.11	PB2.0
AD7m	Address/Data Bit 7		PB2.10	PB1.15
AD8m	Address/Data Bit 8	EMIFWIDTH = 1	PB2.9	PB1.14
AD9m	Address/Data Bit 9	EMIFWIDTH = 2	PB2.8	PB1.13
AD10m	Address/Data Bit 10	EMIFWIDTH = 3	PB2.7	PB1.12
AD11m	Address/Data Bit 11	EMIFWIDTH = 4	PB2.6	PB1.11
AD12m	Address/Data Bit 12	EMIFWIDTH = 5	PB2.5	PB1.10
AD13m	Address/Data Bit 13	EMIFWIDTH = 6	PB2.4	PB1.9
AD14m	Address/Data Bit 14	EMIFWIDTH = 7	PB2.3	PB1.8
AD15m	Address/Data Bit 15	EMIFWIDTH = 8	PB2.2	PB1.7
A16m	Address Bit 16	EMIFWIDTH = 9	PB2.1	
A17m	Address Bit 17	EMIFWIDTH = 10	PB2.0	

Table 8.8. EMIF Muxed Configuration Pin Information (Continued)

EMIF Signal	Description	Conditions	SiM3U1x7/C1x7 Pin Name	SiM3U1x6/C1x6 Pin Name
A18m	Address Bit 18	EMIFWIDTH = 11	PB1.15	
A19m	Address Bit 19	EMIFWIDTH = 12	PB1.14	
A20m	Address Bit 20	EMIFWIDTH = 13	PB1.13	
A21m	Address Bit 21	EMIFWIDTH = 14	PB1.12	
A22m	Address Bit 22	EMIFWIDTH = 15	PB1.11	
A23m	Address Bit 23	EMIFWIDTH = 16	PB1.10	
$\overline{WR}$	Active-Low Write		PB3.3	PB3.3
$\overline{OE}$	Active-Low Output Enable		PB3.4	PB3.4
ALEm	Address Latch Enable		PB3.5	PB3.5
CS0	Chip Select 0		PB3.6	PB3.6
$\overline{BE1}$	Active-Low Byte Enable 1		PB3.7	PB3.7
CS1	Chip Select 1	EMIFCS1EN = 1	PB3.8	PB3.8
$\overline{BE0}$	Active-Low Byte Enable 0	EMIFBE0BEN = 1	PB3.9	PB3.9

Table 8.9. EMIF Non-Muxed Configuration Pin Information

EMIF Signal	Description	Conditions	SiM3U1x7/C1x7 Pin Name	SiM3U1x6/C1x6 Pin Name
A0	Address Bit 0	EMIFWIDTH = 1	PB2.9	PB1.14
A1	Address Bit 1	EMIFWIDTH = 2	PB2.8	PB1.13
A2	Address Bit 2	EMIFWIDTH = 3	PB2.7	PB1.12
A3	Address Bit 3	EMIFWIDTH = 4	PB2.6	PB1.11
A4	Address Bit 4	EMIFWIDTH = 5	PB2.5	PB1.10
A5	Address Bit 5	EMIFWIDTH = 6	PB2.4	PB1.9
A6	Address Bit 6	EMIFWIDTH = 7	PB2.3	PB1.8
A7	Address Bit 7	EMIFWIDTH = 8	PB2.2	PB1.7
A8	Address Bit 8	EMIFWIDTH = 9	PB2.1	
A9	Address Bit 9	EMIFWIDTH = 10	PB2.0	
A10	Address Bit 10	EMIFWIDTH = 11	PB1.15	
A11	Address Bit 11	EMIFWIDTH = 12	PB1.14	
A12	Address Bit 12	EMIFWIDTH = 13	PB1.13	
A13	Address Bit 13	EMIFWIDTH = 14	PB1.12	
A14	Address Bit 14	EMIFWIDTH = 15	PB1.11	
A15	Address Bit 15	EMIFWIDTH = 16	PB1.10	
D0	Data Bit 0		PB3.2	PB3.2
D1	Data Bit 1		PB3.1	PB3.1
D2	Data Bit 2		PB3.0	PB3.0
D3	Data Bit 3		PB2.14	PB2.3
D4	Data Bit 4		PB2.13	PB2.2
D5	Data Bit 5		PB2.12	PB2.1
D6	Data Bit 6		PB2.11	PB2.0
D7	Data Bit 7		PB2.10	PB1.15
$\overline{WR}$	Active-Low Write		PB3.3	PB3.3
$\overline{OE}$	Active-Low Output Enable		PB3.4	PB3.4
	unused by EMIF		PB3.5	PB3.5
CS0	Chip Select 0		PB3.6	PB3.6
$\overline{BE1}$	Active-Low Byte Enable 1		PB3.7	PB3.7
CS1	Chip Select 1	EMIFCS1EN = 1	PB3.8	PB3.8
$\overline{BE0}$	Active-Low Byte Enable 0	EMIFBE0BEN = 1	PB3.9	PB3.9

The EMIF is not supported on SiM3U1x4 devices.

The EMIF pins should be skipped in the corresponding PBSKIPEN register and set to push-pull or open-drain using the PBOUTMD registers, as appropriate. Any pins not used by the EMIF as a result of the EMIFWIDTH, EMIFCS1EN, and EMIFBE0BEN bit settings can be used for crossbar or GPIO functions. When the EMIF is not actively transferring data, the EMIF pin state is controlled by the PB registers, providing a mechanism to place the pins at a stable or desired value when the EMIF is idle.

8.16. External Interrupts

8.16.1. External Interrupt Features

The External Interrupts (INT0/INT1) on the SiM3U1xx/SiM3C1xx devices have the following features:

- Level (high or low) or edge (rising or falling) detection.
- Can select one of up to 16 inputs to monitor.
- Separate from the crossbars so can be used in conjunction with other peripherals on the same pins.

The INT0 and INT1 external interrupt sources are configurable as active high or low, edge or level sensitive. The INT0POL (INT0 Polarity) and INT1POL (INT1 Polarity) bits in the CONTROL0 register select active high or active low; the INT0MD and INT1MD bits in the same register select level or edge sensitive. The table below lists the possible configurations.

Table 8.10. External Interrupt Configuration

INTnMD	INTnPOL	INTn Configuration
1	0	Active low, edge sensitive (falling)
1	1	Active high, edge sensitive (rising)
0	0	Active low, level sensitive (low level)
0	1	Active high, level sensitive (high level)

INT0 and INT1 are assigned to Port Bank pins as defined in the CONTROL0 register. These pin assignments are independent of any crossbar assignments. The External Interrupts will monitor their assigned Port Bank pins without disturbing the peripheral that was assigned to the pin via the crossbars. To assign a Port Bank pin only to INT0 or INT1, configure the crossbar to skip the selected pins by setting the associated bit in PBSKIPEN register. The pins available for use as external interrupt sources vary by package, and are defined in Table 8.11.

Table 8.11. External Interrupt Triggers

INT0 and INT1 Trigger	INT0 and INT1 Trigger Description	SiM3U1x7/C1x7 Pin Name	SiM3U1x6/C1x6 Pin Name	SiM3U1x4/C1x4 Pin Name
INT0.0 / INT1.0	External Interrupt Trigger	PB2.0	PB2.0	PB3.0
INT0.1 / INT1.1	External Interrupt Trigger	PB2.1	PB2.1	PB3.1
INT0.2 / INT1.2	External Interrupt Trigger	PB2.2	PB2.2	PB3.2
INT0.3 / INT1.3	External Interrupt Trigger	PB2.3	PB2.3	PB3.3
INT0.4 / INT1.4	External Interrupt Trigger	PB2.4	PB3.3	Reserved

Table 8.11. External Interrupt Triggers

INT0 and INT1 Trigger	INT0 and INT1 Trigger Description	SiM3U1x7/C1x7 Pin Name	SiM3U1x6/C1x6 Pin Name	SiM3U1x4/C1x4 Pin Name
INT0.5 / INT1.5	External Interrupt Trigger	PB2.5	PB3.4	Reserved
INT0.6 / INT1.6	External Interrupt Trigger	PB2.6	PB3.5	Reserved
INT0.7 / INT1.7	External Interrupt Trigger	PB2.7	PB3.6	Reserved
INT0.8 / INT1.8	External Interrupt Trigger	PB3.3	PB3.7	Reserved
INT0.9 / INT1.9	External Interrupt Trigger	PB3.4	PB3.8	Reserved
INT0.10 / INT1.10	External Interrupt Trigger	PB3.5	PB3.9	Reserved
INT0.11 / INT1.11	External Interrupt Trigger	PB3.6	Reserved	Reserved
INT0.12 / INT1.12	External Interrupt Trigger	PB3.7	Reserved	Reserved
INT0.13 / INT1.13	External Interrupt Trigger	PB3.8	Reserved	Reserved
INT0.14 / INT1.14	External Interrupt Trigger	PB3.9	Reserved	Reserved
INT0.15 / INT1.15	External Interrupt Trigger	PB3.10	Reserved	Reserved

If an INT0 or INT1 external interrupt is configured as edge-sensitive, the corresponding interrupt-pending flag is set once per edge. When configured as level sensitive, the interrupt-pending flag remains logic 1 while the input is active as defined by the corresponding polarity bit (INT0POL or INT1POL); the flag remains cleared while the input is inactive. The external interrupt source must hold the input active until the interrupt request is recognized. It must then deactivate the interrupt request before execution of the ISR completes or another interrupt request will be generated.

8.17. PBCFG0 Registers

This section contains the detailed register descriptions for PBCFG0 registers.

Register 8.1. PBCFG0_CONTROL0: Global Port Control 0

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	PGDONEF	Reserved		PGTIMER					Reserved							
Type	R	R		RW					R							
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	INT1EN	INT1MD		INT1POL	INT1SEL				INT0EN	INT0MD		INT0POL	INT0SEL			
Type	RW	RW		RW	RW				RW	RW		RW	RW			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

PBCFG0_CONTROL0 = 0x4002_A000

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 8.12. PBCFG0_CONTROL0 Register Bit Descriptions

Bit	Name	Function
31	PGDONEF	**Pulse Generator Timer Done Flag.** This bit is cleared by hardware when firmware writes to the PBPGPHASE register. This bit is set when the Pulse Generator timer expires. While PGDONEF is 0, the PB and PBPGPHASE bits that have been enabled for pulse generation with the PBPGMSK cannot be updated with a register write.
30:29	Reserved	Must write reset value.
28:24	PGTIMER	**Pulse Generator Timer.** Count down timer value for supported toggle ports.
23:16	Reserved	Must write reset value.
15	INT1EN	**External Interrupt 1 Enable.** 0: Disable external interrupt 1. 1: Enable external interrupt 1.
14:13	INT1MD	**External Interrupt 1 Mode.** 00: Interrupt based on level sensitivity. 01: Interrupt based on edge sensitivity. 10-11: Reserved.

Table 8.12. PBCFG0_CONTROL0 Register Bit Descriptions

Bit	Name	Function
12	INT1POL	**External Interrupt 1 Polarity.** 0: A low value or falling edge on the selected pin will cause interrupt. 1: A high value or rising edge on the selected pin will cause interrupt.
11:8	INT1SEL	**External Interrupt 1 Pin Selection.** Selects the external pin to use as external interrupt 1. (INT1SEL = 0 selects INT1.0).
7	INT0EN	**External Interrupt 0 Enable.** 0: Disable external interrupt 0. 1: Enable external interrupt 0.
6:5	INT0MD	**External Interrupt 0 Mode.** 00: Interrupt based on level sensitivity. 01: Interrupt based on edge sensitivity. 10-11: Reserved.
4	INT0POL	**External Interrupt 0 Polarity.** 0: A low value or falling edge on the selected pin will cause interrupt. 1: A high value or rising edge on the selected pin will cause interrupt.
3:0	INT0SEL	**External Interrupt 0 Pin Selection.** Selects the external pin to use as external interrupt 0 (INT0SEL = 0 selects INT0.0).

Register 8.2. PBCFG0_CONTROL1: Global Port Control 1

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	LOCK	Reserved							EVREGRMD	Reserved					MATMD	
Type	RW	R							RW	R					RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	EMIFWIDTH						EMIFEN	EMIFCS1EN	EMIFBE0BEN	Reserved					ETMEN	JTAGEN
Type	RW						RW	RW	RW	R	RW				RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Register ALL Access Address

PBCFG0_CONTROL1 = 0x4002_A010

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 8.13. PBCFG0_CONTROL1 Register Bit Descriptions

Bit	Name	Function
31	LOCK	**Port Bank Configuration Lock.** This bit controls the lock for the port bank configuration and control registers. 0: Port Bank Configuration and Control registers are unlocked. 1: The following registers are locked from write access: CONTROL1, XBAR0L, XBAR0H, XBAR1, and all PBSKIP registers.
30:24	Reserved	Must write reset value.
23	EVREGRMD	**External Regulator Reset Mode.** 0: The pins used by the external regulator will default to digital inputs with weak pull-up enabled on any reset. 1: The pins used by the external regulator will default to digital inputs with weak pull-up enabled only on Power-On Reset. Their configured mode will be preserved through all other resets.
22:18	Reserved	Must write reset value.

Table 8.13. PBCFG0_CONTROL1 Register Bit Descriptions

Bit	Name	Function
17:16	MATMD	**Match Mode.** Determines how the port match registers are used. 00: Port Match registers used to provide interrupt / wake sources. 01: Port Match registers used to monitor output pin activity for Capacitive Sensing measurements. 10: Port Match registers used to monitor input pin activity for Capacitive Sensing measurements. 11: Reserved.
15:10	EMIFWIDTH	**EMIF Width.** This field is the number of additional pins allocated to EMIF address. The number of total address lines will be 8 + EMIFWIDTH.
9	EMIFEN	**EMIF Enable.** 0: Disable the EMIF pins. 1: EMIF is enabled and pinned out.
8	EMIFCS1EN	**EMIF CS1 Pin Enable.** 0: Disable the EMIF CS1 pin. 1: Enable the CS1 pin if EMIFEN is also set to 1.
7	EMIFBE0BEN	**EMIF $\overline{\text{BE0}}$ Pin Enable.** 0: Disable the EMIF $\overline{\text{BE0}}$ pin. 1: Enable the $\overline{\text{BE0}}$ pin if EMIFEN is also set to 1.
6:2	Reserved	Must write reset value.
1	ETMEN	**ETM Enable.** 0: ETM not pinned out. 1: ETM is enabled and pinned out.
0	JTAGEN	**JTAG Enable.** 0: JTAG functionality is not pinned out. 1: JTAG functionality is pinned out.

Register 8.3. PBCFG0_XBAR0L: Crossbar 0 Control (Low)

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved	TMR1EXEN	TMR1CTEN	TMR0EXEN	TMR0CTEN	CMP1AEN	CMP1SEN	CMP0AEN	CMP0SEN	I2C0EN	I2S0TXEN	ECI1EN	ECI0EN	EECI0EN	PCA1EN	
Type	R	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	PCA0EN		Reserved			EPCA0EN			USART1CEN	USART1FCEN	USART1EN	SPI0NSSEN	SPI0EN	USART0CEN	USART0FCEN	USART0EN
Type	RW		R			RW			RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

PBCFG0_XBAR0L = 0x4002_A020

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 8.14. PBCFG0_XBAR0L Register Bit Descriptions

Bit	Name	Function
31	Reserved	Must write reset value.
30	TMR1EXEN	**TIMER1 T1EX Enable.** 0: Disable TIMER1 EX on Crossbar 0. 1: Enable TIMER1 EX on Crossbar 0.
29	TMR1CTEN	**TIMER1 T1CT Enable.** 0: Disable TIMER1 CT on Crossbar 0. 1: Enable TIMER1 CT on Crossbar 0.
28	TMR0EXEN	**TIMER0 T0EX Enable.** 0: Disable TIMER0 EX on Crossbar 0. 1: Enable TIMER0 EX on Crossbar 0.
27	TMR0CTEN	**TIMER0 T0CT Enable.** 0: Disable TIMER0 CT on Crossbar 0. 1: Enable TIMER0 CT on Crossbar 0.
26	CMP1AEN	**Comparator 1 Asynchronous Output (CMP1A) Enable.** 0: Disable Comparator 1 Asynchronous Output (CMP1A) on Crossbar 0. 1: Enable Comparator 1 Asynchronous Output (CMP1A) on Crossbar 0.

Table 8.14. PBCFG0_XBAR0L Register Bit Descriptions

Bit	Name	Function
25	CMP1SEN	**Comparator 1 Synchronous Output (CMP1S) Enable.** 0: Disable Comparator 1 Synchronous Output (CMP1S) on Crossbar 0. 1: Enable Comparator 1 Synchronous Output (CMP1S) on Crossbar 0.
24	CMP0AEN	**Comparator 0 Asynchronous Output (CMP0A) Enable.** 0: Disable Comparator 0 Asynchronous Output (CMP0A) on Crossbar 0. 1: Enable Comparator 0 Asynchronous Output (CMP0A) on Crossbar 0.
23	CMP0SEN	**Comparator 0 Synchronous Output (CMP0S) Enable.** 0: Disable Comparator 0 Synchronous Output (CMP0S) on Crossbar 0. 1: Enable Comparator 0 Synchronous Output (CMP0S) on Crossbar 0.
22	I2C0EN	**I2C0 Enable.** 0: Disable I2C0 SDA and SCL on Crossbar 0. 1: Enable I2C0 SDA and SCL on Crossbar 0.
21	I2S0TXEN	**I2S0 TX Enable.** 0: Disable I2S0 TX on Crossbar 0. 1: Enable I2S0 TX on Crossbar 0.
20	ECI1EN	**PCA1 ECI Enable.** 0: Disable PCA1 ECI on Crossbar 0. 1: Enable PCA1 ECI on Crossbar 0.
19	ECI0EN	**PCA0 ECI Enable.** 0: Disable PCA0 ECI on Crossbar 0. 1: Enable PCA0 ECI on Crossbar 0.
18	EECI0EN	**EPCA0 ECI Enable.** 0: Disable EPCA0 ECI on Crossbar 0. 1: Enable EPCA0 ECI on Crossbar 0.
17:16	PCA1EN	**PCA1 Channel Enable.** 00: Disable all PCA1 channels on Crossbar 0. 01: Enable PCA1 CEX0 on Crossbar 0. 10: Reserved. 11: Enable PCA1 CEX0 and CEX1 on Crossbar 0.
15:14	PCA0EN	**PCA0 Channel Enable.** 00: Disable all PCA0 channels on Crossbar 0. 01: Enable PCA0 CEX0 on Crossbar 0. 10: Reserved. 11: Enable PCA0 CEX0 and CEX1 on Crossbar 0.
13:11	Reserved	Must write reset value.

Table 8.14. PBCFG0_XBAR0L Register Bit Descriptions

Bit	Name	Function
10:8	EPCA0EN	**EPCA0 Channel Enable.** 000: Disable all EPCA0 channels on Crossbar 0. 001: Enable EPCA0 STD_CEX0 on Crossbar 0. 010: Enable EPCA0 STD_CEX0 and STD_CEX1 on Crossbar 0. 011: Enable EPCA0 STD_CEX0, STD_CEX1, and STD_CEX2 on Crossbar 0. 100: Enable EPCA0 STD_CEX0, STD_CEX1, STD_CEX2, and STD_CEX3 on Crossbar 0. 101: Enable EPCA0 STD_CEX0, STD_CEX1, STD_CEX2, STD_CEX3, and STD_CEX4 on Crossbar 0. 110: Enable EPCA0 STD_CEX0, STD_CEX1, STD_CEX2, STD_CEX3, STD_CEX4, and STD_CEX5 on Crossbar 0. 111: Reserved.
7	USART1CEN	**USART1 Clock Signal Enable.** 0: Disable USART1 clock on Crossbar 0. 1: Enable USART1 clock on Crossbar 0.
6	USART1FCEN	**USART1 Flow Control Enable.** 0: Disable USART1 flow control on Crossbar 0. 1: Enable USART1 flow control on Crossbar 0.
5	USART1EN	**USART1 Enable.** 0: Disable USART1 RX and TX on Crossbar 0. 1: Enable USART1 RX and TX on Crossbar 0.
4	SPI0NSSEN	**SPI0 NSS Pin Enable.** 0: Disable SPI0 NSS on Crossbar 0. 1: Enable SPI0 NSS on Crossbar 0.
3	SPI0EN	**SPI0 Enable.** 0: Disable SPI0 SCK, MISO, and MOSI on Crossbar 0. 1: Enable SPI0 SCK, MISO, and MOSI on Crossbar 0.
2	USART0CEN	**USART0 Clock Signal Enable.** 0: Disable USART0 clock on Crossbar 0. 1: Enable USART0 clock on Crossbar 0.
1	USART0FCEN	**USART0 Flow Control Enable.** 0: Disable USART0 flow control on Crossbar 0. 1: Enable USART0 flow control on Crossbar 0.
0	USART0EN	**USART0 Enable.** 0: Disable USART0 RX and TX on Crossbar 0. 1: Enable USART0 RX and TX on Crossbar 0.

Register 8.4. PBCFG0_XBAR0H: Crossbar 0 Control (High)

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	XBAR0EN	Reserved														
Type	RW	R														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved								AHBEN	SPI2NSSEN	SPI2EN	SPI1NSSEN	SPI1EN	UART1EN	UART0FCEN	UART0EN
Type	R								RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

PBCFG0_XBAR0H = 0x4002_A030

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 8.15. PBCFG0_XBAR0H Register Bit Descriptions

Bit	Name	Function
31	XBAR0EN	**Crossbar 0 Enable.** 0: Disable Crossbar 0. 1: Enable Crossbar 0.
30:8	Reserved	Must write reset value.
7	AHBEN	**AHB Clock Output Enable.** 0: Disable the AHB Clock / 16 output on Crossbar 0. 1: Enable the AHB Clock / 16 output on Crossbar 0.
6	SPI2NSSEN	**SPI2 NSS Pin Enable.** 0: Disable SPI2 NSS on Crossbar 0. 1: Enable SPI2 NSS on Crossbar 0.
5	SPI2EN	**SPI2 Enable.** 0: Disable SPI2 SCK, MISO, and MOSI on Crossbar 0. 1: Enable SPI2 SCK, MISO, and MOSI on Crossbar 0.
4	SPI1NSSEN	**SPI1 NSS Pin Enable.** 0: Disable SPI1 NSS on Crossbar 0. 1: Enable SPI1 NSS on Crossbar 0.

Table 8.15. PBCFG0_XBAR0H Register Bit Descriptions

Bit	Name	Function
3	SPI1EN	**SPI1 Enable.** 0: Disable SPI1 SCK, MISO, and MOSI on Crossbar 0. 1: Enable SPI1 SCK, MISO, and MOSI on Crossbar 0.
2	UART1EN	**UART1 Enable.** 0: Disable UART1 RX and TX on Crossbar 0. 1: Enable UART1 RX and TX on Crossbar 0.
1	UART0FCEN	**UART0 Flow Control Enable.** 0: Disable UART0 flow control on Crossbar 0. 1: Enable UART0 flow control on Crossbar 0.
0	UART0EN	**UART0 Enable.** 0: Disable UART0 RX and TX on Crossbar 0. 1: Enable UART0 RX and TX on Crossbar 0.

Register 8.5. PBCFG0_XBAR1: Crossbar 1 Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	XBAR1EN	Reserved									KILLHDEN	I2C1EN	LPT0OEN	Reserved	I2S0RXEN	UART1EN
Type	RW	R									RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	I2C0EN	I2S0TXEN	UART0FCEN	UART0EN	USART1CEN	USART1FCEN	USART1EN	SPI2NSSEN	SPI2EN	RTC0EN	SPI1NSSEN	SPI1EN	CMP1SEN	CMP0SEN	SSG0EN	
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

PBCFG0_XBAR1 = 0x4002_A040

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 8.16. PBCFG0_XBAR1 Register Bit Descriptions

Bit	Name	Function
31	XBAR1EN	**Crossbar 1 Enable.** 0: Disable Crossbar 1. 1: Enable Crossbar 1.
30:22	Reserved	Must write reset value.
21	KILLHDEN	**High Drive Kill Pin Enable.** 0: Disable the PB High Drive Kill Pin on Crossbar 1. 1: Enable the PB High Drive Kill Pin on Crossbar 1.
20	I2C1EN	**I2C1 Enable.** 0: Disable I2C1 SDA and SCL on Crossbar 1. 1: Enable I2C1 SDA and SCL on Crossbar 1.
19	LPT0OEN	**LPTIMER0 Output Enable.** 0: Disable LPTIMER0 Output on Crossbar 1. 1: Enable LPTIMER0 Output on Crossbar 1.
18	Reserved	Must write reset value.

Table 8.16. PBCFG0_XBAR1 Register Bit Descriptions

Bit	Name	Function
17	I2S0RXEN	**I2S0 RX Enable.** 0: Disable I2S0 RX on Crossbar 1. 1: Enable I2S0 RX on Crossbar 1.
16	UART1EN	**UART1 Enable.** 0: Disable UART1 RX and TX on Crossbar 1. 1: Enable UART1 RX and TX on Crossbar 1.
15	I2C0EN	**I2C0 Enable.** 0: Disable I2C0 SDA and SCL on Crossbar 1. 1: Enable I2C0 SDA and SCL on Crossbar 1.
14	I2S0TXEN	**I2S0 TX Enable.** 0: Disable I2S0 TX on Crossbar 1. 1: Enable I2S0 TX on Crossbar 1.
13	UART0FCEN	**UART0 Flow Control Enable.** 0: Disable UART0 flow control on Crossbar 1. 1: Enable UART0 flow control on Crossbar1.
12	UART0EN	**UART0 Enable.** 0: Disable UART0 RX and TX on Crossbar 1. 1: Enable UART0 RX and TX on Crossbar 1.
11	USART1CEN	**USART1 Clock Signal Enable.** 0: Disable USART1 clock on Crossbar 1. 1: Enable USART1 clock on Crossbar 1.
10	USART1FCEN	**USART1 Flow Control Enable.** 0: Disable USART1 flow control on Crossbar 1. 1: Enable USART1 flow control on Crossbar 1.
9	USART1EN	**USART1 Enable.** 0: Disable USART1 RX and TX on Crossbar 1. 1: Enable USART1 RX and TX on Crossbar 1.
8	SPI2NSSEN	**SPI2 NSS Pin Enable.** 0: Disable SPI2 NSS on Crossbar 1. 1: Enable SPI2 NSS on Crossbar 1.
7	SPI2EN	**SPI2 Enable.** 0: Disable SPI2 SCK, MISO, and MOSI on Crossbar 1. 1: Enable SPI2 SCK, MISO, and MOSI on Crossbar 1.
6	RTC0EN	**RTC0 Output Enable.** 0: Disable RTC0 Output on Crossbar 1. 1: Enable RTC0 Output on Crossbar 1.
5	SPI1NSSEN	**SPI1 NSS Pin Enable.** 0: Disable SPI1 NSS on Crossbar 1. 1: Enable SPI1 NSS on Crossbar 1.

Table 8.16. PBCFG0_XBAR1 Register Bit Descriptions

Bit	Name	Function
4	SPI1EN	**SPI1 Enable.** 0: Disable SPI1 SCK, MISO, and MOSI on Crossbar 1. 1: Enable SPI1 SCK, MISO, and MOSI on Crossbar 1.
3	CMP1SEN	**Comparator 1 Synchronous Output (CMP1S) Enable.** 0: Disable Comparator 1 Synchronous Output (CMP1S) on Crossbar 1. 1: Enable Comparator 1 Synchronous Output (CMP1S) on Crossbar 1.
2	CMP0SEN	**Comparator 0 Synchronous Output (CMP0S) Enable.** 0: Disable Comparator 0 Synchronous Output (CMP0S) on Crossbar 1. 1: Enable Comparator 0 Synchronous Output (CMP0S) on Crossbar 1.
1:0	SSG0EN	**SSG0 Enable.** 00: Disable all SSG0 channels on Crossbar 1. 01: Enable SSG0 EX0 on Crossbar 1. 10: Enable SSG0 EX0 and EX1 on Crossbar 1. 11: Enable SSG0 EX0, EX1, EX2, and EX3 on Crossbar 1.

Register 8.6. PBCFG0_PBKEY: Global Port Key

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved								KEY							
Type	R								RW							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

PBCFG0_PBKEY = 0x4002_A050

Table 8.17. PBCFG0_PBKEY Register Bit Descriptions

Bit	Name	Function
31:8	Reserved	Must write reset value.
7:0	KEY	**Port Bank 2, 3, and 4 Key.** When a port pin on bank 2, 3, or 4 is locked for access, firmware must write the value 0xA5 followed by a write of 0xF1 to this field to unlock the pin. Reading this register returns the current status of lock (0 = Locked with no keys written, 1 = Locked with first key written, 2 = Unlocked). Once unlocked, any write to a port bank 2, 3, or 4 register or writing the incorrect key sequence value will re-lock the interface. Locked bits are determined by the PBLOCK settings for the specific port bank.

8.18. PBCFG0 Register Memory Map

Table 8.18. PBCFG0 Memory Map

Register Name	PBCFG0_XBAR0H	PBCFG0_XBAR0L	PBCFG0_CONTROL1	PBCFG0_CONTROL0
ALL Address	0x4002_A030	0x4002_A020	0x4002_A010	0x4002_A000
Access Methods	ALL | SET | CLR	ALL | SET | CLR	ALL | SET | CLR	ALL | SET | CLR
Bit 31	XBAR0EN	Reserved	LOCK	PGDONEF
Bit 30	Reserved	TMR1EXEN	Reserved	Reserved
Bit 29		TMR1CTEN		
Bit 28		TMR0EXEN		PGTIMER
Bit 27		TMR0CTEN		
Bit 26		CMP1AEN		
Bit 25		CMP1SEN		
Bit 24		CMP0AEN		
Bit 23		CMP0SEN	EVREGRMD	Reserved
Bit 22		I2C0EN	Reserved	
Bit 21		I2S0TXEN		
Bit 20		ECI1EN		
Bit 19		ECI0EN		
Bit 18		EECI0EN		
Bit 17		PCA1EN	MATMD	
Bit 16				
Bit 15		PCA0EN	EMIFWIDTH	INT1EN
Bit 14		Reserved		INT1MD
Bit 13				INT1POL
Bit 12		EPCA0EN		INT1SEL
Bit 11				
Bit 10				
Bit 9			EMIFEN	INT0EN
Bit 8			EMIFCS1EN	
Bit 7	AHBEN	USART1CEN	EMIFBE0BEN	INT0MD
Bit 6	SPI2NSSEN	USART1FCEN	Reserved	
Bit 5	SPI2EN	USART1EN		INT0POL
Bit 4	SPI1NSSEN	SPI0NSSEN		
Bit 3	SPI1EN	SPI0EN		INT0SEL
Bit 2	UART1EN	USART0CEN		
Bit 1	UART0FCEN	USART0FCEN	ETMEN	
Bit 0	UART0EN	USART0EN	JTAGEN	

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

Table 8.18. PBCFG0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBCFG0_XBAR1	0x4002_A040	ALL | SET | CLR	XBAR1EN	Reserved									KILLHDEN	I2C1EN	LPT0OEN	Reserved	I2S0RXEN	UART1EN	I2C0EN	I2S0TXEN	UART0FCEN	UART0EN	USART1CEN	USART1FCEN	USART1EN	SPI2NSSEN	SPI2EN	RTC0EN	SPI1NSSEN	SPI1EN	CMP1SEN	CMP0SEN	SSG0EN	
PBCFG0_PBKEY	0x4002_A050	ALL	Reserved																								KEY							

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

8.19. PBSTD0, PBSTD1, PBSTD2, and PBSTD3 Registers

This section contains the detailed register descriptions for PBSTD0, PBSTD1, PBSTD2 and PBSTD3 registers.

Register 8.7. PBSTDn_PB: Output Latch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	PB															
Type	RW															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Register ALL Access Addresses

PBSTD0_PB = 0x4002_A0A0
PBSTD1_PB = 0x4002_A140
PBSTD2_PB = 0x4002_A1E0
PBSTD3_PB = 0x4002_A320
This register also supports SET access at (ALL+0x4), CLR access at (ALL+0x8) and MSK access at (ALL+0xC)

Table 8.19. PBSTDn_PB Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15:0	PB	**Output Latch.** These bits define the logic level of the port bank output latch. Each bit in this field controls the output latch value for the corresponding port bank pin (bit x controls the latch for pin PBn.x). Digital input pins should be written to 1 and configured for open drain mode. When using this register via the MSK address, the upper 16 bits can be used to mask writes of the lower 16 bits to the corresponding port bank latches.

Notes:

1. On SiM3x1x7 devices, PB0 and PB1 are full ports (PBx.0-PBx.15), PB2 consists of 15 pins (PB2.0-PB2.14), and PB3 consists of 12 pins (PB3.0-PB3.11). On SiM3x1x6 devices, PB0 and PB1 are full ports (PBx.0-PBx.15), PB2 consists of 4 pins (PB2.0-PB2.3), and PB3 consists of 10 pins (PB3.0-PB3.9). On SiM3x1x4 devices, PB0 is a full port (PB0.0-PB0.15), PB1 consists of 4 pins (PB1.0-PB1.3), PB2 is not implemented, and PB3 consists of 4 pins (PB3.0-PB3.3). Any bits in this register controlling unimplemented pins are reserved.

Register 8.8. PBSTDn_PBPIN: Pin Value

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	PBPIN															
Type	R															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Register ALL Access Addresses

PBSTD0_PBPIN = 0x4002_A0B0
PBSTD1_PBPIN = 0x4002_A150
PBSTD2_PBPIN = 0x4002_A1F0
PBSTD3_PBPIN = 0x4002_A330

Table 8.20. PBSTDn_PBPIN Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15:0	PBPIN	**Pin Value.** These bits read the digital logic level present at the corresponding port bank pin (bit x reads the logic level of pin PBn.x). Pins configured for analog mode will always read back 0.

Notes:

1. On SiM3x1x7 devices, PB0 and PB1 are full ports (PBx.0-PBx.15), PB2 consists of 15 pins (PB2.0-PB2.14), and PB3 consists of 12 pins (PB3.0-PB3.11). On SiM3x1x6 devices, PB0 and PB1 are full ports (PBx.0-PBx.15), PB2 consists of 4 pins (PB2.0-PB2.3), and PB3 consists of 10 pins (PB3.0-PB3.9). On SiM3x1x4 devices, PB0 is a full port (PB0.0-PB0.15), PB1 consists of 4 pins (PB1.0-PB1.3), PB2 is not implemented, and PB3 consists of 4 pins (PB3.0-PB3.3). Any bits in this register controlling unimplemented pins are reserved.

Register 8.9. PBSTDn_PBMDSEL: Mode Select

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	PBMDSEL															
Type	RW															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Register ALL Access Addresses

PBSTD0_PBMDSEL = 0x4002_A0C0
PBSTD1_PBMDSEL = 0x4002_A160
PBSTD2_PBMDSEL = 0x4002_A200
PBSTD3_PBMDSEL = 0x4002_A340
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 8.21. PBSTDn_PBMDSEL Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15:0	PBMDSEL	**Mode Select.** These bits configure the mode of the corresponding port bank pin (bit x controls the mode of pin PBn.x). Setting a bit to 1 configures the pin for digital mode, while clearing a bit to 0 configures the pin for analog mode. Pins configured in analog mode have their digital input paths and weak pullup disconnected.

Notes:

1. On SiM3x1x7 devices, PB0 and PB1 are full ports (PBx.0-PBx.15), PB2 consists of 15 pins (PB2.0-PB2.14), and PB3 consists of 12 pins (PB3.0-PB3.11). On SiM3x1x6 devices, PB0 and PB1 are full ports (PBx.0-PBx.15), PB2 consists of 4 pins (PB2.0-PB2.3), and PB3 consists of 10 pins (PB3.0-PB3.9). On SiM3x1x4 devices, PB0 is a full port (PB0.0-PB0.15), PB1 consists of 4 pins (PB1.0-PB1.3), PB2 is not implemented, and PB3 consists of 4 pins (PB3.0-PB3.3). Any bits in this register controlling unimplemented pins are reserved.

Register 8.10. PBSTDn_PBSKIPEN: Crossbar Pin Skip Enable

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	PBSKIPEN															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

PBSTD0_PBSKIPEN = 0x4002_A0D0
PBSTD1_PBSKIPEN = 0x4002_A170
PBSTD2_PBSKIPEN = 0x4002_A210
PBSTD3_PBSKIPEN = 0x4002_A350
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 8.22. PBSTDn_PBSKIPEN Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15:0	PBSKIPEN	**Crossbar Pin Skip Enable.** These bits configure the crossbar to skip over the corresponding port bank pin (bit x skips over pin PBn.x). Setting a bit to 1 prevents the crossbar from assigning a peripheral to the pin.

Notes:

1. On SiM3x1x7 devices, PB0 and PB1 are full ports (PBx.0-PBx.15), PB2 consists of 15 pins (PB2.0-PB2.14), and PB3 consists of 12 pins (PB3.0-PB3.11). On SiM3x1x6 devices, PB0 and PB1 are full ports (PBx.0-PBx.15), PB2 consists of 4 pins (PB2.0-PB2.3), and PB3 consists of 10 pins (PB3.0-PB3.9). On SiM3x1x4 devices, PB0 is a full port (PB0.0-PB0.15), PB1 consists of 4 pins (PB1.0-PB1.3), PB2 is not implemented, and PB3 consists of 4 pins (PB3.0-PB3.3). Any bits in this register controlling unimplemented pins are reserved.

Register 8.11. PBSTDn_PBOUTMD: Output Mode

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	PBOUTMD															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

PBSTD0_PBOUTMD = 0x4002_A0E0
PBSTD1_PBOUTMD = 0x4002_A180
PBSTD2_PBOUTMD = 0x4002_A220
PBSTD3_PBOUTMD = 0x4002_A360
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 8.23. PBSTDn_PBOUTMD Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15:0	PBOUTMD	**Output Mode.** These bits configure the digital output mode of the corresponding port bank pin (bit x configures the output mode of pin PBn.x). Setting a bit to 1 configures the pin for push-pull operation. Clearing a bit to 0 configures the pin for open-drain operation. Digital inputs should be configured for open drain mode, with a 1 written to the corresponding output latch.

Notes:

1. On SiM3x1x7 devices, PB0 and PB1 are full ports (PBx.0-PBx.15), PB2 consists of 15 pins (PB2.0-PB2.14), and PB3 consists of 12 pins (PB3.0-PB3.11). On SiM3x1x6 devices, PB0 and PB1 are full ports (PBx.0-PBx.15), PB2 consists of 4 pins (PB2.0-PB2.3), and PB3 consists of 10 pins (PB3.0-PB3.9). On SiM3x1x4 devices, PB0 is a full port (PB0.0-PB0.15), PB1 consists of 4 pins (PB1.0-PB1.3), PB2 is not implemented, and PB3 consists of 4 pins (PB3.0-PB3.3). Any bits in this register controlling unimplemented pins are reserved.

Register 8.12. PBSTDn_PBDRV: Drive Strength

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															PBPUEN
Type	R															RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	PBDRV															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

PBSTD0_PBDRV = 0x4002_A0F0
PBSTD1_PBDRV = 0x4002_A190
PBSTD2_PBDRV = 0x4002_A230
PBSTD3_PBDRV = 0x4002_A370
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 8.24. PBSTDn_PBDRV Register Bit Descriptions

Bit	Name	Function
31:17	Reserved	Must write reset value.
16	PBPUEN	**Port Bank Weak Pull-up Enable.** Globally enables the weak pull-up for all pins on this port bank. Pins in analog mode will have their individual weak pullups automatically disabled.
15:0	PBDRV	**Drive Strength.** These bits configure the output drive strength of the corresponding port bank pin (bit x configures the drive strength of pin PBn.x). Setting a bit to 1 enables high drive output on the pin. Clearing a bit to 0 selects low drive output for the pin.

Notes:

1. On SiM3x1x7 devices, PB0 and PB1 are full ports (PBx.0-PBx.15), PB2 consists of 15 pins (PB2.0-PB2.14), and PB3 consists of 12 pins (PB3.0-PB3.11). On SiM3x1x6 devices, PB0 and PB1 are full ports (PBx.0-PBx.15), PB2 consists of 4 pins (PB2.0-PB2.3), and PB3 consists of 10 pins (PB3.0-PB3.9). On SiM3x1x4 devices, PB0 is a full port (PB0.0-PB0.15), PB1 consists of 4 pins (PB1.0-PB1.3), PB2 is not implemented, and PB3 consists of 4 pins (PB3.0-PB3.3). Any bits in this register controlling unimplemented pins are reserved.

Register 8.13. PBSTDn_PM: Port Match Value

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	PM															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

PBSTD0_PM = 0x4002_A100
PBSTD1_PM = 0x4002_A1A0
PBSTD2_PM = 0x4002_A240
PBSTD3_PM = 0x4002_A380
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 8.25. PBSTDn_PM Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15:0	PM	**Port Match Value.** These bits serve a dual purpose as either a Port Match function, or an Activity Monitor function. When used in Port Match mode (MATMD is 00b), the bits in this register determine the match value for individual pins (bit x configures the match value for pin PBn.x). If the corresponding bit in PMEN is set to 1, a port match event will be triggered when the value in PM matches the logic level at the pin. If Activity Monitoring is enabled (MATMD is 01b or 10b), the Port Match registers are used for Activity Monitoring. In this mode, PM is read-only and reports the Activity Monitoring status on a pin, PMEN selects the pins for Activity Monitoring, and Port Match is unavailable.

Notes:

1. On SiM3x1x7 devices, PB0 and PB1 are full ports (PBx.0-PBx.15), PB2 consists of 15 pins (PB2.0-PB2.14), and PB3 consists of 12 pins (PB3.0-PB3.11). On SiM3x1x6 devices, PB0 and PB1 are full ports (PBx.0-PBx.15), PB2 consists of 4 pins (PB2.0-PB2.3), and PB3 consists of 10 pins (PB3.0-PB3.9). On SiM3x1x4 devices, PB0 is a full port (PB0.0-PB0.15), PB1 consists of 4 pins (PB1.0-PB1.3), PB2 is not implemented, and PB3 consists of 4 pins (PB3.0-PB3.3). Any bits in this register controlling unimplemented pins are reserved.

Register 8.14. PBSTDn_PMEN: Port Match Enable

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	PMEN															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

PBSTD0_PMEN = 0x4002_A110
PBSTD1_PMEN = 0x4002_A1B0
PBSTD2_PMEN = 0x4002_A250
PBSTD3_PMEN = 0x4002_A390
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 8.26. PBSTDn_PMEN Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15:0	PMEN	**Port Match Enable.** These bits enable Port Match or Activity Monitoring on the corresponding port bank pin (bit x enables these functions for pin PBn.x). Setting a bit to 1 enables the pin to be used for Port Match or Activity Monitoring. Clearing the bit to 0 disables these functions for the pin.

Notes:

1. On SiM3x1x7 devices, PB0 and PB1 are full ports (PBx.0-PBx.15), PB2 consists of 15 pins (PB2.0-PB2.14), and PB3 consists of 12 pins (PB3.0-PB3.11). On SiM3x1x6 devices, PB0 and PB1 are full ports (PBx.0-PBx.15), PB2 consists of 4 pins (PB2.0-PB2.3), and PB3 consists of 10 pins (PB3.0-PB3.9). On SiM3x1x4 devices, PB0 is a full port (PB0.0-PB0.15), PB1 consists of 4 pins (PB1.0-PB1.3), PB2 is not implemented, and PB3 consists of 4 pins (PB3.0-PB3.3). Any bits in this register controlling unimplemented pins are reserved.

Register 8.15. PBSTDn_PBLOCK: Lock Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	PBLOCK															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

PBSTD2_PBLOCK = 0x4002_A260

PBSTD3_PBLOCK = 0x4002_A3A0

Table 8.27. PBSTDn_PBLOCK Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15:0	PBLOCK	**Port Bank Lock.** These bits lock the corresponding port bank pins (bit x locks functions for pin PBn.x). Setting a bit to 1 prevents modification of that bit in all of the port bank control registers unless the port bank security interface is unlocked.

Notes:

1. Only standard ports PB2 and PB3 implement this register. On SiM3x1x7 devices, PB2 consists of 15 pins (PB2.0-PB2.14) and PB3 consists of 12 pins (PB3.0-PB3.11). On SiM3x1x6 devices, PB2 consists of 4 pins (PB2.0-PB2.3) and PB3 consists of 10 pins (PB3.0-PB3.9). On SiM3x1x4 devices, PB2 is not implemented and PB3 consists of 4 pins (PB3.0-PB3.3). Any bits in this register controlling unimplemented pins are reserved.

Register 8.16. PBSTDn_PBPGEN: Pulse Generator Pin Enable

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	PBPGEN															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

PBSTD2_PBPGEN = 0x4002_A270

Table 8.28. PBSTDn_PBPGEN Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15:0	PBPGEN	**Pulse Generator Pin Enable.** These bits enable the pulse generator function on the corresponding port bank pin (bit x enables the pulse generator for pin PBn.x). Setting a bit to 1 enables pulse generation for the pin and clearing a bit to 0 disables pulse generation for the pin.

Notes:

1. Only standard port PB2 implements this register. On SiM3x1x7 devices, PB2 consists of 15 pins (PB2.0-PB2.14). On SiM3x1x6 devices, PB2 consists of 4 pins (PB2.0-PB2.3). On SiM3x1x4 devices, PB2 is not implemented. Any bits in this register controlling unimplemented pins are reserved.

Register 8.17. PBSTDn_PBPGPHASE: Pulse Generator Phase

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	PBPGPH1															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	PBPGPH0															
Type	RW															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Register ALL Access Addresses

PBSTD2_PBPGPHASE = 0x4002_A280

Table 8.29. PBSTDn_PBPGPHASE Register Bit Descriptions

Bit	Name	Function
31:16	PBPGPH1	**Pulse Generator Phase 1.** These bits set the logic level for phase 1 of the pulse generator on the corresponding port bank pin (bit x defines phase 1 for pin PBn.x). If pulse generation is enabled on the pin, writing to this register will trigger the beginning of the pulse. The value in PBPGPH0 will be applied to the enabled pins immediately, and when the pulse generator counter times out, the enabled pins will be set to the value in PBPGPH1.
15:0	PBPGPH0	**Pulse Generator Phase 0.** These bits set the logic level for phase 0 of the pulse generator on the corresponding port bank pin (bit x defines phase 0 for pin PBn.x). If pulse generation is enabled on the pin, writing to this register will trigger the beginning of the pulse. The value in PBPGPH0 will be applied to the enabled pins immediately, and when the pulse generator counter times out, the enabled pins will be set to the value in PBPGPH1.

Notes:

1. Only standard port PB2 implements this register. On SiM3x1x7 devices, PB2 consists of 15 pins (PB2.0-PB2.14). On SiM3x1x6 devices, PB2 consists of 4 pins (PB2.0-PB2.3). On SiM3x1x4 devices, PB2 is not implemented. Any bits in this register controlling unimplemented pins are reserved.

8.20. PBSTDn Register Memory Map

Table 8.30. PBSTDn Memory Map

Register Name	ALL Offset	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBSTDn_PB	0x0	ALL | SET | CLR | MASK	Reserved																PB																
PBSTDn_PBPIN	0x10	ALL	Reserved																PBPIN																
PBSTDn_PBMDSEL	0x20	ALL | SET | CLR	Reserved																PBMDSEL																

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: PBSTD0 = 0x4002_A0A0, PBSTD1 = 0x4002_A140, PBSTD2 = 0x4002_A1E0, PBSTD3 = 0x4002_A320

Table 8.30. PBSTDn Memory Map

Register Name	ALL Offset	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBSTDn_PBSKIPEN	0x30	ALL | SET | CLR	Reserved																PBSKIPEN																
PBSTDn_PBOUTMD	0x40	ALL | SET | CLR	Reserved																PBOUTMD																
PBSTDn_PBDRV	0x50	ALL | SET | CLR	Reserved															PBPUEN	PBDRV																
PBSTDn_PM	0x60	ALL | SET | CLR	Reserved																PM																

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: PBSTD0 = 0x4002_A0A0, PBSTD1 = 0x4002_A140, PBSTD2 = 0x4002_A1E0, PBSTD3 = 0x4002_A320

Table 8.30. PBSTDn Memory Map

Register Name	ALL Offset	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PBSTDn_PMEN	0x70	ALL | SET | CLR	Reserved																PMEN																
PBSTDn_PBLOCK	0x80	ALL	Reserved																PBLOCK																
PBSTDn_PBPGEN	0x90	ALL	Reserved																PBPGEN																
PBSTDn_PBPGPHASE	0xA0	ALL	PBPGPH1																PBPGPH0																

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: PBSTD0 = 0x4002_A0A0, PBSTD1 = 0x4002_A140, PBSTD2 = 0x4002_A1E0, PBSTD3 = 0x4002_A320

8.21. PBHD4 Registers

This section contains the detailed register descriptions for PBHD4 registers.

Register 8.18. PBHD4_PB: Output Latch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved										PB					
Type	R										RW					
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Register ALL Access Address

PBHD4_PB = 0x4002_A3C0

This register also supports SET access at (ALL+0x4), CLR access at (ALL+0x8) and MSK access at (ALL+0xC)

Table 8.31. PBHD4_PB Register Bit Descriptions

Bit	Name	Function
31:6	Reserved	Must write reset value.
5:0	PB	**Output Latch.** These bits define the logic level of the port bank output latch. Each bit in this field controls the output latch value for the corresponding port bank pin (bit x controls the latch for pin PBn.x). Digital input pins should be written to 1 and configured for open drain mode. When using this register via the MSK address, the upper 16 bits can be used to mask writes of the lower 16 bits to the corresponding port bank latches.

Notes:

1. On SiM3x1x7 devices, PB4 is a full high drive port (PB4.0-PB4.5). On SiM3x1x6 devices, PB4 consists of 4 pins (PB4.0-PB4.3). On SiM3x1x4 devices, PB4 consists of 4 pins (PB4.0-PB4.3). Any bits in this register controlling unimplemented pins are reserved.

Register 8.19. PBHD4_PBPIN: Pin Value

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved										PBPIN					
Type	R										R					
Reset	0	0	0	0	0	0	0	0	0	0	X	X	X	X	X	X

Register ALL Access Address

PBHD4_PBPIN = 0x4002_A3D0

Table 8.32. PBHD4_PBPIN Register Bit Descriptions

Bit	Name	Function
31:6	Reserved	Must write reset value.
5:0	PBPIN	**Pin Value.** These bits read the digital logic level present at the corresponding port bank pin (bit x reads the logic level of pin PBn.x). Pins configured for analog mode will always read back 0.

Notes:

1. On SiM3x1x7 devices, PB4 is a full high drive port (PB4.0-PB4.5). On SiM3x1x6 devices, PB4 consists of 4 pins (PB4.0-PB4.3). On SiM3x1x4 devices, PB4 consists of 4 pins (PB4.0-PB4.3). Any bits in this register controlling unimplemented pins are reserved.

Register 8.20. PBHD4_PBMDSEL: Mode Select

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved										PBMDSEL					
Type	R										RW					
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Register ALL Access Address

PBHD4_PBMDSEL = 0x4002_A3E0

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 8.33. PBHD4_PBMDSEL Register Bit Descriptions

Bit	Name	Function
31:6	Reserved	Must write reset value.
5:0	PBMDSEL	**Mode Select.** These bits configure the mode of the corresponding port bank pin (bit x controls the mode of pin PBn.x). Setting a bit to 1 configures the pin for digital mode, while clearing a bit to 0 configures the pin for analog mode. Pins configured in analog mode have their digital input paths and weak pullup disconnected.

Notes:

1. On SiM3x1x7 devices, PB4 is a full high drive port (PB4.0-PB4.5). On SiM3x1x6 devices, PB4 consists of 4 pins (PB4.0-PB4.3). On SiM3x1x4 devices, PB4 consists of 4 pins (PB4.0-PB4.3). Any bits in this register controlling unimplemented pins are reserved.

Register 8.21. PBHD4_PBDEN: Driver Enable

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved										PBPDEN					
Type	R										RW					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved										PBNDEN					
Type	R										RW					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

PBHD4_PBDEN = 0x4002_A3F0

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 8.34. PBHD4_PBDEN Register Bit Descriptions

Bit	Name	Function
31:22	Reserved	Must write reset value.
21:16	PBPDEN	**Port Bank P-Channel Driver Enable.** These bits enable the P-channel (high-side) driver for the corresponding port bank pin (bit x enables the P-channel driver for pin PBn.x). When a bit is set to 1, the P-channel driver is enabled.
15:6	Reserved	Must write reset value.
5:0	PBNDEN	**Port Bank N-Channel Driver Enable.** These bits enable the N-channel (low-side) driver for the corresponding port bank pin (bit x enables the N-channel driver for pin PBn.x). When a bit is set to 1, the N-channel driver is enabled.

Notes:

1. On SiM3x1x7 devices, PB4 is a full high drive port (PB4.0-PB4.5). On SiM3x1x6 devices, PB4 consists of 4 pins (PB4.0-PB4.3). On SiM3x1x4 devices, PB4 consists of 4 pins (PB4.0-PB4.3). Any bits in this register controlling unimplemented pins are reserved.

Register 8.22. PBHD4_PBDRV: Drive Strength

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved								PBVTRKEN	PBDRVEN	PBBIASEN	Reserved	PBSLEW		PBLVMD	PBPUEN
Type	R								RW	RW	RW	RW	RW		RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved										PBDRV					
Type	R										RW					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

PBHD4_PBDRV = 0x4002_A400

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 8.35. PBHD4_PBDRV Register Bit Descriptions

Bit	Name	Function
31:24	Reserved	Must write reset value.
23	PBVTRKEN	**Port Voltage Supply Tracking Enable.** When set to 1, this bit enables an internal 1/4 voltage divider on the VIOHD supply. The output of this divider can be used by the Comparator and SARADC modules. 0: Disable VIOHD tracking. 1: Enable VIOHD tracking.
22	PBDRVEN	**Port Drive Enable.** 0: Disable the port drivers. 1: Enable the port drivers.
21	PBBIASEN	**Port Bias Enable.** When set to 1, this bit enables the bias generator for high drive capability on this port. When enabling High Drive Port Bank pins, firmware must set these bits in the following sequence with separate register writes: 1) PBBIASEN 2) PBLVMD (optional) 3) PBDRVEN
20	Reserved	Must write reset value.

Notes:

1. On SiM3x1x7 devices, PB4 is a full high drive port (PB4.0-PB4.5). On SiM3x1x6 devices, PB4 consists of 4 pins (PB4.0-PB4.3). On SiM3x1x4 devices, PB4 consists of 4 pins (PB4.0-PB4.3). Any bits in this register controlling unimplemented pins are reserved.

Table 8.35. PBHD4_PBDRV Register Bit Descriptions

Bit	Name	Function
19:18	PBSLEW	**Port Slew Control.** Selects the slew rate for this port bank. Four slew rates are available, from 0 (fastest) to 3 (slowest).
17	PBLVMD	**Port Low Voltage Mode.** Low power mode (PBLVMD = 1) must be used if VIOHD is less than 2.7 V and is recommended if VIOHD is less than 3.0 V. PBLVMD must be cleared to 0 if VIOHD is greater than 3.6 V. 0: Port configured for normal mode. 1: Port configured for low power mode.
16	PBPUEN	**Port Bank Weak Pull-up Enable.** Globally enables the weak pull-up for all pins on this port bank. Pins in analog mode will have their individual weak pullups automatically disabled.
15:6	Reserved	Must write reset value.
5:0	PBDRV	**Drive Strength.** These bits configure the output drive strength of the corresponding port bank pin (bit x configures the drive strength of pin PBn.x). Setting a bit to 1 enables high drive output on the pin. Clearing a bit to 0 selects low drive output for the pin. PBDRV must be cleared to 0 to use the current-limiting feature.

Notes:

1. On SiM3x1x7 devices, PB4 is a full high drive port (PB4.0-PB4.5). On SiM3x1x6 devices, PB4 consists of 4 pins (PB4.0-PB4.3). On SiM3x1x4 devices, PB4 consists of 4 pins (PB4.0-PB4.3). Any bits in this register controlling unimplemented pins are reserved.

Register 8.23. PBHD4_PBILIMIT: Current Limit

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved								PILIMIT				NILIMIT			
Type	R								RW				RW			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved										PBILEN					
Type	R										RW					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

PBHD4_PBILIMIT = 0x4002_A410

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 8.36. PBHD4_PBILIMIT Register Bit Descriptions

Bit	Name	Function
31:24	Reserved	Must write reset value.
23:20	PILIMIT	**P-Channel Current Limit.** Current limit value for P-channel devices for all pins in this port bank. Each pin can individually have current limiting enabled or disabled. There are 16 current limit levels, from 0 (lowest) to 15 (highest).
19:16	NILIMIT	**N-Channel Current Limit.** This field is the current limit value for N-channel devices for all pins in this port bank. Each pin can individually have current limiting enabled or disabled. There are 16 current limit levels, from 0 (lowest) to 15 (highest).
15:6	Reserved	Must write reset value.
5:0	PBILEN	**Current Limit Enable.** These bits enable the current-limiting circuitry of the corresponding port bank pin (bit x enables the current limiter for pin PBn.x). Setting a bit to 1 enables the current limiting circuitry for that pin. The drive strength must be set to low (PBDRV=0) to use current-limiting.

Notes:

1. On SiM3x1x7 devices, PB4 is a full high drive port (PB4.0-PB4.5). On SiM3x1x6 devices, PB4 consists of 4 pins (PB4.0-PB4.3). On SiM3x1x4 devices, PB4 consists of 4 pins (PB4.0-PB4.3). Any bits in this register controlling unimplemented pins are reserved.

Register 8.24. PBHD4_PBFSEL: Function Select

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved			PB5SEL			PB4SEL		PB3SEL		PB2SEL		PB1SEL		PB0SEL	
Type	R			RW			RW		RW		RW		RW		RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

PBHD4_PBFSEL = 0x4002_A430

Table 8.37. PBHD4_PBFSEL Register Bit Descriptions

Bit	Name	Function
31:13	Reserved	Must write reset value.
12:10	PB5SEL	**Port Bank n.5 Function Select.** Digital functional selection for pin PBn.5. 000: Pin configured for GPIO. 001: Pin configured for Port Mapped Level Shift. 010: Pin configured for EPCA0 output. 011: Pin configured for UART1 CTS. 100: Pin configured for LPTIMER0 toggle output. 101-111: Reserved.
9:8	PB4SEL	**Port Bank n.4 Function Select.** Digital functional selection for pin PBn.4. 00: Pin configured for GPIO. 01: Pin configured for Port Mapped Level Shift. 10: Pin configured for EPCA0 output. 11: Pin configured for UART1 RTS.
7:6	PB3SEL	**Port Bank n.3 Function Select.** Digital functional selection for pin PBn.3. 00: Pin configured for GPIO. 01: Pin configured for Port Mapped Level Shift. 10: Pin configured for EPCA0 output. 11: Pin configured for UART1 RX.

Notes:

1. On SiM3x1x7 devices, PB4 is a full high drive port (PB4.0-PB4.5). On SiM3x1x6 devices, PB4 consists of 4 pins (PB4.0-PB4.3). On SiM3x1x4 devices, PB4 consists of 4 pins (PB4.0-PB4.3). Any bits in this register controlling unimplemented pins are reserved.

Table 8.37. PBHD4_PBFSEL Register Bit Descriptions

Bit	Name	Function
5:4	PB2SEL	**Port Bank n.2 Function Select.** Digital functional selection for pin PBn.2. 00: Pin configured for GPIO. 01: Pin configured for Port Mapped Level Shift. 10: Pin configured for EPCA0 output. 11: Pin configured for UART1 TX.
3:2	PB1SEL	**Port Bank n.1 Function Select.** Digital functional selection for pin PBn.1. 00: Pin configured for GPIO. 01: Pin configured for Port Mapped Level Shift. 10: Pin configured for EPCA0 output. 11: Reserved.
1:0	PB0SEL	**Port Bank n.0 Function Select.** Digital functional selection for pin PBn.0. 00: Pin configured for GPIO. 01: Pin configured for Port Mapped Level Shift. 10: Pin configured for EPCA0 output. 11: Reserved.

Notes:

1. On SiM3x1x7 devices, PB4 is a full high drive port (PB4.0-PB4.5). On SiM3x1x6 devices, PB4 consists of 4 pins (PB4.0-PB4.3). On SiM3x1x4 devices, PB4 consists of 4 pins (PB4.0-PB4.3). Any bits in this register controlling unimplemented pins are reserved.

Register 8.25. PBHD4_PBSS: Safe State Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved														PBSSSMD	SSMDEN
Type	R														RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved				PB5SSSEL		PB4SSSEL		PB3SSSEL		PB2SSSEL		PB1SSSEL		PB0SSSEL	
Type	R				RW		RW		RW		RW		RW		RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

PBHD4_PBSS = 0x4002_A440

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 8.38. PBHD4_PBSS Register Bit Descriptions

Bit	Name	Function
31:18	Reserved	Must write reset value.
17	PBSSSMD	**Safe State Signal Mode.** When PBSSSMD is cleared to 0, the kill signal from from a Crossbar must be asserted for two APB clocks before the pins will switch to their designated safe state. When PBSSSMD is set to 1, the kill signal will bypass this deglitching logic and immediately take effect.
16	SSMDEN	**Enter Safe State Mode.** Set by firmware or the $\overline{\text{PB_HDKill}}$ signal to place the high drive port in the Safe State specified by the PBxSSSEL fields. Once set, this bit must be cleared by firmware or a device reset. 0: Disable Safe State. 1: Enter Safe State. Each PBn.x pin will enter the states defined by PBxSSSEL.
15:12	Reserved	Must write reset value.

Notes:

1. On SiM3x1x7 devices, PB4 is a full high drive port (PB4.0-PB4.5). On SiM3x1x6 devices, PB4 consists of 4 pins (PB4.0-PB4.3). On SiM3x1x4 devices, PB4 consists of 4 pins (PB4.0-PB4.3). Any bits in this register controlling unimplemented pins are reserved.

Table 8.38. PBHD4_PBSS Register Bit Descriptions

Bit	Name	Function
11:10	PB5SSSEL	**Port Bank n.5 Safe State Select.** Safe state for pin PBn.5. 00: Place PBn.5 in a High Impedance state. 01: Drive PBn.5 High. 10: Drive PBn.5 Low. 11: Ignore the safe state signal (weak pull-ups disabled).
9:8	PB4SSSEL	**Port Bank n.4 Safe State Select.** Safe state for pin PBn.4. 00: Place PBn.4 in a High Impedance state. 01: Drive PBn.4 High. 10: Drive PBn.4 Low. 11: Ignore the safe state signal (weak pull-ups disabled).
7:6	PB3SSSEL	**Port Bank n.3 Safe State Select.** Safe state for pin PBn.3. 00: Place PBn.3 in a High Impedance state. 01: Drive PBn.3 High. 10: Drive PBn.3 Low. 11: Ignore the safe state signal (weak pull-ups disabled).
5:4	PB2SSSEL	**Port Bank n.2 Safe State Select.** Safe state for pin PBn.2. 00: Place PBn.2 in a High Impedance state. 01: Drive PBn.2 High. 10: Drive PBn.2 Low. 11: Ignore the safe state signal (weak pull-ups disabled).
3:2	PB1SSSEL	**Port Bank n.1 Safe State Select.** Safe state for pin PBn.1. 00: Place PBn.1 in a High Impedance state. 01: Drive PBn.1 High. 10: Drive PBn.1 Low. 11: Ignore the safe state signal (weak pull-ups disabled).
1:0	PB0SSSEL	**Port Bank n.0 Safe State Select.** Safe state for pin PBn.0. 00: Place PBn.0 in a High Impedance state. 01: Drive PBn.0 High. 10: Drive PBn.0 Low. 11: Ignore the safe state signal (weak pull-ups disabled).

Notes:

1. On SiM3x1x7 devices, PB4 is a full high drive port (PB4.0-PB4.5). On SiM3x1x6 devices, PB4 consists of 4 pins (PB4.0-PB4.3). On SiM3x1x4 devices, PB4 consists of 4 pins (PB4.0-PB4.3). Any bits in this register controlling unimplemented pins are reserved.

Register 8.26. PBHD4_PBLOCK: Lock Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved										PBLOCK					
Type	R										RW					
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

Register ALL Access Address

PBHD4_PBLOCK = 0x4002_A450

Table 8.39. PBHD4_PBLOCK Register Bit Descriptions

Bit	Name	Function
31:6	Reserved	Must write reset value.
5:0	PBLOCK	**Port Bank Lock.** These bits lock the corresponding port bank pins (bit x locks functions for pin PBn.x). Setting a bit to 1 prevents modification of that bit in all of the port bank control registers unless the port bank security interface is unlocked.

Notes:

1. On SiM3x1x7 devices, PB4 is a full high drive port (PB4.0-PB4.5). On SiM3x1x6 devices, PB4 consists of 4 pins (PB4.0-PB4.3). On SiM3x1x4 devices, PB4 consists of 4 pins (PB4.0-PB4.3). Any bits in this register controlling unimplemented pins are reserved.

8.22. PBHD4 Register Memory Map

Table 8.40. PBHD4 Memory Map

Register Name	PBHD4_PB	PBHD4_PBPIN	PBHD4_PBMDSEL	PBHD4_PBDEN
ALL Address	0x4002_A3C0	0x4002_A3D0	0x4002_A3E0	0x4002_A3F0
Access Methods	ALL | SET | CLR | MASK	ALL	ALL | SET | CLR	ALL | SET | CLR
Bit 31	Reserved	Reserved	Reserved	Reserved
Bit 30				
Bit 29				
Bit 28				
Bit 27				
Bit 26				
Bit 25				
Bit 24				
Bit 23				
Bit 22				
Bit 21				
Bit 20				PBPDEN
Bit 19				
Bit 18				
Bit 17				
Bit 16				
Bit 15				Reserved
Bit 14				
Bit 13				
Bit 12				
Bit 11				
Bit 10				
Bit 9				
Bit 8				
Bit 7				
Bit 6				
Bit 5	PB	PBPIN	PBMDSEL	PBNDEN
Bit 4				
Bit 3				
Bit 2				
Bit 1				
Bit 0				

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

Table 8.40. PBHD4 Memory Map

Register Name	PBHD4_PBLOCK	PBHD4_PBSS	PBHD4_PBFSEL	PBHD4_PBILIMIT	PBHD4_PBDRV	Register Name
ALL Address	0x4002_A450	0x4002_A440	0x4002_A430	0x4002_A410	0x4002_A400	ALL Address
Access Methods	ALL	ALL | SET | CLR	ALL	ALL | SET | CLR	ALL | SET | CLR	Access Methods
Bit 31	Reserved	Reserved	Reserved	Reserved	Reserved	Bit 31
Bit 30						Bit 30
Bit 29						Bit 29
Bit 28						Bit 28
Bit 27						Bit 27
Bit 26						Bit 26
Bit 25						Bit 25
Bit 24						Bit 24
Bit 23					PBVTRKEN	Bit 23
Bit 22					PBDRVEN	Bit 22
Bit 21					PBBIASEN	Bit 21
Bit 20					Reserved	Bit 20
Bit 19				NILIMIT	PBSLEW	Bit 19
Bit 18						Bit 18
Bit 17					PBLVMD	Bit 17
Bit 16					PBPUEN	Bit 16
Bit 15				Reserved	Reserved	Bit 15
Bit 14						Bit 14
Bit 13						Bit 13
Bit 12						Bit 12
Bit 11						Bit 11
Bit 10						Bit 10
Bit 9			PB5SEL	PBILEN	PBDRV	Bit 9
Bit 8			PB4SEL			Bit 8
Bit 7			PB3SEL			Bit 7
Bit 6			PB2SEL			Bit 6
Bit 5			PB1SEL			Bit 5
Bit 4			PB0SEL			Bit 4
Bit 3		PB5SSSEL				Bit 3
Bit 2		PB4SSSEL				Bit 2
Bit 1		PB3SSSEL				Bit 1
Bit 0	PBLOCK	PB2SSSEL				Bit 0
		PB1SSSEL				
		PB0SSSEL				
		Reserved				
		SSMDEN				
		PBSSSMD				

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

9. Power

This section describes the power modes and Power Management Unit of SiM3U1xx/SiM3C1xx devices.

9.1. Power Modes

The SiM3U1xx/SiM3C1xx devices have the power modes defined in Table 9.1.

Table 9.1. SiM3U1xx/SiM3C1xx Power Modes

Mode	Description	Mode Entrance	Mode Exit
Normal	■ Core operating at full speed ■ Code executing from flash	—	—
Power Mode 1 (PM1)	■ Core operating at full speed ■ Code executing from RAM	—	—
Power Mode 2 (PM2)	■ Core halted ■ AHB and APB clocks on selectively to support peripherals	WFI or WFE instruction	NVIC or WIC wakeup
Power Mode 3 (PM3)	■ Core halted ■ All AHB and APB clocks off	■ DMACTRL0 AHB clock disabled ■ All APB clocks disabled ■ PMSEL bit in RSTSRC0_CONTROL must be cleared to 0 ■ WFI or WFE instruction	Reset event, or RTC0ALRM, RTC0FAIL, LPTIMER0, VDDLOW or VREGLOW interrupt
Power Mode 9 (PM9)	Low power shutdown	■ SLEEPDEEP set in the ARM System Control Register ■ PMSEL bit in RSTSRC0_CONTROL must be set to 1 ■ WFI or WFE instruction	Requires a reset defined by the PMU as a wake up source

In addition to the power modes described in Table 9.1, all peripherals can have their clocks disconnected in the Clock Control module (CLKCTRL) to reduce power consumption whenever a peripheral is not being used.

9.1.1. Normal Mode (Power Mode 0)

Normal Mode is the default mode of the device. The core and peripherals are fully operational, and instructions are executed from flash memory.

9.1.2. Power Mode 1

In Power Mode 1 the core and peripherals are fully operational, with instructions executing from RAM. Compared with Normal Mode, the active power consumption of the device in PM1 is reduced. Additionally, at higher speeds in PM1, the core throughput can also be increased because RAM does not require additional wait states that reduce the instruction fetch speed.

9.1.3. Power Mode 2

In Power Mode 2 the core halts and any enabled peripherals continue to run at the selected clock speed. The power consumption in PM2 corresponds to the AHB and APB clocks left enabled, thus the power can be tuned to the optimal level for the needs of the application. To place the device in PM2, the core should execute a wait-for-interrupt (WFI) or wait-for-event (WFE) instruction. If the WFI instruction is called from an interrupt service routine, the interrupt that wakes the device from PM2 must be of a sufficient priority to be recognized by the core. It is recommended to perform both a DSB (Data Synchronization Barrier) and an ISB (Instruction Synchronization Barrier) operation prior to the WFI to ensure all bus accesses complete. When operating from the LFOSC0 with the DMACTRL0 AHB clock disabled, PM2 can achieve similar power consumption to PM3, but with the ability to wake on APB-clocked interrupts. For example, enabling only the APB clock to the Ports will allow the firmware to wake on a PMATCH0, PBEXT0 or PBEXT1 interrupt with minimal impact on the supply current.

9.1.4. Power Mode 3

In Power Mode 3, the AHB and APB clocks are halted. The device may only wake from enabled interrupt sources which do not require the APB clock (RTC0ALRM, RTC0FAIL, LPTIMER0, VDDLOW and VREGLOW). A special fast wake option allows the device to operate at a very low level from the RTC0TCLK or LFOSC0 oscillator while in PM3, but quickly switch to the faster LPOSC0 when the wake event occurs. Because the current consumption of these blocks is minimal, it is recommended to use the fast wake option.

Before entering PM3, the desired wake source interrupt(s) should be configured, and the AHB clock to the DMA controller and all APB clocks should be disabled. The PMSEL bit in the RSTSRC0_CONFIG register must be cleared to indicate that PM3 is the desired power mode. For fast wake, the core clocks (AHB and APB) should be configured to run from the LPOSC, and the PM3 Fast wake option and PM3 clock source should be selected in the PM3CN register.

The device will enter PM3 on a WFI or WFE instruction. Because all AHB master clocks are disabled, the LPOSC will automatically halt and go into a low-power suspended state. If the WFI instruction is called from an interrupt service routine, the interrupt that wakes the device from PM3 must be of a sufficient priority to be recognized by the core. It is recommended to perform both a DSB (Data Synchronization Barrier) and an ISB (Instruction Synchronization Barrier) operation prior to the WFI to ensure all bus access is complete.

9.1.5. Power Mode 9

In Power Mode 9, the core and all peripherals are halted, all clocks are stopped, and the pins and peripherals are set to a lower power mode. In addition, standard RAM contents are not preserved, though retention RAM contents are still available after exiting the power mode. This mode provides the lowest power consumption for the device, but requires an appropriate reset to exit. The available reset sources to wake from PM9 are controlled by the Power Management Unit (PMU).

Before entering PM9, the desired reset source(s) should be configured in the PMU. The SLEEPDEEP bit in the ARM System Control Register should be set, and the PMSEL bit in the RSTSRC0_CONFIG register must be set to indicate that PM9 is the desired power mode.

The device will enter PM9 on a WFI or WFE instruction, and remain in PM9 until a reset configured by the PMU occurs. It is recommended to perform both a DSB (Data Synchronization Barrier) and an ISB (Instruction Synchronization Barrier) operation prior to the WFI to ensure all bus access is complete. More information on the wake up sources and wake up procedure can be found in Section 9.2.

9.2. Power Management Unit (PMU0)

This section describes the Power Management Unit (PMU) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the PMU block, which is used by all device families covered in this document.

The PMU module includes the following features:

- Up to 16 pin wake inputs can wake the PMU from Power Mode 9.
- The Low Power Timer, RTC0 (alarms and oscillator fail), Comparator 0, and the $\overline{\text{RESET}}$ pin can also serve as wake sources for Power Mode 9.
- All PM9 wake sources (except for the $\overline{\text{RESET}}$ pin) can also reset the Low Power Timer or RTC0 modules.
- Disables the level shifters to pins and peripherals to further reduce power usage in PM9. These level shifters must be re-enabled by firmware after exiting PM9.
- Provides a PMU_Asleep signal to a pin as an indicator that the device is in PM9.

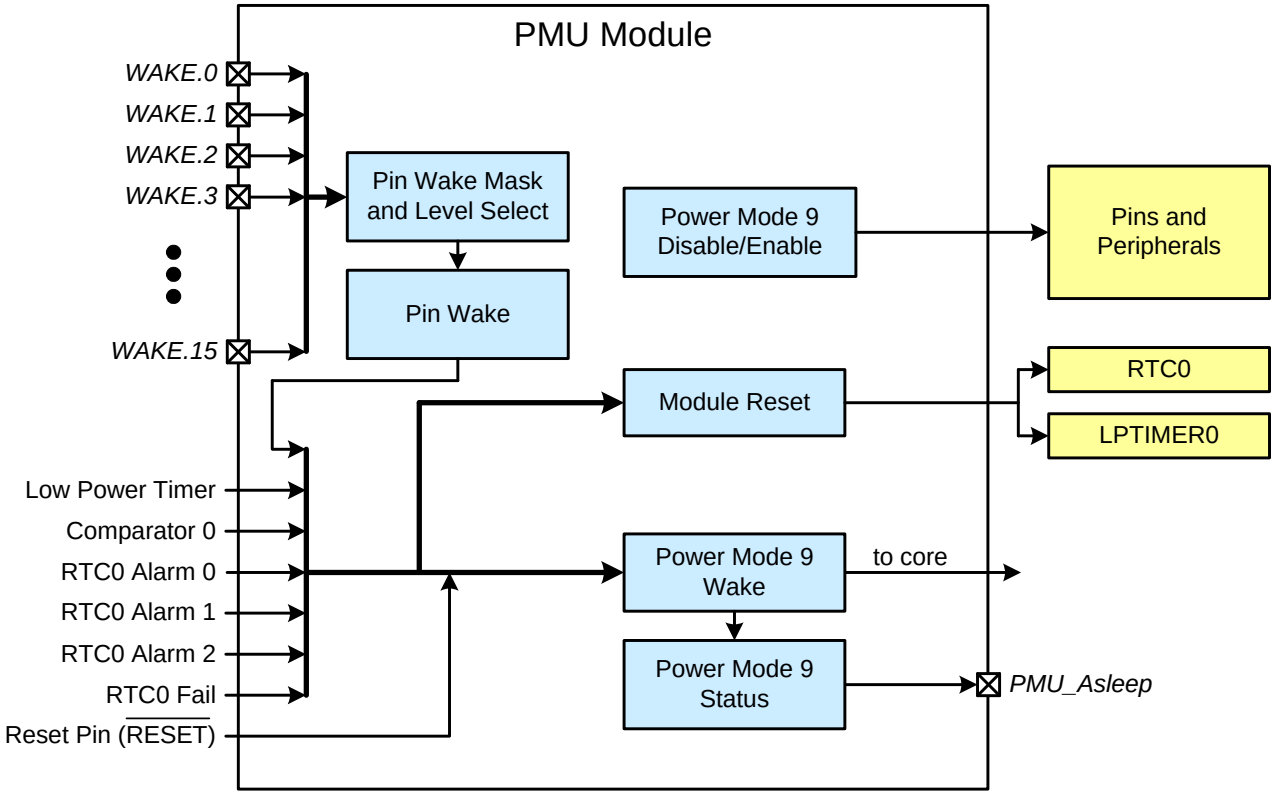


Figure 9.1. PMU Block Diagram

The PMU manages the power-up sequence on power on and the wake up sources from PM9. On power-up, the PMU ensures the core voltages are of a proper value before core instruction execution begins.

9.2.1. Waking from Power Mode 9

The reset wake for PM9 can be sourced from pins (Pin Wake), the Low Power Timer, Comparator 0, RTC0 Alarms (0, 1, or 2), RTC0 Fail, or the Reset Pin (RESET). In most cases, the corresponding interrupt enable must be set in the module in order for an event to be a wakeup source. The Comparator module is the exception and the wakeup event will occur even if the interrupt is disabled. These wakeup sources (except for the reset pin) can also be optionally used to reset RTC0 or the Low Power Timer while the device remains in PM9.

Firmware can check the PM9EF bit during the initialization sequence to determine if the device reset because of a wake from Power Mode 9. If the device did reset because of a wake from PM9, firmware must clear the bits keeping the peripheral and pin interfaces in a lower power state (PERILPEN and PINLPEN), and the WAKESTATUS register provides status flags to indicate the wakeup source. The WAKESTATUS register can be cleared by writing 0 to the WAKECLR bit.

9.2.2. Pin Wake

Up to 16 pins are available as Pin Wake sources (WAKE.0-WAKE.15). any one of these signals can be used to wake the device from low power sleep modes. The pin wake function can be enabled for a pin by setting the corresponding bit in the PWEN register. The desired polarity of the pin to wake the device should be programmed into the corresponding bit in the PWPOL register. Note that if multiple pins are selected as wake sources using PWEN, the polarity of all of the selected pins must match PWPOL in order to wake the device. To serve as a wake source, the pins must remain active in the matching state long enough for the PMU to detect the wake up signal (50ns). The pin wake trigger sources vary by package and are shown in Table 9.2.

Table 9.2. Pin Wake Sources

Pin Wake Source	Trigger Description	SiM3U1x7/C1x7 Pin Name	SiM3U1x6/C1x6 Pin Name	SiM3U1x4/C1x4 Pin Name
WAKE.0	Low Power Wake Pin	PB1.13	PB1.6	PB0.12
WAKE.1	Low Power Wake Pin	PB1.14	PB1.7	PB0.13
WAKE.2	Low Power Wake Pin	PB1.15	PB1.8	PB0.14
WAKE.3	Low Power Wake Pin	PB2.0	PB1.9	PB0.15
WAKE.4	Low Power Wake Pin	PB2.1	PB1.10	PB1.0
WAKE.5	Low Power Wake Pin	PB2.2	PB1.11	PB1.1
WAKE.6	Low Power Wake Pin	PB2.3	PB1.12	Reserved
WAKE.7	Low Power Wake Pin	PB2.4	Reserved	Reserved
WAKE.8	Low Power Wake Pin	PB3.4	PB3.2	Reserved
WAKE.9	Low Power Wake Pin	PB3.5	PB3.3	Reserved
WAKE.10	Low Power Wake Pin	PB3.6	PB3.4	Reserved
WAKE.11	Low Power Wake Pin	PB3.7	PB3.5	Reserved
WAKE.12	Low Power Wake Pin	PB3.8	PB3.6	PB3.0

Table 9.2. Pin Wake Sources

Pin Wake Source	Trigger Description	SiM3U1x7/C1x7 Pin Name	SiM3U1x6/C1x6 Pin Name	SiM3U1x4/C1x4 Pin Name
WAKE.13	Low Power Wake Pin	PB3.9	PB3.7	PB3.1
WAKE.14	Low Power Wake Pin	PB3.10	PB3.8	PB3.2
WAKE.15	Low Power Wake Pin	PB3.11	PB3.9	PB3.3

9.2.3. Low Power Timer

The Low Power Timer wake up source is caused by a Low Power Timer overflow.

9.2.4. Comparator 0

A Comparator 0 (CMP0) event can serve as a wake up or reset source in the PMU. This event can occur from a rising, falling, or either edge on the Comparator 0 output, depending on the settings in the Comparator module.

9.2.5. RTC0

The RTC0 Alarms (0, 1, and 2) and Fail events are wake up sources from PM9 or can automatically reset the LPTIMER0 and RTC0 modules. The Alarms occur from a match between the RTC0 timer and the corresponding Alarm compare value. The Fail event occurs when the RTC0 missing clock detector indicates the RTC0 clock is no longer running.

9.3. PMU0 Registers

This section contains the detailed register descriptions for PMU0 registers.

Register 9.1. PMU0_CONTROL: Module Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved											PMUASLPEN	PWAKEEN	PINLPEN	PERILPEN	WAKECLR
Type	R											RW	RW	RW	RW	W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

PMU0_CONTROL = 0x4004_8000

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 9.3. PMU0_CONTROL Register Bit Descriptions

Bit	Name	Function
31:5	Reserved	Must write reset value.
4	PMUASLPEN	**PMU Asleep Pin Enable.** When set to 1, the PMU Asleep signal will be sent to the appropriate pin. This pin should be skipped by the Crossbar if firmware enables the PMU Asleep signal.
3	PWAKEEN	**Pin Wake Match Enable.** 0: Disable Pin Wake. 1: Enable Pin Wake.
2	PINLPEN	**Pin Low Power Enable.** When this bit is set, the PMU will place the pin interfaces in a low power state before entering Power Mode 9. After exiting PM9, firmware must clear this bit to resume normal operation of the pins.
1	PERILPEN	**Peripheral Low Power Enable.** When this bit is set, the PMU will place the peripheral interfaces in a low power state before entering Power Mode 9. After exiting PM9, firmware must clear this bit to resume normal operation of the peripherals.
0	WAKECLR	**Wakeup Source Clear.** Writing a 0 to this bit clears all wakeup sources. 0: Clear all wakeup sources. 1: Reserved.

Register 9.2. PMU0_CONFIG: Module Configuration

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	LPT0REN	PWAKEREN	CMP0REN	Reserved			RTC0AREN	RTC0FREN	Reserved							
Type	RW	RW	RW	RW			RW	RW	R			RW				
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Register ALL Access Address

PMU0_CONFIG = 0x4004_8010

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 9.4. PMU0_CONFIG Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15	LPT0REN	**Low Power Timer RTC0/LPTIMER0 Reset Enable.** When set to 1, an LPTIMER0 event will cause the RTC0 and LPTIMER0 modules to reset.
14	PWAKEREN	**Pin Wake RTC0/LPTIMER0 Reset Enable.** When set to 1, a Pin Wake event will cause the RTC0 and LPTIMER0 modules to reset.
13	CMP0REN	**Comparator 0 RTC0/LPTIMER0 Reset Enable.** When set to 1, a Comparator 0 event will cause the RTC0 and LPTIMER0 modules to reset.
12:10	Reserved	Must write reset value.
9	RTC0AREN	**RTC0 Alarm RTC0/LPTIMER0 Reset Enable.** When set to 1, an RTC0 alarm event will cause the RTC0 and LPTIMER0 modules to reset. This alarm event can occur from any of the three RTC0 alarms.
8	RTC0FREN	**RTC0 Fail RTC0/LPTIMER0 Reset Enable.** When set to 1, an RTC0 fail event will cause the RTC0 and LPTIMER0 modules to reset.
7:0	Reserved	Must write reset value.

Register 9.3. PMU0_STATUS: Module Status

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved													PORF	PWAKEF	PM9EF
Type	R													RW	R	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Register ALL Access Address

PMU0_STATUS = 0x4004_8020

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 9.5. PMU0_STATUS Register Bit Descriptions

Bit	Name	Function
31:3	Reserved	Must write reset value.
2	PORF	**Power-On Reset Flag.** Hardware sets this bit to 1 to indicate that a power-on reset event occurred. This bit must be cleared by firmware.
1	PWAKEF	**Pin Wake Status Flag.** When cleared to 0, this flag indicates that a pin wake event has occured.
0	PM9EF	**Power Mode 9 Exited Flag.** When set to 1, this flag indicates that the device exited Power Mode 9. Firmware must clear this flag.

Register 9.4. PMU0_WAKEEN: Wake Source Enable

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved							RSTWEN	LPT0WEN	PWAKEWEN	CMP0WEN	Reserved			RTC0AWEN	RTC0FWEN
Type	R							RW	RW	RW	RW	RW			RW	RW
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Register ALL Access Address

PMU0_WAKEEN = 0x4004_8030

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 9.6. PMU0_WAKEEN Register Bit Descriptions

Bit	Name	Function
31:9	Reserved	Must write reset value.
8	RSTWEN	**Reset Pin Wake Enable.** When set to 1, a $\overline{\text{RESET}}$ Pin event will wake the device from Power Mode 9.
7	LPT0WEN	**Low Power Timer Wake Enable.** When set to 1, an LPTIMER0 event will wake the device from Power Mode 9.
6	PWAKEWEN	**Pin Wake Wake Enable.** When set to 1, a Pin Wake event will wake the device from Power Mode 9.
5	CMP0WEN	**Comparator 0 Wake Enable.** When set to 1, a Comparator 0 event will wake the device from Power Mode 9.
4:2	Reserved	Must write reset value.
1	RTC0AWEN	**RTC0 Alarm Wake Enable.** When set to 1, an RTC0 alarm event will wake the device from Power Mode 9. This alarm event can occur from any of the three RTC0 alarms.
0	RTC0FWEN	**RTC0 Fail Wake Enable.** When set to 1, an RTC0 fail event will wake the device from Power Mode 9.

Register 9.5. PMU0_WAKESTATUS: Wake Source Status

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved							RSTWF	LPT0WF	PWAKEWF	CMP0WF	Reserved			RTC0AWF	RTC0FWF
Type	R							R	R	R	R	R			R	R
Reset	0	0	0	0	0	0	0	X	X	X	0	0	0	0	X	X

Register ALL Access Address

PMU0_WAKESTATUS = 0x4004_8040

Table 9.7. PMU0_WAKESTATUS Register Bit Descriptions

Bit	Name	Function
31:9	Reserved	Must write reset value.
8	RSTWF	**Reset Pin Wake Flag.** When set to 1, this flag indicates that the $\overline{\text{RESET}}$ pin woke the device. Firmware must clear this flag.
7	LPT0WF	**Low Power Timer Wake Flag.** When set to 1, this flag indicates that a LPTIMER0 event woke the device. Firmware must clear this flag.
6	PWAKEWF	**Pin Wake Wake Flag.** When set to 1, this flag indicates that a Pin Wake event woke the device. Firmware must clear this flag.
5	CMP0WF	**Comparator 0 Wake Flag.** When set to 1, this flag indicates that a Comparator 0 event woke the device. Firmware must clear this flag.
4:2	Reserved	Must write reset value.
1	RTC0AWF	**RTC0 Alarm Wake Flag.** When set to 1, this flag indicates that an RTC0 alarm event woke the device. This alarm event can occur from any of the three RTC0 alarms. Firmware must clear this flag.
0	RTC0FWF	**RTC0 Fail Wake Flag.** When set to 1, this flag indicates that an RTC0 fail event woke the device. Firmware must clear this flag.

Register 9.6. PMU0_PWEN: Pin Wake Pin Enable

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	PW15EN	PW14EN	PW13EN	PW12EN	PW11EN	PW10EN	PW9EN	PW8EN	PW7EN	PW6EN	PW5EN	PW4EN	PW3EN	PW2EN	PW1EN	PW0EN
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

PMU0_PWEN = 0x4004_8050

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 9.8. PMU0_PWEN Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15	PW15EN	**WAKE.15 Enable.** When set to 1, this bit enables the WAKE.15 signal to wake the device.
14	PW14EN	**WAKE.14 Enable.** When set to 1, this bit enables the WAKE.14 signal to wake the device.
13	PW13EN	**WAKE.13 Enable.** When set to 1, this bit enables the WAKE.13 signal to wake the device.
12	PW12EN	**WAKE.12 Enable.** When set to 1, this bit enables the WAKE.12 signal to wake the device.
11	PW11EN	**WAKE.11 Enable.** When set to 1, this bit enables the WAKE.11 signal to wake the device.
10	PW10EN	**WAKE.10 Enable.** When set to 1, this bit enables the WAKE.10 signal to wake the device.
9	PW9EN	**WAKE.9 Enable.** When set to 1, this bit enables the WAKE.9 signal to wake the device.
8	PW8EN	**WAKE.8 Enable.** When set to 1, this bit enables the WAKE.8 signal to wake the device.
7	PW7EN	**WAKE.7 Enable.** When set to 1, this bit enables the WAKE.7 signal to wake the device.

Table 9.8. PMU0_PWEN Register Bit Descriptions

Bit	Name	Function
6	PW6EN	**WAKE.6 Enable.** When set to 1, this bit enables the WAKE.6 signal to wake the device.
5	PW5EN	**WAKE.5 Enable.** When set to 1, this bit enables the WAKE.5 signal to wake the device.
4	PW4EN	**WAKE.4 Enable.** When set to 1, this bit enables the WAKE.4 signal to wake the device.
3	PW3EN	**WAKE.3 Enable.** When set to 1, this bit enables the WAKE.3 signal to wake the device.
2	PW2EN	**WAKE.2 Enable.** When set to 1, this bit enables the WAKE.2 signal to wake the device.
1	PW1EN	**WAKE.1 Enable.** When set to 1, this bit enables the WAKE.1 signal to wake the device.
0	PW0EN	**WAKE.0 Enable.** When set to 1, this bit enables the WAKE.0 signal to wake the device.

Register 9.7. PMU0_PWPOL: Pin Wake Pin Polarity Select

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	PW15POL	PW14POL	PW13POL	PW12POL	PW11POL	PW10POL	PW9POL	PW8POL	PW7POL	PW6POL	PW5POL	PW4POL	PW3POL	PW2POL	PW1POL	PW0POL
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

PMU0_PWPOL = 0x4004_8060

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 9.9. PMU0_PWPOL Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15	PW15POL	**WAKE.15 Polarity Select.** If PW15EN is set, this bit selects the logic level of WAKE.15 that will be used in the wake comparison.
14	PW14POL	**WAKE.14 Polarity Select.** If PW14EN is set, this bit selects the logic level of WAKE.14 that will be used in the wake comparison.
13	PW13POL	**WAKE.13 Polarity Select.** If PW13EN is set, this bit selects the logic level of WAKE.13 that will be used in the wake comparison.
12	PW12POL	**WAKE.12 Polarity Select.** If PW12EN is set, this bit selects the logic level of WAKE.12 that will be used in the wake comparison.
11	PW11POL	**WAKE.11 Polarity Select.** If PW11EN is set, this bit selects the logic level of WAKE.11 that will be used in the wake comparison.
10	PW10POL	**WAKE.10 Polarity Select.** If PW10EN is set, this bit selects the logic level of WAKE.10 that will be used in the wake comparison.

Table 9.9. PMU0_PWPOL Register Bit Descriptions

Bit	Name	Function
9	PW9POL	**WAKE.9 Polarity Select.** If PW9EN is set, this bit selects the logic level of WAKE.9 that will be used in the wake comparison.
8	PW8POL	**WAKE.8 Polarity Select.** If PW8EN is set, this bit selects the logic level of WAKE.8 that will be used in the wake comparison.
7	PW7POL	**WAKE.7 Polarity Select.** If PW7EN is set, this bit selects the logic level of WAKE.7 that will be used in the wake comparison.
6	PW6POL	**WAKE.6 Polarity Select.** If PW6EN is set, this bit selects the logic level of WAKE.6 that will be used in the wake comparison.
5	PW5POL	**WAKE.5 Polarity Select.** If PW5EN is set, this bit selects the logic level of WAKE.5 that will be used in the wake comparison.
4	PW4POL	**WAKE.4 Polarity Select.** If PW4EN is set, this bit selects the logic level of WAKE.4 that will be used in the wake comparison.
3	PW3POL	**WAKE.3 Polarity Select.** If PW3EN is set, this bit selects the logic level of WAKE.3 that will be used in the wake comparison.
2	PW2POL	**WAKE.2 Polarity Select.** If PW2EN is set, this bit selects the logic level of WAKE.2 that will be used in the wake comparison.
1	PW1POL	**WAKE.1 Polarity Select.** If PW1EN is set, this bit selects the logic level of WAKE.1 that will be used in the wake comparison.
0	PW0POL	**WAKE.0 Polarity Select.** If PW0EN is set, this bit selects the logic level of WAKE.0 that will be used in the wake comparison.

9.4. PMU0 Register Memory Map

Table 9.10. PMU0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PMU0_CONTROL	0x4004_8000	ALL | SET | CLR	Reserved																											PMUASLPEN	PWAKEEN	PINLPEN	PERILPEN	WAKECLR
PMU0_CONFIG	0x4004_8010	ALL | SET | CLR	Reserved																LPT0REN	PWAKEREN	CMP0REN	Reserved			RTC0AREN	RTC0FREN	Reserved							
PMU0_STATUS	0x4004_8020	ALL | SET | CLR	Reserved																												PORF	PWAKEF	PM9EF	
PMU0_WAKEEN	0x4004_8030	ALL | SET | CLR	Reserved																						RSTWEN	LPT0WEN	PWAKEWEN	CMP0WEN	Reserved			RTC0AWEN	RTC0FWEN	

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

Table 9.10. PMU0 Memory Map

Register Name	PMU0_WAKESTATUS	PMU0_PWEN	PMU0_PWPOL
ALL Address	0x4004_8040	0x4004_8050	0x4004_8060
Access Methods	ALL	ALL | SET | CLR	ALL | SET | CLR
Bit 31	Reserved	Reserved	Reserved
Bit 30			
Bit 29			
Bit 28			
Bit 27			
Bit 26			
Bit 25			
Bit 24			
Bit 23			
Bit 22			
Bit 21			
Bit 20			
Bit 19			
Bit 18			
Bit 17			
Bit 16			
Bit 15		PW15EN	PW15POL
Bit 14		PW14EN	PW14POL
Bit 13		PW13EN	PW13POL
Bit 12		PW12EN	PW12POL
Bit 11		PW11EN	PW11POL
Bit 10		PW10EN	PW10POL
Bit 9		PW9EN	PW9POL
Bit 8	RSTWF	PW8EN	PW8POL
Bit 7	LPT0WF	PW7EN	PW7POL
Bit 6	PWAKEWF	PW6EN	PW6POL
Bit 5	CMP0WF	PW5EN	PW5POL
Bit 4	Reserved	PW4EN	PW4POL
Bit 3		PW3EN	PW3POL
Bit 2		PW2EN	PW2POL
Bit 1	RTC0AWF	PW1EN	PW1POL
Bit 0	RTC0FWF	PW0EN	PW0POL

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

10. Core Voltage Regulator (LDO0)

This section describes the core voltage regulator (LDO) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

10.1. Core Voltage Regulator Features

The core voltage regulator includes the following features:

- Regulates internal supply for core, memory and peripherals.
- Two bias settings for power savings during low-frequency operation (AHB Clock ≤ 2.5 MHz).

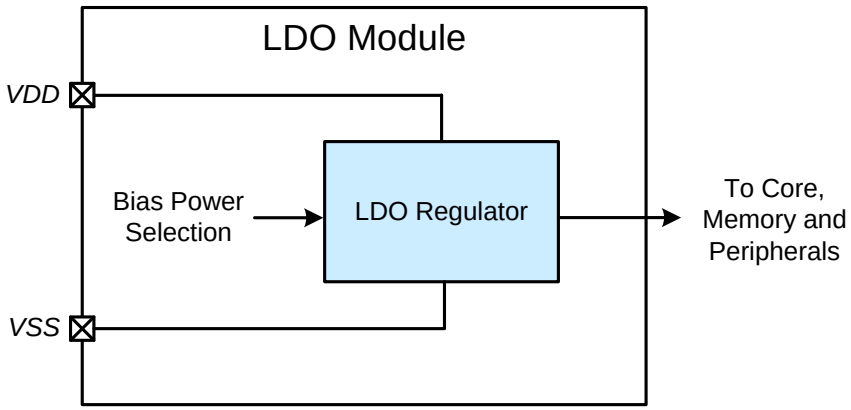


Figure 10.1. Core Regulator Block Diagram

10.2. Functional Description

The core voltage regulator consists of an LDO regulator with a programmable bias current setting. It is powered from the device VDD and VSS supply pins. By default, the core voltage regulator starts up in the high bias setting, allowing the device to quickly come out of reset and begin executing firmware. If the frequency of the AHB clock used in the application is less than or equal to 2.5 MHz, the LDOIBIAS bit in the CONTROL register can be set to 1 for additional power savings. LDOIBIAS should always be cleared to 0 when operating the AHB clock above 2.5 MHz.

10.3. LDO0 Registers

This section contains the detailed register descriptions for LDO0 registers.

Register 10.1. LDO0_CONTROL: Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	LDOAEN	Reserved														
Type	RW	R														
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved														LDOIBIAS	Reserved
Type	R														RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

LDO0_CONTROL = 0x4003_9000

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 10.1. LDO0_CONTROL Register Bit Descriptions

Bit	Name	Function
31	LDOAEN	**LDO Analog Enable.** This bit enables/disables an output from the LDO to the VREF0, LPOSC0, USBOSC and PLLOSC0 blocks. It must be set to 1 when any of these functions is enabled. Clearing this bit to 0 when not using the affected circuits will reduce the current consumption of the device by approximately 15 μA. 0: LDO0 analog output disabled. 1: LDO0 analog output enabled.
30:2	Reserved	Must write reset value.
1	LDOIBIAS	**LDO Bias Current Selection.** Selects between two bias levels for the internal LDO regulator. 0: Select high bias. 1: Select low bias (AHB frequency ≤ 2.5 MHz).
0	Reserved	Must write reset value.

10.4. LDO0 Register Memory Map

Table 10.2. LDO0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
LDO0_CONTROL	0x4003_9000	ALL | SET | CLR	LDOAEN	Reserved																													LDOIBIAS	Reserved

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

11. Device Identification (DEVICEID0) and Universally Unique Identifier

This section describes the Device Identification (DEVICEID) registers and the pre-programmed Unique Identifier, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

11.1. Device ID Features

In full production devices, the Device ID module consists of four static registers which include the following device identification information:

- Silicon Laboratories part number
- Package identification
- Silicon revision
- Custom device ID

Note: For pre-production and early engineering devices, the registers associated with the DEVICEID module contained a 124-bit unique number, as well as the 4-bit REVID field to indicate the silicon revision. The full 128-bit unique identifier is still available in all versions of silicon. See “11.2. Universally Unique Identifier (UUID)” for more details.

11.1.1. Device Identification Encoding

The device identification information is encoded into the DEVICEID0-DEVICEID3 registers. DEVICEID0 contains revision and packaging information. DEVICEID2 and DEVICEID1 contain the part number string, excluding the revision, packaging and “Si” prefix information. DEVICEID3 is set to all zeros for standard-catalog products, and a unique product number for custom-numbered products. FIGURE details the DEVICEID register encoding for an example product (SiM3U167-B-GQ).

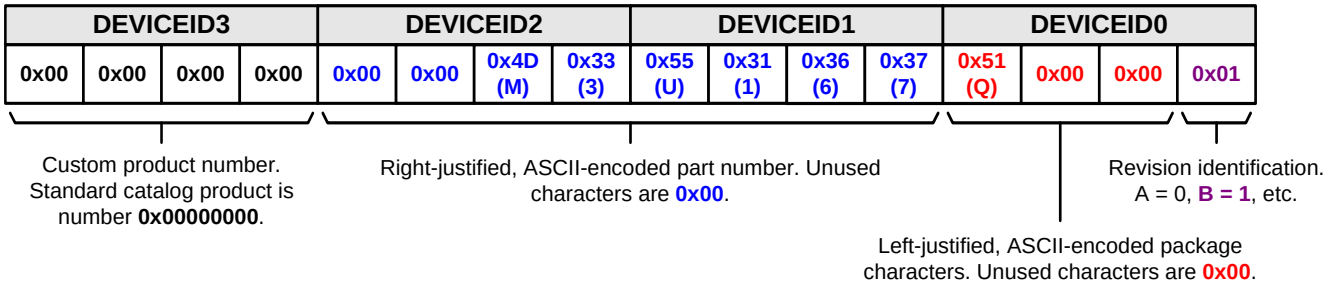


Figure 11.1. Example DEVICEID encoding for part number SiM3U167-B-GQ

11.2. Universally Unique Identifier (UUID)

A128-bit universally unique identifier (UUID) is pre-programmed into all devices. The UUID resides in an area of flash memory which cannot be erased or written in the end application. The UUID can be read by firmware or through the debug port at addresses 0x00040380 through 0x00040383.

11.3. DEVICEID0 Registers

This section contains the detailed register descriptions for DEVICEID0 registers.

Register 11.1. DEVICEID0_DEVICEID0: Device ID Word 0

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	PACKID[23:8]															
Type	RW															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	PACKID[7:0]								Reserved				REVID			
Type	RW								RW				RW			
Reset	X	X	X	X	X	X	X	X	0	0	0	0	X	X	X	X

Register ALL Access Address

DEVICEID0_DEVICEID0 = 0x4004_90C0

Table 11.1. DEVICEID0_DEVICEID0 Register Bit Descriptions

Bit	Name	Function
31:8	PACKID	**Package ID.** This field describes the device package. For standard catalog products, the most significant byte contains the package type letter encoded in ASCII. The unused bytes are encoded as 0x00.
7:4	Reserved	Must write reset value.
3:0	REVID	**Revision ID.** This field provides the revision information for the device. 0000: Revision A. 0001: Revision B. 0010-1111: Reserved.

Register 11.2. DEVICEID0_DEVICEID1: Device ID Word 1

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	DEVICEID1[31:16]															
Type	RW															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	DEVICEID1[15:0]															
Type	RW															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Register ALL Access Address DEVICEID0_DEVICEID1 = 0x4004_90D0																

Table 11.2. DEVICEID0_DEVICEID1 Register Bit Descriptions

Bit	Name	Function
31:0	DEVICEID1	**Device ID 1.** The DEVICEID1 and DEVICEID2 registers contain the ASCII-encoded part number for the silicon, excluding the prefix "Si". The part number is right-justified within the two registers, and any unused characters are set to 0x00.

Register 11.3. DEVICEID0_DEVICEID2: Device ID Word 2

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	DEVICEID2[31:16]															
Type	RW															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	DEVICEID2[15:0]															
Type	RW															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Register ALL Access Address DEVICEID0_DEVICEID2 = 0x4004_90E0																

Table 11.3. DEVICEID0_DEVICEID2 Register Bit Descriptions

Bit	Name	Function
31:0	DEVICEID2	**Device ID 2.** The DEVICEID1 and DEVICEID2 registers contain the ASCII-encoded part number for the silicon, excluding the prefix "Si". The part number is right-justified within the two registers, and any unused characters are set to 0x00.

Register 11.4. DEVICEID0_DEVICEID3: Device ID Word 3

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	DEVICEID3[31:16]															
Type	RW															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	DEVICEID3[15:0]															
Type	RW															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Register ALL Access Address DEVICEID0_DEVICEID3 = 0x4004_90F0																

Table 11.4. DEVICEID0_DEVICEID3 Register Bit Descriptions

Bit	Name	Function
31:0	DEVICEID3	**Device ID 3.** For standard products the DEVICEID3 register will read back all 0's. For custom-programmed MCUs, this register may contain the number assigned to the custom product.

11.4. DEVICEID0 Register Memory Map

Table 11.5. DEVICEID0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DEVICEID0_DEVICEID0	0x4004_90C0	ALL	PACKID																								Reserved				REVID			
DEVICEID0_DEVICEID1	0x4004_90D0	ALL	DEVICEID1																															

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

Table 11.5. DEVICEID0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DEVICEID0_DEVICEID2	0x4004_90E0	ALL	DEVICEID2																															
DEVICEID0_DEVICEID3	0x4004_90F0	ALL	DEVICEID3																															

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

12. Advanced Encryption Standard (AES0)

This section describes the Advanced Encryption Standard (AES) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the AES block, which is used by all device families covered in this document.

12.1. AES Features

The AES module includes the following features:

- Operates on 4-word (16-byte) blocks.
- Supports key sizes of 128, 192, and 256 bits for both encryption and decryption.
- Generates the round key for decryption operations.
- All cipher operations can be performed without any firmware intervention for a set of 4-word blocks (up to 32 kB).
- Support for various chained and stream-ciphering configurations with XOR paths on both the input and output.
- Internal 4-word FIFOs to facilitate DMA operations.
- Integrated key storage.
- Hardware acceleration for Cipher-Block Chaining (CBC) and Counter (CTR) algorithms utilizing integrated counter-block generation and previous-block caching.

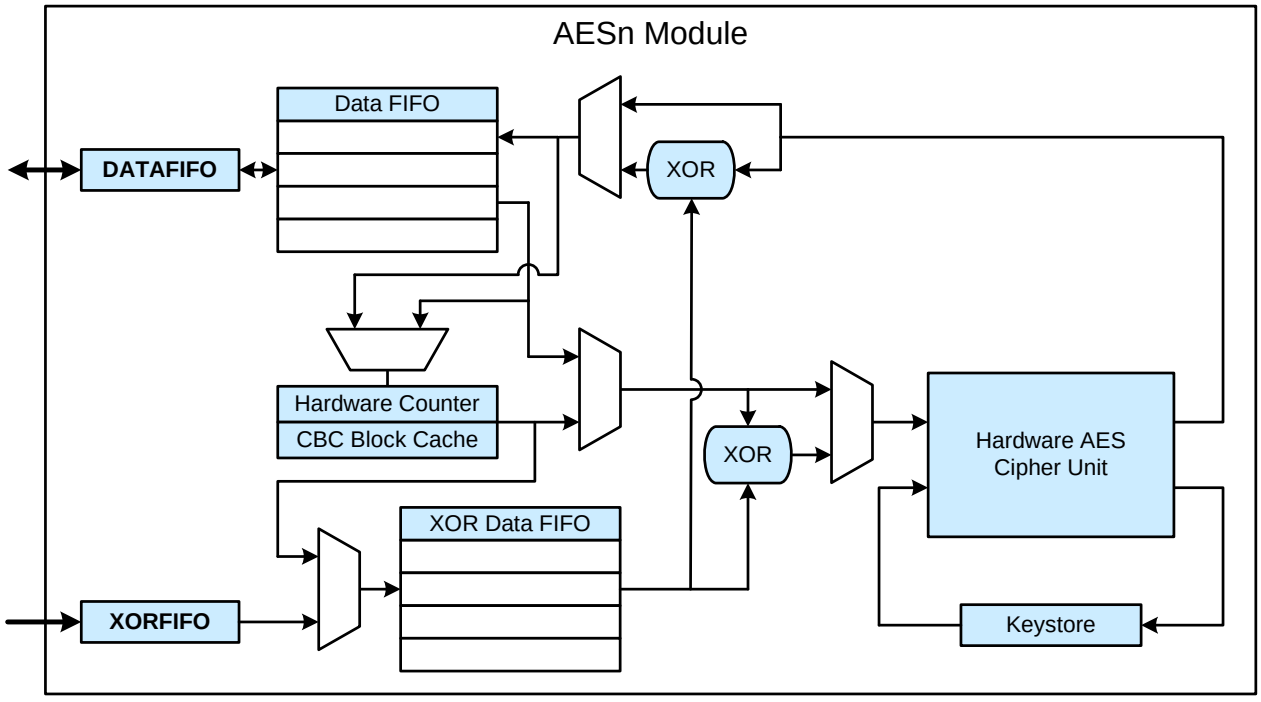


Figure 12.1. AES0 Block Diagram

12.2. Overview

The AES block cipher is a symmetric key encryption algorithm. Symmetric key encryption relies on secret keys that are known by both the sender and receiver. The decryption key may be obtained using a simple transformation of the encryption key. AES is not a public key encryption algorithm.

The AES block cipher uses a fixed 4-word (16-byte) block size. Data segments less than 4 words in length must be padded with zeros to fill the entire block.

Since symmetric key encryption relies on secret keys, the security of the data can only be protected if the key remains secret. If the encryption key is stored in flash memory, then the entire flash should be locked to ensure the encryption key cannot be discovered.

The hardware counter in the AES0 module can be programmed with an initial value using the HWCTR register. Similarly, the hardware keystore can be programmed with an initial value using the HWKEY register.

The basic AES block cipher is implemented in hardware. The integrated hardware acceleration for cipher block chaining (CBC) and counter (CTR) algorithms results in identical performance, memory bandwidth, and memory footprint between the most basic electronic codebook (ECB) algorithm and these more complex algorithms. This hardware accelerator provides performance that is much faster than a software implementation, which translates to more bandwidth available for other functions or a power savings for low-power applications.

12.2.1. Enabling the AES0 Module

Immediately following any device reset, the RESET bit is set to 1 to save power. All register bits except RESET are reset using the combined device reset and the RESET bit. To use the AES0 module, firmware must first clear the RESET bit before initializing the registers.

12.3. Interrupts

The AES0 interrupt flags are located in the STATUS register. The associated interrupt enable bits are located in the CONTROL register. An AES0 completion interrupt can be generated if OCIEN is set to 1 whenever an encryption or decryption operation is complete. An AES0 error interrupt can be generated whenever an input/output data FIFO overrun (DORF = 1) or underrun (DURF = 1) error occurs, or when an XOR data FIFO overrun (XORF = 1) occurs. The completion interrupt should only be used in conjunction with software mode (SWMDEN bit is set to 1) and not with DMA operations, where the DMA completion interrupt should be used. The error interrupt should always be enabled (ERRIEN = 1), even when using the DMA with the AES module.

12.4. Debug Mode

Firmware can clear the DBGMD bit to force the AES0 module to halt on a debug breakpoint. Setting the DBGMD bit forces the module to continue operating while the core halts in debug mode.

12.5. DMA Configuration and Usage

A DMA channel may be used to transfer data for the input/output data FIFO or the XOR data FIFO. The AES module FIFOs only support word reads and writes. Each DMA transfer must consist of 4 words (16 bytes). To write to the input/output data FIFO, the DMA must move data from the source location in memory to the internal DATAFIFO register in non-incrementing mode. To read from the input/output data FIFO, the DMA must move data from the internal DATAFIFO register in non-incrementing mode to the destination location in memory. For the XOR data FIFO, the DMA must move data from the source location in memory to the internal XORFIFO register in non-incrementing mode. Firmware should only directly access the DATAFIFO and XORFIFO registers in software mode (SWMDEN bit is set to 1). AES0 FIFOs targeted by the DMA module should not be directly written to or read from.

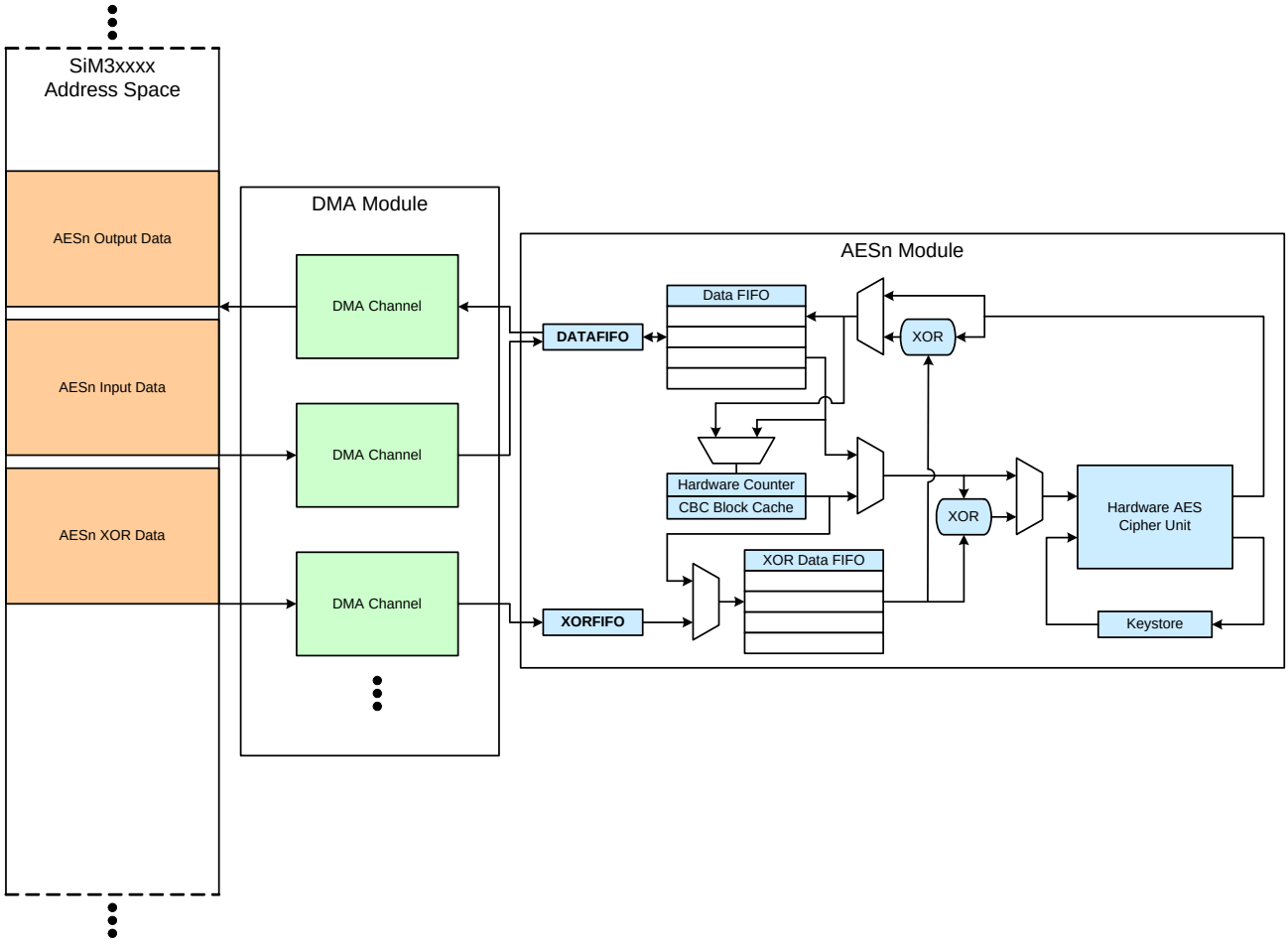


Figure 12.2. AES DMA Configuration

12.5.1. DMA and Interrupts

If a DMA channel is enabled for one of the FIFOs, the AES operation complete and error interrupts will not be suppressed. In software mode (SWMDEN = 1) the AES operation complete interrupt should be used, and in DMA mode (SWMDEN = 0) the DMA complete interrupt should be used. The error interrupt should be used in both software and DMA modes to notify the firmware of error conditions.

12.5.2. General DMA Transfer Setup

For the AES module, the DMA channels have these common settings:

- Source size (SRCSIZE) and destination size (DSTSIZE) are 2 for a word transfer.
- Number of transfers is (4 x N) – 1, where N is the number of 4-byte words.
- RPOWER = 2 (4 data transfers per transaction).
- The size of all memory buffers (input, output, and XOR) modulo 16 bytes must equal 0, so an even number of 4-word transfers must occur.

The input DMA channel should be programmed as follows:

- Destination end pointer set to the DATAFIFO register.
- Source end pointer set to the plain or cipher text input buffer address location + 16 x N – 4, where N is the number of blocks.
- The DSTAIMD field should be set to 011b for no increment.
- The SRCAIMD field should be set to 010b for word increments.

The output DMA channel should be programmed as follows:

- Destination end pointer set to the plain or cipher text output buffer address location + 16 x N – 4, where N is the number of blocks.
- Source end pointer set to the DATAFIFO register.
- The DSTAIMD field should be set to 010b for word increments.
- The SRCAIMD field should be set to 011b for no increment.

The XOR DMA channel should be programmed as follows:

- Destination end pointer set to the XORFIFO register.
- Source end pointer set to the In XOR buffer address location + 16 x N – 4, where N is the number of blocks.
- The DSTAIMD field should be set to 011b for no increment.
- The SRCAIMD field should be set to 010b for word increments.

To start a DMA operation with the AES module out of any device reset:

1. Set up the DMA channels for the input/output and XOR FIFOs depending on the desired cipher algorithm.
2. Clear the soft reset bit (RESET) to 0.
3. Configure the AES peripheral operation in the CONTROL, XFRSIZE, HWKEYx, and HWCTRx registers.
4. Start the AES operation by writing a 1 to XFRSTA.
5. Wait for the DMA completion interrupt.

12.6. Using the AES0 Module for Electronic Codebook (ECB)

The electronic codebook (ECB) cipher algorithm is the most basic block cipher mode since each cipher text output is only a function of its corresponding plain text input and the encryption key. This algorithm is shown in Figure 12.3.

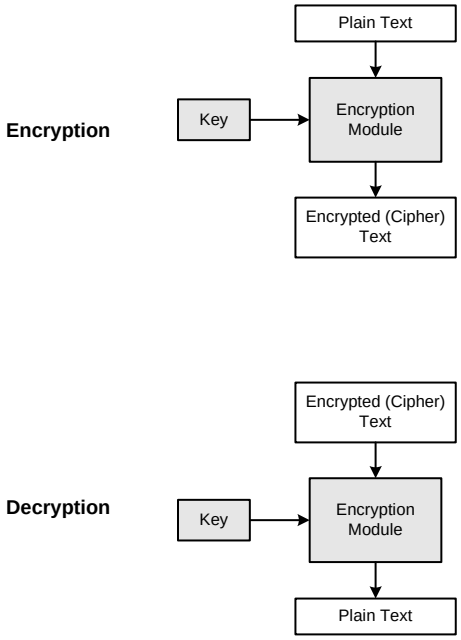


Figure 12.3. Electronic Codebook (ECB) Algorithm Diagram

The block diagram of the AES module performing this algorithm (encryption and decryption) is shown in Figure 12.4. The active data paths in this mode are shown in red.

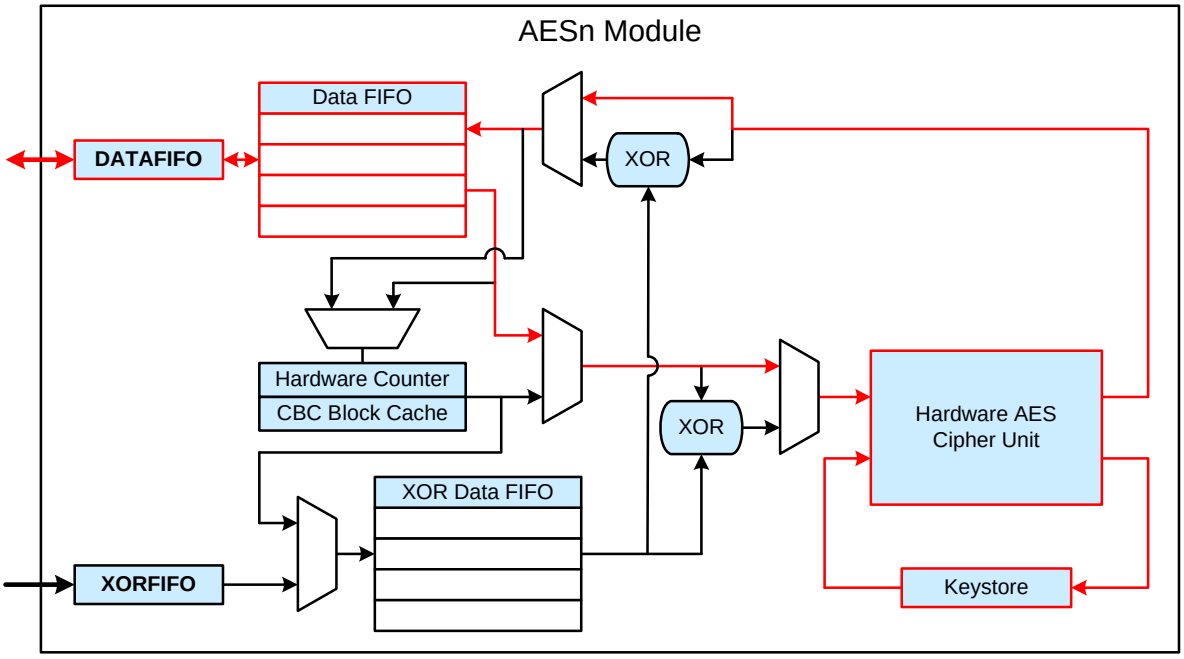


Figure 12.4. Electronic Codebook (ECB) AES Module Block Diagram—Encryption and Decryption

12.6.1. Configuring the DMA for ECB Encryption

To use the DMA with ECB encryption, the DMA and AES modules should be configured as follows:

DMA Input Channel:

1. Destination end pointer set to the DATAFIFO register.
2. Source end pointer set to the plain text input buffer address location + 16 x N – 4, where N is the number of blocks.

DMA Output Channel:

1. Destination end pointer set to the cipher text output buffer address location + 16 x N – 4, where N is the number of blocks.
2. Source end pointer set to the DATAFIFO register.

AES Module:

1. The XFRSIZE register should be set to N-1, where N is the number of 4-word blocks.
2. The HWKEYx registers should be written with the desired key in little endian format.
3. The CONTROL register should be set as follows:
   a. ERRIEN set to 1.
   b. KEYSIZE set to the appropriate number of bits for the key.
   c. EDMD set to 1 for encryption.
   d. KEYCPEN set to 1 to enable key capture at the end of the transaction.
   e. The HCBCEN, HCTREN, XOREN, BEN, SWMDEN bits should all be cleared to 0.

Once the DMA and AES settings have been set, the transfer should be started by writing 1 to the XFRSTA bit. When the encryption process finishes, the decryption key will be available in the HWKEYx registers.

12.6.2. Configuring the DMA for ECB Decryption

To use the DMA with ECB decryption, the DMA and AES modules should be configured as follows:

DMA Input Channel:

1. Destination end pointer set to the DATAFIFO register.
2. Source end pointer set to the cipher text input buffer address location + 16 x N – 4, where N is the number of blocks.

DMA Output Channel:

1. Destination end pointer set to the plain text output buffer address location + 16 x N – 4, where N is the number of blocks.
2. Source end pointer set to the DATAFIFO register.

AES Module:

1. The XFRSIZE register should be set to N-1, where N is the number of 4-word blocks.
2. The HWKEYx registers should be written with decryption key value (automatically generated in the HWKEYx registers after the encryption process).
3. The CONTROL register should be set as follows:
   a. ERRIEN set to 1
   b. KEYSIZE set to the appropriate number of bits for the key.
   c. EDMD set to 1 for encryption.
   d. KEYCPEN set to 1 to enable key capture at the end of the transaction.
   e. The HCBCEN, HCTREN, XOREN, BEN, SWMDEN bits should all be cleared to 0.

Once the DMA and AES settings have been set, the transfer should be started by writing 1 to the XFRSTA bit.

12.7. Using the AES0 Module for Cipher Block Chaining (CBC)

The cipher block chaining (CBC) cipher algorithm significantly improves the strength of basic ECB encryption by making each block encryption be a function of the previous block in addition to the current plain text and key. This algorithm is shown in Figure 12.5.

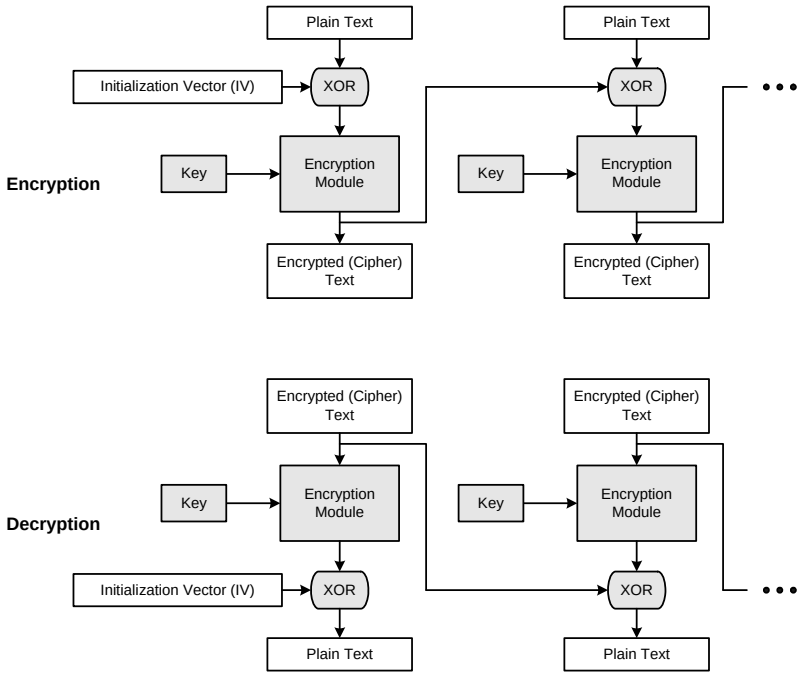


Figure 12.5. Cipher Block Chaining (CBC) Algorithm Diagram

The AES module can perform this algorithm by using a DMA channel or by using the AES hardware to feed the XOR data FIFO. Both of these methods are discussed.

12.7.1. Hardware XOR Channel Cipher Block Chaining

The AES0 module has a hardware chaining mode that accelerates the execution of the CBC algorithm. The module can reuse the hardware counter registers to temporarily store the previous result (CT(n-1)) for use with the next block. This eliminates the need for a third DMA channel, freeing it for other use, and reduces the timing to be identical with the counter (CTR) and much-simpler electronic codebook (ECB) algorithms. Any algorithms that use this hardware path, however, should not use the hardware counter (HCTREN set to 1) feature.

The block diagram of the AES module performing this encryption algorithm using the hardware path is shown in Figure 12.6. This decryption algorithm block diagram using the hardware path is shown in Figure 12.7. The active data paths in this mode are shown in red.

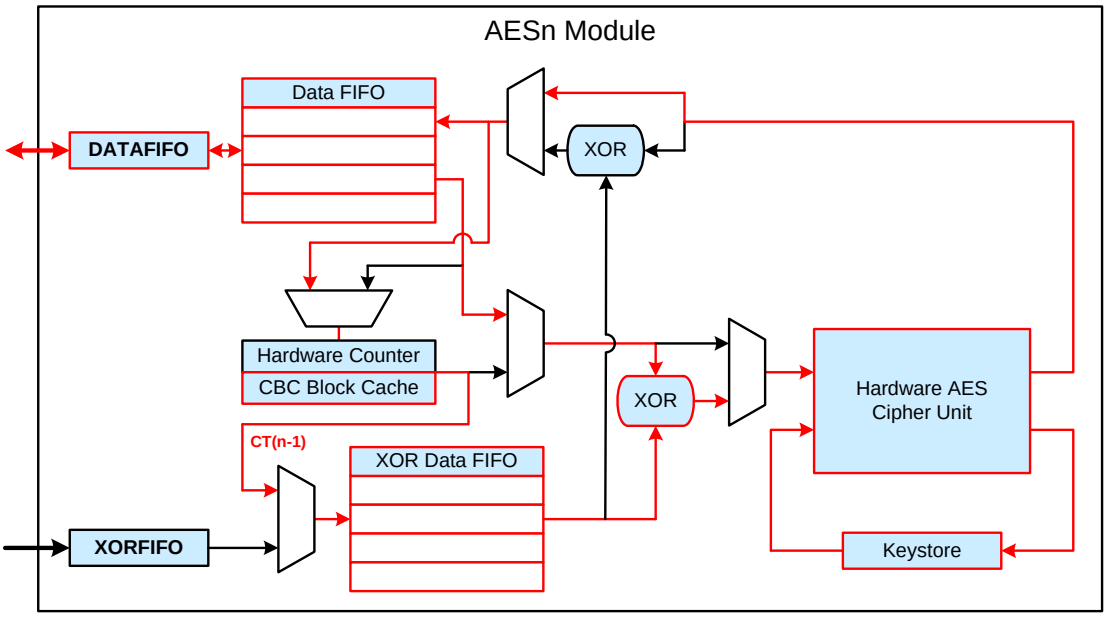


Figure 12.6. Hardware Cipher Block Chaining (CBC) AES Module Block Diagram—Encryption

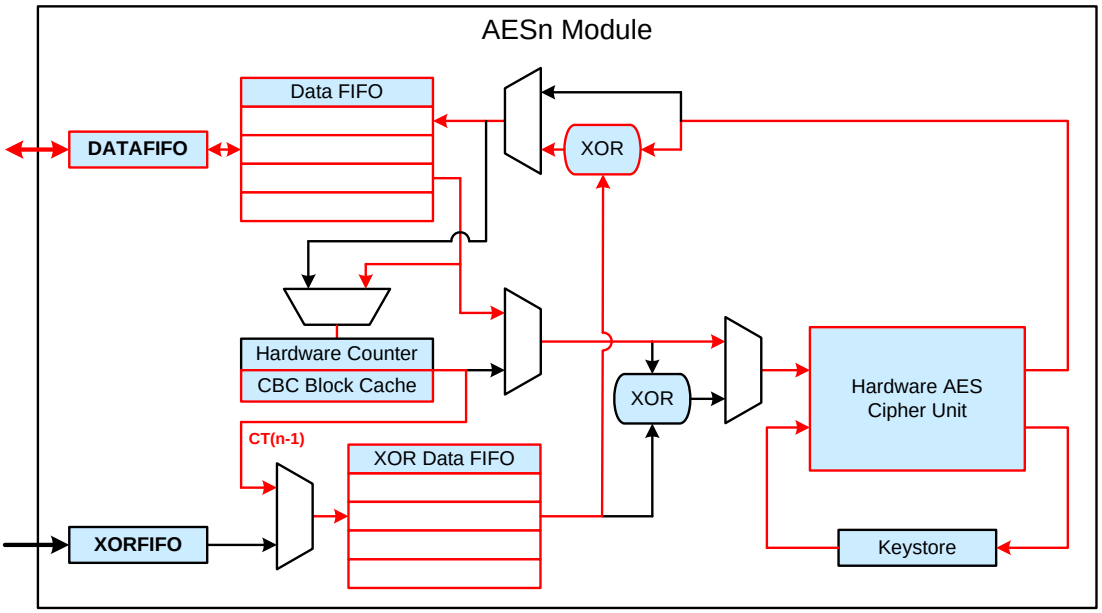


Figure 12.7. Hardware Cipher Block Chaining (CBC) AES Module Block Diagram—Decryption

12.7.1.1. Configuring the DMA for Hardware CBC Encryption

To use the DMA with hardware CBC encryption, the DMA and AES modules should be configured as follows:

DMA Input Channel:

1. Destination end pointer set to the DATAFIFO register.
2. Source end pointer set to the plain text input buffer address location + 16 x N – 4, where N is the number of blocks.

DMA Output Channel:

1. Destination end pointer set to the cipher text output buffer address location + 16 x N – 4, where N is the number of blocks.
2. Source end pointer set to the DATAFIFO register.

Initialization Vector:

The initialization vector should be initialized to the HWCTRx registers.

AES Module:

1. The XFRSIZE register should be set to N-1, where N is the number of 4-word blocks.
2. The HWKEYx registers should be written with the desired key in little endian format.
3. The CONTROL register should be set as follows:
   a. ERRIEN set to 1.
   b. KEYSIZE set to the appropriate number of bits for the key.
   c. XOREN bits set to 01b to enable the XOR input path.
   d. EDMD set to 1 for encryption.
   e. KEYCPEN set to 1 to enable key capture at the end of the transaction.
   f. HCBCEN set to 1 to enable Hardware Cipher Block Chaining mode.
   g. The HCTREN, BEN, SWMDEN bits should all be cleared to 0.

Once the DMA and AES settings have been set, the transfer should be started by writing 1 to the XFRSTA bit. When the encryption process finishes, the decryption key is available in the HWKEYx registers.

12.7.1.2. Configuring the DMA for Hardware CBC Decryption

To use the DMA with Hardware CBC decryption, the DMA and AES modules should be configured as follows:

DMA Input Channel:

1. Destination end pointer set to the DATAFIFO register.
2. Source end pointer set to the cipher text input buffer address location + 16 x N – 4, where N is the number of blocks.

DMA Output Channel:

1. Destination end pointer set to the plain text output buffer address location + 16 x N – 4, where N is the number of blocks.
2. Source end pointer set to the DATAFIFO register.

Initialization Vector:

The initialization vector should be initialized to the HWCTRx registers.

AES Module:

1. The XFRSIZE register should be set to N-1, where N is the number of 4-word blocks.
2. The HWKEYx registers should be written with the desired key in little endian format.
3. The CONTROL register should be set as follows:
   a. ERRIEN set to 1.
   b. KEYSIZE set to the appropriate number of bits for the key.
   c. XOREN set to 10b to enable the XOR output path.
   d. EDMD set to 0 for decryption.
   e. KEYCPEN set to 0 to disable key capture at the end of the transaction.
   f. HCBCEN set to 1 to enable Hardware Cipher Block Chaining mode.
   g. The HCTREN, BEN, SWMDEN bits should all be cleared to 0.

Once the DMA and AES settings have been set, the transfer should be started by writing 1 to the XFRSTA bit.

12.7.2. DMA XOR Channel Cipher Block Chaining

The block diagram of the AES module performing this encryption algorithm is shown in Figure 12.8. The block diagram of the AES module performing this decryption algorithm is shown in Figure 12.9. The active data paths in this mode are shown in red.

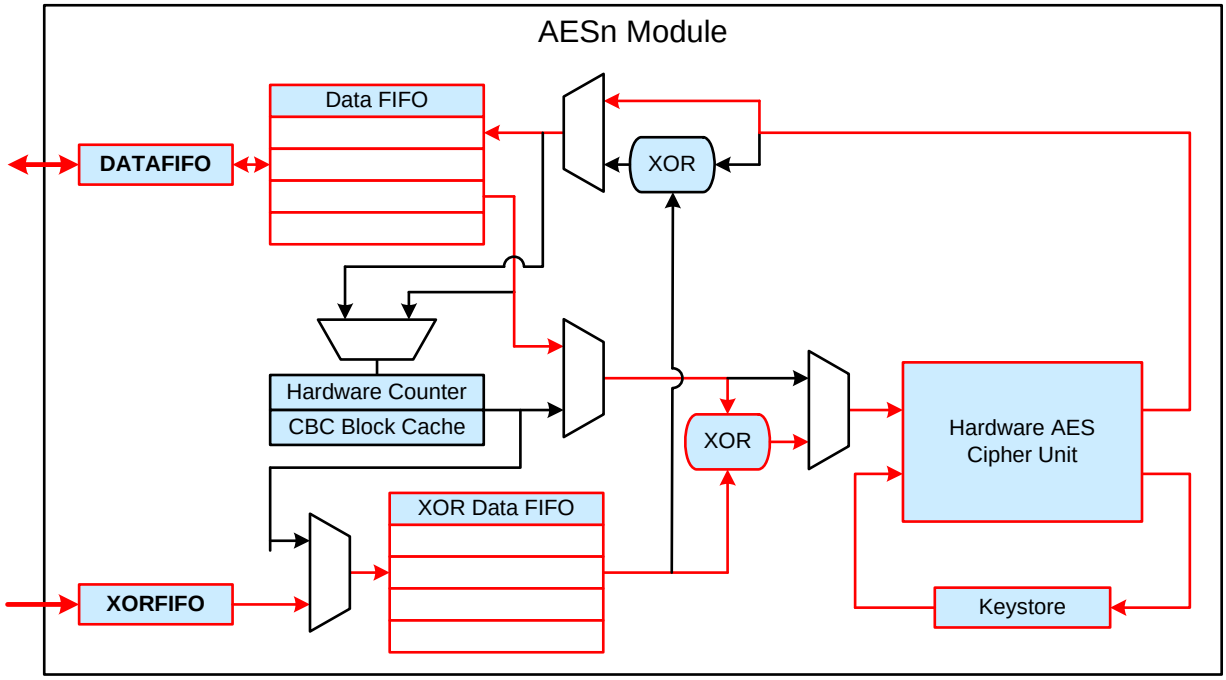


Figure 12.8. Cipher Block Chaining (CBC) AES Module Block Diagram—Encryption

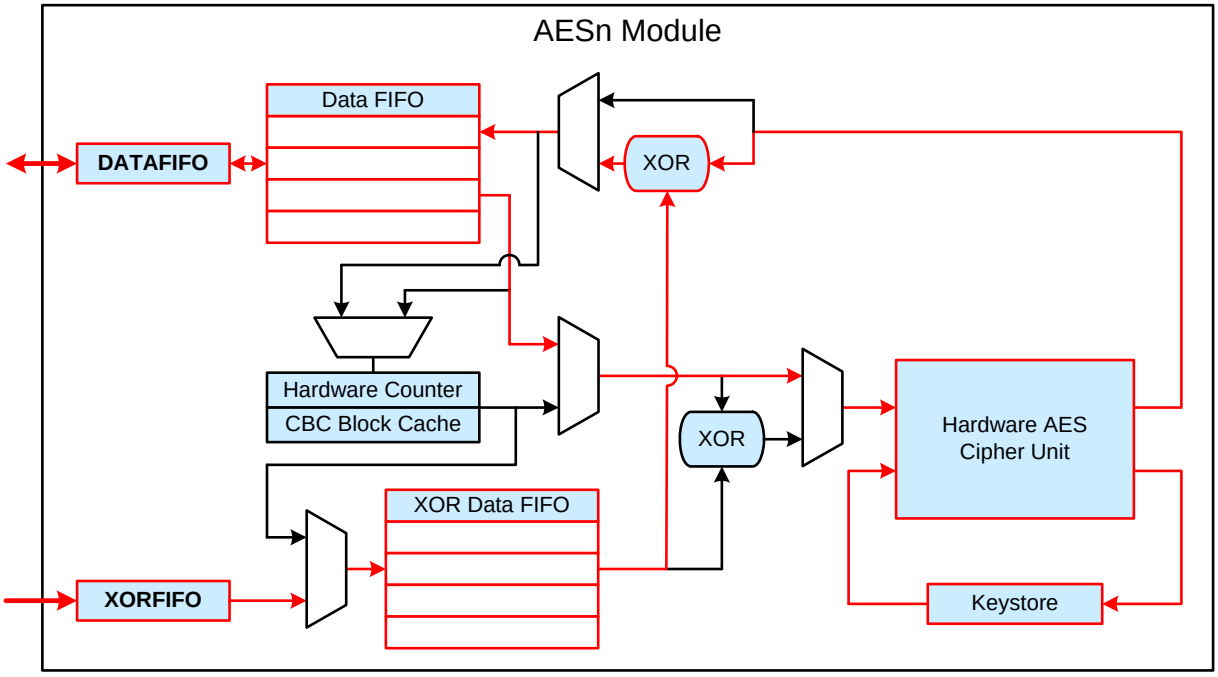


Figure 12.9. Cipher Block Chaining (CBC) AES Module Block Diagram—Decryption

12.7.2.1. Configuring the DMA for CBC Encryption

To use the DMA with CBC encryption, the DMA and AES modules should be configured as follows:

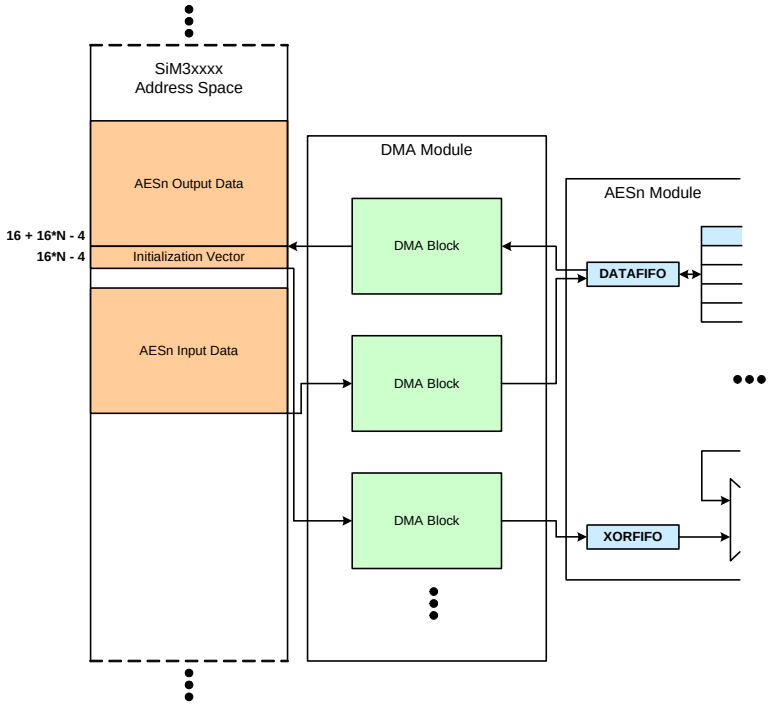


Figure 12.10. DMA XOR Channel Cipher Block Chaining Memory Setup

DMA Input Channel:

1. Destination end pointer set to the DATAFIFO register.
2. Source end pointer set to the plain text input buffer address location + 16 x N – 4, where N is the number of blocks.

DMA Output Channel:

1. Destination end pointer set to the cipher text output buffer address location + 16 + 16 x N – 4, where N is the number of blocks.
2. Source end pointer set to the DATAFIFO register.

DMA XOR Channel:

1. Destination end pointer set to the cipher text output buffer address location + 16 x N – 4, where N is the number of blocks. By programming the output 16 bytes ahead of the XOR channel, the XOR channel is always one block behind (CT(n-1)).
2. Source end pointer set to the DATAFIFO register.

Initialization Vector:

The initialization vector should be initialized to the cipher text output buffer address location + 16 x N – 4.

AES Module:

1. The XFRSIZE register should be set to N-1, where N is the number of 4-word blocks.
2. The HWKEYx registers should be written with the desired key in little endian format.
3. The CONTROL register should be set as follows:
   a. ERRIEN set to 1.
   b. KEYSIZE set to the appropriate number of bits for the key.
   c. XOREN set to 01b to enable the XOR input path.

d. EDMD set to 1 for encryption.

e. KEYCPEN set to 1 to enable key capture at the end of the transaction.

f. The HCBCEN, HCTREN, BEN, SWMDEN bits should all be cleared to 0.

Once the DMA and AES settings have been set, the transfer should be started by writing 1 to the XFRSTA bit. When the encryption process finishes, the decryption key is available in the HWKEYx registers.

12.7.2.2. Configuring the DMA for CBC Decryption

The decryption process is similar to the encryption process, except now the CT(n-1) value is taken before the data is passed through the AES module instead of after, as shown in Figure 12.5. To use the DMA with CBC decryption, the DMA and AES modules should be configured as follows:

DMA Input Channel:

1. Destination end pointer set to the DATAFIFO register.
2. Source end pointer set to the cipher text input buffer address location + 16 + 16 x N – 4, where N is the number of blocks.

DMA Output Channel:

1. Destination end pointer set to the plain text output buffer address location + 16*N – 4, where N is the number of blocks.
2. Source end pointer set to the DATAFIFO register.

DMA XOR Channel:

1. Destination end pointer set to the cipher text input buffer address location + 16 x N – 4, where N is the number of blocks. By programming the DMA input 16 bytes ahead of the XOR channel, the XOR channel is always one block behind (CT(n-1)).
2. Source end pointer set to the DATAFIFO register.

Initialization Vector:

The initialization vector should be initialized to the cipher text input buffer address location + 16*N – 4.

AES Module:

1. The XFRSIZE register should be set to N-1, where N is the number of 4-word blocks.
2. The HWKEYx registers should be written with the desired key in little endian format.
3. The CONTROL register should be set as follows:

   a. ERRIEN set to 1.

   b. KEYSIZE set to the appropriate number of bits for the key.

   c. XOREN set to 10b to enable the XOR output path.

   d. EDMD set to 0 for decryption.

   e. KEYCPEN set to 0 to disable key capture at the end of the transaction.

   f. The HCBCEN, HCTREN, BEN, SWMDEN bits should all be cleared to 0.

Once the DMA and AES settings have been set, the transfer should be started by writing 1 to the XFRSTA bit.

12.8. Using the AES0 Module for Counter (CTR)

The counter (CTR) cipher algorithm is a stream cipher mode which improves upon the basic ECB algorithm by adding a third block variable (a counter block in this case). This algorithm is shown in Figure 12.11.

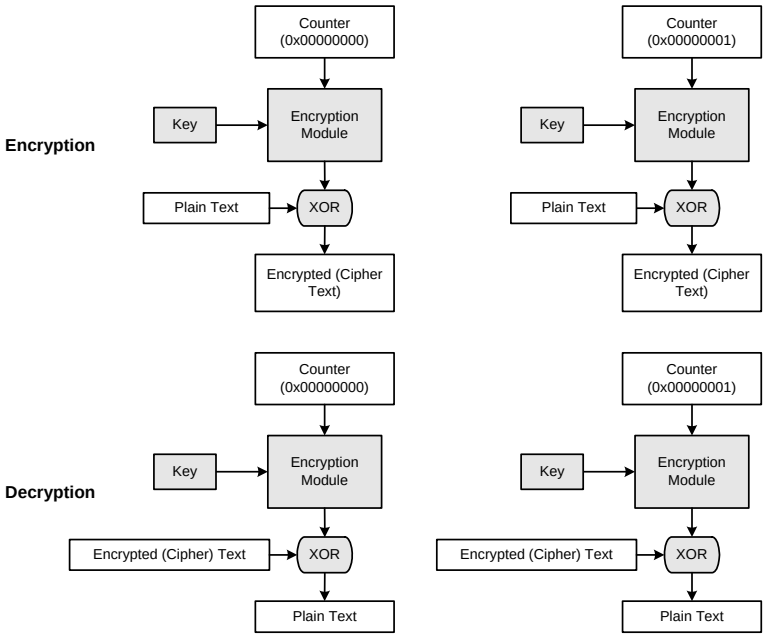


Figure 12.11. Counter (CTR) Algorithm Diagram

Similar to CBC mode, the CTR algorithm requires an initialization vector to encrypt the first block. Unlike CBC, this value is a counter instead of the previous block's output. This counter is implemented in hardware in the AES0 module.

The block diagram of the AES module performing this algorithm (encryption and decryption) is shown in Figure 12.12. The active data paths in this mode are shown in red.

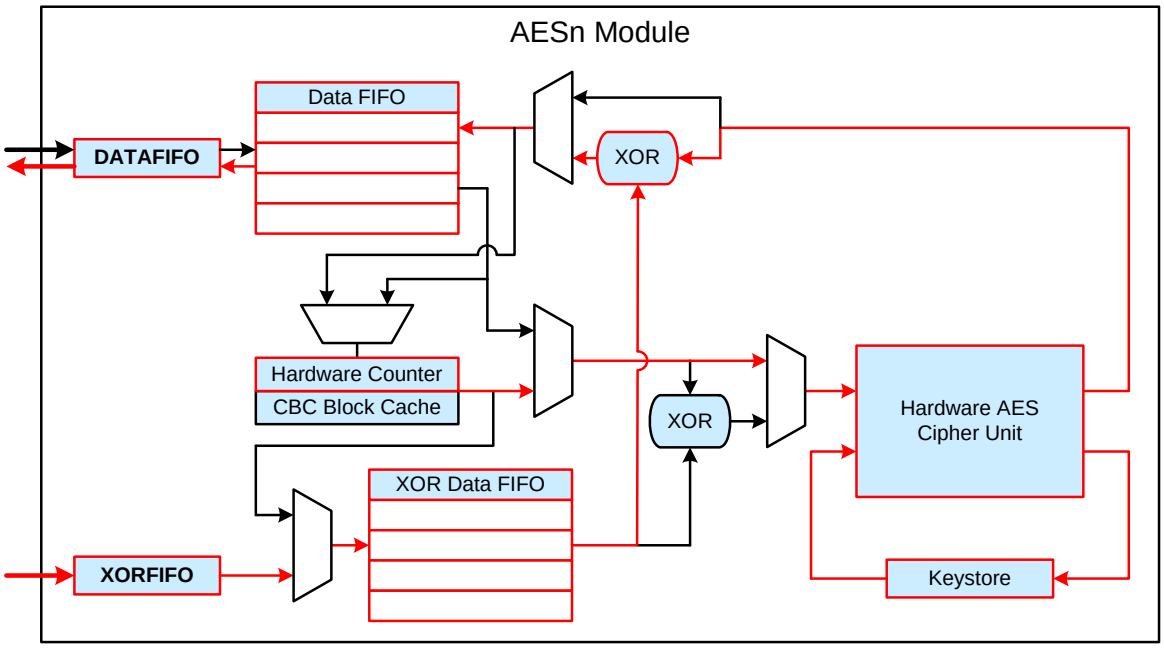


Figure 12.12. Counter (CTR) AES Module Block Diagram—Encryption and Decryption

12.8.1. Configuring the DMA for CTR Encryption

To use the DMA with CTR encryption, the DMA and AES modules should be configured as follows:

DMA Output Channel:

1. Destination end pointer set to the cipher text output buffer address location + 16 x N – 4, where N is the number of blocks.
2. Source end pointer set to the DATAFIFO register.

DMA XOR Channel:

1. Destination end pointer set to the plain text input buffer address location + 16 x N – 4.
2. Source end pointer set to the XORFIFO register.

Initialization Vector:

The initialization vector should be initialized to the HWCTRx registers.

AES Module:

1. The XFRSIZE register should be set to N-1, where N is the number of 4-word blocks.
2. The HWKEYx registers should be written with the desired key in little endian format.
3. The CONTROL register should be set as follows:
   a. ERRIEN set to 1.
   b. KEYSIZE set to the appropriate number of bits for the key.
   c. EDMD set to 1 for encryption.
   d. KEYCPEN set to 0 to disable key capture at the end of the transaction.
   e. HCTREN set to 1 to enable Hardware Counter mode.
   f. XOREN set to 10b to enable the XOR output path.
   g. The HCBCEN, BEN, SWMDEN bits should all be cleared to 0.

Once the DMA and AES settings have been set, the transfer should be started by writing 1 to the XFRSTA bit.

12.8.2. Configuring the DMA for CTR Decryption

This algorithm does not need the key capture output from the encryption process since the encryption and decryption algorithms are exactly the same.

To use the DMA with CTR decryption, the DMA and AES modules should be configured as follows:

DMA Output Channel:

1. Destination end pointer set to the plain text input buffer address location + 16 x N – 4, where N is the number of blocks.
2. Source end pointer set to the DATAFIFO register.

DMA XOR Channel:

1. Destination end pointer set to the cipher text output buffer address location + 16 x N – 4.
2. Source end pointer set to the XORFIFO register.

Initialization Vector:

The initialization vector should be initialized to the HWCTRx registers.

AES Module:

1. The XFRSIZE register should be set to N-1, where N is the number of 4-word blocks.
2. The HWKEYx registers should be written with the desired key in little endian format.
3. The CONTROL register should be set as follows:
   a. ERRIEN set to 1.
   b. KEYSIZE set to the appropriate number of bits for the key.
   c. EDMD set to 1 for encryption.
   d. KEYCPEN set to 0 to disable key capture at the end of the transaction.
   e. HCTREN set to 1 to enable Hardware Counter mode.
   f. XOREN set to 10b to enable the XOR output path.
   g. The HCBCEN, BEN, SWMDEN bits should all be cleared to 0.

Once the DMA and AES settings have been set, the transfer should be started by writing 1 to the XFRSTA bit.

12.9. Performing “In-Place” Ciphers

For the cipher examples described in Section 12.6, Section 12.7, and Section 12.8, the DMA channels are described using plain text and cipher text address offsets. However, these addresses can be the same instead of separate entities to reduce general-purpose memory usage. These in-place ciphers overwrite the plain text input with the cipher text output data.

For example, the hardware cipher block chaining algorithm executed in place is shown in Figure 12.13.

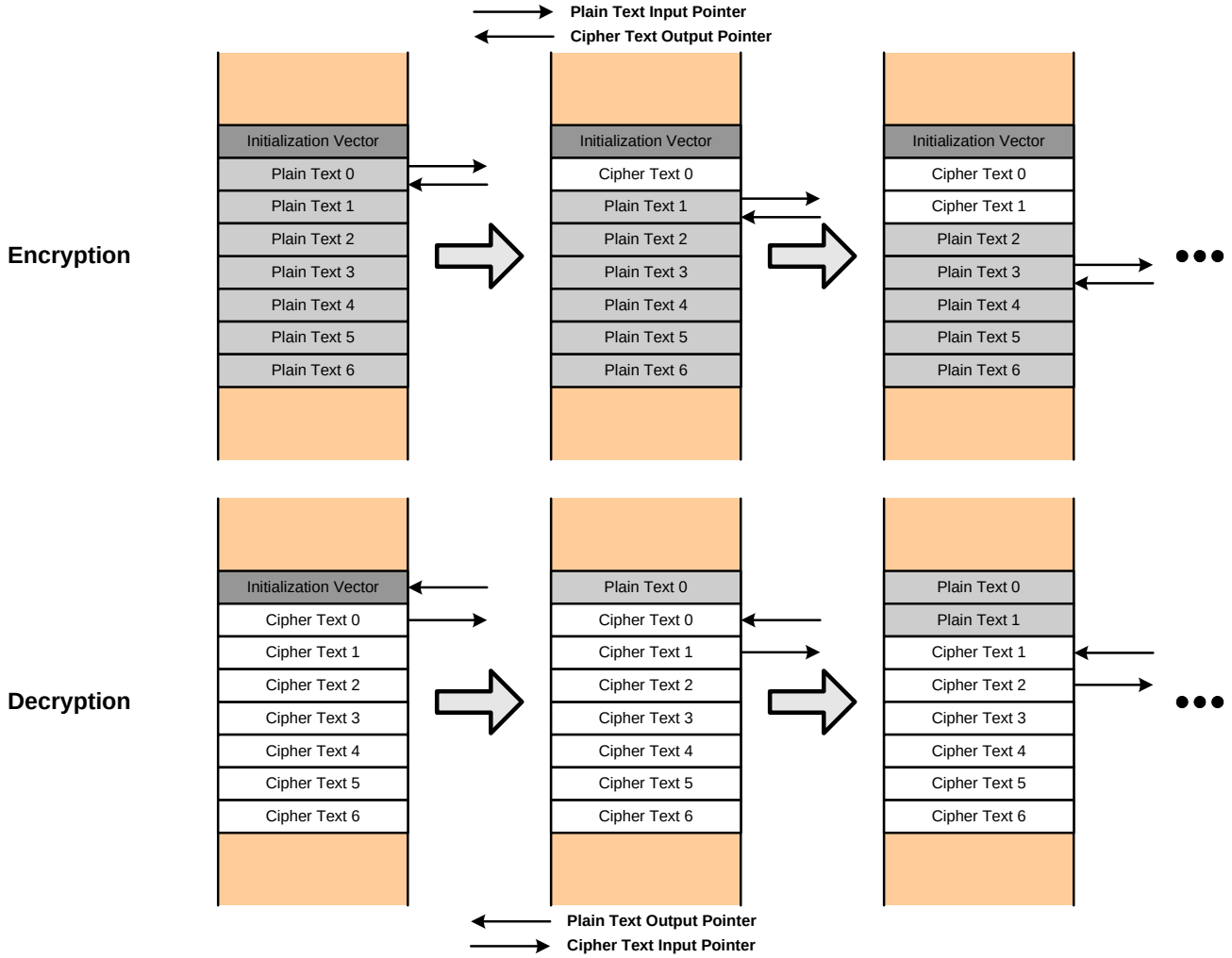


Figure 12.13. Memory Map of Hardware CBC Algorithm Performed In Place

12.10. Using the AES0 Module in Software Mode

Software mode (SWMDEN bit set to 1) allows the firmware to perform smaller or more custom operations with the AES module. When software mode is enabled, the AES module will not generate any DMA requests.

In software mode, each operation must consist of 4 words and follows this general encryption or decryption sequence out of any device reset:

1. The RESET bit must be cleared to access the AES registers.
2. Configure the operation, including setting SWMDEN to 1.
3. Load the input/output data FIFO (DATAFIFO) with four words. These writes must be completed in word form, since byte and half-word writes are not supported.
4. Load the XOR data FIFO (XORFIFO) with four words if XOREN is set to 01b or 10b. These writes must be completed in word form, since byte and half-word writes are not supported.
5. Set KEYCPEN to 1 if key capture is required (EDMD must also be set to 1 for the key capture to occur).
6. Enable the operation complete interrupt by setting OCIEN to 1. Alternatively, firmware can poll XFRSTA or BUSYF.
7. Set XFRSTA to 1 to start the AES operation on the 4-word block.
8. Wait for the completion interrupt or poll until the operation completes.
9. Read the input/output data FIFO (DATAFIFO) with four word reads to obtain the resulting cipher text output.

If key capture (KEYCPEN set to 1) was enabled for an encryption operation, then the key is overwritten. The key must be re-written if a subsequent operation is also an encryption.

The hardware counter and hardware cipher block chaining modes can be used in conjunction with software mode, but bypass mode (BEN) is not available.

12.10.1. Software Mode Error Conditions

Care must be taken when reading or writing the input/output or XOR FIFOs:

- Loading more than four words into the input/output or XOR data FIFOs results in an overrun error.
- Loading less than four words into these data FIFOs prevents an operation from starting when the XFRSTA bit is set to 1.
- Reading the input/output data FIFO with more than four word reads results in an underrun error.

Firmware can check the current status of the FIFOs using the DFIFOLVL and XFIFOLVL fields in the STATUS register. Failure to read the data from the input/output data FIFO leaves the FIFO full, so any subsequent writes to this FIFO with new input data causes an overrun event. Any unwanted data in the data or XOR FIFOs can be discarded by using a soft reset of the AES module (RESET).

12.11. AES0 Registers

This section contains the detailed register descriptions for AES0 registers.

Register 12.1. AES0_CONTROL: Module Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	RESET	DBGMD	Reserved				OCIEN	ERRIEN	Reserved						KEYSIZE	
Type	RW	RW	R				RW	RW	R						RW	
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved		HCBCEN	HCTREN	XOREN		BEN	SWMDEN	Reserved					EDMD	KEYCPEN	XFRSTA
Type	R		RW	RW	RW		RW	RW	R					RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

AES0_CONTROL = 0x4002_7000

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 12.1. AES0_CONTROL Register Bit Descriptions

Bit	Name	Function
31	RESET	**Module Soft Reset.** Must be cleared to access any other AES module bit. 0: AES module is not in soft reset. 1: AES module is in soft reset and none of the module bits can be accessed.
30	DBGMD	**AES Debug Mode.** 0: A debug breakpoint will cause the AES module to halt. 1: The AES module will continue to operate while the core is halted in debug mode.
29:26	Reserved	Must write reset value.
25	OCIEN	**Operation Complete Interrupt Enable.** Enables the completion interrupt for each 4-word block. 0: Disable the operation complete interrupt. 1: Enable the operation complete interrupt.
24	ERRIEN	**Error Interrupt Enable.** 0: Disable the error interrupt. 1: Enable the error interrupt.
23:18	Reserved	Must write reset value.

Table 12.1. AES0_CONTROL Register Bit Descriptions

Bit	Name	Function
17:16	KEYSIZE	**Keystore Size Select.** Selects the size of the key used in the AES encryption or decryption process. 00: Key is composed of 128 bits. 01: Key is composed of 192 bits. 10: Key is composed of 256 bits. 11: Reserved.
15:14	Reserved	Must write reset value.
13	HCBCEN	**Hardware Cipher-Block Chaining Mode Enable.** Enables the Hardware Cipher-Clock Chaining (CBC) mode. This causes the XOR path to be fed automatically from hardware with CT(n-1), so there is no need for firmware or the DMA to feed the XOR path in this mode. 0: Disable hardware cipher-block chaining (CBC) mode. 1: Enable hardware cipher-block chaining (CBC) mode.
12	HCTREN	**Hardware Counter Mode Enable.** Enables the Hardware Counter Mode. 0: Disable hardware counter mode. 1: Enable hardware counter mode.
11:10	XOREN	**XOR Enable.** Enables the input or output XOR path. 00: Disable the XOR paths. 01: Enable the XOR input path, disable the XOR output path. 10: Disable the XOR input path, enable the XOR output path. 11: Reserved.
9	BEN	**Bypass AES Operation Enable.** If this bit is set to 1, the AES module hardware is bypassed, which allows firmware to use the module as a memory copy. The XOR paths (output and input) are not available in this mode. 0: Do not bypass AES operations. 1: Bypass AES operations.
8	SWMDEN	**Software Mode Enable.** Setting this bit to 1 stops DMA requests from being generated. When this bit is 1, firmware is responsible for loading the input FIFO with 4 words, loading the XOR FIFO with 4 words (if applicable), and reading 4 words from the output FIFO after receiving the done interrupt. If this bit is 1, the transfer size register (XFRSIZE) should be cleared to 0. Bypass mode is not supported in conjunction with software mode. 0: Disable software mode. 1: Enable software mode.
7:3	Reserved	Must write reset value.
2	EDMD	**Encryption/Decryption Mode.** 0: AES module performs a decryption operation 1: AES module performs an encryption operation.

Table 12.1. AES0_CONTROL Register Bit Descriptions

Bit	Name	Function
1	KEYCPEN	**Key Capture Enable.** If this bit is set to 1, the current key in the keystore is overwritten during the module operation. In the case where EDMD is set to 1 (encryption operation), this generated key is the proper decryption key after the last block is complete. If SWMDEN is cleared to 0, the hardware only overwrites the key on the last block of the DMA transfer. If SWMDEN is set to 1, the hardware overwrites the key for each operation KEYCPEN is set to 1. 0: Disable key capture. 1: Enable key capture.
0	XFRSTA	**AES Transfer Start.** If SWMDEN is set to 1, setting this bit to 1 starts an AES module operation on the 4-word block. This bit is automatically cleared when the 4-word operation completes. If SWMDEN is cleared to 0, setting this bit to 1 starts a series of AES module operations until the XFRSIZE register counts down to 0. This bit is automatically cleared when the XFRSIZE regiser is 0 and the current operation completes.

Register 12.2. AES0_XFRSIZE: Number of Blocks

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved					XFRSIZE										
Type	R					RW										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

AES0_XFRSIZE = 0x4002_7010

Table 12.2. AES0_XFRSIZE Register Bit Descriptions

Bit	Name	Function
31:11	Reserved	Must write reset value.
10:0	XFRSIZE	**Transfer Size.** The number of 4-word blocks such that XFRSIZE + 1 blocks will be processed and transferred by the AES module. This value must match the DMA transfer size for each relevant channel. This value is automatically decremented by hardware as each block operation completes. When the SWMDEN bit is set to 1, this register should be set to 0.

Register 12.3. AES0_DATAFIFO: Input/Output Data FIFO Access

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	DATAFIFO[31:16]															
Type	RW															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	DATAFIFO[15:0]															
Type	RW															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Register ALL Access Address

AES0_DATAFIFO = 0x4002_7020

Table 12.3. AES0_DATAFIFO Register Bit Descriptions

Bit	Name	Function
31:0	DATAFIFO	**Input/Output Data FIFO Access.** This data register interfaces with the AES Input/Output data FIFO. For DMA operations, this register should be targeted by the DMA input data and DMA output data (4-word transfers) in non-incrementing mode. For software operations (SWMDEN bit is set to 1), this register can be written to set the input data or read to access the output data in 4-word blocks. When doing soft transfers, the register must be written or read 4 times to fill or empty the FIFO before the AES operation can be initiated using the XFRSTA bit. All reads and writes of this register must be word reads and writes. Byte and half word reads and writes are not allowed. Any read or write operation will add or remove an entire word to or from the FIFO, so non-word reads or writes may result in data loss or FIFO errors. Reads from this register will result in data pops from the Input/Output FIFO. Writes to this register will result in data pushes to the Input/Output FIFO. Input/Output data FIFO overflows and underruns will result in an error interrupt (if enabled).

Notes:

1. Reads of this register modify the state of hardware. Debug logic should take care when reading this register.
2. The access methods for this register are restricted. Do not use half-word or byte access methods on this register.

Register 12.4. AES0_XORFIFO: XOR Data FIFO Access

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	XORFIFO[31:16]															
Type	RW															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	XORFIFO[15:0]															
Type	RW															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Register ALL Access Address

AES0_XORFIFO = 0x4002_7030

Table 12.4. AES0_XORFIFO Register Bit Descriptions

Bit	Name	Function
31:0	XORFIFO	**XOR Data FIFO Access.** This data register interfaces with the AES XOR data FIFO. For DMA operations, this register should be targeted by the DMA input XOR data (4-word transfers) in non-incrementing mode. For software operations (SWMDEN bit is set to 1), this register can be written to set the input XOR data in 4-word blocks. When doing soft transfers, the register must be written 4 times to fill or empty the FIFO before the AES operation can be initiated using the XFRSTA bit. All reads and writes of this register must be word reads and writes. Byte and half word reads and writes are not allowed. Any write operation will add an entire word to the FIFO. Reads from this register have no effect. Writes to this register will result in data pushes to the Input/Output FIFO. XOR data FIFO overflows will result in an error interrupt (if enabled).

Notes:

1. Reads of this register modify the state of hardware. Debug logic should take care when reading this register.
2. The access methods for this register are restricted. Do not use half-word or byte access methods on this register.

Register 12.5. AES0_HWKEY0: Hardware Key Word 0

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	HWKEY0[31:16]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	HWKEY0[15:0]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

AES0_HWKEY0 = 0x4002_7040

Table 12.5. AES0_HWKEY0 Register Bit Descriptions

Bit	Name	Function
31:0	HWKEY0	**Hardware Key Word 0.** This register contains word 0 of the keystore. If the KEYCPEN bit is set to 1, the new key will be accessible using these registers after the old key is overwritten.

Register 12.6. AES0_HWKEY1: Hardware Key Word 1

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	HWKEY1[31:16]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	HWKEY1[15:0]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

AES0_HWKEY1 = 0x4002_7050

Table 12.6. AES0_HWKEY1 Register Bit Descriptions

Bit	Name	Function
31:0	HWKEY1	**Hardware Key Word 1.** This register contains word 1 of the keystore. If the KEYCPEN bit is set to 1, the new key will be accessible using these registers after the old key is overwritten.

Register 12.7. AES0_HWKEY2: Hardware Key Word 2

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	HWKEY2[31:16]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	HWKEY2[15:0]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

AES0_HWKEY2 = 0x4002_7060

Table 12.7. AES0_HWKEY2 Register Bit Descriptions

Bit	Name	Function
31:0	HWKEY2	**Hardware Key Word 2.** This register contains word 2 of the keystore. If the KEYCPEN bit is set to 1, the new key will be accessible using these registers after the old key is overwritten.

Register 12.8. AES0_HWKEY3: Hardware Key Word 3

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	HWKEY3[31:16]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	HWKEY3[15:0]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

AES0_HWKEY3 = 0x4002_7070

Table 12.8. AES0_HWKEY3 Register Bit Descriptions

Bit	Name	Function
31:0	HWKEY3	**Hardware Key Word 3.** This register contains word 3 of the keystore. If the KEYCPEN bit is set to 1, the new key will be accessible using these registers after the old key is overwritten.

Register 12.9. AES0_HWKEY4: Hardware Key Word 4

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	HWKEY4[31:16]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	HWKEY4[15:0]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

AES0_HWKEY4 = 0x4002_7080

Table 12.9. AES0_HWKEY4 Register Bit Descriptions

Bit	Name	Function
31:0	HWKEY4	**Hardware Key Word 4.** This register contains word 4 of the keystore. If the KEYCPEN bit is set to 1, the new key will be accessible using these registers after the old key is overwritten.

Register 12.10. AES0_HWKEY5: Hardware Key Word 5

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	HWKEY5[31:16]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	HWKEY5[15:0]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

AES0_HWKEY5 = 0x4002_7090

Table 12.10. AES0_HWKEY5 Register Bit Descriptions

Bit	Name	Function
31:0	HWKEY5	**Hardware Key Word 5.** This register contains word 5 of the keystore. If the KEYCPEN bit is set to 1, the new key will be accessible using these registers after the old key is overwritten.

Register 12.11. AES0_HWKEY6: Hardware Key Word 6

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	HWKEY6[31:16]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	HWKEY6[15:0]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

AES0_HWKEY6 = 0x4002_70A0

Table 12.11. AES0_HWKEY6 Register Bit Descriptions

Bit	Name	Function
31:0	HWKEY6	**Hardware Key Word 6.** This register contains word 6 of the keystore. If the KEYCPEN bit is set to 1, the new key will be accessible using these registers after the old key is overwritten.

Register 12.12. AES0_HWKEY7: Hardware Key Word 7

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	HWKEY7[31:16]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	HWKEY7[15:0]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

AES0_HWKEY7 = 0x4002_70B0

Table 12.12. AES0_HWKEY7 Register Bit Descriptions

Bit	Name	Function
31:0	HWKEY7	**Hardware Key Word 7.** This register contains word 7 of the keystore. If the KEYCPEN bit is set to 1, the new key will be accessible using these registers after the old key is overwritten.

Register 12.13. AES0_HWCTR0: Hardware Counter Word 0

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	HWCTR0[31:16]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	HWCTR0[15:0]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

AES0_HWCTR0 = 0x4002_70C0

Table 12.13. AES0_HWCTR0 Register Bit Descriptions

Bit	Name	Function
31:0	HWCTR0	**Hardware Counter Word 0.** This register contains word 0 of the 128-bit hardware counter. These registers are little endian. When the HCTREN bit is set to 1, this register should be written with the initial counter value to seed the encryption or decryption process for a block of operations. Reading this register always reflects the current value of the hardware counter. Firmware should not modify the contents of this register when using hardware CBC mode (HCBCEN = 1).

Register 12.14. AES0_HWCTR1: Hardware Counter Word 1

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	HWCTR1[31:16]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	HWCTR1[15:0]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

AES0_HWCTR1 = 0x4002_70D0

Table 12.14. AES0_HWCTR1 Register Bit Descriptions

Bit	Name	Function
31:0	HWCTR1	**Hardware Counter Word 1.** This register contains word 1 of the 128-bit hardware counter. These registers are little endian. When the HCTREN bit is set to 1, this register should be written with the initial counter value to seed the encryption or decryption process for a block of operations. Reading this register always reflects the current value of the hardware counter. Firmware should not modify the contents of this register when using hardware CBC mode (HCBCEN = 1).

Register 12.15. AES0_HWCTR2: Hardware Counter Word 2

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	HWCTR2[31:16]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	HWCTR2[15:0]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

AES0_HWCTR2 = 0x4002_70E0

Table 12.15. AES0_HWCTR2 Register Bit Descriptions

Bit	Name	Function
31:0	HWCTR2	**Hardware Counter Word 2.** This register contains word 2 of the 128-bit hardware counter. These registers are little endian. When the HCTREN bit is set to 1, this register should be written with the initial counter value to seed the encryption or decryption process for a block of operations. Reading this register always reflects the current value of the hardware counter. Firmware should not modify the contents of this register when using hardware CBC mode (HCBCEN = 1).

Register 12.16. AES0_HWCTR3: Hardware Counter Word 3

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	HWCTR3[31:16]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	HWCTR3[15:0]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

AES0_HWCTR3 = 0x4002_70F0

Table 12.16. AES0_HWCTR3 Register Bit Descriptions

Bit	Name	Function
31:0	HWCTR3	**Hardware Counter Word 3.** This register contains word 3 of the 128-bit hardware counter. These registers are little endian. When the HCTREN bit is set to 1, this register should be written with the initial counter value to seed the encryption or decryption process for a block of operations. Reading this register always reflects the current value of the hardware counter. Firmware should not modify the contents of this register when using hardware CBC mode (HCBCEN = 1).

Register 12.17. AES0_STATUS: Module Status

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	OCI	ERRI	Reserved													BUSYF
Type	RW	RW	R													R
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved					XFIFOLVL			Reserved	DFIFOLVL			Reserved	XORF	DORF	DURF
Type	R					R			R	R			R	R	R	R
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

AES0_STATUS = 0x4002_7100

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 12.17. AES0_STATUS Register Bit Descriptions

Bit	Name	Function
31	OCI	**Operation Complete Interrupt Flag.** This bit is set to 1 by hardware when the current AES operation is complete. This bit must be cleared by software.
30	ERRI	**Error Interrupt Flag.** This bit is set to 1 by hardware when an AES error has occurred (DURF, DORF, or XORF set to 1). This bit must be cleared by software.
29:17	Reserved	Must write reset value.
16	BUSYF	**Module Busy Flag.** 0: AES module is not busy. 1: AES module is completing an operation.
15:11	Reserved	Must write reset value.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 12.17. AES0_STATUS Register Bit Descriptions

Bit	Name	Function
10:8	XFIFOLVL	**XOR Data FIFO Level.** 000: XOR data FIFO is empty. 001: XOR data FIFO contains 1 word. 010: XOR data FIFO contains 2 words. 011: XOR data FIFO contains 3 words. 100: XOR data FIFO contains 4 words (full). 101-111: Reserved.
7	Reserved	Must write reset value.
6:4	DFIFOLVL	**Input/Output Data FIFO Level.** 000: Input/Output data FIFO is empty. 001: Input/Output data FIFO contains 1 word. 010: Input/Output data FIFO contains 2 words. 011: Input/Output data FIFO contains 3 words. 100: Input/Output data FIFO contains 4 words (full). 101-111: Reserved.
3	Reserved	Must write reset value.
2	XORF	**XOR Data FIFO Overrun Flag.** Firmware can recover from this condition by performing a soft reset on the AES module by using the RESET bit. 0: No XOR data FIFO overrun. 1: An XOR data FIFO overrun has occurred.
1	DORF	**Input/Output Data FIFO Overrun Flag.** Firmware can recover from this condition by performing a soft reset on the AES module by using the RESET bit. 0: No input/output data FIFO overrun. 1: An input/output data FIFO overrun has occurred.
0	DURF	**Input/Output Data FIFO Underrun Flag.** Firmware can recover from this condition by performing a soft reset on the AES module by using the RESET bit. 0: No input/output data FIFO underrun. 1: An input/output data FIFO underrun has occurred.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

12.12. AES0 Register Memory Map

Table 12.18. AES0 Memory Map

Register Name	AES0_CONTROL	AES0_XFRSIZE	AES0_DATAFIFO	AES0_XORFIFO	AES0_HWKEY0
ALL Address	0x4002_7000	0x4002_7010	0x4002_7020	0x4002_7030	0x4002_7040
Access Methods	ALL | SET | CLR	ALL	ALL	ALL	ALL
Bit 31	RESET	Reserved	DATAFIFO	XORFIFO	HWKEY0
Bit 30	DBGMD				
Bit 29	Reserved				
Bit 28					
Bit 27					
Bit 26					
Bit 25	OCIEN				
Bit 24	ERRIEN				
Bit 23	Reserved				
Bit 22					
Bit 21					
Bit 20					
Bit 19	KEYSIZE				
Bit 18					
Bit 17					
Bit 16					
Bit 15	Reserved				
Bit 14					
Bit 13	HCBCEN				
Bit 12	HCTREN				
Bit 11	XOREN	XFRSIZE			
Bit 10					
Bit 9	BEN				
Bit 8	SWMDEN				
Bit 7	Reserved				
Bit 6					
Bit 5					
Bit 4					
Bit 3					
Bit 2	EDMD				
Bit 1	KEYCPEN				
Bit 0	XFRSTA				

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

Table 12.18. AES0 Memory Map

Register Name	AES0_HWKEY1	AES0_HWKEY2	AES0_HWKEY3	AES0_HWKEY4	AES0_HWKEY5
ALL Address	0x4002_7050	0x4002_7060	0x4002_7070	0x4002_7080	0x4002_7090
Access Methods	ALL	ALL	ALL	ALL	ALL
Bit 31	HWKEY1	HWKEY2	HWKEY3	HWKEY4	HWKEY5
Bit 30					
Bit 29					
Bit 28					
Bit 27					
Bit 26					
Bit 25					
Bit 24					
Bit 23					
Bit 22					
Bit 21					
Bit 20					
Bit 19					
Bit 18					
Bit 17					
Bit 16					
Bit 15					
Bit 14					
Bit 13					
Bit 12					
Bit 11					
Bit 10					
Bit 9					
Bit 8					
Bit 7					
Bit 6					
Bit 5					
Bit 4					
Bit 3					
Bit 2					
Bit 1					
Bit 0					

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

Table 12.18. AES0 Memory Map

Register Name	AES0_HWKEY6	AES0_HWKEY7	AES0_HWCTR0	AES0_HWCTR1	AES0_HWCTR2
ALL Address	0x4002_70A0	0x4002_70B0	0x4002_70C0	0x4002_70D0	0x4002_70E0
Access Methods	ALL	ALL	ALL	ALL	ALL
Bit 31	HWKEY6	HWKEY7	HWCTR0	HWCTR1	HWCTR2
Bit 30					
Bit 29					
Bit 28					
Bit 27					
Bit 26					
Bit 25					
Bit 24					
Bit 23					
Bit 22					
Bit 21					
Bit 20					
Bit 19					
Bit 18					
Bit 17					
Bit 16					
Bit 15					
Bit 14					
Bit 13					
Bit 12					
Bit 11					
Bit 10					
Bit 9					
Bit 8					
Bit 7					
Bit 6					
Bit 5					
Bit 4					
Bit 3					
Bit 2					
Bit 1					
Bit 0					

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

Table 12.18. AES0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
AES0_HWCTR3	0x4002_70F0	ALL	HWCTR3																															
AES0_STATUS	0x4002_7100	ALL | SET | CLR	OCI	ERRI	Reserved													BUSYF	Reserved					XFIFOLVL		Reserved		DFIFOLVL			Reserved		XORF	DORF	DURF

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

13. Capacitive Sensing (CAPSENSE0)

This section describes the Capacitive Sensing (CAPSENSE) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the CAPSENSE block, which is used by all device families covered in this document.

13.1. Capacitive Sensing Features

The CAPSENSE0 module measures capacitance on external pins and converts it to a digital value. The module includes the following features:

- Nine start-of-conversion sources, including timers, I2C timers and software “on-demand” triggers.
- Sixteen external I/O pin input channels.
- Option to convert to 12, 13, 14, or 16 bits.
- Automatic threshold comparison with programmable polarity (“less than or equal” or “greater than”).
- Four operation modes: single conversion, single scan, continuous single conversion, and continuous scan.
- Auto-accumulate mode that averages multiple samples from a single start of conversion signal.
- Single bit retry options available to reduce the effect of noise during a conversion.
- Supports channel bonding to monitor multiple channels connected together with a single conversion.
- Scanning option allows the module to convert a single or series of channels and compare against the threshold while the AHB clock is stopped and the core is in a low power mode.

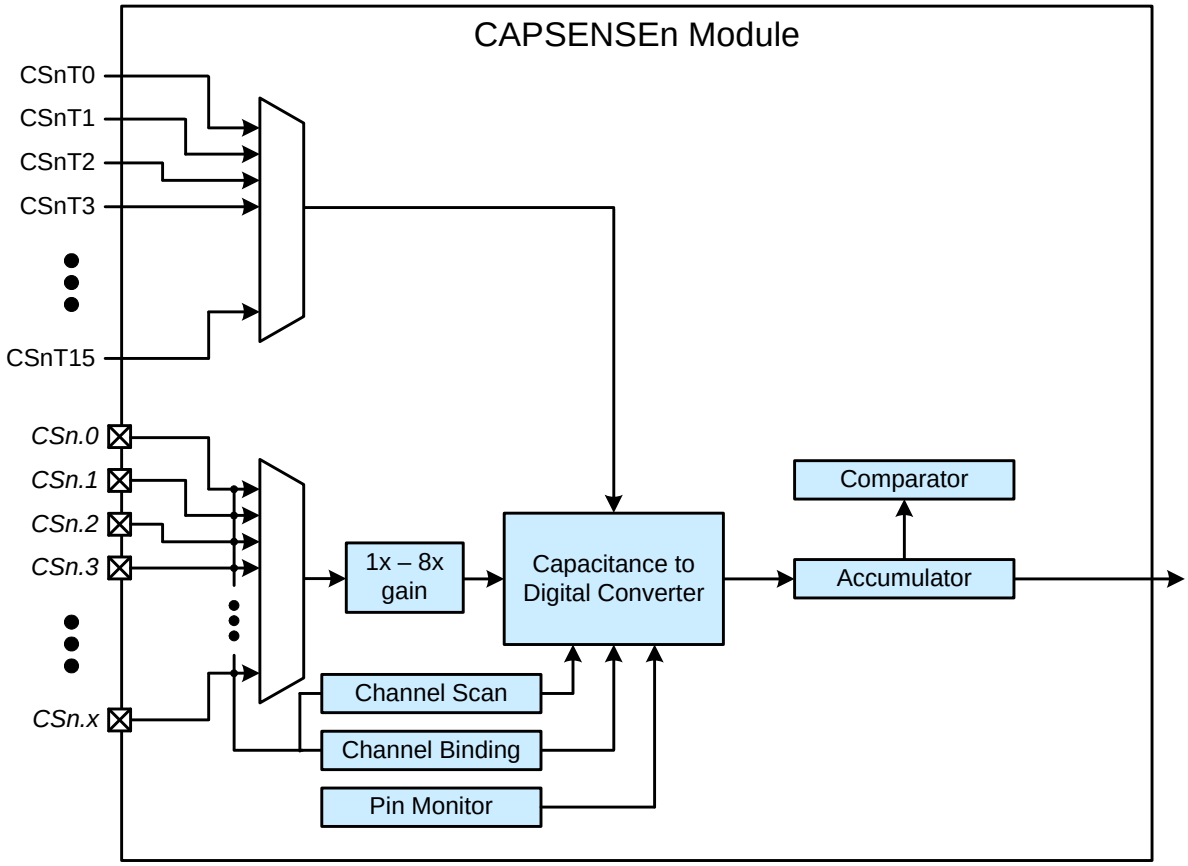


Figure 13.1. CAPSENSE Block Diagram

13.2. Overview

The capacitive sensing (CAPSENSE) module uses a capacitance-to-digital circuit to determine the capacitance on an input pin. The module can take measurements from different physical pins using the module's analog multiplexer. In addition, the module can measure multiple pins in sequence using the scan modes, or multiple pins at the same time using the multiple-channel measurement feature. The module can continue to make measurements in one of the continuous modes if the AHB clock stops, allowing the module to wake a device from a low power mode. The module is enabled only when the CSEN and BIASEN bits are set to 1; otherwise, the module is in a low-power shutdown state.

A selectable gain circuit allows the module to adjust the maximum allowable capacitance. An accumulator is also included, which can be configured to average multiple conversions on an input channel.

The hardware can generate an interrupt after completing a conversion, a series of conversions, or when the measured value crosses a threshold defined in the CSTH register.

All pins used as inputs to the CAPSENSE module should be configured as analog inputs as described in the port configuration section.

13.3. Measurement Overview

In single-conversion mode, the CAPSENSE module measures the channel selected by the CSMX field. This measurement begins when the start of conversion source set by the CSCM field triggers. Hardware sets the BUSYF flag to 1 while a conversion is in progress and automatically clears this bit when the conversion completes, in addition to setting the conversion done (CDI) flag. Firmware can read the conversion result from the DATA register. Figure 13.2 depicts this basic CAPSENSE measurement process.

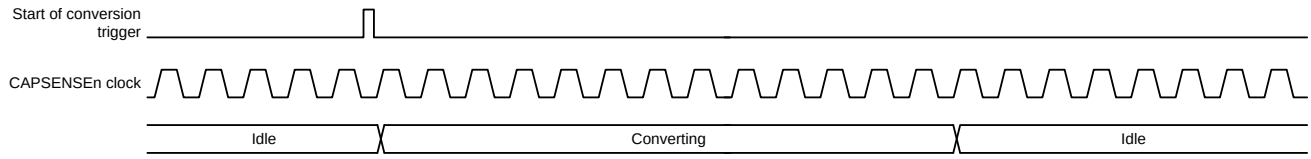


Figure 13.2. Basic CAPSENSE Measurement

Firmware can disconnect the pin from the measurement circuitry by setting the CSDISC bit to 1.

Note: Enable the CAPSENSE bias (BIASEN = 1) before enabling the module (CSEN = 1). Firmware must wait at least 2 µs after enabling the bias before starting a conversion.

13.3.1. Start of Conversion Sources

A capacitive sense conversion can be initiated in one of nine ways, depending on the programmed state of the start of conversion (CSCM) field in the CONTROL register. The available options are detailed in Table 13.1.

Table 13.1. CAPSENSE0 Start of Conversion Sources

Trigger	Convert Start Description	Internal Signal
CS0T0	Internal Convert Start	“On Demand” by writing 1 to CSBUSY
CS0T1	Internal Convert Start	Timer 0 Low overflow
CS0T2	Internal Convert Start	Timer 0 High overflow
CS0T3	Internal Convert Start	Timer 1 Low overflow
CS0T4	Internal Convert Start	Timer 1 High overflow
CS0T5	Internal Convert Start	I2C0 Timer Byte 1 overflow
CS0T6	Internal Convert Start	I2C0 Timer Byte 3 overflow
CS0T7	Internal Convert Start	I2C1 Timer Byte 1 overflow
CS0T8	Internal Convert Start	I2C1 Timer Byte 3 overflow

13.3.2. Inputs

The CAPSENSE module can select between multiple external inputs (up to 16 are available, depending on the package type). These inputs can be measured individually, scanned using the single scan or continuous scan modes, or connected together using the CONTROL.CSMCEN bit. The CAPSENSE module input channel settings are located in the MUX.CSMX field and SCANM register. The CAPSENSE0 input channels vary between package options, and are shown in Table 13.2.

Table 13.2. CAPSENSE0 Input Channels

CAPSENSE0 Input	CAPSENSE0 Input Description	SiM3U1x7/C1x7 Pin Name	SiM3U1x6/C1x6 Pin Name	SiM3U1x4/C1x4 Pin Name
CS0.0	External Channel	PB0.1	Reserved	PB0.2
CS0.1	External Channel	PB0.2	PB0.0	PB0.3
CS0.2	External Channel	PB0.3	PB0.1	PB0.4
CS0.3	External Channel	PB0.4	PB0.2	PB0.6
CS0.4	External Channel	PB0.5	PB0.3	PB0.7
CS0.5	External Channel	PB0.6	PB0.4	Reserved
CS0.6	External Channel	PB0.7	PB0.5	Reserved
CS0.7	External Channel	PB0.8	PB0.6	PB0.0
CS0.8	External Channel	PB1.7	PB1.2	Reserved
CS0.9	External Channel	PB1.8	PB1.3	Reserved
CS0.10	External Channel	PB1.13	PB1.6	PB0.12
CS0.11	External Channel	PB1.14	PB1.7	PB0.13
CS0.12	External Channel	PB1.15	PB1.8	PB0.14
CS0.13	External Channel	PB2.0	PB1.9	PB0.15
CS0.14	External Channel	PB2.1	PB1.10	PB1.0
CS0.15	External Channel	PB2.2	PB1.11	PB1.1

13.4. Conversion and Input Modes

The CAPSENSE module has four conversion modes: single conversion, single scan, continuous single, and continuous scan.

13.4.1. Single Conversion Mode

For single conversion mode (CMD = 0), the module takes a single conversion for each start of conversion trigger. Hardware sets the conversion done (CDI) flag when the single conversion completes.

Firmware can use single conversion mode and multiplex through the input channels to measure multiple channels. After polling on conversion done (CDI) or waiting for the conversion done interrupt, firmware can switch the CSMX input multiplexor to the next input before moving the data out of the DATA register. Then, the next measurement can be initiated by the start of conversion trigger after disabling (CSEN = 0) and reenabling (CSEN = 1) the module.

13.4.2. Single Scan Mode

Scan mode is enabled by setting CMD to 1 for single scan mode or 3 for continuous scan mode. When operating in scan mode and MCEN is cleared to 0, the CAPSENSE module will cycle through the channels defined by the SCANEN field. Each bit in SCANEN represents a capacitive sensing channel, and setting a bit to 1 in the SCANEN field enables that channel for the scan. The accumulator setting (ACCMD) determines the number of samples takes for each channel in scan mode. Figure 13.3 illustrates single scan mode.

The module will stop taking measurements in single scan mode after measuring the last channel or if a compare threshold event occurs. Using scan mode with the CAPSENSE comparator interrupt enabled allows a system to detect a change in measured capacitance without requiring any additional dedicated core resources.

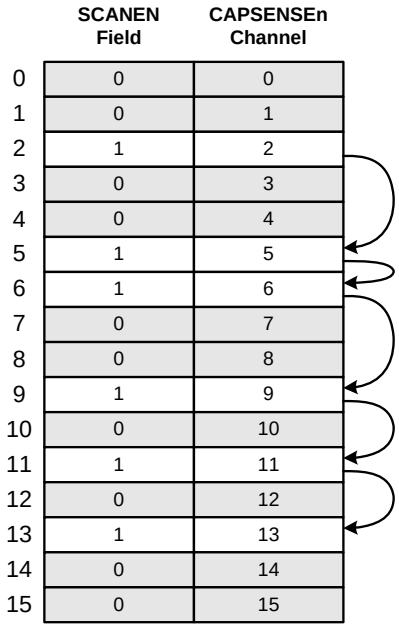


Figure 13.3. Single Scan Mode

13.4.3. Continuous Single and Scan Modes

Conversions can be configured to be initiated continuously through one of two methods. The module can convert on a single channel continuously (CMD = 1), or it can be configured to convert continuously with scan enabled (CMD = 3).

The module will continue to convert on the single or scan channels until a compare threshold event occurs, or firmware disables the module by clearing CSEN to 0.

The CAPSENSE module can continue to make conversions in the continuous modes even if the AHB clock is stopped. A new capacitive measurement operation cannot be started if the AHB clock is not running.

13.5. Conversion Rate

The CAPSENSE module uses a method of successive approximation to determine the value of an external capacitance. The number of bits the module converts is adjustable using the CNVR field. Conversions are 13 bits long by default, but they can be adjusted to 12, 13, 14, or 16 bits. Unconverted bits will be set to 0. Shorter conversion lengths produce faster conversion rates, and longer conversion lengths increase the conversion time. Applications can take advantage of faster conversion rates when the unconverted bits fall below the noise floor.

13.6. Accumulation Modes

The capacitive sensing module can measure a channel multiple times and internally accumulate those measurements. The ACCMD field selects the accumulation mode, and the module can accumulate 1, 4, 8, 16, 32, or 64 samples. After the defined number of samples have been accumulated, the result is divided by either 1, 4, 8, 16, 32, or 64 depending on the ACCMD setting and copied to the DATA register in right-justified format. The accumulator setting determines the number of samples takes for each channel in both single and scan modes.

The conversion done (CDI) flag behavior depends on the settings of the CAPSENSE accumulator. If the module is configured to accumulate multiple conversions on an input channel, the hardware will set the conversion done (CDI) flag after last conversion completes. Each measurement requires a separate start of conversion trigger.

13.7. Measuring Multiple Channels in a Single Measurement

The CAPSENSE module has the capability of measuring the total capacitance of multiple channels using a single conversion. When the multiple channel feature is enabled (MCEN = 1), channels selected by SCANEN field are internally shorted together, and the combined node is selected as the module input. This mode can be used to detect a capacitance change on multiple channels using a single conversion.

The single scan and continuous scan modes should not be used when MCEN is set to 1.

13.8. Pin Monitoring

The CAPSENSE module provides accurate conversions in all operating modes of the core, peripherals, and pins. The pin monitoring feature improves interference immunity from high-current output pin switching. The port match register in the port configuration module controls the pin selection of the monitors.

Note: The APB clock must be greater than 5 MHz when using the pin monitoring feature with the capacitive sensing module.

Conversions in the capacitive sensing module are immune to any change on digital inputs and immune to most output switching. Even high-speed serial data transmission will not affect CAPSENSE operation as long as the output load is limited. Output changes that switch large loads, such as LEDs and heavily-loaded communications lines, can affect conversion accuracy. For this reason, the module includes pin monitoring circuits that will, if enabled, automatically adjust conversion timing if necessary to eliminate any effect from high-current output pin switching.

The pin monitor enable bit in the device port configuration module should be set for any output signal that is expected to drive a large load.

Pin monitors should not be enabled unless they are required. The pin monitor works by repeating any portion of a conversion that may have been corrupted by a change on an output pin. As a result, enabling a pin monitor will slow capacitive sensing conversions.

The frequency of CAPSENSE retry operations can be limited by setting the PMMD field. In the default state, all converter retry requests will be performed. This is the recommended setting for all applications. The number of retries per conversion can be limited to either two (PMMD = 1) or four (PMMD = 2) retries. Limiting the number of retries per conversion ensures that conversions will be completed even in circumstances where extremely frequent high-power output switching occurs, though there may be some loss of accuracy due to switching noise.

Activity of the pin monitor circuit can be detected by reading the pin monitor event (PMEF) flag. Hardware will set this flag if any CAPSENSE converter retries have occurred. It remains set until cleared by firmware or a device reset.

13.9. Compare Threshold

When enabled (CMPEN = 1), the module comparator compares the latest capacitive sensing conversion result with the value stored in CSTH. The polarity of this conversion is controlled with the CMPPOL bit. If CMPPOL is set to 1, hardware sets the compare threshold (CMPI) flag to 1 if the result is less than or equal to the stored value. If CMPPOL is cleared to 0, hardware sets the CMPI flag to 1 if the result is greater than the threshold.

If the accumulator is configured to accumulate multiple conversions, a comparison will not be made until the last conversion has been accumulated.

If the module is operating in scan mode, a compare threshold event will cause the module to stop converting until a new start of conversion trigger occurs.

13.10. Interrupts

Hardware sets the conversion done (CDI) flag when a single conversion completes. The CDI flag behavior depends on the settings of the CAPSENSE accumulator (ACCMD). If the module is configured to accumulate multiple conversions on an input channel, the hardware will set the conversion done (CDI) flag after last conversion completes. Hardware can also generate an interrupt when setting CDI, if enabled (CDIEN = 1).

If the module comparator is enabled (CMPEN = 1), the compare threshold (CMPI) flag sets when the result is less than or equal to the threshold if CMPPOL is cleared to 0 or greater than the threshold if CMPPOL is set to 1. This event will also generate an interrupt if the comparator is enabled (CMPEN = 1). The CMPI flag must be cleared by disabling the module (CSEN = 1).

Hardware sets the end-of-scan (EOSI) flag when a single scan operation completes and can generate an interrupt if EOSIEN is set to 1. This flag must be cleared by disabling the module (CSEN = 0).

13.11. Additional Options

13.11.1. Gain Adjustment

The gain of the capacitive measurement circuit can be adjusted in integer increments from 1x to 8x, where 8x is the default setting. High gain gives the best sensitivity and resolution for small capacitors, such as those typically implemented as touch-sensitive PCB features. To measure larger capacitance values, the gain should be lowered accordingly. The CGSEL field sets the gain value.

13.11.2. Measuring Non-Typical Systems

There are several configuration options in the CAPSENSE module designed to modify the operation of the circuit and address special situations. In particular, any circuit with more than 500 ohms of series impedance between the sensor and the device pin may require adjustments for optimal performance. Typical applications which may require adjustments include:

- Touch panel sensors fabricated using a resistive conductor such as indium-tin-oxide (ITO).
- Circuits using a high-value series resistor to isolate the sensor element for high ESD protection.

Most systems will require no fine tuning, and the default settings for RAMPSEL, IASEL, and DTSEL should be used.

13.12. Taking a Measurement

13.12.1. Initializing the Module

To configure the CAPSENSE module to take a measurement:

1. Enable the CAPSENSE bias (BIASEN = 1) before enabling the module. Firmware must wait at least 2 µs after enabling the bias before starting a conversion.
2. Initialize the start of conversion mode select field (CSCM) to the desired trigger source.
3. Set the input channel using the CSMX field. Any pins used as an input must be set to analog mode.
4. (Optional) If using scan or multiple-channel measurement modes, set the SCANEN field to enable the channels. Any pins used as an input must be set to analog mode.
5. Set the conversion mode using the CMD field.
6. Set the conversion rate using the CNVR field.
7. Set the accumulation mode using the ACCMD field.
8. (Optional) If using the compare threshold, set the threshold (CSTH), set the threshold polarity (CMPPOL), and enable the comparator (CMPEN = 1).
9. Enable the module (CSEN = 1).
10. Initiate the conversions using the selected trigger source.
11. Disable the module (CSEN = 0) after the measurement.
12. Reenable the module (CSEN = 1) and start the next measurement operation.

13.12.2. Reconfiguring the Module

To reconfigure the capacitive sensing module after conversions have already taken place:

1. Disable the module (CSEN = 0) to clear any pending flags and reset the module.
2. Update any fields in the module.
3. Enable the module (CSEN = 1).
4. Initiate the conversions using the selected trigger source.
5. Disable the module after the operation (CSEN = 0).

13.13. CAPSENSE0 Registers

This section contains the detailed register descriptions for CAPSENSE0 registers.

Register 13.1. CAPSENSE0_CONTROL: Module Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved					EOSI	CDI	CMPI	Reserved	EOSIEN	CDIEN	CMPEN	Reserved	PMEF	PMMD	
Type	R					R	RW	R	R	RW	RW	RW	R	RW	RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	CSCM				MCEN	ACCMD			CNVR		CMD		CMPPOL	BIASEN	CSEN	BUSYF
Type	RW				RW	RW			RW		RW		RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Register ALL Access Address

CAPSENSE0_CONTROL = 0x4002_3000

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 13.3. CAPSENSE0_CONTROL Register Bit Descriptions

Bit	Name	Function
31:27	Reserved	Must write reset value.
26	EOSI	**End-of-Scan Interrupt Flag.** This bit is set to 1 by hardware when a single scan operation is complete. This will only occur in single scan mode. This bit can only be cleared by disabling the module (CSEN = 0).
25	CDI	**Conversion Done Interrupt Flag.** This bit is set to 1 by hardware when a data conversion is complete in single conversion mode. This bit must be cleared by firmware.
24	CMPI	**Threshold Comparator Interrupt Flag.** Hardware sets this bit to 1 when a the compare threshold event occurs based on the polarity set by CMPPOL if the internal comparator is enabled (CMPEN = 1). This bit can only be cleared by disabling the module (CSEN = 0).
23	Reserved	Must write reset value.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 13.3. CAPSENSE0_CONTROL Register Bit Descriptions

Bit	Name	Function
22	EOSIEN	**End-of-Scan Interrupt Enable.** 0: Disable the single scan end-of-scan interrupt. 1: Enable the single scan end-of-scan interrupt.
21	CDIEN	**Conversion Done Interrupt Enable.** 0: Disable the single conversion done interrupt. 1: Enable the single conversion done interrupt.
20	CMPEN	**Threshold Comparator Enable.** Enables the threshold comparator, which compares the accumulated capacitive sensing conversion output to the value stored in CSTH. When enabled, the comparator threshold event will stop the continuous conversion modes. This bit also enables the compare threshold interrupt. 0: Disable the threshold comparator. 1: Enable the threshold comparator.
19	Reserved	Must write reset value.
18	PMEF	**Pin Monitor Event Flag.** Hardware sets this flag if any measurement retries have occurred due to a pin monitor event. This bit remains set until cleared by firmware. 0: A retry did not occur due to a pin monitor event during the last conversion. 1: A retry occurred due to a pin monitor event during the last conversion.
17:16	PMMD	**Pin Monitor Mode.** 00: Always retry on a pin state change. 01: Retry up to twice on consecutive bit cycles. 10: Retry up to four times on consecutive bit cycles. 11: Ignore monitored signal state change.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 13.3. CAPSENSE0_CONTROL Register Bit Descriptions

Bit	Name	Function
15:12	CSCM	**Start of Conversion Mode Select.** 0000: The CSnT0 trigger source starts conversions. 0001: The CSnT1 trigger source starts conversions. 0010: The CSnT2 trigger source starts conversions. 0011: The CSnT3 trigger source starts conversions. 0100: The CSnT4 trigger source starts conversions. 0101: The CSnT5 trigger source starts conversions. 0110: The CSnT6 trigger source starts conversions. 0111: The CSnT7 trigger source starts conversions. 1000: The CSnT8 trigger source starts conversions. 1001: The CSnT9 trigger source starts conversions. 1010: The CSnT10 trigger source starts conversions. 1011: The CSnT11 trigger source starts conversions. 1100: The CSnT12 trigger source starts conversions. 1101: The CSnT13 trigger source starts conversions. 1110: The CSnT14 trigger source starts conversions. 1111: The CSnT15 trigger source starts conversions.
11	MCEN	**Multiple Channel Enable.** When this bit is set to 1, the channels selected by SCANEN are internally shorted together and the combined node is selected as the measurement input. This mode can be used to detect a capacitance change on multiple channels using a single conversion. 0: Disable the multiple channel measurement feature. 1: Enable the multiple channel measurement feature.
10:8	ACCMD	**Accumulator Mode Select.** When ACCMD is set to a value other than 0, the hardware will not set the conversion done or end-of-scan flags until all samples are taken and accumulated. 000: Accumulate 1 sample. 001: Accumulate 4 samples. 010: Accumulate 8 samples. 011: Accumulate 16 samples. 100: Accumulate 32 samples. 101: Accumulate 64 samples. 110-111: Reserved.
7:6	CNVR	**Conversion Rate.** 00: Conversions last 12 internal CAPSENSE clocks and results are 12 bits in length. 01: Conversions last 13 internal CAPSENSE clocks and results are 13 bits in length. 10: Conversions last 14 internal CAPSENSE clocks and results are 14 bits in length. 11: Conversions last 16 internal CAPSENSE clocks and results are 16 bits in length.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 13.3. CAPSENSE0_CONTROL Register Bit Descriptions

Bit	Name	Function
5:4	CMD	**Conversion Mode Select.** 00: Single Conversion Mode: One conversion occurs on a single channel. 01: Single Scan Mode: One conversion on each channel selected by SCANEN occurs. An end-of-scan interrupt indicates all channels have been measured. 10: Continuous Single Conversion Mode: Continuously converts on a single channel. This operation ends only if the module is disabled (CSEN = 0) or if a compare threshold event occurs (CMPI = 1). 11: Continuous Scan Mode: Continuously loops through and converts on all the channels selected by SCANEN. This operation ends only if the module is disabled (CSEN = 0) or if a compare threshold event occurs (CMPI = 1).
3	CMPPOL	**Digital Comparator Polarity Select.** 0: The digital comparator generates an interrupt if the conversion is greater than the CSTH threshold. 1: The digital comparator generates an interrupt if the conversion is less than or equal to the CSTH threshold.
2	BIASEN	**Bias Enable.** Enable the bias to the CAPSENSEn core. Firmware must wait at least 2 microseconds after enabling the bias before starting a conversion. Disabling the bias will save power if no measurement is being taken. 0: Disable the bias. 1: Enable the bias.
1	CSEN	**Module Enable.** The bit should be set to 1 after setting BIASEN to 1. The capacitive sensing module should be disabled after each measurement operation, and all settings in the module should be adjusted while the module is disabled. The CMPI and EOSI flags can only be cleared by clearing this bit to 0. 0: Disable the capacitive sensing module. 1: Enable the capacitive sensing module.
0	BUSYF	**Start and Busy Flag.** Hardware sets this bit to 1 when a conversion in progress and clears this bit automatically when the conversion completes. Firmware can set this bit to 1 to initiate a conversion if BUSYF is selected as the start of conversion source by the CSCM field.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Register 13.2. CAPSENSE0_MODE: Measurement Mode

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R													RW		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved	DTSEL			Reserved	IASEL			RAMPSEL		Reserved			CGSEL		
Type	R	RW			R	RW			RW		RW		R	RW		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Register ALL Access Address

CAPSENSE0_MODE = 0x4002_3010

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 13.4. CAPSENSE0_MODE Register Bit Descriptions

Bit	Name	Function
31:15	Reserved	Must write reset value.
14:12	DTSEL	**Discharge Time Select.** This field adjusts the primary CAPSENSEn reset time. For most touch-sensitive switches, the default (fastest) value is sufficient.
11	Reserved	Must write reset value.
10:8	IASEL	**Output Current Select.** These bits allow firmware to adjust the output current used to charge up the capacitive sensor element. For most touch-sensitive switches, the default (highest) current is sufficient.
7:6	RAMPSEL	**Ramp Selection.** These bits are used to compensate capacitive sensing conversions for circuits requiring slower ramp times. For most touch-sensitive switches, the default (fastest) value is sufficient.
5:3	Reserved	Must write reset value.
2:0	CGSEL	**Capacitance Gain Select.** These bits select the gain applied to the capacitance measurement. Lower gain values increase the size of the capacitance that can be measured with the CAPSENSE module. The capacitance gain is equivalent to CGSEL + 1.

Register 13.3. CAPSENSE0_DATA: Measurement Data

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	DATA															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

CAPSENSE0_DATA = 0x4002_3020

Table 13.5. CAPSENSE0_DATA Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15:0	DATA	**Capacitive Sensing Data.** This field stores the result from the last completed 12- to 16-bit capacitive sensing conversion.

Register 13.4. CAPSENSE0_SCAN: Channel Scan Enable

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	SCANEN															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

CAPSENSE0_SCAN = 0x4002_3030

Table 13.6. CAPSENSE0_SCAN Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15:0	SCANEN	**Channel Scan Enable.** This field selects which capacitive sensing channels will be included in the scan for single scan or continuous scan modes. When MCEN is set to 1, the selected channels will be connected together internally for one capacitive measurement. A 1 in this field enables the corresponding CSn.x channels. For example, a 1 in bit position 3 will enable the scan operation on CSn.3.

Register 13.5. CAPSENSE0_CSTH: Compare Threshold

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	CSTH															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

CAPSENSE0_CSTH = 0x4002_3040

Table 13.7. CAPSENSE0_CSTH Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15:0	CSTH	**Compare Threshold.** When CMPEN is set to 1, this value is compared against the capacitive sensing conversion result to generate an interrupt according to the polarity setting of CMPPOL.

Register 13.6. CAPSENSE0_MUX: Mux Channel Select

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved								CSDISC	Reserved			CSMX			
Type	R								RW	R			RW			
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Register ALL Access Address
CAPSENSE0_MUX = 0x4002_3050

Table 13.8. CAPSENSE0_MUX Register Bit Descriptions

Bit	Name	Function
31:8	Reserved	Must write reset value.
7	CSDISC	**Channel Disconnect.** 0: Connect the capacitive sensing circuit to the selected channel. 1: Disconnect the capacitive sensing input channel.
6:4	Reserved	Must write reset value.

Table 13.8. CAPSENSE0_MUX Register Bit Descriptions

Bit	Name	Function
3:0	CSMX	**Mux Channel Select.** This field selects one of the 16 input channels for the capacitive sensing conversion. This selection is only used in single conversion and continuous single modes. In scan modes, input selection is determined by the SCANEN field in the SCAN register. 0000: Select CSn.0. 0001: Select CSn.1. 0010: Select CSn.2. 0011: Select CSn.3. 0100: Select CSn.4. 0101: Select CSn.5. 0110: Select CSn.6. 0111: Select CSn.7. 1000: Select CSn.8. 1001: Select CSn.9. 1010: Select CSn.10. 1011: Select CSn.11. 1100: Select CSn.12. 1101: Select CSn.13. 1110: Select CSn.14. 1111: Select CSn.15.

13.14. CAPSENSE0 Register Memory Map

Table 13.9. CAPSENSE0 Memory Map

Register Name	CAPSENSE0_SCAN	CAPSENSE0_DATA	CAPSENSE0_MODE	CAPSENSE0_CONTROL
ALL Address	0x4002_3030	0x4002_3020	0x4002_3010	0x4002_3000
Access Methods	ALL	ALL	ALL | SET | CLR	ALL | SET | CLR
Bit 31	Reserved	Reserved	Reserved	Reserved
Bit 30				
Bit 29				
Bit 28				
Bit 27				
Bit 26				EOSI
Bit 25				CDI
Bit 24				CMPI
Bit 23				Reserved
Bit 22				EOSIEN
Bit 21				CDIEN
Bit 20				CMPEN
Bit 19				Reserved
Bit 18				PMEF
Bit 17				PMMD
Bit 16				
Bit 15	SCANEN	DATA	DTSEL	CSCM
Bit 14				
Bit 13				
Bit 12				
Bit 11			Reserved	MCEN
Bit 10			IASEL	ACCMD
Bit 9				
Bit 8				
Bit 7			RAMPSEL	CNVR
Bit 6				
Bit 5			Reserved	CMD
Bit 4				
Bit 3			CGSEL	CMPPOL
Bit 2				BIASEN
Bit 1				CSEN
Bit 0				BUSYF

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

Table 13.9. CAPSENSE0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CAPSENSE0_CSTH	0x4002_3040	ALL	Reserved																CSTH															
CAPSENSE0_MUX	0x4002_3050	ALL	Reserved																								CSDISC	Reserved				CSMX		

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

14. Comparator (CMP0 and CMP1)

This section describes the Comparator (CMP) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the CMP block, which is used by both CMP0 and CMP1 on all device families covered in this document.

14.1. Comparator Features

The comparator takes two analog input voltages and outputs the relationship between these voltages (less than or greater than). The comparator module includes the following features:

- Multiple sources for the positive and negative poles, including VDD, VREF, and up to 8 I/O pins.
- Two outputs are available: a digital synchronous latched output and a digital asynchronous raw output.
- Programmable hysteresis and response time.
- Falling or rising edge interrupt options on the comparator output.

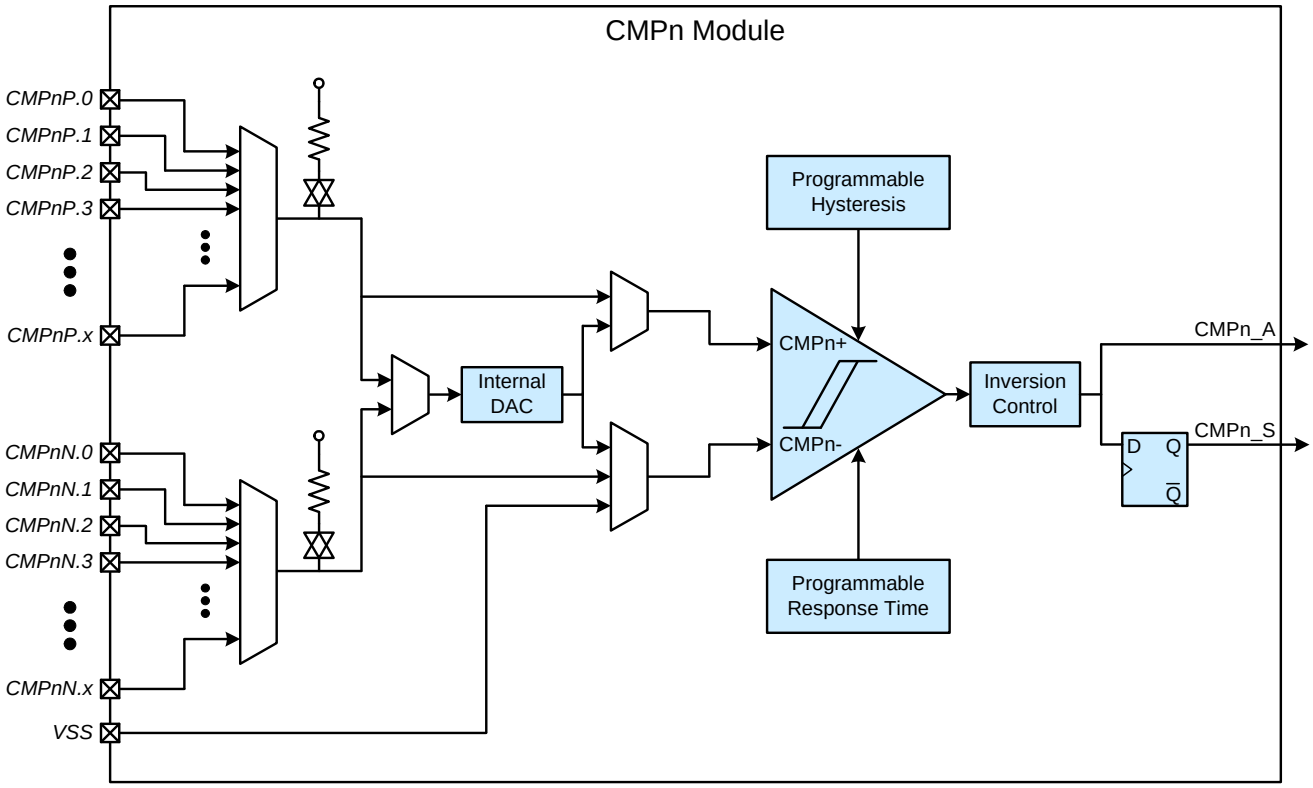


Figure 14.1. Comparator Block Diagram

14.2. Overview

The comparator offers programmable response time and hysteresis, an analog input multiplexer, and two outputs that are optionally available at the port bank pins: a digital synchronous latched output (CMPn_S) and a digital asynchronous raw output (CMPn_A). The asynchronous CMPn_A signal is available even when the system clock is not active, allowing the comparator to operate and generate an output when the device is in some low power modes.

The comparator also features an internal DAC that may be used to create a firmware-programmable threshold voltage. The comparator DAC output level (DACLVL) field configures the DAC output voltage, and the input mux select (INMUX) field enables the DAC output to the input of the comparator.

14.3. Inputs

When enabled (CMPEN = 1), the comparator performs an analog comparison of the voltage levels at its positive (CMPn+) and negative (CMPn–) inputs. The CMPn+ and CMPn– inputs connect to select internal supplies or external port pins through analog input multiplexers, configured by the positive analog input mux select (PMUX) and negative analog input mux select (NMUX) fields in the MODE register. Any port bank pins selected as comparator inputs should be configured as analog inputs, as described in the port configuration section. The CMP0 and CMP1 input channels vary between different package options, and are shown in table Table 14.1 and Table 14.2. Note that for some selections, other device circuitry must be enabled.

The CMPn+ and CMPn– inputs have weak internal pull-ups that may be enabled by setting the positive input weak pullup enable (PWPUEN) and negative input weak pullup enable (NWPUEN) bits.

Table 14.1. CMP0 Input Channels

CMP0 Input	CMP0 Input Description	SiM3U1x7/C1x7 Pin Name	SiM3U1x6/C1x6 Pin Name	SiM3U1x4/C1x4 Pin Name
CMP0P.0	External Positive Input	PB2.11	PB2.3	PB1.2
CMP0P.1	External Positive Input	PB2.13	PB3.0	PB3.0
CMP0P.2	External Positive Input	PB3.0	PB3.2	PB3.2
CMP0P.3	External Positive Input	PB3.2	PB3.4	Reserved
CMP0P.4	External Positive Input	PB3.4	PB3.6	Reserved
CMP0P.5	External Positive Input	PB3.6	PB3.8	Reserved
CMP0P.6	External Positive Input	PB3.8	Reserved	Reserved
CMP0P.7	External Positive Input	PB3.10	Reserved	Reserved
CMP0P.8	Internal Positive Input	Voltage at VREGIN / 4 (1)		
CMP0P.9	Internal Positive Input	EXTVREG0 Current Sense (2)		
CMP0P.10	Internal Positive Input	1.8 V Output of LDO		
CMP0P.11	Internal Positive Input	Supply for Internal Oscillators (VDDOSC)		
CMP0P.12	Internal Positive Input	VREF		

Table 14.1. CMP0 Input Channels (Continued)

CMP0 Input	CMP0 Input Description	SiM3U1x7/C1x7 Pin Name	SiM3U1x6/C1x6 Pin Name	SiM3U1x4/C1x4 Pin Name
CMP0P.13	Internal Positive Input	VIO		
CMP0P.14	Internal Positive Input	Voltage at VIOHD / 4 (3)		
CMP0N.0	External Negative Input	PB2.12	PB2.2	PB1.3
CMP0N.1	External Negative Input	PB2.14	PB3.1	PB3.1
CMP0N.2	External Negative Input	PB3.1	PB3.3	PB3.3
CMP0N.3	External Negative Input	PB3.3	PB3.5	Reserved
CMP0N.4	External Negative Input	PB3.5	PB3.7	Reserved
CMP0N.5	External Negative Input	PB3.7	PB3.9	Reserved
CMP0N.6	External Negative Input	PB3.9	Reserved	Reserved
CMP0N.7	External Negative Input	PB3.11	Reserved	Reserved
CMP0N.8	Internal Negative Input	VDD		
CMP0N.9	Internal Negative Input	VREF		

Notes:
1. The VREGIN/4 option requires the VREGIN sense circuitry to be enabled within the VREG0 block (VREG0_CONTROL.SENSEEN).
2. The EXTVREG0 Current Sense option requires the current sense circuit in the EXTVREG0 block to be enabled (EXTVREG0_CSCONTROL.ADCISNSEN).
3. The VIOHD/4 option requires the VIOHD divider to be enabled within the PBHD block (PBHD4_PBDRV.PBVTRKEN).

Table 14.2. CMP1 Input Channels

CMP1 Input	CMP1 Input Description	SiM3U1x7/C1x7 Pin Name	SiM3U1x6/C1x6 Pin Name	SiM3U1x4/C1x4 Pin Name
CMP1P.0	External Positive Input	PB2.11	PB2.3	PB1.2
CMP1P.1	External Positive Input	PB2.13	PB3.0	PB3.0
CMP1P.2	External Positive Input	PB3.0	PB3.2	PB3.2
CMP1P.3	External Positive Input	PB3.2	PB3.4	Reserved
CMP1P.4	External Positive Input	PB3.4	PB3.6	Reserved

Table 14.2. CMP1 Input Channels (Continued)

CMP1 Input	CMP1 Input Description	SiM3U1x7/C1x7 Pin Name	SiM3U1x6/C1x6 Pin Name	SiM3U1x4/C1x4 Pin Name
CMP1P.5	External Positive Input	PB3.6	PB3.8	Reserved
CMP1P.6	External Positive Input	PB3.8	Reserved	Reserved
CMP1P.7	External Positive Input	PB3.10	Reserved	Reserved
CMP1P.8	Internal Positive Input	Voltage at VREGIN / 4 (1)		
CMP1P.9	Internal Positive Input	EXTVREG0 Current Sense (2)		
CMP1P.10	Internal Positive Input	1.8 V Output of LDO		
CMP1P.11	Internal Positive Input	Supply for Internal Oscillators (VDDOSC)		
CMP1P.12	Internal Positive Input	VREF		
CMP1P.13	Internal Positive Input	VIO		
CMP1P.14	Internal Positive Input	Voltage at VIOHD / 4 (3)		
CMP1N.0	External Negative Input	PB2.12	PB2.2	PB1.3
CMP1N.1	External Negative Input	PB2.14	PB3.1	PB3.1
CMP1N.2	External Negative Input	PB3.1	PB3.3	PB3.3
CMP1N.3	External Negative Input	PB3.3	PB3.5	Reserved
CMP1N.4	External Negative Input	PB3.5	PB3.7	Reserved
CMP1N.5	External Negative Input	PB3.7	PB3.9	Reserved
CMP1N.6	External Negative Input	PB3.9	Reserved	Reserved
CMP1N.7	External Negative Input	PB3.11	Reserved	Reserved
CMP1N.8	Internal Negative Input	VDD		
CMP1N.9	Internal Negative Input	VREF		

Notes:

1. The VREGIN/4 option requires the VREGIN sense circuitry to be enabled within the VREG0 block (VREG0_CONTROL.SENSEEN).
2. The EXTVREG0 Current Sense option requires the current sense circuit in the EXTVREG0 block to be enabled (EXTVREG0_CSCONTROL.ADCISNSEN).
3. The VIOHD/4 option requires the VIOHD divider to be enabled within the PBHD block (PBHD4_PBDRV.PBVTRKEN).

The CMPm module offers several different input modes shown in Table 14.3, configurable via the input mux select (INMUX) field.

Table 14.3. Comparator Input Modes

INMUX Value	Input to CMPn+	Input to CMPn-
0	positive analog input mux output	negative analog input mux output
1	positive analog input mux output	VSS
2	DAC output with positive analog input mux connected to internal DAC reference	negative analog input mux output
3	positive analog input mux output	DAC output with negative analog input mux connected to internal DAC reference

Figure 14.2, Figure 14.3, Figure 14.4, and Figure 14.5 illustrate each of these configurations.

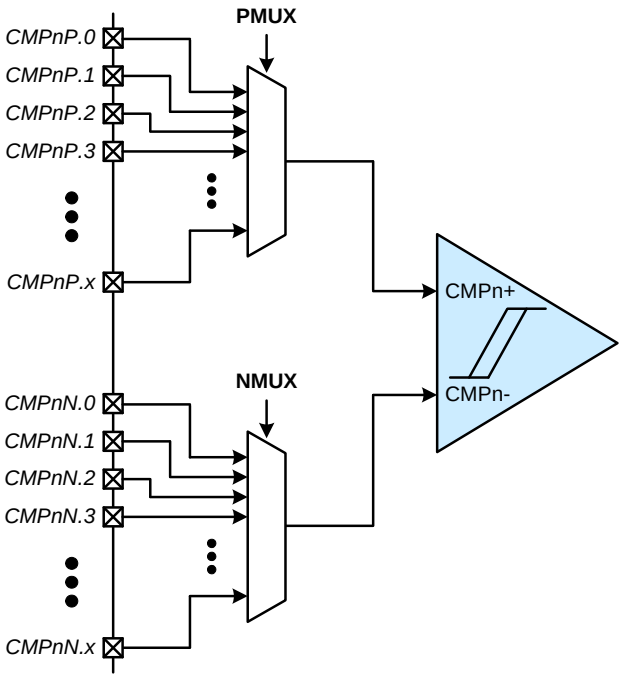


Figure 14.2. Comparator Input Mode 0 (INMUX = 0)

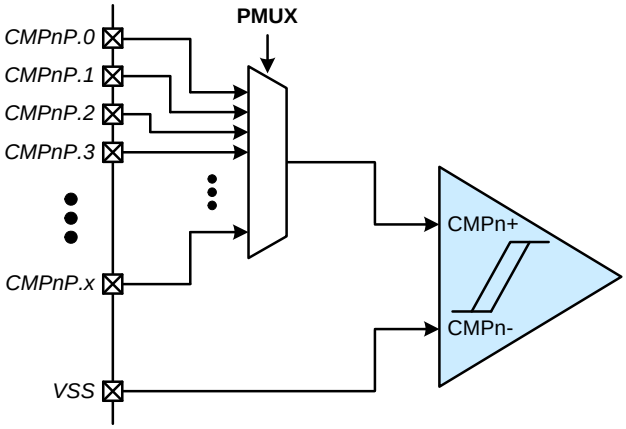


Figure 14.3. Comparator Input Mode 1 (INMUX = 1)

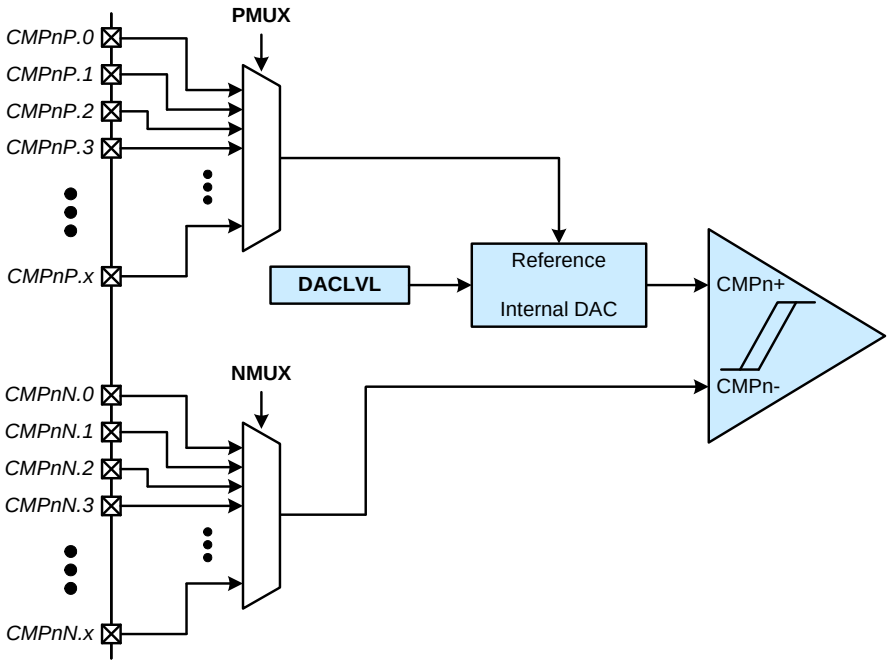


Figure 14.4. Comparator Input Mode 2 (INMUX = 2)

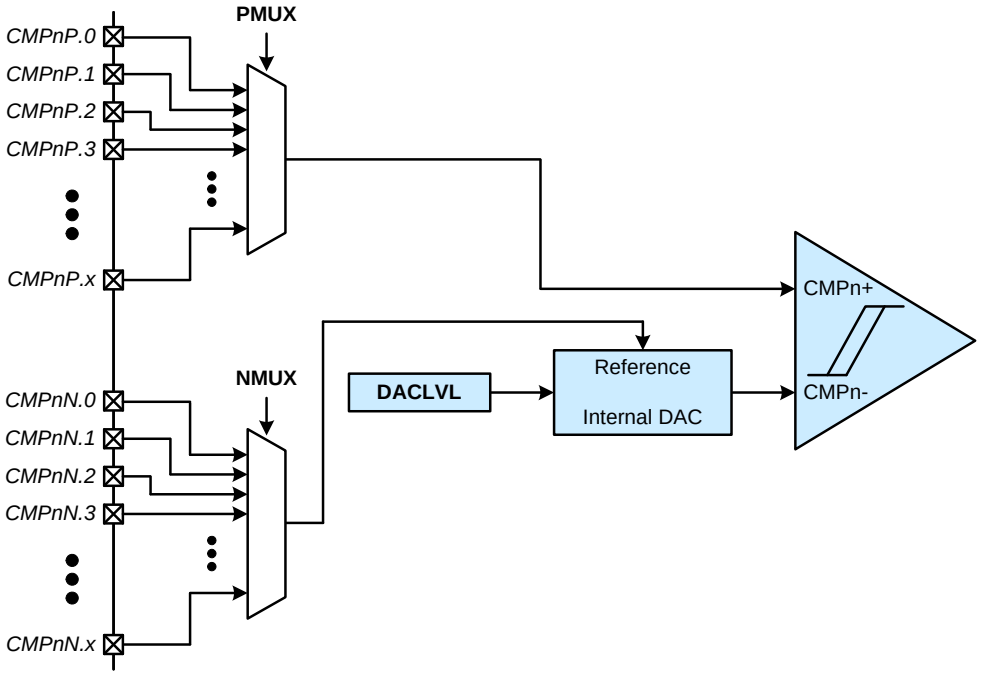


Figure 14.5. Comparator Input Mode 3 (INMUX = 3)

14.4. Outputs

When enabled (CMPEN = 1) and non-inverted (INVEN = 0), the comparator will output a 1 if the voltage at the positive input (CMPn+) is higher than the voltage at the negative input (CMPn–). Firmware can set the INVEN bit to 1 to invert the comparator output polarity.

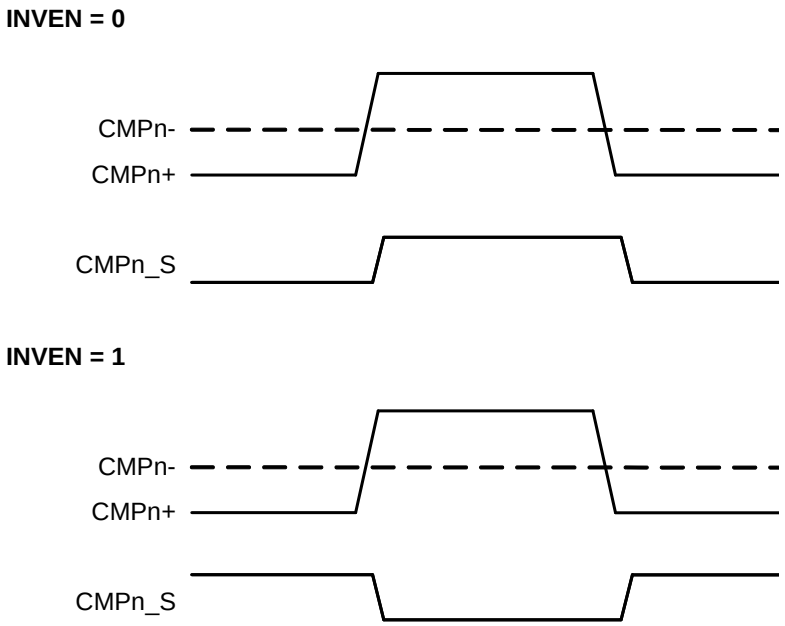


Figure 14.6. Output Configurations

The synchronous output (CMPn_S) is synchronized with the APB clock and may be polled by firmware, used as an interrupt source, or routed to a port bank pin through the crossbar. The asynchronous comparator output (CMPn_A) is not synchronized with the APB clock and is used by low power mode wake-up logic and reset decision logic.

When the module is disabled (CMPEN = 0), the comparator will output a static 0 (INVEN = 0) or 1 (INVEN = 1).

14.5. Response Time

The comparator response time may be configured in firmware using the CMPMD field. There are four settings available, from Mode 0 (fastest response time and highest power) to Mode 3 (slowest response time and lowest power). For lower power applications, selecting a slower response time reduces the module's active supply current.

The comparator rising edge and falling edge response times are typically not equal. The device data sheet contains comparator timing and supply current specifications.

14.6. Hysteresis

The comparator module features programmable hysteresis that can be used to stabilize the comparator output when a transition occurs on the input. The comparator output will not transition until the voltage on the comparator CMPn+ input exceeds the threshold voltage on the CMPn– input by the amount programmed in the positive hysteresis control (CMPHYP) field. Similarly, the comparator output will not transition until the voltage on the comparator CMPn+ input falls below the threshold voltage on the CMPn– input by the amount programmed in the negative hysteresis control (CMPHYN) field.

Figure 14.7 illustrates the programmable comparator hysteresis.

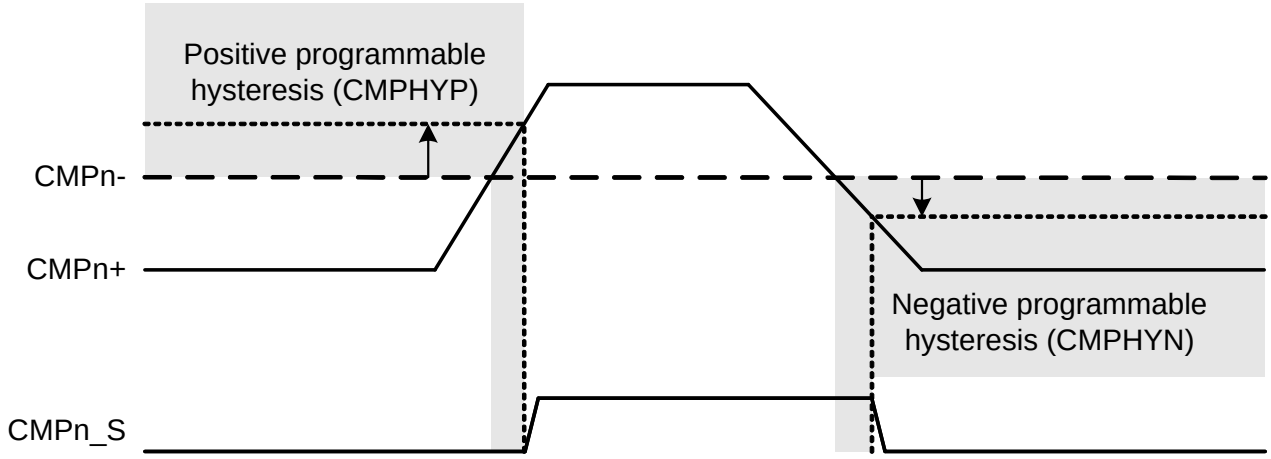


Figure 14.7. Comparator Hysteresis

Both the positive and negative hysteresis control fields may be configured for 5, 10, or 20 mV hysteresis. Firmware can disable positive or negative hysteresis by clearing CMPHYP or CMPHYN to 0.

14.7. Interrupts and Flags

The rising-edge (CMPRI) and falling-edge (CMPFI) interrupt flags allow firmware to determine whether the comparator had a rising-edge or falling-edge output transition. The rising-edge interrupt enable (RIEN) and falling-edge interrupt enable (FIEN) bits can enable these flags as an interrupt source. The module hardware sets the CMPRI and CMPFI flags when a corresponding rising or falling edge is detected, regardless of the interrupt enable state. Once set, these flags remain set until cleared by firmware.

The comparator may detect false rising and falling edges during power-on or after changes are made to the hysteresis or response time control bits. Firmware should explicitly clear the rising-edge and falling-edge flags to 0 a short time after enabling the comparator or changing the mode bits.

14.8. CMP0 and CMP1 Registers

This section contains the detailed register descriptions for CMP0 and CMP1 registers.

Register 14.1. CMPn_CONTROL: Module Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	CMPEN	CMPOUT	Reserved													
Type	RW	R	R													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved	CMPRI	CMPFI	Reserved												
Type	R	RW	RW	R												
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

CMP0_CONTROL = 0x4001_F000
CMP1_CONTROL = 0x4002_0000
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 14.4. CMPn_CONTROL Register Bit Descriptions

Bit	Name	Function
31	CMPEN	**Comparator Enable.** 0: Disable the comparator. 1: Enable the comparator.
30	CMPOUT	**Output State.** This bit indicates the logic level of the comparator output. When INVEN is set, it directly inverts the meaning of this bit. 0: Voltage on CMP+ < CMP- (INVEN = 0). 1: Voltage on CMP+ > CMP- (INVEN = 0).
29:15	Reserved	Must write reset value.
14	CMPRI	**Rising Edge Interrupt Flag.** 0: No comparator rising edge has occurred since this flag was last cleared. 1: A comparator rising edge occurred since last flag was cleared.

Notes:
1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 14.4. CMPn_CONTROL Register Bit Descriptions

Bit	Name	Function
13	CMPFI	**Falling Edge Interrupt Flag.** 0: No comparator falling edge has occurred since this flag was last cleared. 1: A comparator falling edge occurred since last flag was cleared.
12:0	Reserved	Must write reset value.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Register 14.2. CMPn_MODE: Input and Module Mode

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved	INVEN	Reserved		CMPHYP		CMPHYN		PWPUEN	NWPUEN	DACLVL					
Type	RW	RW	R		RW		RW		RW	RW	RW					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved	RIEN	FIEN	Reserved	CMPMD		INMUX		PMUX				NMUX			
Type	R	RW	RW	R	RW		RW		RW				RW			
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

CMP0_MODE = 0x4001_F010

CMP1_MODE = 0x4002_0010

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 14.5. CMPn_MODE Register Bit Descriptions

Bit	Name	Function
31	Reserved	Must write reset value.
30	INVEN	**Invert Comparator Output Enable.** 0: Do not invert the comparator output. 1: Invert the comparator output.
29:28	Reserved	Must write reset value.
27:26	CMPHYP	**Positive Hysteresis Control.** 00: Disable positive hysteresis. 01: Set positive hysteresis to 5 mV. 10: Set positive hysteresis to 10 mV. 11: Set positive hysteresis to 20 mV.
25:24	CMPHYN	**Negative Hysteresis Control.** 00: Disable negative hysteresis. 01: Set negative hysteresis to 5 mV. 10: Set negative hysteresis to 10 mV. 11: Set negative hysteresis to 20 mV.
23	PWPUEN	**Positive Input Weak Pullup Enable.** 0: Disable the positive input weak pull up. 1: Enable the positive input weak pull up.

Table 14.5. CMPn_MODE Register Bit Descriptions

Bit	Name	Function
22	NWPUEN	**Negative Input Weak Pullup Enable.** 0: Disable the negative input weak pull up. 1: Enable the negative input weak pull up.
21:16	DACLVL	**Comparator DAC Output Level.** These bits control the comparator reference DAC's output voltage. The voltage is given by: $\text{DAC Output} = \text{VREF} \times \left(\frac{\text{DACLVL}}{64}\right)$ where VREF is the positive or negative comparator mux selection, as defined by INMUX.
15	Reserved	Must write reset value.
14	RIEN	**Rising Edge Interrupt Enable.** 0: Disable the comparator rising edge interrupt. 1: Enable the comparator rising edge interrupt.
13	FIEN	**Falling Edge Interrupt Enable.** 0: Disable the comparator falling edge interrupt. 1: Enable the comparator falling edge interrupt.
12	Reserved	Must write reset value.
11:10	CMPMD	**Comparator Mode.** 00: Mode 0 (fastest response time, highest power consumption). 01: Mode 1. 10: Mode 2. 11: Mode 3 (slowest response time, lowest power consumption).
9:8	INMUX	**Input MUX Select.** 00: Connects the NMUX signal to CMP- and the PMUX signal to CMP+. 01: Connects VSS to CMP- and the PMUX signal to CMP+. 10: Connects the NMUX signal to CMP-, the PMUX signal to the Comparator DAC voltage reference, and the DAC output to CMP+. 11: Connects the PMUX signal to CMP+, the NMUX signal to the Comparator DAC voltage reference, and the DAC output to CMP-.

Table 14.5. CMPn_MODE Register Bit Descriptions

Bit	Name	Function
7:4	PMUX	**Positive Input Select.** 0000: Select CMPnP.0. 0001: Select CMPnP.1. 0010: Select CMPnP.2. 0011: Select CMPnP.3. 0100: Select CMPnP.4. 0101: Select CMPnP.5. 0110: Select CMPnP.6. 0111: Select CMPnP.7. 1000: Select CMPnP.8. 1001: Select CMPnP.9. 1010: Select CMPnP.10. 1011: Select CMPnP.11. 1100: Select CMPnP.12. 1101: Select CMPnP.13. 1110: Select CMPnP.14. 1111: Select CMPnP.15.
3:0	NMUX	**Negative Input Select.** 0000: Select CMPnN.0. 0001: Select CMPnN.1. 0010: Select CMPnN.2. 0011: Select CMPnN.3. 0100: Select CMPnN.4. 0101: Select CMPnN.5. 0110: Select CMPnN.6. 0111: Select CMPnN.7. 1000: Select CMPnN.8. 1001: Select CMPnN.9. 1010: Select CMPnN.10. 1011: Select CMPnN.11. 1100: Select CMPnN.12. 1101: Select CMPnN.13. 1110: Select CMPnN.14. 1111: Select CMPnN.15.

14.9. CMPn Register Memory Map

Table 14.6. CMPn Memory Map

Register Name	ALL Offset	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CMPn_CONTROL	0x0	ALL | SET | CLR	CMPEN	CMPOUT	Reserved															CMPRI	CMPFI	Reserved													
CMPn_MODE	0x10	ALL | SET | CLR	Reserved	INVEN	Reserved		CMPHYP		CMPHYN		PWPUEN	NWPUEN	DACLVL						Reserved	RIEN	FIEN	Reserved	CMPMD		INMUX		PMUX				NMUX			

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: CMP0 = 0x4001_F000, CMP1 = 0x4002_0000

15. Cyclic Redundancy Check (CRC0)

This section describes the Cyclic Redundancy Check (CRC) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the CRC block, which is used by all device families covered in this document.

15.1. CRC Features

The CRC module includes the following features:

- Support for four common polynomials (one 32-bit and three 16-bit options).
- Byte-level bit reversal for the CRC input.
- Byte-order reorientation of words for the CRC input.
- Word or half-word bit reversal of the CRC result.
- Ability to configure and seed an operation in a single register write.
- Support for single-cycle parallel (unrolled) CRC computation for 32- or 8-bit blocks.
- Capability to CRC 32 bits of data per peripheral bus (APB) clock.

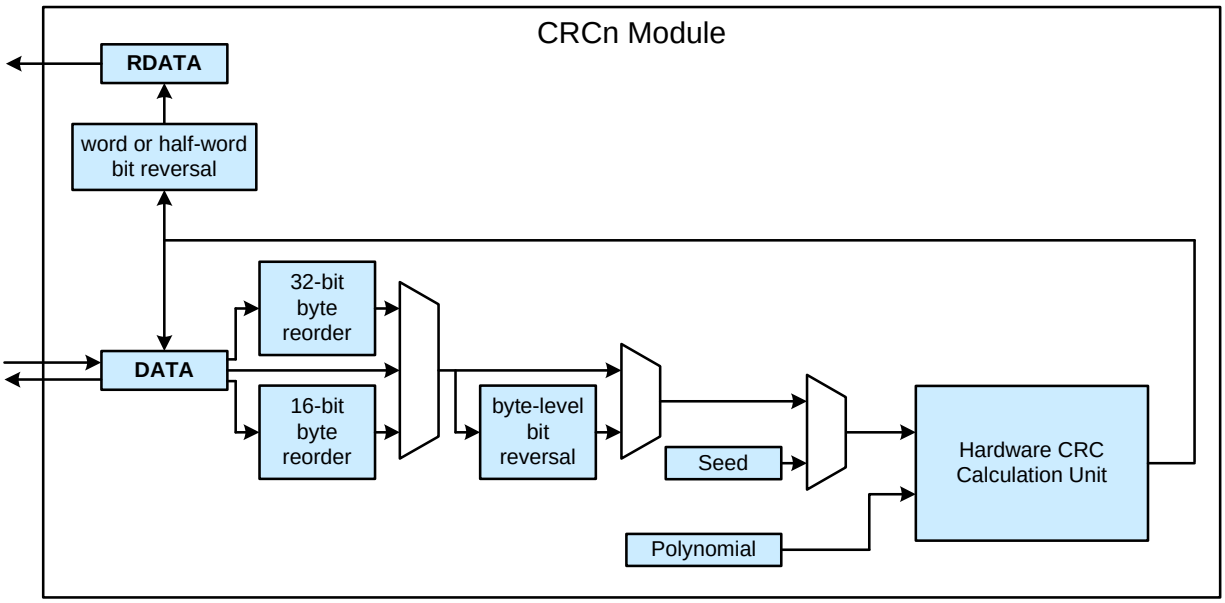


Figure 15.1. CRC0 Block Diagram

15.2. Overview

The CRC module is designed to provide hardware calculations for flash memory verification and communications protocols.

The CRC module supports four common polynomials. The supported 32-bit polynomial is 0x04C11DB7 (IEEE 802.3). The three supported 16-bit polynomials are 0x1021 (CCITT–16), 0x3D65 (IEC16–MBus), and 0x8005 (ZigBee, 802.15.4, and USB).

The CRC module will automatically detect non-word writes (byte, half-word, and non-aligned byte) and adjust the internal calculation to only process the least-significant byte. Any word writes are treated as an entire word and all 32 bits will be considered in the CRC calculation update. The BMDEN bit can be used to force the CRC module to treat all writes as bytes.

15.3. Interrupts

The CRC module does not have any interrupts associated with it.

15.4. DMA Configuration and Usage

A DMA channel may be used to transfer data into the CRC module. The CRC DATA register only supports word and byte writes. Half-word writes to this register will be treated as byte writes. The recommended DMA usage model is to use the DMA to transfer all available words of data and use firmware writes to capture any remaining bytes. To write data into the CRC module, the DMA must move one word of data at a time from the source location in memory to the internal DATA register in non-incrementing mode. Firmware can then write any remaining bytes to the DATA register and read the CRC result from the DATA register. The CRC DATA register should not be directly written to or read from when targeted by a DMA channel.

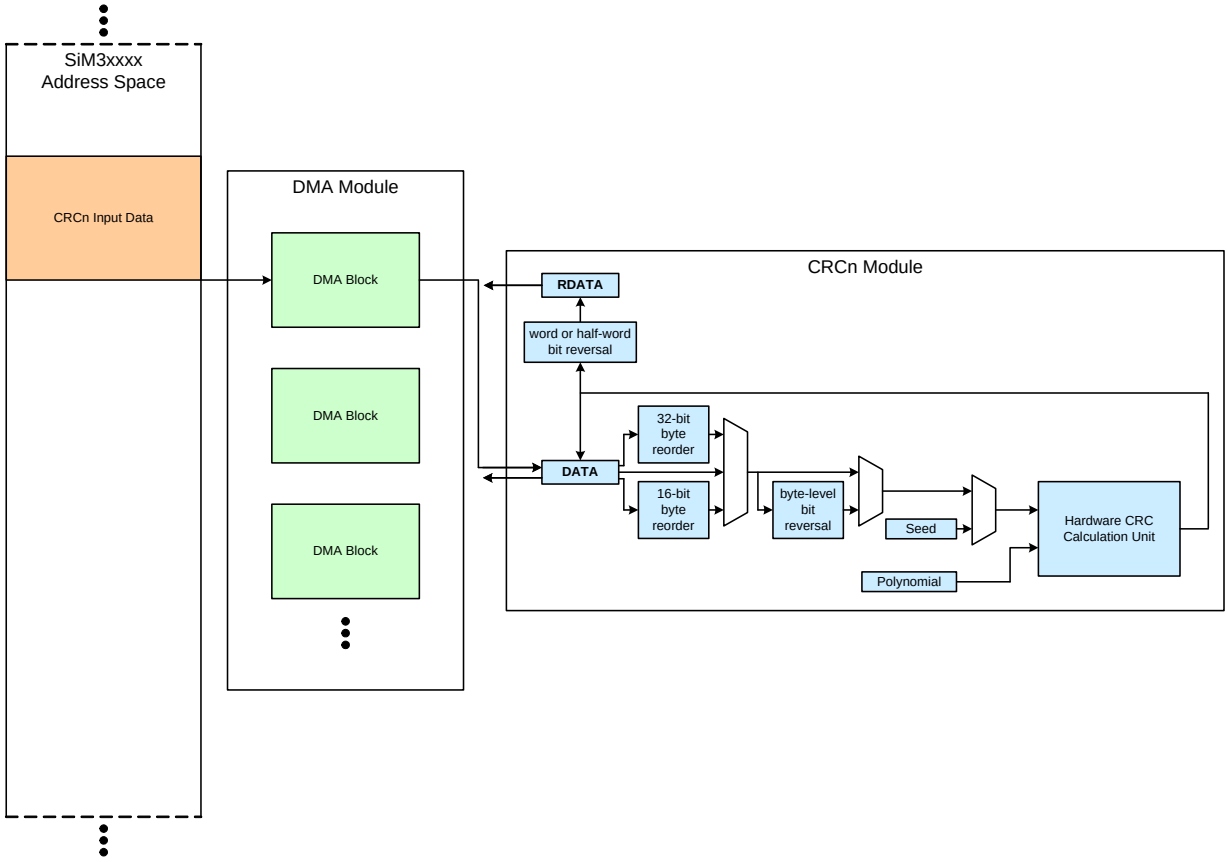


Figure 15.2. CRC DMA Configuration

For the CRC module, the DMA channel should be set up as follows:

- Source size and Destination Size are 2 for a word transfer.
- Number of transfers is N - 1, where N is the number of 4-byte words.
- RPOWER = 0 (1 word transfer per transaction).

To start a DMA operation with the CRC module out of any device reset:

1. Set up the DMA channel for the CRC Input.
2. Configure the CRC peripheral operation in the CONTROL register.
3. Start the DMA transfer.
4. Wait for DMA completion interrupt.
5. Write any remaining bytes to the DATA register to complete the CRC calculation for the memory block.
6. Read the CRC result from DATA or the bit-reversed CRC result from RDATA.

15.5. Byte-Level Bit Reversal and Byte Reordering

The byte-level bit reversal and byte reordering operations occur before the data is used in the CRC calculation. These operations can occur on words or half words. The hardware ignores the ORDER field with any byte writes or operations with byte mode enabled (BMDEN = 1), but the bit reversal settings (BBREN) are still applied to the byte.

Using an ORDER bits setting of 10b for 32-bit big endian byte order with BBREN set to 1 for byte-level bit reversal allows big endian data to be treated like 32-bit little endian data with MSB-first bit ordering, as shown in Figure 15.3.

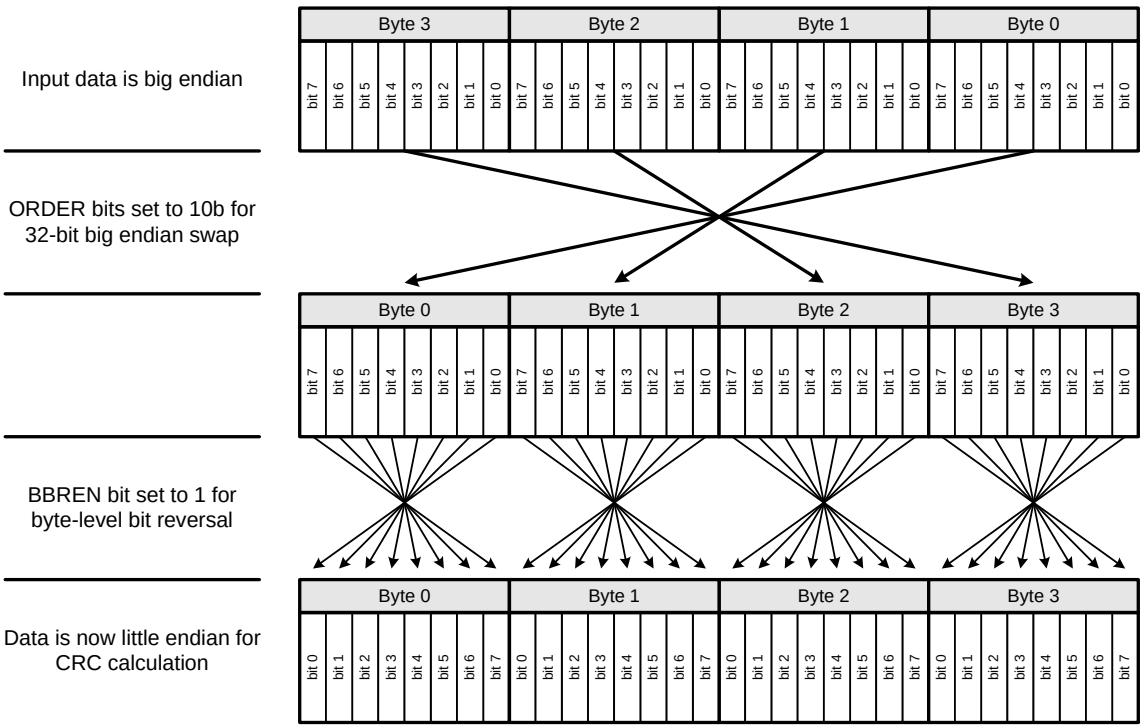


Figure 15.3. CRC Data Ordering Example—Big Endian to Little Endian

Using an ORDER bits setting of 01b for 16-bit big endian byte order with BBREN set to 1 for byte-level bit reversal allows big endian data to be treated by 16-bit little endian data with MSB-first bit ordering, as shown in Figure 15.4.

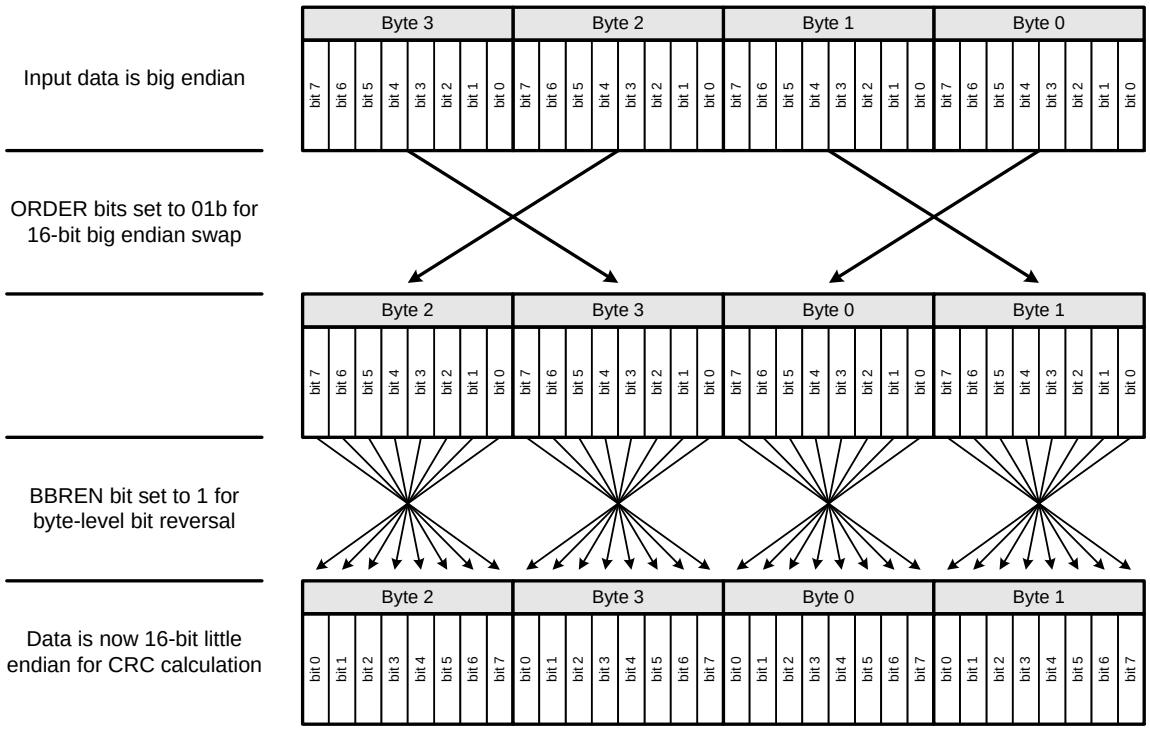


Figure 15.4. CRC Data Ordering Example—Big Endian to 16-bit Little Endian

Assuming a word input byte order of B3 B2 B1 B0, the values used in the CRC calculation for the various settings of the byte-level bit reversal and byte reordering are shown in Table 15.1.

Table 15.1. Byte-Level Bit Reversal and Byte Reordering Results (B3 B2 B1 B0 Input Order)

Original CRC Calculation Method			Equivalent CRC Settings		
Polynomial Width (bits)	**Byte Order**	**Bit Order (MSB/LSB first)**	**ORDER bits setting**	**BBREN bit setting**	**Input to CRC calculation**
32	little endian	LSB	00 (none)	0	B3 B2 B1 B0
32	little endian	MSB	10 (32-bit)	1	‘B0 ‘B1 ‘B2 ‘B3
32	big endian	LSB	10 (32-bit)	0	B0 B1 B2 B3
32	big endian	MSB	00 (none)	1	‘B3 ‘B2 ‘B1 ‘B0
16	little endian	LSB	00 (none)	0	B3 B2 B1 B0
16	little endian	MSB	01 (16-bit)	1	‘B2 ‘B3 ‘B0 ‘B1
16	big endian	LSB	01 (16-bit)	0	B2 B3 B0 B1
16	big endian	MSB	00 (none)	1	‘B3 ‘B2 ‘B1 ‘B0
8	—	LSB	—	0	XX XX XX B0
8	—	MSB	—	1	XX XX XX ‘B0

Notes:

3. X indicates a “don’t care.”

4. Bn is the byte field within the word.

5. ‘Bn is the bit-reversed byte field within the word.

15.6. CRC0 Registers

This section contains the detailed register descriptions for CRC0 registers.

Register 15.1. CRC0_CONTROL: Module Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved				ORDER		BBREN	BMDEN	Reserved		POLYSEL		Reserved	CRCEN	SEED	SINITEN
Type	R				RW		RW	RW	R		RW		R	RW	W	W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

CRC0_CONTROL = 0x4002_8000

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 15.2. CRC0_CONTROL Register Bit Descriptions

Bit	Name	Function
31:12	Reserved	Must write reset value.
11:10	ORDER	**Input Processing Order.** Controls the input order of the bytes in a 32-bit word to the CRC calculation unit. This setting has no effect in byte mode. 00: No byte reorientation (output is same order as input). 01: Swap for 16-bit big endian order (input: B3 B2 B1 B0, output: B2 B3 B0 B1). 10: Swap for 32-bit big endian order (input: B3 B2 B1 B0, output: B0 B1 B2 B3). 11: Reserved.
9	BBREN	**Byte-Level Bit Reversal Enable.** Controls the input order of the bits in each byte to the CRC calculation unit. When set to 1, byte-level bit reversal is enabled and the bits in each byte are reversed. 0: No byte-level bit reversal (input is same order as written). 1: Byte-level bit reversal enabled (the bits in each byte are reversed).
8	BMDEN	**Byte Mode Enable.** Enables byte mode, where all writes to the DATA register are considered as 8-bit writes. When set to 1, only the least-significant byte of the data word will be used for the CRC calculation. 0: Disable byte mode (word/byte width is determined automatically by the hardware). 1: Enable byte mode (all writes are considered as bytes).

Table 15.2. CRC0_CONTROL Register Bit Descriptions

Bit	Name	Function
7:6	Reserved	Must write reset value.
5:4	POLYSEL	**Polynomial Selection.** Selects the polynomial used by the CRC calculations. 00: Select 32-bit polynomial: 0x04C11DB7. 01: Select 16-bit polynomial: 0x1021. 10: Select 16-bit polynomial: 0x3D65. 11: Select 16-bit polynomial: 0x8005.
3	Reserved	Must write reset value.
2	CRCEN	**CRC Enable.** Enables CRC functionality. Until this bit is set to 1, writes to the DATA register do not result in CRC operations. Byte reorientation and bit reversal is still active even if CRCEN is cleared to 0. 0: Disable CRC operations. 1: Enable CRC operations.
1	SEED	**Seed Setting.** Determines the value of all CRC register bits when a 1 is written to the SINITEN bit. This bit always reads back as 0. 0: CRC seed value is all 0's (0x00000000) 1: CRC seed value is all 1's (0xFFFFFFFF).
0	SINITEN	**Seed Initialization Enable.** When this bit is set to 1, all bits of the CRC register are initialized to the value determined by the SEED bit setting. This bit always reads back as 0. 0: Do not initialize the CRC module to the value set by the SEED bit. 1: Initialize the CRC module to the value set by the SEED bit.

Register 15.2. CRC0_DATA: Input/Result Data

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	DATA[31:16]															
Type	RW															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	DATA[15:0]															
Type	RW															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Register ALL Access Address

CRC0_DATA = 0x4002_8010

Table 15.3. CRC0_DATA Register Bit Descriptions

Bit	Name	Function
31:0	DATA	**Input/Result Data.** If the CRCEN bit is set to 1, data written to this register will be used for CRC calculations, byte-level bit reversal, and byte reordering. The current CRC result can be read from this register at any time. If the CRCEN bit is cleared to 0, data written to this register will be used for byte-level bit reversal and byte reordering only. No delay is required between writing the data and reading the CRC or reordering result.

Notes:

1. Reads of this register modify the state of hardware. Debug logic should take care when reading this register.
2. The access methods for this register are restricted. Do not use half-word access methods on this register.

Register 15.3. CRC0_RDATA: Bit-Reversed Output Data

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	RDATA[31:16]															
Type	R															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	RDATA[15:0]															
Type	R															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Register ALL Access Address

CRC0_RDATA = 0x4002_8020

Table 15.4. CRC0_RDATA Register Bit Descriptions

Bit	Name	Function
31:0	RDATA	**Bit-Reversed Output Data.** This register provides the bit reversed version of the DATA register. When the 32-bit CRC polynomial is selected, the reversal occurs on the entire 32-bit word. When a 16-bit CRC polynomial is selected, the least-significant half-word (bits [15:0]) is reversed.

15.7. CRC0 Register Memory Map

Table 15.5. CRC0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
CRC0_CONTROL	0x4002_8000	ALL | SET | CLR	Reserved																				ORDER		BBREN	BMDEN	Reserved		POLYSEL		Reserved	CRCEN	SEED	SINITEN
CRC0_DATA	0x4002_8010	ALL	DATA																															
CRC0_RDATA	0x4002_8020	ALL	RDATA																															

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

16. DMA Controller (DMACTRL0)

This section describes the DMA Controller (DMACTRL) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the DMATRL block, which is used by all device families covered in this document.

16.1. DMA Controller Features

The DMA Controller module includes the following features:

- Utilizes ARM PrimeCell uDMA architecture.
- Implements 16 channels.
- DMA crossbar supports direct peripheral data requests and maps peripherals to each channel.
- Supports primary, alternate, and scatter-gather channel transfer structures to implement various types of transfers.
- Access allowed to all APB and AHB memory space.

The DMA facilitates autonomous peripheral operation, allowing the core to finish tasks more quickly without spending time polling or waiting for peripherals to interrupt. This helps reduce the overall power consumption of the system, as the device can spend more time in low-power modes.

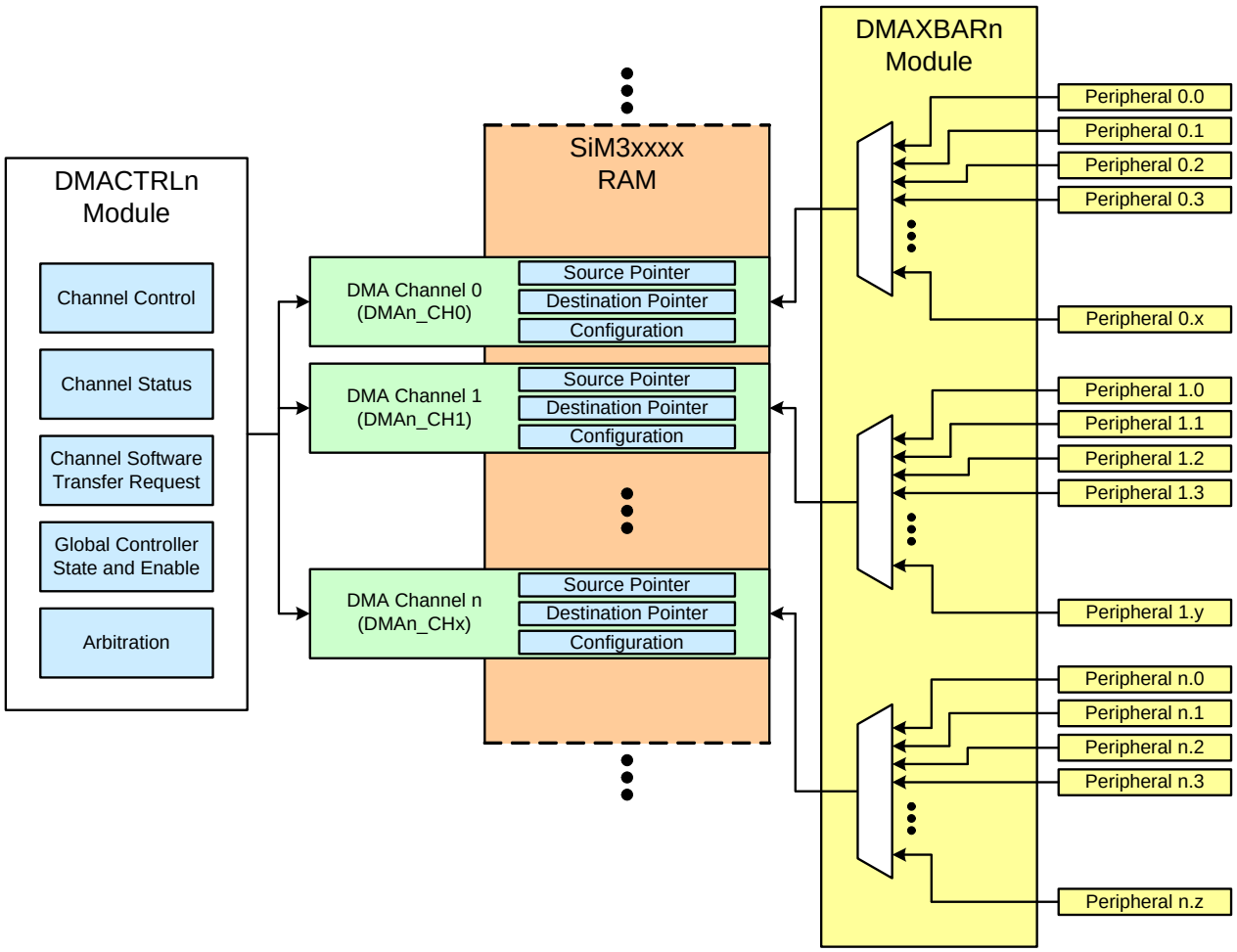


Figure 16.1. DMACTRL and DMACH Block Diagram

16.2. Overview

The DMA controller provides a single access point for all 16 DMA channels and the global DMA controls. The controller is also responsible for handling arbitration between channels.

Each channel has separate enables, alternate enables, masks, software requests, programmable priority, and status flags. The channels operate independently, but have a fixed arbitration order.

The channels have controls and flags in the DMACTRL registers. In addition, each channel has several transfer structures stored in memory that describe the data transfer in detail. Each channel can have a primary, alternate, or scatter-gather structures. The BASEPTR and ABASEPTR registers point to the starting address of these structures in memory. Firmware sets the BASEPTR field, and the controller hardware automatically sets the ABASEPTR field based on the number of channels implemented in the module.

The NUMCHAN field in the STATUS register reports the number of channels implemented on a device. The STATE field reports the current status of the DMA controller, and the DMAENS bit indicates whether the global DMA enable is set.

16.3. Interrupts

Each DMA channel has a separate interrupt vector, and a channel will generate an interrupt at the end of the current transfer. Firmware can enable or disable the interrupts in the NVIC.

16.4. Configuring a DMA Channel

To configure a DMA channel for a data transfer:

1. Enable the DMA module (DMAEN = 1).
2. Set the address location of the channel transfer structures (BASEPTR) according to the restrictions in Table 16.1.
3. Create the primary and any alternate or scatter-gather data structures in memory for the desired transfer.
4. Enable the DMA channel using the CHENSET register.
5. Submit a request to start the transfer.

For software-initiated transfers, a request starts by setting the channel's software request in the CHSWRCN register. It is recommended that firmware set the channel request mask (CHREQMSET) for channels using software-initiated transfers to avoid any peripherals connected to the channel from requesting DMA transfers.

For peripheral transfers, firmware should configure the peripheral for the DMA transfer and set the device's DMA crossbar (DMAXBAR) to map a DMA channel to the peripheral. The peripheral will request data as needed. The channel request mask (CHREQMCLR) must be cleared for the channel to use peripheral transfers.

The CHALTSET register can set a DMA channel to use the alternate structure instead of the primary structure. Firmware can use the CHALTCLR register to set the channel back to the primary structure. The controller automatically updates the CHALTSET fields to indicate which structure is in use during transfers that use the alternate structure (ping-pong and scatter-gather).

16.5. DMA Channel Transfer Structures

Each channel has transfer structures stored in memory that describe the data transfer in detail. Each structure is composed of four 32-bit words in memory organized as follows:

1. Source End Pointer (word 1): The address of the last source data in the transfer.
2. Destination End Pointer (word 2): The last destination address of the transfer.
3. Channel Configuration (word 3): Configuration details for the transfer.
4. Alignment padding (word 4): Not used by the DMA controller. Firmware my use this word for any purpose.

Each channel can have a primary, alternate, and scatter-gather structures. The primary and alternate structures are organized in contiguous blocks in memory for each of the channels. The spacing for these structures is fixed, so any unused channels must still be accounted for when placing structures in memory. In addition to the fixed structure, the base address (BASEPTR) supports between 22- and 27-bit addresses, depending on the number of channels implemented. The primary structures must be placed at the start of an address block sized for both the primary and alternate structures. Table 16.1 shows the valid base pointer addresses for each range of implemented channels.

The scatter-gather structures are more flexible and can appear anywhere in memory.

Table 16.1. Valid Base Pointer Addresses

Number of Channels Implemented	Base Pointer Size (BASEPTR)	Valid Primary Channel Transfer Structure Addresses	Required Number of Bytes (Primary and Alternate)
1	27 bits [31:5]	multiples of 16 (0x00000010)	32
2	26 bits [31:6]	multiples of 32 (0x00000020)	64
3-4	25 bits [31:7]	multiples of 64 (0x00000040)	128
5-8	24 bits [31:8]	multiples of 128 (0x00000080)	256
9-16	23 bits [31:9]	multiples of 256 (0x00000100)	512
17-32	22 bits [31:10]	multiples of 512 (0x00000200)	1024

Channel 0's primary structure begins at address offset 0x0000, Channel 1's primary structure starts at offset 0x0010, and so on. The alternate structures begin after the last primary structure location for the number of implemented channels, regardless of whether or not the channels are in use.

Firmware originally sets the channel configuration descriptor; the DMA controller will modify this word as the transfer progresses, so firmware should not access this descriptor until any active transfers for the channel complete.

Figure 16.2 shows the fixed memory configuration for the structures.

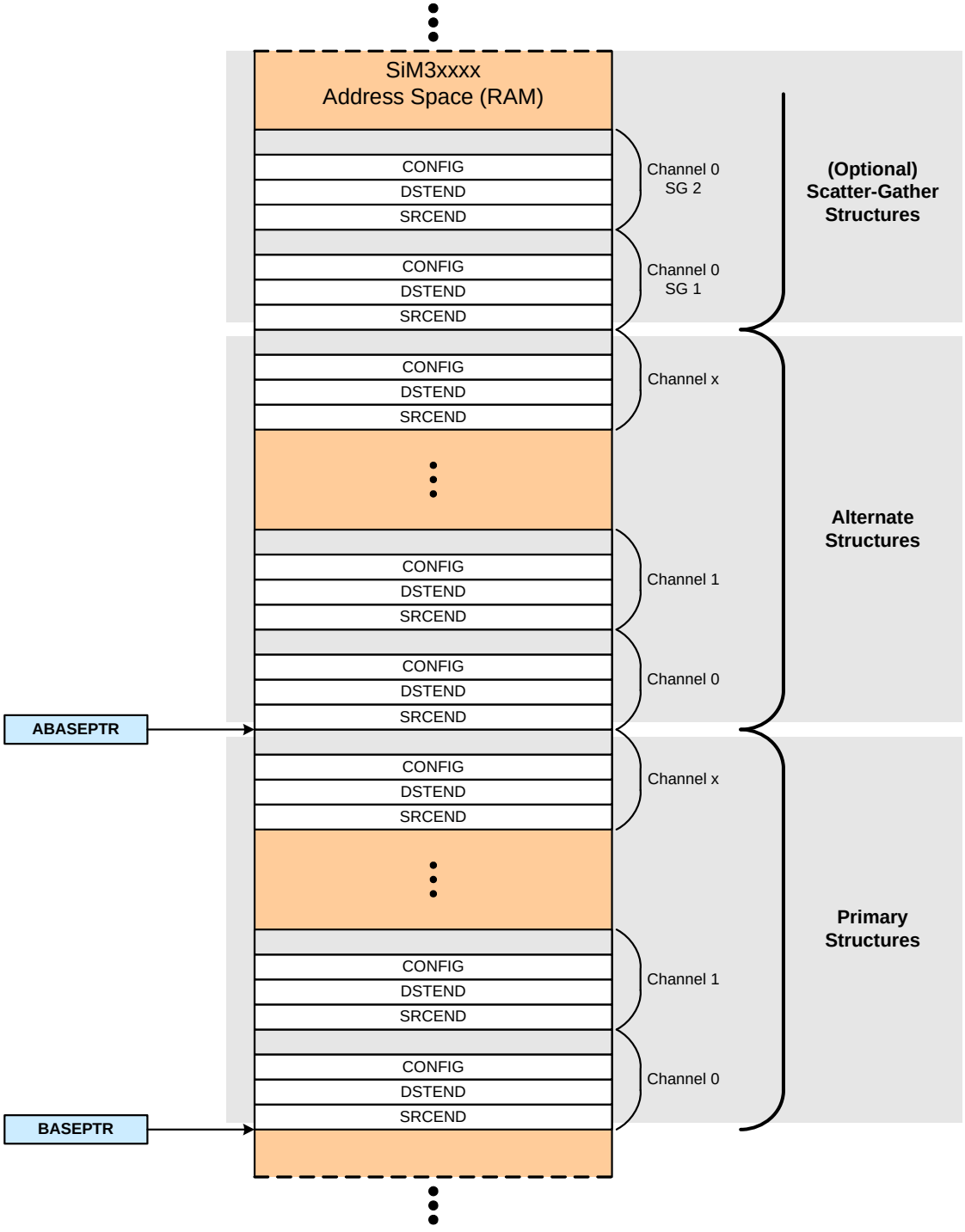


Figure 16.2. Channel Transfer Structure Memory Configuration

16.5.1. Channel Transfer Structure Descriptors

Table 16.2, Table 16.3, Table 16.4 describe the source end pointer, destination pointer, and configuration descriptors for the primary, alternate, and scatter-gather DMA channel structures.

Table 16.2. DMA0_CHx_SRCEND: Source End Pointer

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	SRCEND[31:16]															
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	SRCEND[15:0]															

Address in Channel Transfer Structure: 0x0000

Bit	Name	Function
31:0	SRCEND	**Source End Pointer.** This field is the address of the last source data in the DMA transfer.

Table 16.3. DMA0_CHx_DSTEND: Destination End Pointer

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	DSTEND[31:16]															
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	DSTEND[15:0]															

Address in Channel Transfer Structure: 0x0004

Bit	Name	Function
31:0	DSTEND	**Destination End Pointer.** This field is the last destination address of the DMA transfer.

Table 16.4. DMA0_CHx_CONFIG: Channel Configuration

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	DSTAIMD		DSTSIZE		SRCAIMD		SRCSIZE		Reserved						RPOWER[3:2]	
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	RPOWER[1:0]		NCOUNT										Reserved	TMD		

Address in Channel Transfer Structure: 0x0008

Bit	Name	Function
31:30	DSTAIMD	**Destination Address Increment Mode.** This field must be set to a value that's equal to or greater than the DSTSIZE setting. 00: The destination address increments by one byte after each data transfer. 01: The destination address increments by one half-word after each data transfer. 10: The destination address increments by one word after each data transfer. 11: The destination address does not increment.
29:28	DSTSIZE	**Destination Data Size Select.** The destination size (DSTSIZE) must equal the source size (SRCSIZE). 00: Each DMA destination data transfer writes a byte. 01: Each DMA destination data transfer writes a half-word. 10: Each DMA destination data transfer writes a word. 11: Reserved.
27:26	SRCAIMD	**Source Address Increment Mode.** This field must be set to a value that's equal to or greater than the SRCSIZE setting. 00: The source address increments by one byte after each data transfer. 01: The source address increments by one half-word after each data transfer. 10: The source address increments by one word after each data transfer. 11: The source address does not increment.
25:24	SRCSIZE	**Source Data Size Select.** The destination size (DSTSIZE) must equal the source size (SRCSIZE). 00: Each DMA source data transfer reads a byte. 01: Each DMA source data transfer reads a half-word. 10: Each DMA source data transfer reads a word. 11: Reserved.
23:18	Reserved	Must write 0 to this field.

Bit	Name	Function
17:14	RPOWER	**Transfer Size Select.** This field determines the number of data transfers between each DMA channel re-arbitration. The number of data transfers is given by: $\text{Number of Transfers} = 2^{\text{RPOWER}}$ This field is ignored for peripherals that support single data requests only. A value of 0 for RPOWER should be used for channels interfacing with these types of peripherals.
13:4	NCOUNT	**Transfer Total.** This field is the total number of transfers for the DMA channel. The total number is NCOUNT + 1, so software requiring a total of 4 transfers would set the NCOUNT field to 3. The DMA controller decrements this field as transfers are made.
3	Reserved	Must write 0 to this bit.
2:0	TMD	**Transfer Mode.** 000: Stop the DMA channel. 001: Use the Basic transfer type (single structure only). 010: Use the Auto-Request transfer type (single structure only). 011: Use the Ping-Pong transfer type (primary and alternate structures). 100: Use the Memory Scatter-Gather Primary transfer type (primary, alternate, and scattered structures). 101: Use the Memory Scatter-Gather Alternate transfer type (primary, alternate, and scattered structures). 110: Use the Peripheral Scatter-Gather Primary transfer type (primary, alternate, and scattered structures). 111: Use the Peripheral Scatter-Gather Alternate transfer type (primary, alternate, and scattered structures).

16.6. Transfer Types

The DMA channels support five transfer types: basic, auto-request, ping-pong, memory scatter-gather, and peripheral scatter-gather. Table 16.5 shows the memory requirements for each transfer type.

Table 16.5. Transfer Memory Requirements

Transfer Type	Transfer Structures Required			Memory (RAM) Required (bytes)		
	Primary	Alternate	Scatter-Gather	5-8 Channels Implemented	9-16 Channels Implemented	17-32 Channels Implemented
Basic	✓			128	256	512
Auto-Request	✓			128	256	512
Ping-Pong	✓	✓		256	512	1024
Memory Scatter-Gather	✓	✓	✓	256 + SG	512 + SG	1024 + SG
Peripheral Scatter-Gather	✓	✓	✓	256 + SG	512 + SG	1024 + SG

16.6.1. Basic Transfers

The basic transfer type uses only one structure (primary or alternate). In this mode, the channel will make NCOUNT + 1 data moves in 2^{RPOWER} bursts. Each data request moves one 2^{RPOWER} set of data. The number of requests required for a transfer is:

$$\text{Number of Requests} = \frac{NCOUNT + 1}{2^{RPOWER}}$$

Equation 16.1. Number of Requests for Basic Transfers

Any data remaining can be transferred by firmware or use an extra DMA data request.

After the final data transfer:

1. The DMA channel will write the primary structure TMD field with 0.
2. The primary structure NCOUNT field will contain 0.
3. The controller automatically disables the channel (the channel bit in CHENSET will read 0).

Figure 16.3 illustrates the DMA memory structures for a basic transfer.

This transfer type is recommended for peripheral to memory or memory to peripheral transfers.

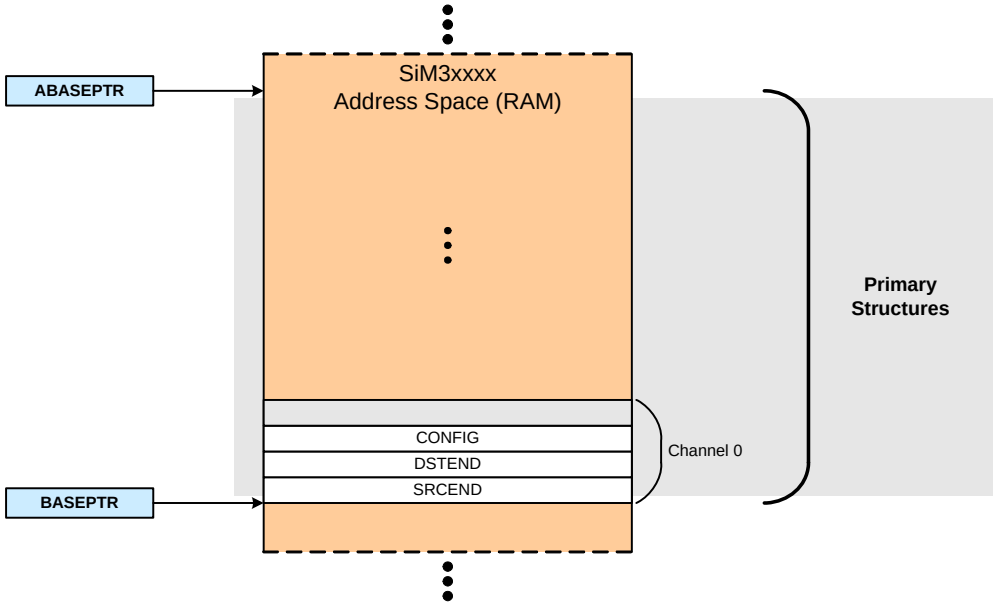


Figure 16.3. Basic and Auto-Request Transfer Memory Configuration

16.6.2. Auto-Request Transfers

Auto-request transfers use only one structure (primary or alternate). This transfer type only requires one data request to transfer all of the data. The controller will arbitrate as normal (every 2^{RPOWER} transfers), and a channel interrupt will occur when the transfer completes. This transfer type is recommended for memory to memory transfers.

After the final data transfer:

1. The DMA channel will write the primary structure TMD field with 0.
2. The primary structure NCOUNT field will contain 0.
3. The controller automatically disables the channel (the channel bit in CHENSET will read 0).

The auto-request memory configuration is identical to the basic transfer shown in Figure 16.3.

16.6.3. Ping-Pong Transfers

Ping-pong transfers use both the primary and alternate channel structures. When the channel completes the transfer described by the first structure, it clears the TMD field in the original structure to 0 and toggles to point to the other structure. A channel interrupt will occur to allow firmware to update the completed transfer's structure, as the ping-pong operation will stop without intervention.

As with basic transfers, each 2^{RPOWER} data moves require a new data request. The number of requests is given by Equation 16.1.

Figure 16.4 shows an example where a channel's primary structure has an RPOWER of 1 with an NCOUNT of 3 and the alternate structure has an RPOWER of 0 with an NCOUNT of 4. These structures are both configured to move words (DSTSIZE and SRCSIZE set to 2) in ping-pong mode (TMD = 3).

Figure 16.5 illustrates the ping-pong memory configuration.

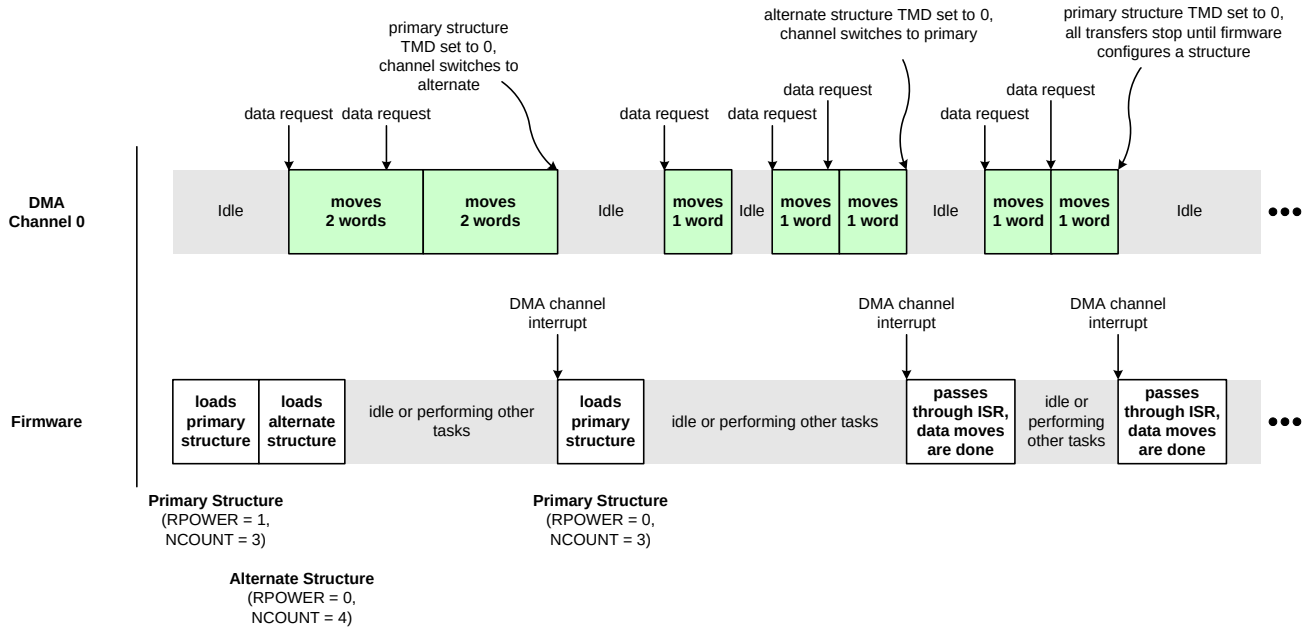


Figure 16.4. Ping-Pong Transfer Example

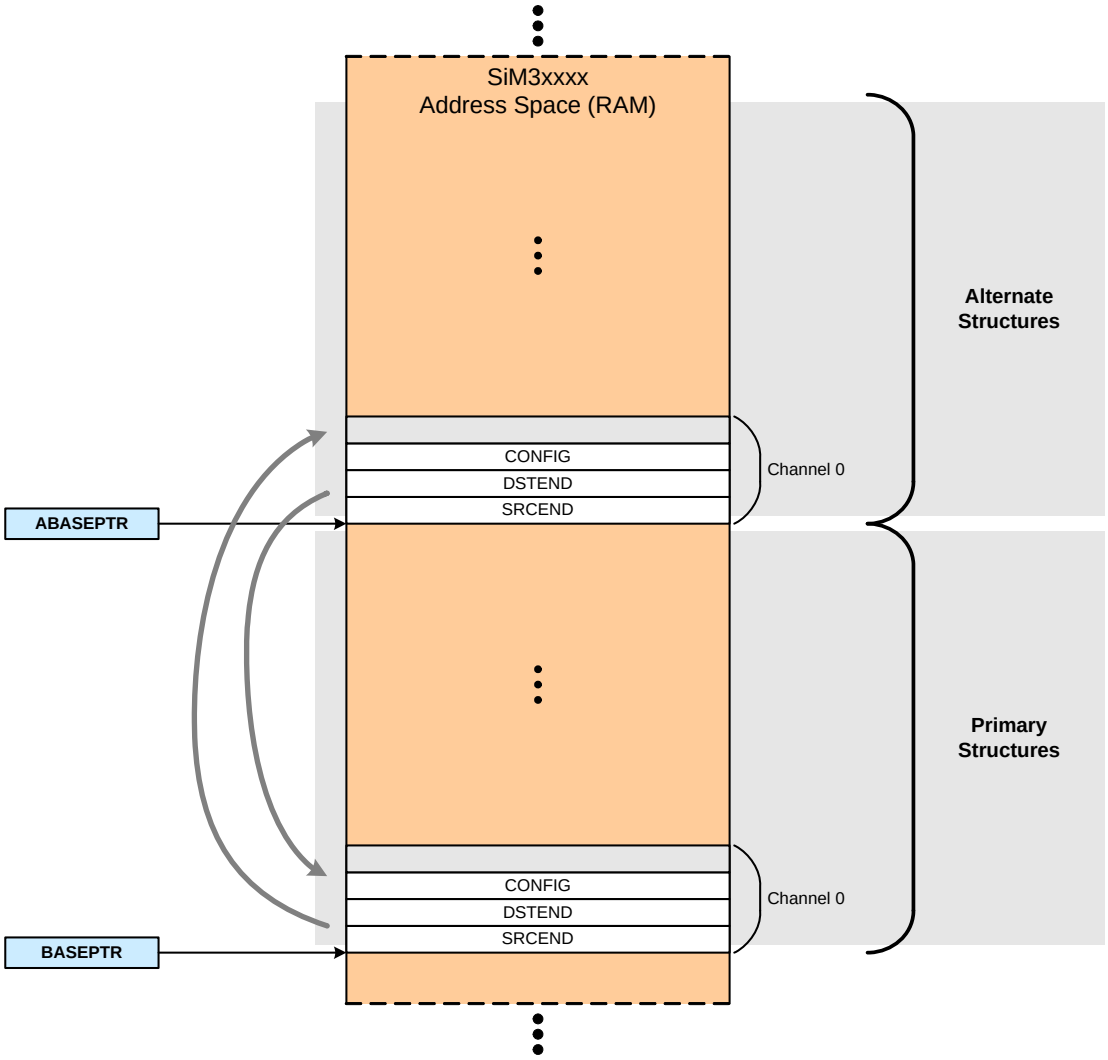


Figure 16.5. Ping-Pong Transfer Memory Configuration

16.6.4. Memory Scatter-Gather Transfers

The memory scatter-gather transfer uses primary, alternate, and scatter-gather structures. This transfer type allows a DMA channel to be set for multiple transfers at once without core intervention at the price of extra memory for the scatter-gather structures.

The primary structure in this mode contains the number and location of the scatter-gather structures. The primary structure should be programmed as follows:

1. Memory scatter-gather primary mode (TMD = 4).
2. RPOWER = 2.
3. NCOUNT set to the value specified by Equation 16.2.
4. SRCEND is set to the location of the last word of all the scatter-gather structures.
5. DSTEND is set to the location of the last word in the single alternate structure.

$$NCOUNT = (\text{Number of SG Structures} \times 4) - 1$$

Equation 16.2. NCOUNT Value for Scatter-Gather Transfers

The scatter-gather structures must be stacked contiguously in memory. The channel will copy the scatter-gather structures into the alternate structure location and execute them one by one. The scatter-gather structures should be programmed to memory scatter-gather alternate mode (TMD = 5), except for the last structure, which should use the basic or auto-request transfer types (TMD = 1 or 2).

Once started, the DMA channel execution process is as follows:

1. Copy scatter-gather 1 (SG1) to the alternate structure.
2. Jump to the alternate structure and execute.
3. Jump back to the primary structure.
4. Copy scatter-gather 2 (SG2) to the alternate structure.
5. Jump to the alternate structure and execute.
6. Jump back to the primary structure.

The channel will continue in this pattern until the channel encounters a scatter-gather structure set to a basic or auto-request transfer.

Only one data request is required to execute all of the scattered transactions. The channel interrupt will occur once the last scatter-gather structure (programmed to a basic transfer) executes, if enabled. Arbitration occurs every 2^{RPOWER} of the scatter-gather structures.

Figure 16.6 shows the memory scatter-gather memory configuration.

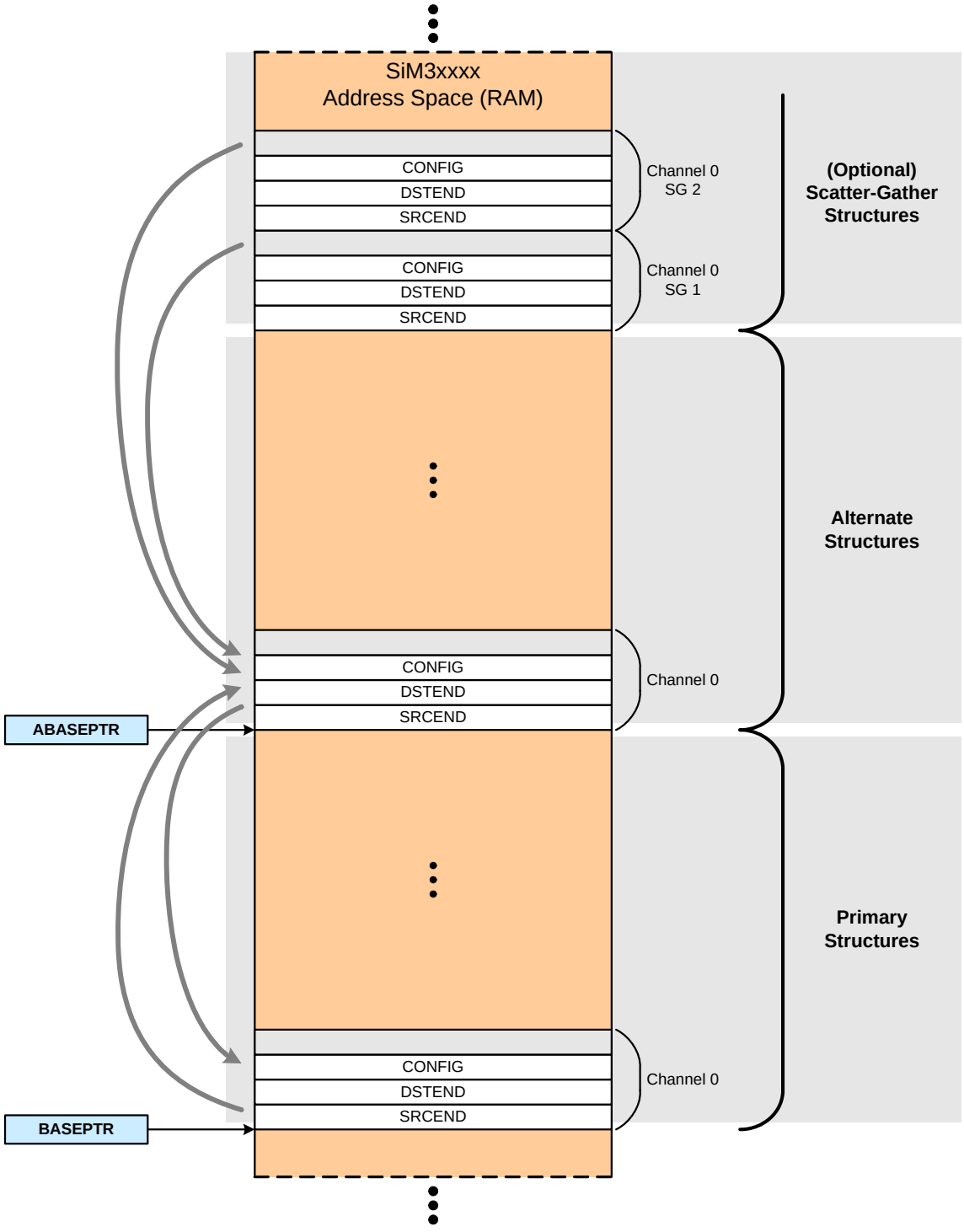


Figure 16.6. Memory and Peripheral Scatter-Gather Transfer Memory Configuration

16.6.5. Peripheral Scatter-Gather Transfers

The peripheral scatter-gather transfer is very similar to the memory scatter-gather transfer and uses primary, alternate, and scatter-gather structures. This transfer type allows a DMA channel to be set for multiple transfers at once without core intervention at the price of extra memory for the scatter-gather structures. A data request is required for each 2^{RPOWER} data move of the scatter-gather structure tasks. The RPOWER value can be different for each scatter-gather task. Equation 16.1 describes the total number of data requests required to complete a transfer.

The primary structure in this mode contains the number and location of the scatter-gather structures. The primary structure should be programmed as follows:

1. Peripheral scatter-gather primary mode (TMD = 6).
2. RPOWER = 2.
3. NCOUNT set to the value specified by Equation 16.2.
4. SRCEND is set to the location of the last word of all the scatter-gather structures.
5. DSTEND is set to the location of the last word in the single alternate structure.

The scatter-gather structures must be stacked contiguously in memory. The channel will copy the scatter-gather structures into the alternate structure location and execute them one by one. The scatter-gather structures should be programmed to peripheral scatter-gather alternate mode (TMD = 7), except for the last structure, which should use the basic or auto-request transfer types (TMD = 1 or 2).

Once started, the DMA channel execution process is as follows:

1. Copy scatter-gather 1 (SG1) to the alternate structure.
2. Jump to the alternate structure and execute.
3. Jump back to the primary structure.
4. Copy scatter-gather 2 (SG2) to the alternate structure.
5. Jump to the alternate structure and execute.
6. Jump back to the primary structure.

The channel will continue in this pattern until the channel encounters a scatter-gather structure set to a basic or auto-request transfer.

The channel interrupt will occur once the last scatter-gather structure (programmed to a basic transfer) executes, if enabled.

Figure 16.6 shows the peripheral scatter-gather memory configuration.

16.7. Data Requests

Each DMA channel has two data requests: single and burst. Peripherals can support single requests, burst requests, or both. If configured to use a DMA channel, peripherals request data as needed using the appropriate request type. Table 16.6 lists the supported requests for the supported triggers and peripherals.

The RPOWER field is only valid for peripherals that support burst requests. For peripherals that only support single requests, the RPOWER field is ignored and re-arbitration occurs after every single data move.

Table 16.6. Supported Trigger or Peripheral Data Requests

Peripheral Module	Supported Request Types
AESn	burst only
EPCAn	burst only
I2Cn	single only
I2Sn	burst only
IDACn	burst only
SARADCn	burst only
SPIn	burst only
TIMERn overflow	burst only
USARTn	single only
USBn	both
External Trigger	burst only
Software Trigger	burst only

In addition to peripheral-initiated transfers, all of the supported DMA channels can select the rising or falling edges of one of the DMA external transfer start signals to initiate data transfers. When the selected edge occurs on the external signal, the DMA channels with the DMA0T0/1 signals selected in the DMAXBARx.CHANnSEL field will start the corresponding channel's data transfer as defined by the DMA channel data structure in memory. The DMA module external trigger sources vary between package options, and are detailed in Table 16.7.

Table 16.7. DMA External Triggers

DMA Trigger	DMA Trigger Description	SiM3U1x7/C1x7 Pin Name	SiM3U1x6/C1x6 Pin Name	SiM3U1x4/C1x4 Pin Name
DMA0T0	External Transfer Start	PB1.10	PB1.4	PB0.10
DMA0T1	External Transfer Start	PB1.11	PB1.5	PB0.11

16.8. Masking Channels

DMA channels can be temporarily disabled by setting the channel bit in CHREQMSET. Setting this bit to 1 causes the DMA channel to no longer respond to data requests from peripherals. The channel will always respond to software-initiated transfer requests, even if CHREQMSET is set for the channel. Firmware can write a 1 to the CHREQMCLR register to clear the mask for a channel.

It is recommended that firmware set the channel request mask (CHREQMSET) for channels using software-initiated transfers to avoid any peripherals connected to the channel from requesting DMA transfers.

16.9. Errors

The ERROR bit in the BERRCLR register indicates when a DMA bus error occurs. This bit may or may not generate an interrupt on a SiM3U1xx/SiM3C1xx device. For devices that do not support DMA bus error interrupts, firmware should check this flag after a DMA transfer to determine if an error occurred.

16.10. Arbitration

The DMA controller is a master on the AHB bus. This allows the module to control data transfers without any interaction with the core.

The channels are in a fixed priority order. Channel 0 has the highest priority, and the last implemented channel has the lowest priority. This fixed order can be superceded by using the programmable high priority setting (CHHPSET). At each re-arbitration period, the controller gives control of the bus to the highest priority channel with a pending data request.

The RPOWER field in the channel transfer structures determines when the re-arbitration periods occur. The channel in control of the bus will make 2^{RPOWER} data moves before the controller re-arbitrates. If the channel still has the highest priority, it can transfer again until the next re-arbitration period. The RPOWER field is only valid for peripherals that support burst requests. For peripherals that only support single requests, re-arbitration will occur after each single data move.

Figure 16.7 shows an example controller arbitration with two channels active. Channel 0 has an RPOWER of 1 (2 data moves), and channel 1 has an RPOWER of 0 (1 data move). Both channels are set to move words (DSTSIZE and SRCSIZE set to 2).

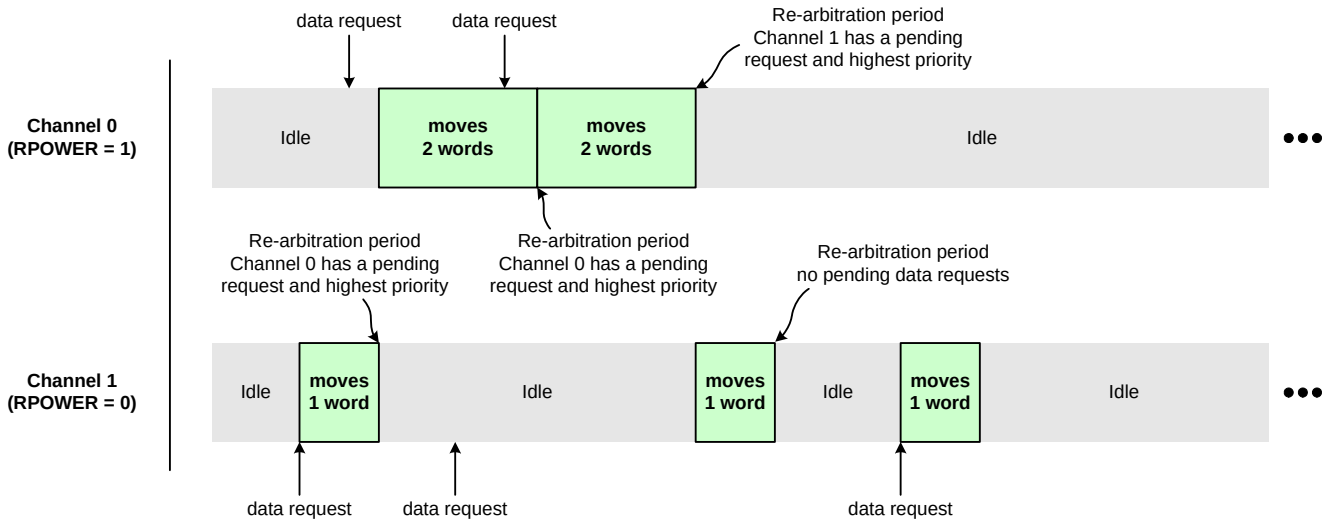


Figure 16.7. DMA Arbitration Example

16.11. DMACTRL0 Registers

This section contains the detailed register descriptions for DMACTRL0 registers.

Register 16.1. DMACTRL0_STATUS: Controller Status

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved											NUMCHAN				
Type	R											R				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved								STATE				Reserved			DMAENSTS
Type	R								R				R			R
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

DMACTRL0_STATUS = 0x4003_6000

Table 16.8. DMACTRL0_STATUS Register Bit Descriptions

Bit	Name	Function
31:21	Reserved	Must write reset value.
20:16	NUMCHAN	**Number of Supported DMA Channels.** This value represents one less than the number of supported DMA channels on the device. For example, a value of 15 in this field means 16 channels are supported on the device.
15:8	Reserved	Must write reset value.
7:4	STATE	**State Machine State.** This field indicates the current state of the control state machine. 0000: Idle. 0001: Reading channel controller data. 0010: Reading source data end pointer. 0011: Reading destination data end pointer. 0100: Reading source data. 0101: Writing destination data. 0110: Waiting for a DMA request to clear. 0111: Writing channel controller data. 1000: Stalled. 1001: Done. 1010: Peripheral scatter-gather transition. 1011-1111: Reserved.

Table 16.8. DMACTRL0_STATUS Register Bit Descriptions

Bit	Name	Function
3:1	Reserved	Must write reset value.
0	DMAENSTS	**DMA Enable Status.** 0: DMA controller is disabled 1: DMA controller is enabled.

Register 16.2. DMACTRL0_CONFIG: Controller Configuration

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	W															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved															DMAEN
Type	W															W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

DMACTRL0_CONFIG = 0x4003_6004

Table 16.9. DMACTRL0_CONFIG Register Bit Descriptions

Bit	Name	Function
31:1	Reserved	Must write reset value.
0	DMAEN	**DMA Enable.** 0: Disable the DMA controller. 1: Enable the DMA controller.

Register 16.3. DMACTRL0_BASEPTR: Base Pointer

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	BASEPTR[26:11]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	BASEPTR[10:0]											Reserved				
Type	RW											R				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

DMACTRL0_BASEPTR = 0x4003_6008

Table 16.10. DMACTRL0_BASEPTR Register Bit Descriptions

Bit	Name	Function
31:5	BASEPTR	**Control Base Pointer.** This field points to the base location in memory for the DMA channel control data structure. This field is left justified and is a different size depending on the number of channels implemented on the device. 1 channel: bits [31:5] specify the base address (multiples of 16). 2 channels: bits [31:6] specify the base address (multiples of 32). 3-4 channels: bits [31:7] specify the base address (multiples of 64). 5-8 channels: bits [31:8] specify the base address (multiples of 128). 9-16 channels: bits [31:9] specify the base address (multiples of 256). 17-32 channels: bits [31:10] specify the base address (multiples of 512). Any unused bits are undefined and should be written to 0. The primary structures must be placed at the start of an address block sized for both the primary and alternate structures.
4:0	Reserved	Must write reset value.

Register 16.4. DMACTRL0_ABASEPTR: Alternate Base Pointer

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	ABASEPTR[31:16]															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	ABASEPTR[15:0]															
Type	R															
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

Register ALL Access Address

DMACTRL0_ABASEPTR = 0x4003_600C

Table 16.11. DMACTRL0_ABASEPTR Register Bit Descriptions

Bit	Name	Function
31:0	ABASEPTR	**Alternate Control Base Pointer.** This read-only field points to the base location in memory for the alternate DMA channel control data structure. This address depends on the number of channels implemented on the device.

Register 16.5. DMACTRL0_CHSTATUS: Channel Status

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	CH15	CH14	CH13	CH12	CH11	CH10	CH9	CH8	CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0
Type	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Register ALL Access Address

DMACTRL0_CHSTATUS = 0x4003_6010

Table 16.12. DMACTRL0_CHSTATUS Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15	CH15	**Channel 15 Status.** 0: DMA Channel 15 is not waiting for a data request. 1: DMA Channel 15 is waiting for a data request.
14	CH14	**Channel 14 Status.** 0: DMA Channel 14 is not waiting for a data request. 1: DMA Channel 14 is waiting for a data request.
13	CH13	**Channel 13 Status.** 0: DMA Channel 13 is not waiting for a data request. 1: DMA Channel 13 is waiting for a data request.
12	CH12	**Channel 12 Status.** 0: DMA Channel 12 is not waiting for a data request. 1: DMA Channel 12 is waiting for a data request.
11	CH11	**Channel 11 Status.** 0: DMA Channel 11 is not waiting for a data request. 1: DMA Channel 11 is waiting for a data request.
10	CH10	**Channel 10 Status.** 0: DMA Channel 10 is not waiting for a data request. 1: DMA Channel 10 is waiting for a data request.
9	CH9	**Channel 9 Status.** 0: DMA Channel 9 is not waiting for a data request. 1: DMA Channel 9 is waiting for a data request.

Table 16.12. DMACTRL0_CHSTATUS Register Bit Descriptions

Bit	Name	Function
8	CH8	**Channel 8 Status.** 0: DMA Channel 8 is not waiting for a data request. 1: DMA Channel 8 is waiting for a data request.
7	CH7	**Channel 7 Status.** 0: DMA Channel 7 is not waiting for a data request. 1: DMA Channel 7 is waiting for a data request.
6	CH6	**Channel 6 Status.** 0: DMA Channel 6 is not waiting for a data request. 1: DMA Channel 6 is waiting for a data request.
5	CH5	**Channel 5 Status.** 0: DMA Channel 5 is not waiting for a data request. 1: DMA Channel 5 is waiting for a data request.
4	CH4	**Channel 4 Status.** 0: DMA Channel 4 is not waiting for a data request. 1: DMA Channel 4 is waiting for a data request.
3	CH3	**Channel 3 Status.** 0: DMA Channel 3 is not waiting for a data request. 1: DMA Channel 3 is waiting for a data request.
2	CH2	**Channel 2 Status.** 0: DMA Channel 2 is not waiting for a data request. 1: DMA Channel 2 is waiting for a data request.
1	CH1	**Channel 1 Status.** 0: DMA Channel 1 is not waiting for a data request. 1: DMA Channel 1 is waiting for a data request.
0	CH0	**Channel 0 Status.** 0: DMA Channel 0 is not waiting for a data request. 1: DMA Channel 0 is waiting for a data request.

Register 16.6. DMACTRL0_CHSWRCN: Channel Software Request Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	W															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	CH15	CH14	CH13	CH12	CH11	CH10	CH9	CH8	CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0
Type	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

DMACTRL0_CHSWRCN = 0x4003_6014

Table 16.13. DMACTRL0_CHSWRCN Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15	CH15	**Channel 15 Software Request.** 0: DMA Channel 15 does not generate a software data request. 1: DMA Channel 15 generates a software data request.
14	CH14	**Channel 14 Software Request.** 0: DMA Channel 14 does not generate a software data request. 1: DMA Channel 14 generates a software data request.
13	CH13	**Channel 13 Software Request.** 0: DMA Channel 13 does not generate a software data request. 1: DMA Channel 13 generates a software data request.
12	CH12	**Channel 12 Software Request.** 0: DMA Channel 12 does not generate a software data request. 1: DMA Channel 12 generates a software data request.
11	CH11	**Channel 11 Software Request.** 0: DMA Channel 11 does not generate a software data request. 1: DMA Channel 11 generates a software data request.
10	CH10	**Channel 10 Software Request.** 0: DMA Channel 10 does not generate a software data request. 1: DMA Channel 10 generates a software data request.
9	CH9	**Channel 9 Software Request.** 0: DMA Channel 9 does not generate a software data request. 1: DMA Channel 9 generates a software data request.

Table 16.13. DMACTRL0_CHSWRCN Register Bit Descriptions

Bit	Name	Function
8	CH8	**Channel 8 Software Request.** 0: DMA Channel 8 does not generate a software data request. 1: DMA Channel 8 generates a software data request.
7	CH7	**Channel 7 Software Request.** 0: DMA Channel 7 does not generate a software data request. 1: DMA Channel 7 generates a software data request.
6	CH6	**Channel 6 Software Request.** 0: DMA Channel 6 does not generate a software data request. 1: DMA Channel 6 generates a software data request.
5	CH5	**Channel 5 Software Request.** 0: DMA Channel 5 does not generate a software data request. 1: DMA Channel 5 generates a software data request.
4	CH4	**Channel 4 Software Request.** 0: DMA Channel 4 does not generate a software data request. 1: DMA Channel 4 generates a software data request.
3	CH3	**Channel 3 Software Request.** 0: DMA Channel 3 does not generate a software data request. 1: DMA Channel 3 generates a software data request.
2	CH2	**Channel 2 Software Request.** 0: DMA Channel 2 does not generate a software data request. 1: DMA Channel 2 generates a software data request.
1	CH1	**Channel 1 Software Request.** 0: DMA Channel 1 does not generate a software data request. 1: DMA Channel 1 generates a software data request.
0	CH0	**Channel 0 Software Request.** 0: DMA Channel 0 does not generate a software data request. 1: DMA Channel 0 generates a software data request.

Register 16.7. DMACTRL0_CHREQMSET: Channel Request Mask Set

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	CH15	CH14	CH13	CH12	CH11	CH10	CH9	CH8	CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

DMACTRL0_CHREQMSET = 0x4003_6020

Table 16.14. DMACTRL0_CHREQMSET Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15	CH15	**Channel 15 Request Mask Enable.** Read: 0: DMA Channel 15 peripheral data requests enabled. 1: DMA Channel 15 peripheral data requests disabled. Write: 0: No effect (use CHREQMCLR to clear). 1: Disable DMA Channel 15 peripheral data requests.
14	CH14	**Channel 14 Request Mask Enable.** Read: 0: DMA Channel 14 peripheral data requests enabled. 1: DMA Channel 14 peripheral data requests disabled. Write: 0: No effect (use CHREQMCLR to clear). 1: Disable DMA Channel 14 peripheral data requests.
13	CH13	**Channel 13 Request Mask Enable.** Read: 0: DMA Channel 13 peripheral data requests enabled. 1: DMA Channel 13 peripheral data requests disabled. Write: 0: No effect (use CHREQMCLR to clear). 1: Disable DMA Channel 13 peripheral data requests.

Table 16.14. DMACTRL0_CHREQMSET Register Bit Descriptions

Bit	Name	Function
12	CH12	**Channel 12 Request Mask Enable.** Read: 0: DMA Channel 12 peripheral data requests enabled. 1: DMA Channel 12 peripheral data requests disabled. Write: 0: No effect (use CHREQMCLR to clear). 1: Disable DMA Channel 12 peripheral data requests.
11	CH11	**Channel 11 Request Mask Enable.** Read: 0: DMA Channel 11 peripheral data requests enabled. 1: DMA Channel 11 peripheral data requests disabled. Write: 0: No effect (use CHREQMCLR to clear). 1: Disable DMA Channel 11 peripheral data requests.
10	CH10	**Channel 10 Request Mask Enable.** Read: 0: DMA Channel 10 peripheral data requests enabled. 1: DMA Channel 10 peripheral data requests disabled. Write: 0: No effect (use CHREQMCLR to clear). 1: Disable DMA Channel 10 peripheral data requests.
9	CH9	**Channel 9 Request Mask Enable.** Read: 0: DMA Channel 9 peripheral data requests enabled. 1: DMA Channel 9 peripheral data requests disabled. Write: 0: No effect (use CHREQMCLR to clear). 1: Disable DMA Channel 9 peripheral data requests.
8	CH8	**Channel 8 Request Mask Enable.** Read: 0: DMA Channel 8 peripheral data requests enabled. 1: DMA Channel 8 peripheral data requests disabled. Write: 0: No effect (use CHREQMCLR to clear). 1: Disable DMA Channel 8 peripheral data requests.
7	CH7	**Channel 7 Request Mask Enable.** Read: 0: DMA Channel 7 peripheral data requests enabled. 1: DMA Channel 7 peripheral data requests disabled. Write: 0: No effect (use CHREQMCLR to clear). 1: Disable DMA Channel 7 peripheral data requests.

Table 16.14. DMACTRL0_CHREQMSET Register Bit Descriptions

Bit	Name	Function
6	CH6	**Channel 6 Request Mask Enable.** Read: 0: DMA Channel 6 peripheral data requests enabled. 1: DMA Channel 6 peripheral data requests disabled. Write: 0: No effect (use CHREQMCLR to clear). 1: Disable DMA Channel 6 peripheral data requests.
5	CH5	**Channel 5 Request Mask Enable.** Read: 0: DMA Channel 5 peripheral data requests enabled. 1: DMA Channel 5 peripheral data requests disabled. Write: 0: No effect (use CHREQMCLR to clear). 1: Disable DMA Channel 5 peripheral data requests.
4	CH4	**Channel 4 Request Mask Enable.** Read: 0: DMA Channel 4 peripheral data requests enabled. 1: DMA Channel 4 peripheral data requests disabled. Write: 0: No effect (use CHREQMCLR to clear). 1: Disable DMA Channel 4 peripheral data requests.
3	CH3	**Channel 3 Request Mask Enable.** Read: 0: DMA Channel 3 peripheral data requests enabled. 1: DMA Channel 3 peripheral data requests disabled. Write: 0: No effect (use CHREQMCLR to clear). 1: Disable DMA Channel 3 peripheral data requests.
2	CH2	**Channel 2 Request Mask Enable.** Read: 0: DMA Channel 2 peripheral data requests enabled. 1: DMA Channel 2 peripheral data requests disabled. Write: 0: No effect (use CHREQMCLR to clear). 1: Disable DMA Channel 2 peripheral data requests.
1	CH1	**Channel 1 Request Mask Enable.** Read: 0: DMA Channel 1 peripheral data requests enabled. 1: DMA Channel 1 peripheral data requests disabled. Write: 0: No effect (use CHREQMCLR to clear). 1: Disable DMA Channel 1 peripheral data requests.

Table 16.14. DMACTRL0_CHREQMSET Register Bit Descriptions

Bit	Name	Function
0	CH0	**Channel 0 Request Mask Enable.** Read: 0: DMA Channel 0 peripheral data requests enabled. 1: DMA Channel 0 peripheral data requests disabled. Write: 0: No effect (use CHREQMCLR to clear). 1: Disable DMA Channel 0 peripheral data requests.

Register 16.8. DMACTRL0_CHREQMCLR: Channel Request Mask Clear

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	W															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	CH15	CH14	CH13	CH12	CH11	CH10	CH9	CH8	CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0
Type	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

DMACTRL0_CHREQMCLR = 0x4003_6024

Table 16.15. DMACTRL0_CHREQMCLR Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15	CH15	**Channel 15 Request Mask Disable.** 0: No effect. 1: Enable DMA Channel 15 peripheral data requests.
14	CH14	**Channel 14 Request Mask Disable.** 0: No effect. 1: Enable DMA Channel 14 peripheral data requests.
13	CH13	**Channel 13 Request Mask Disable.** 0: No effect. 1: Enable DMA Channel 13 peripheral data requests.
12	CH12	**Channel 12 Request Mask Disable.** 0: No effect. 1: Enable DMA Channel 12 peripheral data requests.
11	CH11	**Channel 11 Request Mask Disable.** 0: No effect. 1: Enable DMA Channel 11 peripheral data requests.
10	CH10	**Channel 10 Request Mask Disable.** 0: No effect. 1: Enable DMA Channel 10 peripheral data requests.
9	CH9	**Channel 9 Request Mask Disable.** 0: No effect. 1: Enable DMA Channel 9 peripheral data requests.

Table 16.15. DMACTRL0_CHREQMCLR Register Bit Descriptions

Bit	Name	Function
8	CH8	**Channel 8 Request Mask Disable.** 0: No effect. 1: Enable DMA Channel 8 peripheral data requests.
7	CH7	**Channel 7 Request Mask Disable.** 0: No effect. 1: Enable DMA Channel 7 peripheral data requests.
6	CH6	**Channel 6 Request Mask Disable.** 0: No effect. 1: Enable DMA Channel 6 peripheral data requests.
5	CH5	**Channel 5 Request Mask Disable.** 0: No effect. 1: Enable DMA Channel 5 peripheral data requests.
4	CH4	**Channel 4 Request Mask Disable.** 0: No effect. 1: Enable DMA Channel 4 peripheral data requests.
3	CH3	**Channel 3 Request Mask Disable.** 0: No effect. 1: Enable DMA Channel 3 peripheral data requests.
2	CH2	**Channel 2 Request Mask Disable.** 0: No effect. 1: Enable DMA Channel 2 peripheral data requests.
1	CH1	**Channel 1 Request Mask Disable.** 0: No effect. 1: Enable DMA Channel 1 peripheral data requests.
0	CH0	**Channel 0 Request Mask Disable.** 0: No effect. 1: Enable DMA Channel 0 peripheral data requests.

Register 16.9. DMACTRL0_CHENSET: Channel Enable Set

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	CH15	CH14	CH13	CH12	CH11	CH10	CH9	CH8	CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

DMACTRL0_CHENSET = 0x4003_6028

Table 16.16. DMACTRL0_CHENSET Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15	CH15	**Channel 15 Enable.** Read: 0: DMA Channel 15 disabled. 1: DMA Channel 15 enabled. Write: 0: No effect (use CHENCLR to clear). 1: Enable DMA Channel 15.
14	CH14	**Channel 14 Enable.** Read: 0: DMA Channel 14 disabled. 1: DMA Channel 14 enabled. Write: 0: No effect (use CHENCLR to clear). 1: Enable DMA Channel 14.

Notes:

1. The controller will automatically disable a channel when: 1) the controller completes the DMA cycle, 2) it reads a channel configuration memory location and the cycle control field is 0, or 3) an error (ERROR = 1) occurs on the AHB bus.

Table 16.16. DMACTRL0_CHENSET Register Bit Descriptions

Bit	Name	Function
13	CH13	**Channel 13 Enable.** Read: 0: DMA Channel 13 disabled. 1: DMA Channel 13 enabled. Write: 0: No effect (use CHENCLR to clear). 1: Enable DMA Channel 13.
12	CH12	**Channel 12 Enable.** Read: 0: DMA Channel 12 disabled. 1: DMA Channel 12 enabled. Write: 0: No effect (use CHENCLR to clear). 1: Enable DMA Channel 12.
11	CH11	**Channel 11 Enable.** Read: 0: DMA Channel 11 disabled. 1: DMA Channel 11 enabled. Write: 0: No effect (use CHENCLR to clear). 1: Enable DMA Channel 11.
10	CH10	**Channel 10 Enable.** Read: 0: DMA Channel 10 disabled. 1: DMA Channel 10 enabled. Write: 0: No effect (use CHENCLR to clear). 1: Enable DMA Channel 10.
9	CH9	**Channel 9 Enable.** Read: 0: DMA Channel 9 disabled. 1: DMA Channel 9 enabled. Write: 0: No effect (use CHENCLR to clear). 1: Enable DMA Channel 9.

Notes:

1. The controller will automatically disable a channel when: 1) the controller completes the DMA cycle, 2) it reads a channel configuration memory location and the cycle control field is 0, or 3) an error (ERROR = 1) occurs on the AHB bus.

Table 16.16. DMACTRL0_CHENSET Register Bit Descriptions

Bit	Name	Function
8	CH8	**Channel 8 Enable.** Read: 0: DMA Channel 8 disabled. 1: DMA Channel 8 enabled. Write: 0: No effect (use CHENCLR to clear). 1: Enable DMA Channel 8.
7	CH7	**Channel 7 Enable.** Read: 0: DMA Channel 7 disabled. 1: DMA Channel 7 enabled. Write: 0: No effect (use CHENCLR to clear). 1: Enable DMA Channel 7.
6	CH6	**Channel 6 Enable.** Read: 0: DMA Channel 6 disabled. 1: DMA Channel 6 enabled. Write: 0: No effect (use CHENCLR to clear). 1: Enable DMA Channel 6.
5	CH5	**Channel 5 Enable.** Read: 0: DMA Channel 5 disabled. 1: DMA Channel 5 enabled. Write: 0: No effect (use CHENCLR to clear). 1: Enable DMA Channel 5.
4	CH4	**Channel 4 Enable.** Read: 0: DMA Channel 4 disabled. 1: DMA Channel 4 enabled. Write: 0: No effect (use CHENCLR to clear). 1: Enable DMA Channel 4.

Notes:

1. The controller will automatically disable a channel when: 1) the controller completes the DMA cycle, 2) it reads a channel configuration memory location and the cycle control field is 0, or 3) an error (ERROR = 1) occurs on the AHB bus.

Table 16.16. DMACTRL0_CHENSET Register Bit Descriptions

Bit	Name	Function
3	CH3	**Channel 3 Enable.** Read: 0: DMA Channel 3 disabled. 1: DMA Channel 3 enabled. Write: 0: No effect (use CHENCLR to clear). 1: Enable DMA Channel 3.
2	CH2	**Channel 2 Enable.** Read: 0: DMA Channel 2 disabled. 1: DMA Channel 2 enabled. Write: 0: No effect (use CHENCLR to clear). 1: Enable DMA Channel 2.
1	CH1	**Channel 1 Enable.** Read: 0: DMA Channel 1 disabled. 1: DMA Channel 1 enabled. Write: 0: No effect (use CHENCLR to clear). 1: Enable DMA Channel 1.
0	CH0	**Channel 0 Enable.** Read: 0: DMA Channel 0 disabled. 1: DMA Channel 0 enabled. Write: 0: No effect (use CHENCLR to clear). 1: Enable DMA Channel 0.

Notes:

1. The controller will automatically disable a channel when: 1) the controller completes the DMA cycle, 2) it reads a channel configuration memory location and the cycle control field is 0, or 3) an error (ERROR = 1) occurs on the AHB bus.

Register 16.10. DMACTRL0_CHENCLR: Channel Enable Clear

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	W															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	CH15	CH14	CH13	CH12	CH11	CH10	CH9	CH8	CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0
Type	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

DMACTRL0_CHENCLR = 0x4003_602C

Table 16.17. DMACTRL0_CHENCLR Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15	CH15	**Channel 15 Disable.** 0: No effect. 1: Disable DMA Channel 15.
14	CH14	**Channel 14 Disable.** 0: No effect. 1: Disable DMA Channel 14.
13	CH13	**Channel 13 Disable.** 0: No effect. 1: Disable DMA Channel 13.
12	CH12	**Channel 12 Disable.** 0: No effect. 1: Disable DMA Channel 12.
11	CH11	**Channel 11 Disable.** 0: No effect. 1: Disable DMA Channel 11.
10	CH10	**Channel 10 Disable.** 0: No effect. 1: Disable DMA Channel 10.

Notes:

1. The controller will automatically disable a channel by setting the appropriate bit when: 1) the controller completes the DMA cycle, 2) it reads a channel configuration memory location and the cycle control field is 0, or 3) an error (ERROR = 1) occurs on the AHB bus.

Table 16.17. DMACTRL0_CHENCLR Register Bit Descriptions

Bit	Name	Function
9	CH9	**Channel 9 Disable.** 0: No effect. 1: Disable DMA Channel 9.
8	CH8	**Channel 8 Disable.** 0: No effect. 1: Disable DMA Channel 8.
7	CH7	**Channel 7 Disable.** 0: No effect. 1: Disable DMA Channel 7.
6	CH6	**Channel 6 Disable.** 0: No effect. 1: Disable DMA Channel 6.
5	CH5	**Channel 5 Disable.** 0: No effect. 1: Disable DMA Channel 5.
4	CH4	**Channel 4 Disable.** 0: No effect. 1: Disable DMA Channel 4.
3	CH3	**Channel 3 Disable.** 0: No effect. 1: Disable DMA Channel 3.
2	CH2	**Channel 2 Disable.** 0: No effect. 1: Disable DMA Channel 2.
1	CH1	**Channel 1 Disable.** 0: No effect. 1: Disable DMA Channel 1.
0	CH0	**Channel 0 Disable.** 0: No effect. 1: Disable DMA Channel 0.

Notes:

1. The controller will automatically disable a channel by setting the appropriate bit when: 1) the controller completes the DMA cycle, 2) it reads a channel configuration memory location and the cycle control field is 0, or 3) an error (ERROR = 1) occurs on the AHB bus.

Register 16.11. DMACTRL0_CHALTSET: Channel Alternate Select Set

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	CH15	CH14	CH13	CH12	CH11	CH10	CH9	CH8	CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

DMACTRL0_CHALTSET = 0x4003_6030

Table 16.18. DMACTRL0_CHALTSET Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15	CH15	**Channel 15 Alternate Enable.** Read: 0: DMA Channel 15 is using primary data structure. 1: DMA Channel 15 is using alternate data structure. Write: 0: No effect (use CHALTCLR to clear). 1: Use the alternate data structure for DMA Channel 15.
14	CH14	**Channel 14 Alternate Enable.** Read: 0: DMA Channel 14 is using primary data structure. 1: DMA Channel 14 is using alternate data structure. Write: 0: No effect (use CHALTCLR to clear). 1: Use the alternate data structure for DMA Channel 14.

Notes:

1. The controller toggles the value of the channel bit after it completes: 1) the four transfers that the primary data structure specifies for a memory scatter-gather or peripheral scatter-gather DMA cycle, 2) all the transfers that the primary data structure specifies for a ping-pong DMA cycle, or 3) all the transfers that the alternate data structure specifies for the following DMA cycle types (ping-pong, memory scatter-gather, peripheral scatter-gather).

Table 16.18. DMACTRL0_CHALTSET Register Bit Descriptions

Bit	Name	Function
13	CH13	**Channel 13 Alternate Enable.** Read: 0: DMA Channel 13 is using primary data structure. 1: DMA Channel 13 is using alternate data structure. Write: 0: No effect (use CHALTCLR to clear). 1: Use the alternate data structure for DMA Channel 13.
12	CH12	**Channel 12 Alternate Enable.** Read: 0: DMA Channel 12 is using primary data structure. 1: DMA Channel 12 is using alternate data structure. Write: 0: No effect (use CHALTCLR to clear). 1: Use the alternate data structure for DMA Channel 12.
11	CH11	**Channel 11 Alternate Enable.** Read: 0: DMA Channel 11 is using primary data structure. 1: DMA Channel 11 is using alternate data structure. Write: 0: No effect (use CHALTCLR to clear). 1: Use the alternate data structure for DMA Channel 11.
10	CH10	**Channel 10 Alternate Enable.** Read: 0: DMA Channel 10 is using primary data structure. 1: DMA Channel 10 is using alternate data structure. Write: 0: No effect (use CHALTCLR to clear). 1: Use the alternate data structure for DMA Channel 10.
9	CH9	**Channel 9 Alternate Enable.** Read: 0: DMA Channel 9 is using primary data structure. 1: DMA Channel 9 is using alternate data structure. Write: 0: No effect (use CHALTCLR to clear). 1: Use the alternate data structure for DMA Channel 9.

Notes:

1. The controller toggles the value of the channel bit after it completes: 1) the four transfers that the primary data structure specifies for a memory scatter-gather or peripheral scatter-gather DMA cycle, 2) all the transfers that the primary data structure specifies for a ping-pong DMA cycle, or 3) all the transfers that the alternate data structure specifies for the following DMA cycle types (ping-pong, memory scatter-gather, peripheral scatter-gather).

Table 16.18. DMACTRL0_CHALTSET Register Bit Descriptions

Bit	Name	Function
8	CH8	**Channel 8 Alternate Enable.** Read: 0: DMA Channel 8 is using primary data structure. 1: DMA Channel 8 is using alternate data structure. Write: 0: No effect (use CHALTCLR to clear). 1: Use the alternate data structure for DMA Channel 8.
7	CH7	**Channel 7 Alternate Enable.** Read: 0: DMA Channel 7 is using primary data structure. 1: DMA Channel 7 is using alternate data structure. Write: 0: No effect (use CHALTCLR to clear). 1: Use the alternate data structure for DMA Channel 7.
6	CH6	**Channel 6 Alternate Enable.** Read: 0: DMA Channel 6 is using primary data structure. 1: DMA Channel 6 is using alternate data structure. Write: 0: No effect (use CHALTCLR to clear). 1: Use the alternate data structure for DMA Channel 6.
5	CH5	**Channel 5 Alternate Enable.** Read: 0: DMA Channel 5 is using primary data structure. 1: DMA Channel 5 is using alternate data structure. Write: 0: No effect (use CHALTCLR to clear). 1: Use the alternate data structure for DMA Channel 5.
4	CH4	**Channel 4 Alternate Enable.** Read: 0: DMA Channel 4 is using primary data structure. 1: DMA Channel 4 is using alternate data structure. Write: 0: No effect (use CHALTCLR to clear). 1: Use the alternate data structure for DMA Channel 4.

Notes:

1. The controller toggles the value of the channel bit after it completes: 1) the four transfers that the primary data structure specifies for a memory scatter-gather or peripheral scatter-gather DMA cycle, 2) all the transfers that the primary data structure specifies for a ping-pong DMA cycle, or 3) all the transfers that the alternate data structure specifies for the following DMA cycle types (ping-pong, memory scatter-gather, peripheral scatter-gather).

Table 16.18. DMACTRL0_CHALTSET Register Bit Descriptions

Bit	Name	Function
3	CH3	**Channel 3 Alternate Enable.** Read: 0: DMA Channel 3 is using primary data structure. 1: DMA Channel 3 is using alternate data structure. Write: 0: No effect (use CHALTCLR to clear). 1: Use the alternate data structure for DMA Channel 3.
2	CH2	**Channel 2 Alternate Enable.** Read: 0: DMA Channel 2 is using primary data structure. 1: DMA Channel 2 is using alternate data structure. Write: 0: No effect (use CHALTCLR to clear). 1: Use the alternate data structure for DMA Channel 2.
1	CH1	**Channel 1 Alternate Enable.** Read: 0: DMA Channel 1 is using primary data structure. 1: DMA Channel 1 is using alternate data structure. Write: 0: No effect (use CHALTCLR to clear). 1: Use the alternate data structure for DMA Channel 1.
0	CH0	**Channel 0 Alternate Enable.** Read: 0: DMA Channel 0 is using primary data structure. 1: DMA Channel 0 is using alternate data structure. Write: 0: No effect (use CHALTCLR to clear). 1: Use the alternate data structure for DMA Channel 0.

Notes:

1. The controller toggles the value of the channel bit after it completes: 1) the four transfers that the primary data structure specifies for a memory scatter-gather or peripheral scatter-gather DMA cycle, 2) all the transfers that the primary data structure specifies for a ping-pong DMA cycle, or 3) all the transfers that the alternate data structure specifies for the following DMA cycle types (ping-pong, memory scatter-gather, peripheral scatter-gather).

Register 16.12. DMACTRL0_CHALTCLR: Channel Alternate Select Clear

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	W															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	CH15	CH14	CH13	CH12	CH11	CH10	CH9	CH8	CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0
Type	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

DMACTRL0_CHALTCLR = 0x4003_6034

Table 16.19. DMACTRL0_CHALTCLR Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15	CH15	**Channel 15 Alternate Disable.** 0: No effect. 1: Use the primary data structure for DMA Channel 15.
14	CH14	**Channel 14 Alternate Disable.** 0: No effect. 1: Use the primary data structure for DMA Channel 14.
13	CH13	**Channel 13 Alternate Disable.** 0: No effect. 1: Use the primary data structure for DMA Channel 13.
12	CH12	**Channel 12 Alternate Disable.** 0: No effect. 1: Use the primary data structure for DMA Channel 12.
11	CH11	**Channel 11 Alternate Disable.** 0: No effect. 1: Use the primary data structure for DMA Channel 11.
10	CH10	**Channel 10 Alternate Disable.** 0: No effect. 1: Use the primary data structure for DMA Channel 10.
9	CH9	**Channel 9 Alternate Disable.** 0: No effect. 1: Use the primary data structure for DMA Channel 9.

Table 16.19. DMACTRL0_CHALTCLR Register Bit Descriptions

Bit	Name	Function
8	CH8	**Channel 8 Alternate Disable.** 0: No effect. 1: Use the primary data structure for DMA Channel 8.
7	CH7	**Channel 7 Alternate Disable.** 0: No effect. 1: Use the primary data structure for DMA Channel 7.
6	CH6	**Channel 6 Alternate Disable.** 0: No effect. 1: Use the primary data structure for DMA Channel 6.
5	CH5	**Channel 5 Alternate Disable.** 0: No effect. 1: Use the primary data structure for DMA Channel 5.
4	CH4	**Channel 4 Alternate Disable.** 0: No effect. 1: Use the primary data structure for DMA Channel 4.
3	CH3	**Channel 3 Alternate Disable.** 0: No effect. 1: Use the primary data structure for DMA Channel 3.
2	CH2	**Channel 2 Alternate Disable.** 0: No effect. 1: Use the primary data structure for DMA Channel 2.
1	CH1	**Channel 1 Alternate Disable.** 0: No effect. 1: Use the primary data structure for DMA Channel 1.
0	CH0	**Channel 0 Alternate Disable.** 0: No effect. 1: Use the primary data structure for DMA Channel 0.

Register 16.13. DMACTRL0_CHHPSET: Channel High Priority Set

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	CH15	CH14	CH13	CH12	CH11	CH10	CH9	CH8	CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

DMACTRL0_CHHPSET = 0x4003_6038

Table 16.20. DMACTRL0_CHHPSET Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15	CH15	**Channel 15 High Priority Enable.** Read: 0: DMA Channel 15 is using the default priority level. 1: DMA Channel 15 is using the high priority level. Write: 0: No effect (use CHHPCLR to clear). 1: Use the high priority level for DMA Channel 15.
14	CH14	**Channel 14 High Priority Enable.** Read: 0: DMA Channel 14 is using the default priority level. 1: DMA Channel 14 is using the high priority level. Write: 0: No effect (use CHHPCLR to clear). 1: Use the high priority level for DMA Channel 14.
13	CH13	**Channel 13 High Priority Enable.** Read: 0: DMA Channel 13 is using the default priority level. 1: DMA Channel 13 is using the high priority level. Write: 0: No effect (use CHHPCLR to clear). 1: Use the high priority level for DMA Channel 13.

Table 16.20. DMACTRL0_CHHPSET Register Bit Descriptions

Bit	Name	Function
12	CH12	**Channel 12 High Priority Enable.** Read: 0: DMA Channel 12 is using the default priority level. 1: DMA Channel 12 is using the high priority level. Write: 0: No effect (use CHHPCLR to clear). 1: Use the high priority level for DMA Channel 12.
11	CH11	**Channel 11 High Priority Enable.** Read: 0: DMA Channel 11 is using the default priority level. 1: DMA Channel 11 is using the high priority level. Write: 0: No effect (use CHHPCLR to clear). 1: Use the high priority level for DMA Channel 11.
10	CH10	**Channel 10 High Priority Enable.** Read: 0: DMA Channel 10 is using the default priority level. 1: DMA Channel 10 is using the high priority level. Write: 0: No effect (use CHHPCLR to clear). 1: Use the high priority level for DMA Channel 10.
9	CH9	**Channel 9 High Priority Enable.** Read: 0: DMA Channel 9 is using the default priority level. 1: DMA Channel 9 is using the high priority level. Write: 0: No effect (use CHHPCLR to clear). 1: Use the high priority level for DMA Channel 9.
8	CH8	**Channel 8 High Priority Enable.** Read: 0: DMA Channel 8 is using the default priority level. 1: DMA Channel 8 is using the high priority level. Write: 0: No effect (use CHHPCLR to clear). 1: Use the high priority level for DMA Channel 8.
7	CH7	**Channel 7 High Priority Enable.** Read: 0: DMA Channel 7 is using the default priority level. 1: DMA Channel 7 is using the high priority level. Write: 0: No effect (use CHHPCLR to clear). 1: Use the high priority level for DMA Channel 7.

Table 16.20. DMACTRL0_CHHPSET Register Bit Descriptions

Bit	Name	Function
6	CH6	**Channel 6 High Priority Enable.** Read: 0: DMA Channel 6 is using the default priority level. 1: DMA Channel 6 is using the high priority level. Write: 0: No effect (use CHHPCLR to clear). 1: Use the high priority level for DMA Channel 6.
5	CH5	**Channel 5 High Priority Enable.** Read: 0: DMA Channel 5 is using the default priority level. 1: DMA Channel 5 is using the high priority level. Write: 0: No effect (use CHHPCLR to clear). 1: Use the high priority level for DMA Channel 5.
4	CH4	**Channel 4 High Priority Enable.** Read: 0: DMA Channel 4 is using the default priority level. 1: DMA Channel 4 is using the high priority level. Write: 0: No effect (use CHHPCLR to clear). 1: Use the high priority level for DMA Channel 4.
3	CH3	**Channel 3 High Priority Enable.** Read: 0: DMA Channel 3 is using the default priority level. 1: DMA Channel 3 is using the high priority level. Write: 0: No effect (use CHHPCLR to clear). 1: Use the high priority level for DMA Channel 3.
2	CH2	**Channel 2 High Priority Enable.** Read: 0: DMA Channel 2 is using the default priority level. 1: DMA Channel 2 is using the high priority level. Write: 0: No effect (use CHHPCLR to clear). 1: Use the high priority level for DMA Channel 2.
1	CH1	**Channel 1 High Priority Enable.** Read: 0: DMA Channel 1 is using the default priority level. 1: DMA Channel 1 is using the high priority level. Write: 0: No effect (use CHHPCLR to clear). 1: Use the high priority level for DMA Channel 1.

Table 16.20. DMACTRL0_CHHPSET Register Bit Descriptions

Bit	Name	Function
0	CH0	**Channel 0 High Priority Enable.** Read: 0: DMA Channel 0 is using the default priority level. 1: DMA Channel 0 is using the high priority level. Write: 0: No effect (use CHHPCLR to clear). 1: Use the high priority level for DMA Channel 0.

Register 16.14. DMACTRL0_CHHPCLR: Channel High Priority Clear

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	W															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	CH15	CH14	CH13	CH12	CH11	CH10	CH9	CH8	CH7	CH6	CH5	CH4	CH3	CH2	CH1	CH0
Type	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W	W
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

DMACTRL0_CHHPCLR = 0x4003_603C

Table 16.21. DMACTRL0_CHHPCLR Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15	CH15	**Channel 15 High Priority Disable.** 0: No effect. 1: Use the high default level for DMA Channel 15.
14	CH14	**Channel 14 High Priority Disable.** 0: No effect. 1: Use the high default level for DMA Channel 14.
13	CH13	**Channel 13 High Priority Disable.** 0: No effect. 1: Use the high default level for DMA Channel 13.
12	CH12	**Channel 12 High Priority Disable.** 0: No effect. 1: Use the high default level for DMA Channel 12.
11	CH11	**Channel 11 High Priority Disable.** 0: No effect. 1: Use the high default level for DMA Channel 11.
10	CH10	**Channel 10 High Priority Disable.** 0: No effect. 1: Use the high default level for DMA Channel 10.
9	CH9	**Channel 9 High Priority Disable.** 0: No effect. 1: Use the high default level for DMA Channel 9.

Table 16.21. DMACTRL0_CHHPCLR Register Bit Descriptions

Bit	Name	Function
8	CH8	**Channel 8 High Priority Disable.** 0: No effect. 1: Use the high default level for DMA Channel 8.
7	CH7	**Channel 7 High Priority Disable.** 0: No effect. 1: Use the high default level for DMA Channel 7.
6	CH6	**Channel 6 High Priority Disable.** 0: No effect. 1: Use the high default level for DMA Channel 6.
5	CH5	**Channel 5 High Priority Disable.** 0: No effect. 1: Use the high default level for DMA Channel 5.
4	CH4	**Channel 4 High Priority Disable.** 0: No effect. 1: Use the high default level for DMA Channel 4.
3	CH3	**Channel 3 High Priority Disable.** 0: No effect. 1: Use the high default level for DMA Channel 3.
2	CH2	**Channel 2 High Priority Disable.** 0: No effect. 1: Use the high default level for DMA Channel 2.
1	CH1	**Channel 1 High Priority Disable.** 0: No effect. 1: Use the high default level for DMA Channel 1.
0	CH0	**Channel 0 High Priority Disable.** 0: No effect. 1: Use the high default level for DMA Channel 0.

Register 16.15. DMACTRL0_BERRCLR: Bus Error Clear

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved															ERROR
Type	R															RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

DMACTRL0_BERRCLR = 0x4003_604C

Table 16.22. DMACTRL0_BERRCLR Register Bit Descriptions

Bit	Name	Function
31:1	Reserved	Must write reset value.
0	ERROR	**DMA Bus Error Clear.** Read: 0: DMA error did not occur. 1: DMA error occurred since the last time ERROR was cleared. Write: 0: No effect. 1: Clear the DMA error flag.

16.12. DMACTRL0 Register Memory Map

Table 16.23. DMACTRL0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DMACTRL0_STATUS	0x4003_6000	ALL	Reserved											NUMCHAN					Reserved								STATE				Reserved			DMAENSTS
DMACTRL0_CONFIG	0x4003_6004	ALL	Reserved																															DMAEN
DMACTRL0_BASEPTR	0x4003_6008	ALL	BASEPTR																												Reserved			

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

Table 16.23. DMACTRL0 Memory Map

Register Name	DMACTRL0_ABASEPTR	DMACTRL0_CHSTATUS	DMACTRL0_CHSWRCN
ALL Address	0x4003_600C	0x4003_6010	0x4003_6014
Access Methods	ALL	ALL	ALL
Bit 31	ABASEPTR	Reserved	Reserved
Bit 30			
Bit 29			
Bit 28			
Bit 27			
Bit 26			
Bit 25			
Bit 24			
Bit 23			
Bit 22			
Bit 21			
Bit 20			
Bit 19			
Bit 18			
Bit 17			
Bit 16			
Bit 15		CH15	CH15
Bit 14		CH14	CH14
Bit 13		CH13	CH13
Bit 12		CH12	CH12
Bit 11		CH11	CH11
Bit 10		CH10	CH10
Bit 9		CH9	CH9
Bit 8		CH8	CH8
Bit 7		CH7	CH7
Bit 6		CH6	CH6
Bit 5		CH5	CH5
Bit 4		CH4	CH4
Bit 3		CH3	CH3
Bit 2		CH2	CH2
Bit 1		CH1	CH1
Bit 0		CH0	CH0

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

Table 16.23. DMACTRL0 Memory Map

Register Name	DMACTRL0_CHREQMSET	DMACTRL0_CHREQMCLR	DMACTRL0_CHENSET
ALL Address	0x4003_6020	0x4003_6024	0x4003_6028
Access Methods	ALL	ALL	ALL
Bit 31	Reserved	Reserved	Reserved
Bit 30			
Bit 29			
Bit 28			
Bit 27			
Bit 26			
Bit 25			
Bit 24			
Bit 23			
Bit 22			
Bit 21			
Bit 20			
Bit 19			
Bit 18			
Bit 17			
Bit 16			
Bit 15	CH15	CH15	CH15
Bit 14	CH14	CH14	CH14
Bit 13	CH13	CH13	CH13
Bit 12	CH12	CH12	CH12
Bit 11	CH11	CH11	CH11
Bit 10	CH10	CH10	CH10
Bit 9	CH9	CH9	CH9
Bit 8	CH8	CH8	CH8
Bit 7	CH7	CH7	CH7
Bit 6	CH6	CH6	CH6
Bit 5	CH5	CH5	CH5
Bit 4	CH4	CH4	CH4
Bit 3	CH3	CH3	CH3
Bit 2	CH2	CH2	CH2
Bit 1	CH1	CH1	CH1
Bit 0	CH0	CH0	CH0

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

Table 16.23. DMACTRL0 Memory Map

Register Name	DMACTRL0_CHALTCLR	DMACTRL0_CHALTSET	DMACTRL0_CHENCLR
ALL Address	0x4003_6034	0x4003_6030	0x4003_602C
Access Methods	ALL	ALL	ALL
Bit 31	Reserved	Reserved	Reserved
Bit 30			
Bit 29			
Bit 28			
Bit 27			
Bit 26			
Bit 25			
Bit 24			
Bit 23			
Bit 22			
Bit 21			
Bit 20			
Bit 19			
Bit 18			
Bit 17			
Bit 16			
Bit 15	CH15	CH15	CH15
Bit 14	CH14	CH14	CH14
Bit 13	CH13	CH13	CH13
Bit 12	CH12	CH12	CH12
Bit 11	CH11	CH11	CH11
Bit 10	CH10	CH10	CH10
Bit 9	CH9	CH9	CH9
Bit 8	CH8	CH8	CH8
Bit 7	CH7	CH7	CH7
Bit 6	CH6	CH6	CH6
Bit 5	CH5	CH5	CH5
Bit 4	CH4	CH4	CH4
Bit 3	CH3	CH3	CH3
Bit 2	CH2	CH2	CH2
Bit 1	CH1	CH1	CH1
Bit 0	CH0	CH0	CH0

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

Table 16.23. DMACTRL0 Memory Map

Register Name	DMACTRL0_CHHPSET	DMACTRL0_CHHPCLR	DMACTRL0_BERRCLR
ALL Address	0x4003_6038	0x4003_603C	0x4003_604C
Access Methods	ALL	ALL	ALL
Bit 31	Reserved	Reserved	Reserved
Bit 30			
Bit 29			
Bit 28			
Bit 27			
Bit 26			
Bit 25			
Bit 24			
Bit 23			
Bit 22			
Bit 21			
Bit 20			
Bit 19			
Bit 18			
Bit 17			
Bit 16			
Bit 15	CH15	CH15	
Bit 14	CH14	CH14	
Bit 13	CH13	CH13	
Bit 12	CH12	CH12	
Bit 11	CH11	CH11	
Bit 10	CH10	CH10	
Bit 9	CH9	CH9	
Bit 8	CH8	CH8	
Bit 7	CH7	CH7	
Bit 6	CH6	CH6	
Bit 5	CH5	CH5	
Bit 4	CH4	CH4	
Bit 3	CH3	CH3	
Bit 2	CH2	CH2	
Bit 1	CH1	CH1	
Bit 0	CH0	CH0	ERROR

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

17. DMA Crossbar (DMAXBAR0)

This section describes the DMA Crossbar, and is applicable to all products in the following device families, unless otherwise stated

- SiM3U1xx
- SiM3C1xx

17.1. DMA Crossbar Features

The DMA Crossbar includes the following features:

- Maps multiple peripherals to 16 DMA channels to provide flexibility.
- Default arbitration priority assignment (Channel 0 highest, Channel 15 lowest).

Peripherals are assigned to various channels, and the DMA Crossbar can be used to assign a channel to a particular peripheral. These assignments are shown in Table 17.1.

Table 17.1. DMA Crossbar Channel Peripheral Assignments

Peripheral	DMA Channel 0	DMA Channel 1	DMA Channel 2	DMA Channel 3	DMA Channel 4	DMA Channel 5	DMA Channel 6	DMA Channel 7	DMA Channel 8	DMA Channel 9	DMA Channel 10	DMA Channel 11	DMA Channel 12	DMA Channel 13	DMA Channel 14	DMA Channel 15
AES0 RX							✓					✓				
AES0 TX						✓					✓					
AES0 XOR								✓					✓			
DMAXT0	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
DMAXT1	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
EPCA0 Capture				✓	✓				✓	✓						
EPCA0 Control		✓	✓												✓	✓
I2C0 RX		✓					✓					✓				
I2C0 TX	✓									✓						
I^2S RX					✓	✓					✓	✓				
I^2S TX			✓	✓									✓	✓		
IDAC0				✓			✓								✓	✓
IDAC1		✓	✓										✓	✓		
SARADC0			✓		✓	✓										
SARADC1				✓							✓					✓
SPI0 RX		✓												✓		
SPI0 TX			✓												✓	

Table 17.1. DMA Crossbar Channel Peripheral Assignments (Continued)

Peripheral	DMA Channel 0	DMA Channel 1	DMA Channel 2	DMA Channel 3	DMA Channel 4	DMA Channel 5	DMA Channel 6	DMA Channel 7	DMA Channel 8	DMA Channel 9	DMA Channel 10	DMA Channel 11	DMA Channel 12	DMA Channel 13	DMA Channel 14	DMA Channel 15
SPI1 RX	✓								✓							
SPI1 TX					✓			✓					✓			
TIMER0L Overflow	✓	✓						✓					✓		✓	
TIMER0H Overflow	✓				✓		✓			✓		✓		✓		✓
TIMER1L Overflow	✓	✓						✓					✓		✓	
TIMER1H Overflow	✓	✓		✓				✓			✓		✓			✓
USART0 RX	✓						✓		✓			✓		✓		
USART0 TX			✓		✓			✓							✓	
USART1 RX		✓							✓			✓				
USART1 TX						✓				✓			✓			
USB0 EP1 IN								✓								✓
USB0 EP2 IN							✓								✓	
USB0 EP3 IN						✓								✓		
USB0 EP4 IN					✓								✓			
USB0 EP1 OUT				✓								✓				
USB0 EP2 OUT			✓								✓					
USB0 EP3 OUT		✓								✓						
USB0 EP4 OUT	✓								✓							
Software Trigger	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Notes:
1. USB0 Peripheral only available on SiM3U1xx devices.

17.2. Channel Priority

In addition to the default priority order where DMA Channel 0 has the highest priority and DMA Channel 15 has the lowest priority, each channel supports a programmable priority. This priority can be programmed in the DMA Controller module (DMACTRLn).

17.3. DMAXBAR0 Registers

This section contains the detailed register descriptions for DMAXBAR0 registers.

Register 17.1. DMAXBAR0_DMAXBAR0: Channel 0-7 Trigger Select

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	CH7SEL				CH6SEL				CH5SEL				CH4SEL			
Type	RW				RW				RW				RW			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	CH3SEL				CH2SEL				CH1SEL				CH0SEL			
Type	RW				RW				RW				RW			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

DMAXBAR0_DMAXBAR0 = 0x4003_7000

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 17.2. DMAXBAR0_DMAXBAR0 Register Bit Descriptions

Bit	Name	Function
31:28	CH7SEL	**DMA Channel 7 Peripheral Select.** 0000: Service USB0 EP1 IN data requests. 0001: Service AES0 XOR data requests. 0010: Service SPI1 TX data requests. 0011: Service USART0 TX data requests. 0100: Service DMAXT0 rising edge data requests. 0101: Service DMAXT0 falling edge data requests. 0110: Service DMAXT1 rising edge data requests. 0111: Service DMAXT1 falling edge data requests. 1000: Service TIMER0L overflow data requests. 1001: Service TIMER1L overflow data requests. 1010: Service TIMER1H overflow data requests. 1011-1111: Reserved.

Table 17.2. DMAXBAR0_DMAXBAR0 Register Bit Descriptions

Bit	Name	Function
27:24	CH6SEL	**DMA Channel 6 Peripheral Select.** 0000: Service USB0 EP2 IN data requests. 0001: Service AES0 RX data requests. 0010: Service USART0 RX data requests. 0011: Service I2C0 RX data requests. 0100: Service IDAC0 data requests. 0101: Service DMAXT0 rising edge data requests. 0110: Service DMAXT0 falling edge data requests. 0111: Service DMAXT1 rising edge data requests. 1000: Service DMAXT1 falling edge data requests. 1001: Service TIMER0H overflow data requests. 1010-1111: Reserved.
23:20	CH5SEL	**DMA Channel 5 Peripheral Select.** 0000: Service USB0 EP3 IN data requests. 0001: Service AES0 TX data requests. 0010: Service USART1 TX data requests. 0011: Service SARADC0 data requests. 0100: Service I2S0 RX data requests. 0101: Service DMAXT0 rising edge data requests. 0110: Service DMAXT0 falling edge data requests. 0111: Service DMAXT1 rising edge data requests. 1000: Service DMAXT1 falling edge data requests. 1001-1111: Reserved.
19:16	CH4SEL	**DMA Channel 4 Peripheral Select.** 0000: Service USB0 EP4 IN data requests. 0001: Service SPI1 TX data requests. 0010: Service USART0 TX data requests. 0011: Service SARADC0 data requests. 0100: Service I2S0 RX data requests. 0101: Service EPCA0 capture data requests. 0110: Service DMAXT0 rising edge data requests. 0111: Service DMAXT0 falling edge data requests. 1000: Service DMAXT1 rising edge data requests. 1001: Service DMAXT1 falling edge data requests. 1010: Service TIMER0H overflow data requests. 1011-1111: Reserved.

Table 17.2. DMAXBAR0_DMAXBAR0 Register Bit Descriptions

Bit	Name	Function
15:12	CH3SEL	**DMA Channel 3 Peripheral Select.** 0000: Service USB0 EP1 OUT data requests. 0001: Service SARADC1 data requests. 0010: Service IDAC0 data requests. 0011: Service I2S0 TX data requests. 0100: Service EPCA0 capture data requests. 0101: Service DMAXT0 rising edge data requests. 0110: Service DMAXT0 falling edge data requests. 0111: Service DMAXT1 rising edge data requests. 1000: Service DMAXT1 falling edge data requests. 1001: Service TIMER1H overflow data requests. 1010-1111: Reserved.
11:8	CH2SEL	**DMA Channel 2 Peripheral Select.** 0000: Service USB0 EP2 OUT data requests. 0001: Service SPI0 TX data requests. 0010: Service USART0 TX data requests. 0011: Service SARADC0 data requests. 0100: Service IDAC1 data requests. 0101: Service I2S0 TX data requests. 0110: Service EPCA0 control data requests. 0111: Service DMAXT0 rising edge data requests. 1000: Service DMAXT0 falling edge data requests. 1001: Service DMAXT1 rising edge data requests. 1010: Service DMAXT1 falling edge data requests. 1011-1111: Reserved.
7:4	CH1SEL	**DMA Channel 1 Peripheral Select.** 0000: Service USB0 EP3 OUT data requests. 0001: Service SPI0 RX data requests. 0010: Service USART1 RX data requests. 0011: Service I2C0 RX data requests. 0100: Service IDAC1 data requests. 0101: Service EPCA0 control data requests. 0110: Service DMAXT0 rising edge data requests. 0111: Service DMAXT0 falling edge data requests. 1000: Service DMAXT1 rising edge data requests. 1001: Service DMAXT1 falling edge data requests. 1010: Service TIMER0L overflow data requests. 1011: Service TIMER1L overflow data requests. 1100: Service TIMER1H overflow data requests. 1101-1111: Reserved.

Table 17.2. DMAXBAR0_DMAXBAR0 Register Bit Descriptions

Bit	Name	Function
3:0	CH0SEL	**DMA Channel 0 Peripheral Select.** 0000: Service USB0 EP4 OUT data requests. 0001: Service SPI1 RX data requests. 0010: Service USART0 RX data requests. 0011: Service I2C0 TX data requests. 0100: Service DMAXT0 rising edge data requests. 0101: Service DMAXT0 falling edge data requests. 0110: Service DMAXT1 rising edge data requests. 0111: Service DMAXT1 falling edge data requests. 1000: Service TIMER0L overflow data requests. 1001: Service TIMER0H overflow data requests. 1010: Service TIMER1L overflow data requests. 1011: Service TIMER1H overflow data requests. 1100-1111: Reserved.

Register 17.2. DMAXBAR0_DMAXBAR1: Channel 8-15 Trigger Select

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	CH15SEL				CH14SEL				CH13SEL				CH12SEL			
Type	RW				RW				RW				RW			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	CH11SEL				CH10SEL				CH9SEL				CH8SEL			
Type	RW				RW				RW				RW			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

DMAXBAR0_DMAXBAR1 = 0x4003_7010

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 17.3. DMAXBAR0_DMAXBAR1 Register Bit Descriptions

Bit	Name	Function
31:28	CH15SEL	**DMA Channel 15 Peripheral Select.** 0000: Service USB0 EP1 IN data requests. 0001: Service SARADC1 data requests. 0010: Service IDAC0 data requests. 0011: Service EPCA0 control data requests. 0100: Service DMAXT0 rising edge data requests. 0101: Service DMAXT0 falling edge data requests. 0110: Service DMAXT1 rising edge data requests. 0111: Service DMAXT1 falling edge data requests. 1000: Service TIMER0H overflow data requests. 1001: Service TIMER1H overflow data requests. 1010-1111: Reserved.
27:24	CH14SEL	**DMA Channel 14 Peripheral Select.** 0000: Service USB0 EP2 IN data requests. 0001: Service SPI0 TX data requests. 0010: Service USART0 TX data requests. 0011: Service IDAC0 data requests. 0100: Service EPCA0 control data requests. 0101: Service DMAXT0 rising edge data requests. 0110: Service DMAXT0 falling edge data requests. 0111: Service DMAXT1 rising edge data requests. 1000: Service DMAXT1 falling edge data requests. 1001: Service TIMER0L overflow data requests. 1010: Service TIMER1L overflow data requests. 1011-1111: Reserved.

Table 17.3. DMAXBAR0_DMAXBAR1 Register Bit Descriptions

Bit	Name	Function
23:20	CH13SEL	**DMA Channel 13 Peripheral Select.** 0000: Service USB0 EP3 IN data requests. 0001: Service SPI0 RX data requests. 0010: Service USART0 RX data requests. 0011: Service IDAC1 data requests. 0100: Service I2S0 TX data requests. 0101: Service DMAXT0 rising edge data requests. 0110: Service DMAXT0 falling edge data requests. 0111: Service DMAXT1 rising edge data requests. 1000: Service DMAXT1 falling edge data requests. 1001: Service TIMER0H overflow data requests. 1010-1111: Reserved.
19:16	CH12SEL	**DMA Channel 12 Peripheral Select.** 0000: Service USB0 EP4 IN data requests. 0001: Service AES0 XOR data requests. 0010: Service USART1 TX data requests. 0011: Service SPI1 TX data requests. 0100: Service IDAC1 data requests. 0101: Service I2S0 TX data requests. 0110: Service DMAXT0 rising edge data requests. 0111: Service DMAXT0 falling edge data requests. 1000: Service DMAXT1 rising edge data requests. 1001: Service DMAXT1 falling edge data requests. 1010: Service TIMER0L overflow data requests. 1011: Service TIMER1L overflow data requests. 1100: Service TIMER1H overflow data requests. 1101-1111: Reserved.
15:12	CH11SEL	**DMA Channel 11 Peripheral Select.** 0000: Service USB0 EP1 OUT data requests. 0001: Service AES0 RX data requests. 0010: Service USART1 RX data requests. 0011: Service USART0 RX data requests. 0100: Service I2C0 RX data requests. 0101: Service I2S0 RX data requests. 0110: Service DMAXT0 rising edge data requests. 0111: Service DMAXT0 falling edge data requests. 1000: Service DMAXT1 rising edge data requests. 1001: Service DMAXT1 falling edge data requests. 1010: Service TIMER0H overflow data requests. 1011-1111: Reserved.

Table 17.3. DMAXBAR0_DMAXBAR1 Register Bit Descriptions

Bit	Name	Function
11:8	CH10SEL	**DMA Channel 10 Peripheral Select.** 0000: Service USB0 EP2 OUT data requests. 0001: Service AES0 TX data requests. 0010: Service SARADC1 data requests. 0011: Service I2S0 RX data requests. 0100: Service DMAXT0 rising edge data requests. 0101: Service DMAXT0 falling edge data requests. 0110: Service DMAXT1 rising edge data requests. 0111: Service DMAXT1 falling edge data requests. 1000: Service TIMER1H overflow data requests. 1001-1111: Reserved.
7:4	CH9SEL	**DMA Channel 9 Peripheral Select.** 0000: Service USB0 EP3 OUT data requests. 0001: Service USART1 TX data requests. 0010: Service I2C0 TX data requests. 0011: Service EPCA0 capture data requests. 0100: Service DMAXT0 rising edge data requests. 0101: Service DMAXT0 falling edge data requests. 0110: Service DMAXT1 rising edge data requests. 0111: Service DMAXT1 falling edge data requests. 1000: Service TIMER0H overflow data requests. 1001-1111: Reserved.
3:0	CH8SEL	**DMA Channel 8 Peripheral Select.** 0000: Service USB0 EP4 OUT data requests. 0001: Service USART1 RX data requests. 0010: Service SPI1 RX data requests. 0011: Service USART0 RX data requests. 0100: Service EPCA0 capture data requests. 0101: Service DMAXT0 rising edge data requests. 0110: Service DMAXT0 falling edge data requests. 0111: Service DMAXT1 rising edge data requests. 1000: Service DMAXT1 falling edge data requests. 1001-1111: Reserved.

17.4. DMAXBAR0 Register Memory Map

Table 17.4. DMAXBAR0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
DMAXBAR0_DMAXBAR0	0x4003_7000	ALL | SET | CLR	CH7SEL				CH6SEL				CH5SEL				CH4SEL				CH3SEL				CH2SEL				CH1SEL				CH0SEL			
DMAXBAR0_DMAXBAR1	0x4003_7010	ALL | SET | CLR	CH15SEL				CH14SEL				CH13SEL				CH12SEL				CH11SEL				CH10SEL				CH9SEL				CH8SEL			

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

18. External Memory Interface (EMIF0)

This section describes the External Memory Interface (EMIF) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the EMIF block, which is used by all device families covered in this document.

18.1. EMIF Features

The External Memory Interface allows external parallel asynchronous devices, like SRAMs, NOR flash memories, and LCD controllers, to appear as part of the system memory map. The EMIF module includes the following features:

- Provides a memory mapped view of multiple external devices.
- Support for byte, half-word and word accesses regardless of external device data-width.
- Error indicator for certain invalid transfers.
- Minimum external timing allows for 3 clocks per write or 4 clocks per read.
- Output bus can be shared between non-muxed and muxed devices.
- Extended address output allows for up to 16-bit (non-muxed mode) or 24-bit (muxed mode) address.
- Support for 8-bit and 16-bit (muxed-mode only) devices with up to two chip-select signals.
- Support for internally muxed devices with dynamic address shifting.
- Fully programmable control signal waveforms.

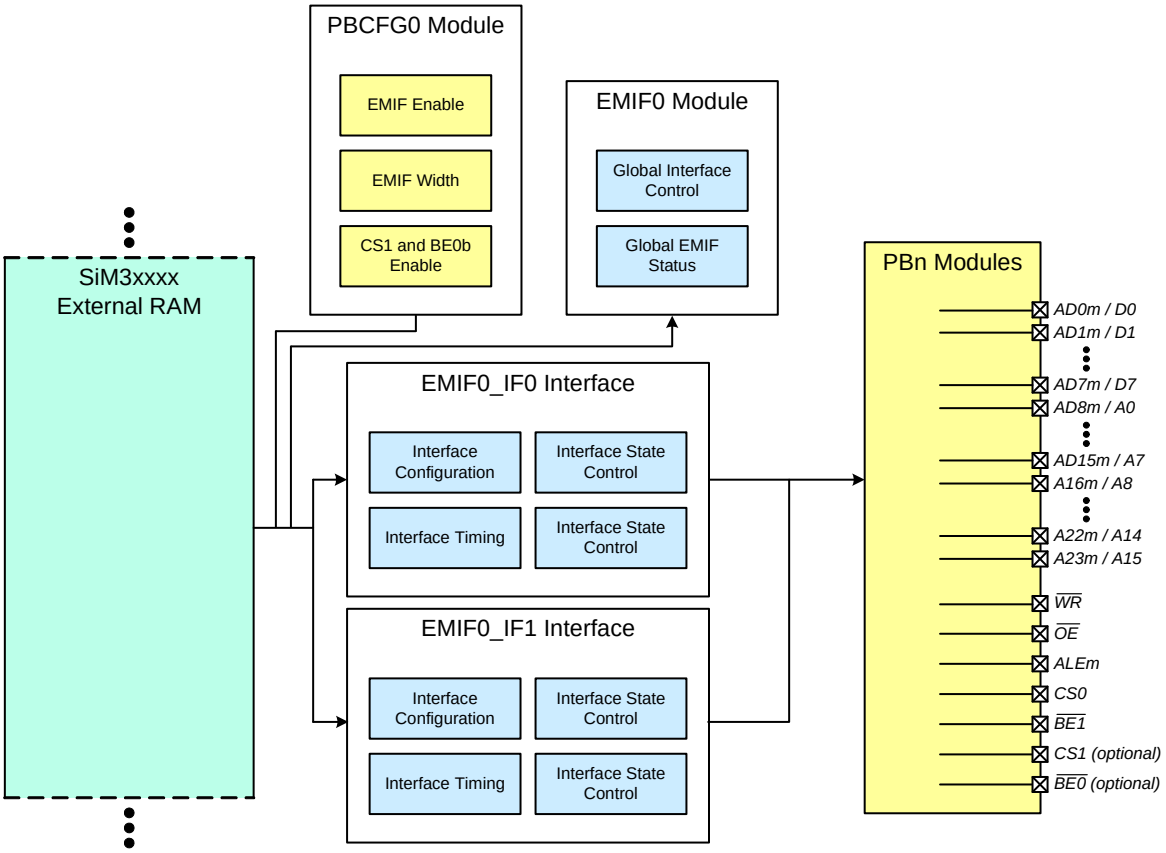


Figure 18.1. EMIF and EMIFIF Block Diagram (Muxed Configuration)

18.2. Overview

The EMIF module is a parallel bus that consists of one global control and multiple, independent interfaces. Each address or data bit requires a separate physical pin to implement the parallel EMIF interface. The global control and status registers enable the interfaces individually (IFxEN) and report the status of the EMIF pins.

The EMIF0_IFx interfaces are accessed using different memory spaces in the external RAM region of the device, starting at address 0x60000000. Each interface is 16 MB in size, so interface 0 begins at address 0x60000000 and interface 1 begins at 0x68000000. The interfaces have separate chip select (CSx) signals. Writing to or reading from the address space of an interface automatically triggers the associated chip select signal. These signals may need to be separately enabled using the device port configuration module.

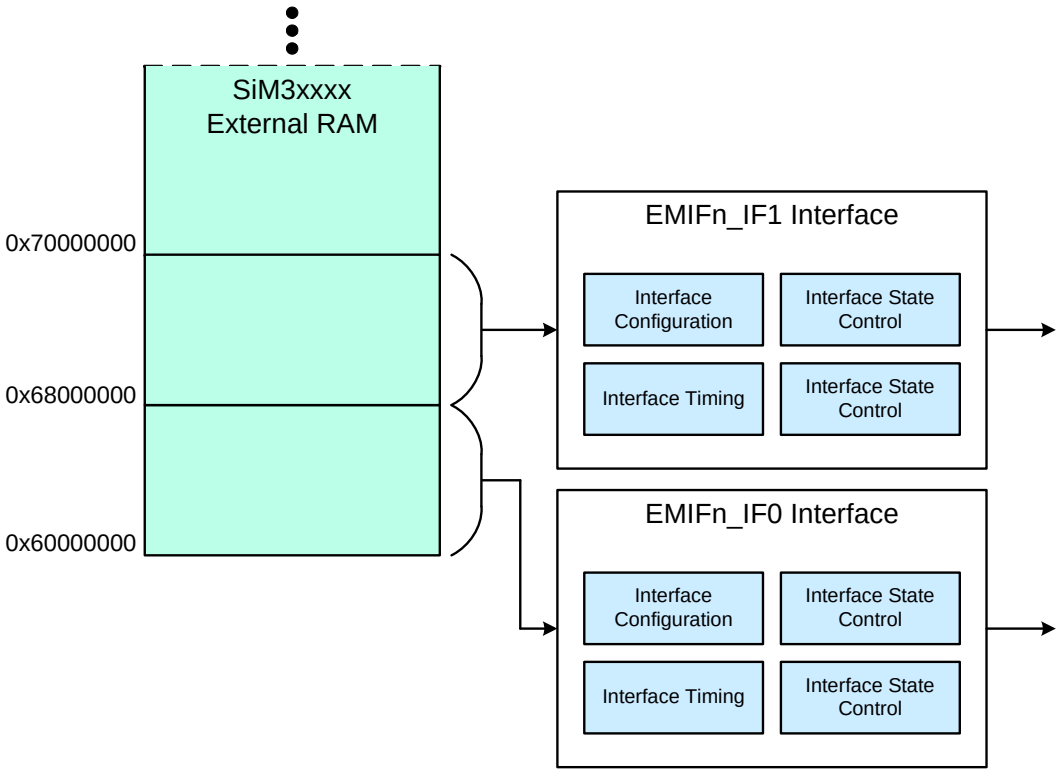


Figure 18.2. External Memory Organization

The EMIF module supports two bus configurations: multiplexed, and non-multiplexed. In non-multiplexed mode, the address and data signals are separate. In multiplexed mode, the address and data signals are shared. An external latch that triggers from the address latch enable (ALEm) control output is required when an interface operates in multiplexed mode and interfaces with a non-multiplexed slave device. This external latch captures the address in the first half of the transaction so the pins can transition to the data phase in the second half of the transaction. For internally multiplexed slave devices, the EMIF module supports dynamic address shifting and separate byte enable signals.

In addition to the two supported bus configurations, when used in multiplexed mode the EMIF interfaces support both 8-bit and 16-bit data bus widths.

The EMIF interfaces can be independently configured to use different bus configurations and bus widths.

18.3. Signal Descriptions

The EMIF consists of the following signals: address and data, write signal ($\overline{WR}$), output enable ($\overline{OE}$), address latch enable (ALEm), chip select (CSx), and byte enable ($\overline{BE0}$, $\overline{BE1}$). The chip select signals are unique for each interface supported, but all other signals are shared between interfaces.

18.3.1. Address and Data

The address and data signals are A16m-A23m, AD0m-AD15m in multiplexed mode and A0-A15, D0-D7 in non-multiplexed mode. Each address or data bit requires a separate physical pin to implement the parallel EMIF interface. The timing of the address and data signals are controlled by the address and data setup and hold times for each interface. Non-multiplexed mode supports an 8-bit data bus. Multiplexed mode supports 8 or 16-bit data buses. The BUSWIDTH bit in the EMIF interfaces selects the width of the data bus for that interface.

The EMIF address width is configured using the EMIFWIDTH field in the PBCFG0_CONTROL1 register. In non-multiplexed mode, the device supports address widths between 0 and 16. The EMIFWIDTH field directly determines how many address lines are used. In multiplexed mode, the device supports address widths between 8 and 24. In muxed mode, the number of address lines are equal to EMIFWIDTH + 8. An additional option when operating in multiplexed mode with a 16-bit data bus is “dynamic address shifting”, in which the address signals A23-A1 are shifted relative to the data signals. More information on dynamic address shifting can be found in “18.6.1. Dynamic Address Shifting” .

Table 18.1 shows the signal mapping for each of the configuration options. These pins are shared between all interfaces supported on the device.

Table 18.1. Address and Data Signal Mapping

Data sheet Pin Name	Non-Multiplexed Mode (MUXMD = 0)	Multiplexed Mode (MUXMD = 1)		
	8-bit Data Bus (BUSWIDTH = 0)	8-bit Data Bus (BUSWIDTH = 0)	16-bit Data Bus (BUSWIDTH = 1)	
			Address Shift Disabled (ASEN = 0)	Address Shift Enabled (ASEN = 1)
AD0m / D0	D0	AD0	AD0	A1 / D0
AD1m / D1	D1	AD1	AD1	A2 / D1
- - -	- - -	- - -	- - -	- - -
AD6m / D6	D6	AD6	AD6	A7 / D6
AD7m / D7	D7	AD7	AD7	A8 / D7
AD8m / A0	A0	A8	AD8	A9 / D8
AD9m / A1	A1	A9	AD9	A10 / D9
- - -	- - -	- - -	- - -	- - -
AD14m / A6	A6	A14	AD14	A15 / D14
AD15m / A7	A7	A15	AD15	A16 / D15
A16m / A8	A8	A16	A16	A17
A17m / A9	A9	A17	A17	A18
- - -	- - -	- - -	- - -	- - -
A22m / A14	A14	A22	A22	A23
A23m / A15	A15	A23	A23	

18.3.2. Write Signal ($\overline{WR}$)

The write signal pin ($\overline{WR}$) indicates when a write to the slave device occurs. The state of the write signal pin during address setup, address hold, data setup, and data hold is programmable for both read (IFRCST) and write (IFWCST) transactions.

This pin is shared between all interfaces supported on the device.

18.3.3. Output Enable ($\overline{OE}$)

The output enable ($\overline{OE}$) signal enables the slave output pins for read operations. The state of the output enable pin during address setup, address hold, data setup, and data hold is programmable for both read (IFRCST) and write (IFWCST) transactions.

In addition to the programmable states, each interface supports an optional output enable delay configured using the DELAYOE bit. When DELAYOE is set to 1, the interface will delay the assertion of the output enable signal by half an AHB clock period.

This pin is shared between all interfaces supported on the device.

18.3.4. Address Latch Enable (ALEm)

The address latch enable (ALEm) signal is used in multiplexed bus modes with non-multiplexed slave devices only. This signal controls the external address latch to capture the address during the first phase of the transaction, so the pins can transition to the data in the second phase of the transaction.

The state of the ALEm pin during address setup, address hold, data setup, and data hold is programmable for both read (IFRCST) and write (IFWCST) transactions.

This signal is also called the address valid (ADVm) signal and is shared between all interfaces supported on the device.

18.3.5. Chip Select (CSx)

The chip select (CSx) signal is separate for each interface. The state of the CSx pin during address setup, address hold, data setup, and data hold is programmable for both read (IFRCST) and write (IFWCST) transactions.

18.3.6. Byte Enable ($\overline{BE0}$, $\overline{BE1}$)

The byte enable signals are used in multiplexed bus modes, and most EMIF slave devices use active-low byte-enable controls. When connecting to a non-multiplexed slave device that requires an external address latch, the lower byte enable signal on the slave device can be connected to the latched version of the least-significant bit of the address.

The EMIF module supports dynamic address shifting to also support internally muxed devices. This means that the least-significant address bit is not available to serve as the lower byte enable. Instead, the EMIF module provides a dedicated byte enable signal that acts as the lower-byte enable. This optional byte enable $\overline{BE0}$ is enabled in the device port configuration module.

The EMIF module also supports an optional upper byte enable ($\overline{BE1}$). If the data size is greater than a byte, this upper byte enable will always be logic 0. Otherwise, this upper byte enable will only be logic 0 when accessing the upper byte in a half word. In the case of read operations from a 16-bit device, the EMIF module always asserts both byte enables.

These pins are shared between all interfaces supported on the device.

18.4. Memory Interface

The EMIF memory interface accepts byte, half-word, or word accesses of any justification. The output of the bus for each of these accesses depends on the bus width of the interface, specified by the BUSWIDTH bit. Figure 18.3 and Figure 18.4 show the generated bus output for the different external RAM accesses.

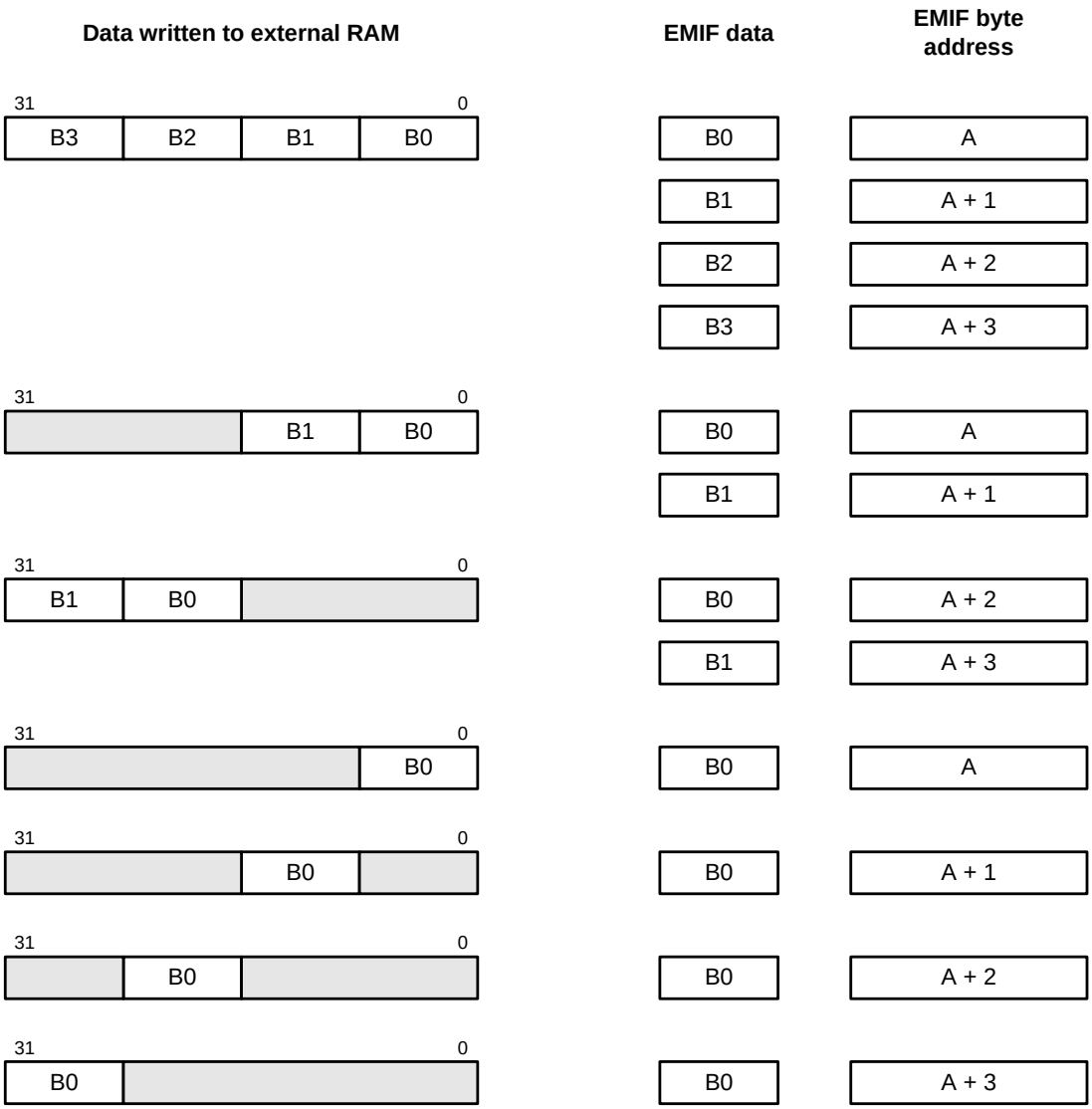


Figure 18.3. EMIF Decoding with 8-bit Bus (BUSWIDTH = 0)

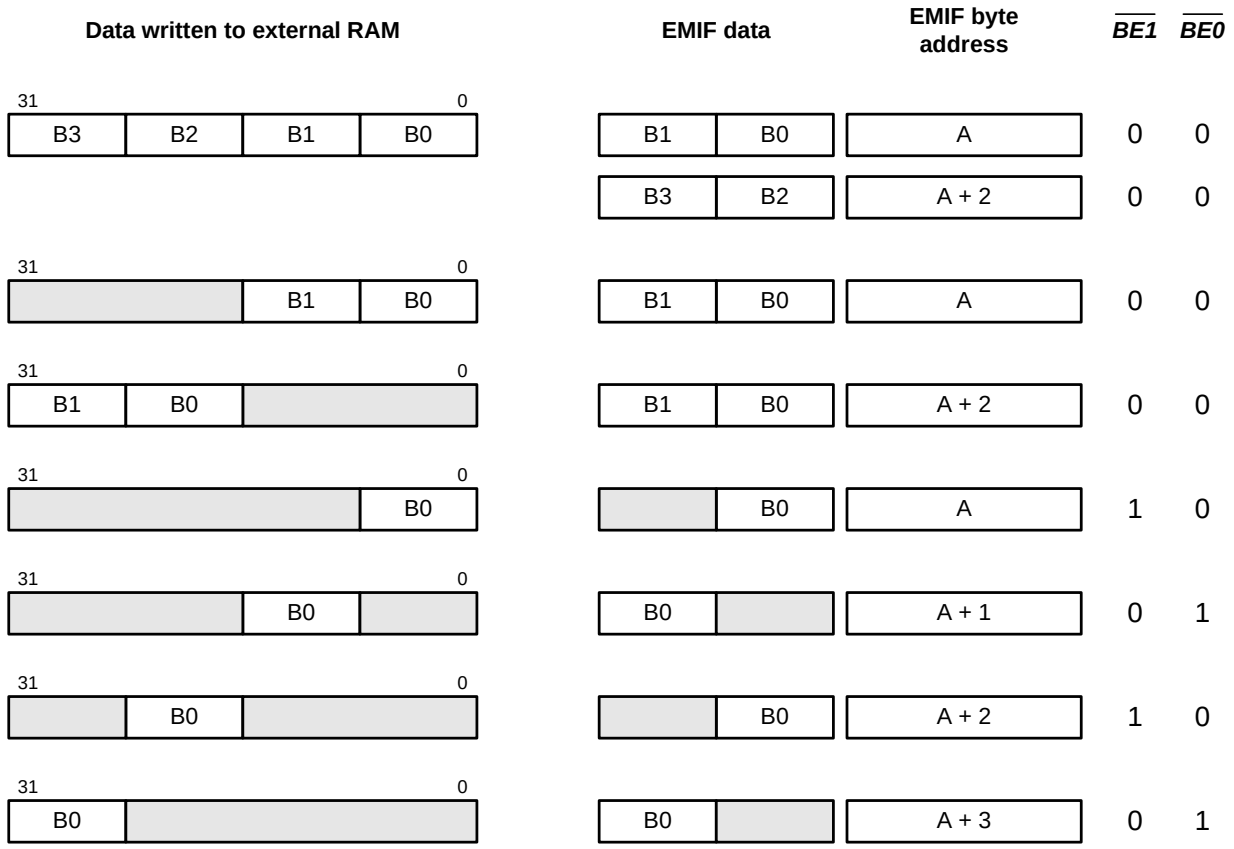


Figure 18.4. EMIF Decoding with 16-bit Bus in Multiplexed Mode (BUSWIDTH = 1)

18.5. Non-Multiplexed Output Mode

Non-multiplexed mode is enabled by setting the MUXMD bit to 0 in the interface registers. Non-multiplexed mode requires more pins than multiplexed mode, but doesn't require an address latch or dynamic address shifting. Figure 18.5 shows an example non-multiplexed hardware configuration.

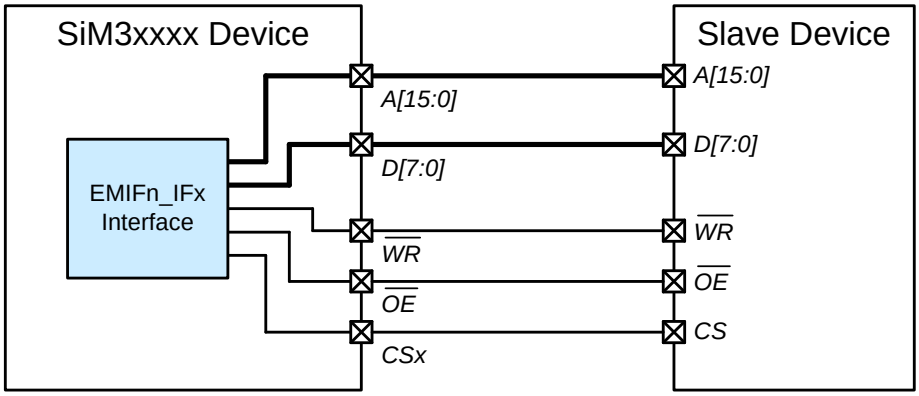


Figure 18.5. Example 8-bit Non-Multiplexed Configuration

The ALEm signal is unused in non-multiplexed mode. In addition, the maximum supported data width is 8 bits, and dynamic address shifting is automatically disabled.

18.6. Multiplexed Output Mode

When the interface MUXMD field is set to 1, the interface operates in multiplexed mode. The address and data pins are shared in this mode. For non-multiplexed slave devices, an external address latch controlled by the ALEm signal is required. Multiplexed mode supports both 8 and 16-bit external data bus widths.

Figure 18.6 shows an example multiplexed hardware configuration with a non-multiplexed slave device supporting 16-bit data and 24-bit address.

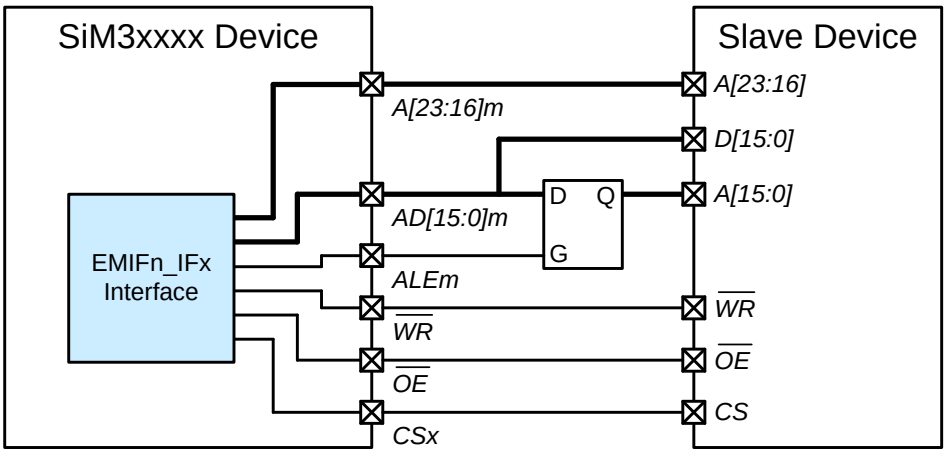


Figure 18.6. Example 8-bit Multiplexed Configuration with Non-Multiplexed Slave Device

18.6.1. Dynamic Address Shifting

The EMIF module supports dynamic address shifting for slave devices that only support an internally multiplexed EMIF interface. These interfaces shift the address A[23:1] to A[22:0]. This address shift cannot be performed using a static external latch like the non-multiplexed slave devices, since the shift would then apply to the data phase and corrupt the data. This dynamic address shifting can be enabled by setting ASEN to 1 in the interface.

All slaves with multiplexed interfaces also include an address valid input (ADVm), which is shared with the EMIF ALEm signal. The ALEm signal pulses low during the address phase, and the slave device samples the address on the rising edge of this signal. When ADVm is de-asserted, the slave device expects valid data to be driven on this same bus.

18.7. Mixing Configurations

Multiple interfaces can mix non-multiplexed and multiplexed modes, in addition to using different bus widths.

In the example shown in Figure 18.7, interface 0 is configured for 8-bit multiplexed mode with a non-multiplexed slave, and interface 1 is configured for 8-bit non-multiplexed mode. Note that the multiplexed ADm[15:8] pins are physically shared with the non-multiplexed A[7:0], and the multiplexed ADm[7:0] pins are physically shared with non-multiplexed D[7:0].

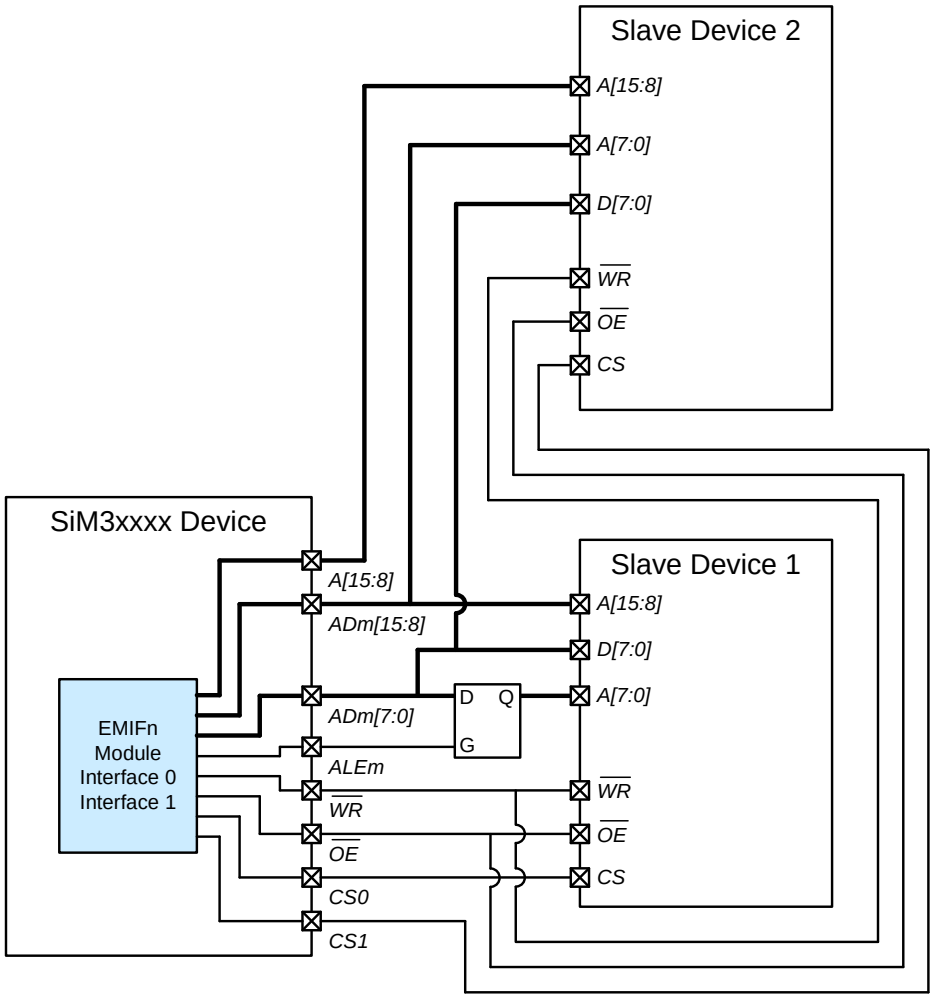


Figure 18.7. Example Mixed Configuration with Multiple Slaves

18.8. Transaction Timing

Each EMIF interface transaction consists of four states: address setup, address hold, data wait, and data hold. The timing for these states is programmable for both read and write operations. Figure 18.8, Figure 18.9, Figure 18.10, and Figure 18.11 illustrate the states of each transaction.

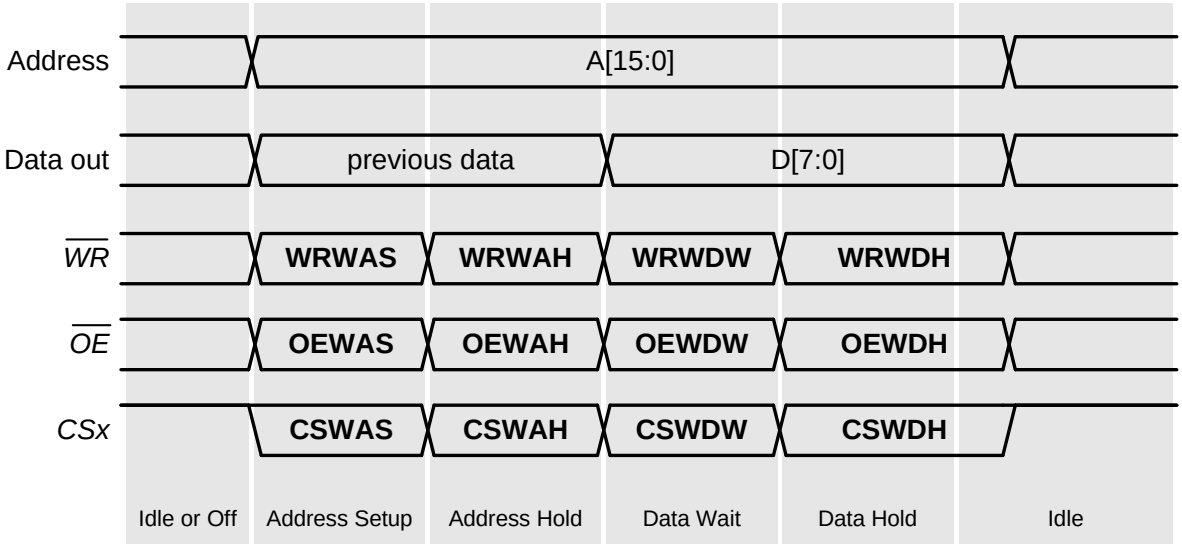


Figure 18.8. EMIF Non-Multiplexed Transaction Timing (8-bit Bus, Write)

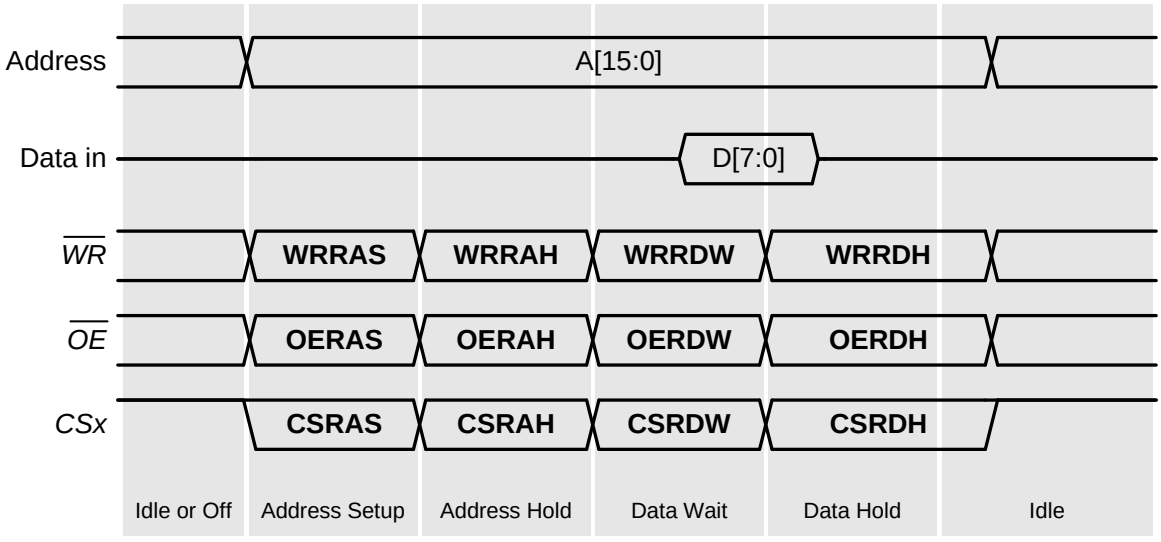


Figure 18.9. EMIF Non-Multiplexed Transaction Timing (8-bit Bus, Read)

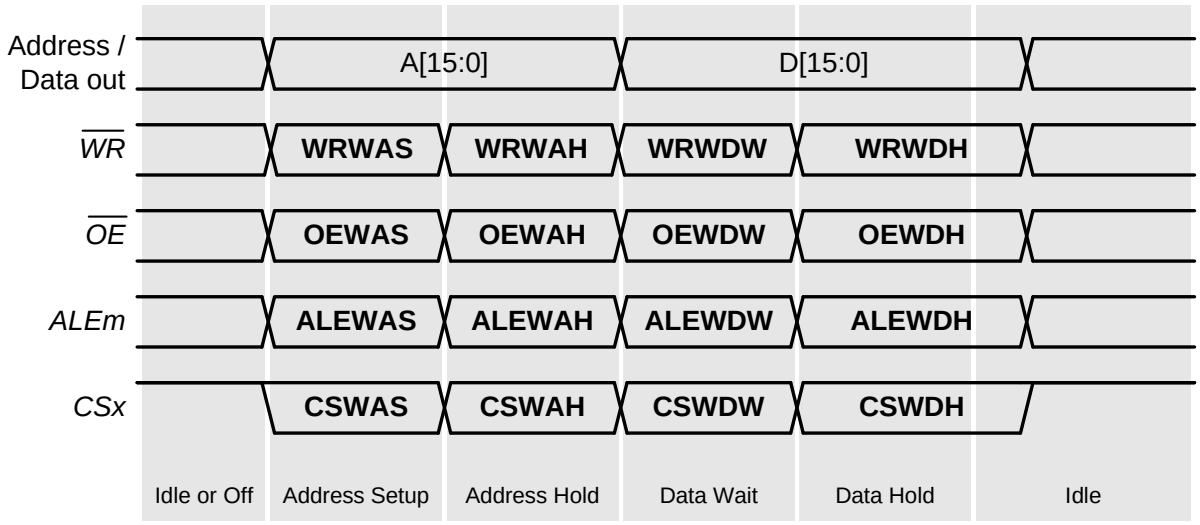


Figure 18.10. EMIF Multiplexed Transaction Timing (16-bit Bus, Write)

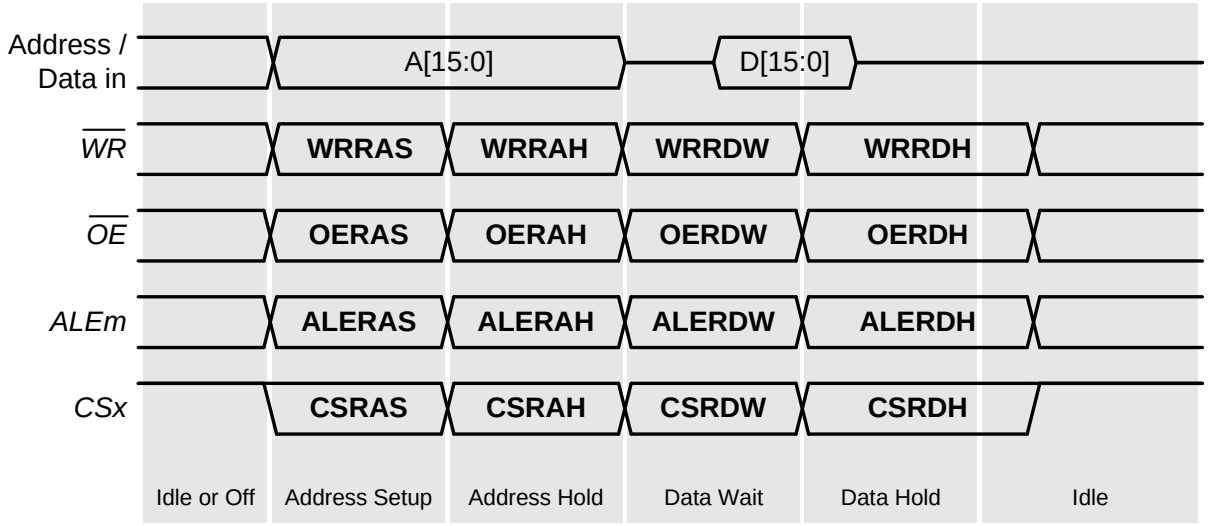


Figure 18.11. EMIF Multiplexed Transaction Timing (16-bit Bus, Read)

The RASET, RAHOLD, RDHOLD, and RDWAIT fields control the state timing for read operations. The WASET, WAHOLD, WDHOLD, and WDWAIT fields control the timing for write operations.

18.9. Idle and Off States

The interface returns to the idle state set by the pin latches in between transactions. Firmware can check whether the EMIF module has been idle for 4 cycles using the IDLESTS bit.

The EMIF module can optionally transition to the off state after 4 idle cycles. When off, the EMIF pins will automatically disable their output drivers, allowing the slave device to be shared amongst multiple master devices. The drivers will re-enable automatically when firmware reads or writes an address in the external RAM memory space. This feature can be enabled by setting OFFSTEN to 1. Firmware can check the status of the EMIF using the OFFSTS bit.

18.10. Additional Features

18.10.1. Read-Only Mode

An interface can be placed in read-only mode by setting the ROEN interface bit to 1. When ROEN is cleared to 0, the interface supports both read and write operations. A write operation initiated with an interface that is read-only can generate an AHB error.

18.10.2. Write Data Hold Inhibit

When WDHINH is set to 1, the interface inhibits the data hold state for write transfers. This can speed up write transactions for slave devices that can support it and allows for a minimum of 3 clocks per write transaction. The WDHOLD field is ignored when WDHINH is set to 1.

18.10.3. Keep Last Read Enable

When KLREN is set to 1, the interface drives the last value read on the interface bus until the next active request. The interface will drive the idle states on the pins when idle until the first read operation.

18.11. Configuring the External Memory Interface

To configure the EMIF module:

1. Configure the output modes of the associated port pins as either push-pull or open-drain (push-pull is most common), and skip the associated pins in the crossbars using the port configuration module.
2. Configure the Port Bank latches in the device port configuration module to set the idle states for the EMIF pins.
3. Select multiplexed mode or non-multiplexed mode using the interface MUXMD bit.
4. If in multiplexed mode, select the data bus width for the interface using the BUSWIDTH bit. BUSWIDTH should be 0 (8-bits) for non-multiplexed mode.
5. Select the desired EMIF address width using the EMIFWIDTH field in the PBCFG0_CONTROL1 register.
6. (Optional) Enable dynamic address shifting for internally multiplexed slave devices (ASEN = 1).
7. (Optional) Enable the off state (OFFSTEN = 1) for multi-master configurations.
8. (Optional) Enable the optional chip select and output byte enable using the device port configuration module.
9. Set the CSx, $\overline{OE}$, $\overline{WR}$, and ALEm states for read and write operations using the IFRCST and IFWCST registers.
10. Configure the interface timing (RASET, RAHOLD, RDHOLD, RDWAIT, WASET, WAHOLD, WDHOLD, and WDWAIT fields) to values compatible with the slave device or devices.
11. Enable the interface using the IFxEN bit.

Firmware can then read and write the slave device by writing or reading the correct addresses in the external RAM memory space.

18.12. EMIF0 Registers

This section contains the detailed register descriptions for EMIF0 registers.

Register 18.1. EMIF0_CONTROL: Module Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved											OFFSTEN	Reserved		IF1EN	IF0EN
Type	R											RW	R		RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Register ALL Access Address

EMIF0_CONTROL = 0x4002_6000

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 18.2. EMIF0_CONTROL Register Bit Descriptions

Bit	Name	Function
31:5	Reserved	Must write reset value.
4	OFFSTEN	**OFF Output State Enable.** This bit causes the EMIF to go into an off state after 4 idle cycles, disabling the signal drivers. 0: EMIF will not enter the off state after 4 idle cycles. 1: EMIF will enter the off state after 4 idle cycles.
3:2	Reserved	Must write reset value.
1	IF1EN	**Interface 1 Enable.** Writing or reading from the interface 1 addresses when interface 1 is enabled will cause the associated chip select (CS1) to activate. If interface 1 is disabled, writing or reading from the interface 1 addresses will cause a Hard Fault error.
0	IF0EN	**Interface 0 Enable.** Writing or reading from the interface 0 addresses when interface 0 is enabled will cause the associated chip select (CS0) to activate. If interface 0 is disabled, writing or reading from the interface 0 addresses will cause a Hard Fault error.

Register 18.2. EMIF0_STATUS: Module Status

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved														IDLESTS	OFFSTS
Type	R														R	R
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Register ALL Access Address EMIF0_STATUS = 0x4002_6020																

Table 18.3. EMIF0_STATUS Register Bit Descriptions

Bit	Name	Function
31:2	Reserved	Must write reset value.
1	IDLESTS	**EMIF IDLE Status.** 0: The EMIF has not been idle for four cycles. 1: The EMIF has been idle for four four cycles.
0	OFFSTS	**EMIF OFF Status.** 0: The EMIF bus is active. 1: The EMIF is in the off bus state.

18.13. EMIF0 Register Memory Map

Table 18.4. EMIF0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EMIF0_CONTROL	0x4002_6000	ALL | SET | CLR	Reserved																											OFFSTEN	Reserved		IF1EN	IF0EN
EMIF0_STATUS	0x4002_6020	ALL	Reserved																														IDLESTS	OFFSTS

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

18.14. EMIF0_IFx Registers

This section contains the detailed register descriptions for EMIF0_IF0 and EMIF0_IF1 registers.

Register 18.3. EMIFn_IFx_CONFIG: Interface Configuration

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															KLREN
Type	R															RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved			DELAYOE	Reserved			WDHINH	Reserved			ROEN	ASEN	MUXMD	Reserved	BUSWIDTH
Type	R			RW	R			RW	R			RW	RW	RW	R	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

EMIF0_IF0_CONFIG = 0x4002_6080
EMIF0_IF1_CONFIG = 0x4002_6100
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 18.5. EMIFn_IFx_CONFIG Register Bit Descriptions

Bit	Name	Function
31:17	Reserved	Must write reset value.
16	KLREN	**Keep Last Read Enable.** When KLREN is set to 1, the interface drives the last read value out on the interface bus until the next active request. 0: The bus is driven to the idle state between active requests. 1: The bus drives the last value read on the interface between active requests.
15:13	Reserved	Must write reset value.
12	DELAYOE	**Output Enable Delay.** When DELAYOE is set to 1, the interface will delay the assertion of the output enable signal ($\overline{OE}$) by 1/2 an AHB cycle. 0: The output enable signal ($\overline{OE}$) is not delayed. 1: The output enable signal ($\overline{OE}$) is delayed.
11:9	Reserved	Must write reset value.

Table 18.5. EMIFn_IFx_CONFIG Register Bit Descriptions

Bit	Name	Function
8	WDHINH	**Write Data Hold State Inhibit.** When WDHINH is set to 1, the interface inhibits the data hold state for write transfers. 0: Enable the write data hold state. 1: Inhibit the write data hold state.
7:5	Reserved	Must write reset value.
4	ROEN	**Interface Read Only Enable.** 0: The interface supports reads and writes. 1: The interface supports only reads.
3	ASEN	**Interface Automatic Address Shift Enable.** When connecting to a 16-bit internally muxed device and ASEN is set, the EMIF will automatically shift the address on the address/data bus during the address phase. This feature must be used with a bus width of 16 bits (BUSWIDTH = 1) and muxed mode (MUXMD = 1). If BUSWIDTH or MUXMD is not set to 1, setting this bit will have no effect. 0: The address is not automatically shifted. 1: The address is automatically shifted.
2	MUXMD	**Interface Mux Mode.** When MUXMD is set to 1, the lower bits of the address are driven on the lower bits of the bus during the address setup and address hold states. 0: The interface operates in non-multiplexed mode. 1: The interface operates in multiplexed mode.
1	Reserved	Must write reset value.
0	BUSWIDTH	**Interface Bus Data Width.** 0: The data bus is 8-bits wide. 1: The data bus is 16-bits wide. This option should only be used in multiplexed mode (MUXMD = 1).

Register 18.4. EMIFn_IFx_IFRT: Interface Read Timing

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved										RDWAIT					
Type	R										RW					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved				RDHOLD				RAHOLD				RASET			
Type	R				RW				RW				RW			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses
EMIF0_IF0_IFRT = 0x4002_6090
EMIF0_IF1_IFRT = 0x4002_6110

Table 18.6. EMIFn_IFx_IFRT Register Bit Descriptions

Bit	Name	Function
31:22	Reserved	Must write reset value.
21:16	RDWAIT	**Interface Read Data Wait Delay.** The read data wait state delay is set by this field according to the equation: $t_{RDW} = \frac{(RDWAIT + 1)}{F_{AHB}}$
15:12	Reserved	Must write reset value.
11:8	RDHOLD	**Interface Read Data Hold Delay.** The read data hold state delay is set by this field according to the equation: $t_{RDH} = \frac{(RDHOLD + 1)}{F_{AHB}}$
7:4	RAHOLD	**Interface Read Address Hold Delay.** The read address hold state delay is set by this field according to the equation: $t_{RAH} = \frac{RAHOLD}{F_{AHB}}$

Table 18.6. EMIFn_IFx_IFRT Register Bit Descriptions

Bit	Name	Function
3:0	RASET	**Interface Read Address Setup Delay .** The read address setup state delay is set by this field according to the equation: $t_{RAS} = \frac{(RASET + 1)}{F_{AHB}}$

Register 18.5. EMIFn_IFx_IFWT: Interface Write Timing

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved										WDWAIT					
Type	R										RW					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved				WDHOLD				WAHOLD				WASET			
Type	R				RW				RW				RW			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

EMIF0_IF0_IFWT = 0x4002_60A0

EMIF0_IF1_IFWT = 0x4002_6120

Table 18.7. EMIFn_IFx_IFWT Register Bit Descriptions

Bit	Name	Function
31:22	Reserved	Must write reset value.
21:16	WDWAIT	**Interface Write Data Wait Delay.** The write data wait state delay is set by this field according to the equation: $t_{WDW} = \frac{(WDWAIT + 1)}{F_{AHB}}$
15:12	Reserved	Must write reset value.
11:8	WDHOLD	**Interface Write Data Hold Delay.** The write data hold state delay is set by this field according to the equation: $t_{WDH} = \frac{(WDHOLD + 1)}{F_{AHB}}$
7:4	WAHOLD	**Interface Write Address Hold Delay.** The write address hold state delay is set by this field according to the equation: $t_{WAH} = \frac{WAHOLD}{F_{AHB}}$

Table 18.7. EMIFn_IFx_IFWT Register Bit Descriptions

Bit	Name	Function
3:0	WASET	**Interface Write Address Setup Delay .** The write address setup state delay is set by this field according to the equation: $t_{WAS} = \frac{(WASET + 1)}{F_{AHB}}$

Register 18.6. EMIFn_IFx_IFRCST: Interface Read Control States

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	ALERDH	ALERDW	ALERAH	ALERAS	WRRDH	WRRDW	WRRAH	WRRAS	OERDH	OERDW	OERAH	OERAS	CSRDH	CSRDW	CSRAH	CSRAS
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	1	1	1	1	1	0	1	1	1	0	1	1

Register ALL Access Addresses

EMIF0_IF0_IFRCST = 0x4002_60B0

EMIF0_IF1_IFRCST = 0x4002_6130

Table 18.8. EMIFn_IFx_IFRCST Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15	ALERDH	**Address Latch Enable Read Data Hold State.** 0: Set address latch enable (ALEm) to low during the read data hold state. 1: Set address latch enable (ALEm) to high during the read data hold state.
14	ALERDW	**Address Latch Enable Read Data Wait State.** 0: Set address latch enable (ALEm) to low during the read data wait state. 1: Set address latch enable (ALEm) to high during the read data wait state.
13	ALERAH	**Address Latch Enable Read Address Hold State.** 0: Set address latch enable (ALEm) to low during the read address hold state. 1: Set address latch enable (ALEm) to high during the read address hold state.
12	ALERAS	**Address Latch Enable Read Address Setup State.** 0: Set address latch enable (ALEm) to low during the read address setup state. 1: Set address latch enable (ALEm) to high during the read address setup state.
11	WRRDH	**Write Signal Read Data Hold State.** 0: Set write signal ($\overline{WR}$) to low during the read data hold state. 1: Set write signal ($\overline{WR}$) to high during the read data hold state.
10	WRRDW	**Write Signal Read Data Wait State.** 0: Set write signal ($\overline{WR}$) to low during the read data wait state. 1: Set write signal ($\overline{WR}$) to high during the read data wait state.

Table 18.8. EMIFn_IFx_IFRCST Register Bit Descriptions

Bit	Name	Function
9	WRRAH	**Write Signal Read Address Hold State.** 0: Set write signal ($\overline{WR}$) to low during the read address hold state. 1: Set write signal ($\overline{WR}$) to high during the read address hold state.
8	WRRAS	**Write Signal Read Address Setup State.** 0: Set write signal ($\overline{WR}$) to low during the read address setup state. 1: Set write signal ($\overline{WR}$) to high during the read address setup state.
7	OERDH	**Output Enable Read Data Hold State.** 0: Set output enable ($\overline{OE}$) to low during the read data hold state. 1: Set output enable ($\overline{OE}$) to high during the read data hold state.
6	OERDW	**Output Enable Read Data Wait State.** 0: Set output enable ($\overline{OE}$) to low during the read data wait state. 1: Set output enable ($\overline{OE}$) to high during the read data wait state.
5	OERAH	**Output Enable Read Address Hold State.** 0: Set output enable ($\overline{OE}$) to low during the read address hold state. 1: Set output enable ($\overline{OE}$) to high during the read address hold state.
4	OERAS	**Output Enable Read Address Setup State.** 0: Set output enable ($\overline{OE}$) to low during the read address setup state. 1: Set output enable ($\overline{OE}$) to high during the read address setup state.
3	CSRDH	**Chip Select Read Data Hold State.** 0: Set chip select (CSx) to low during the read data hold state. 1: Set chip select (CSx) to high during the read data hold state.
2	CSRDW	**Chip Select Read Data Wait State.** 0: Set chip select (CSx) to low during the read data wait state. 1: Set chip select (CSx) to high during the read data wait state.
1	CSRAH	**Chip Select Read Address Hold State.** 0: Set chip select (CSx) to low during the read address hold state. 1: Set chip select (CSx) to high during the read address hold state.
0	CSRAS	**Chip Select Read Address Setup State.** 0: Set chip select (CSx) to low during the read address setup state. 1: Set chip select (CSx) to high during the read address setup state.

Register 18.7. EMIFn_IFx_IFWCST: Interface Write Control States

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	ALEWDH	ALEWDW	ALEWAH	ALEWAS	WRWDH	WRWDW	WRWAH	WRWAS	OEWDH	OEWDW	OEWAH	OEWAS	CSWDH	CSWDW	CSWAH	CSWAS
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	1	0	1	1	1	1	1	1	1	0	1	1

Register ALL Access Addresses

EMIF0_IF0_IFWCST = 0x4002_60C0
EMIF0_IF1_IFWCST = 0x4002_6140

Table 18.9. EMIFn_IFx_IFWCST Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15	ALEWDH	**Address Latch Enable Write Data Hold State.** 0: Set address latch enable (ALEm) to low during the write data hold state. 1: Set address latch enable (ALEm) to high during the write data hold state.
14	ALEWDW	**Address Latch Enable Write Data Wait State.** 0: Set address latch enable (ALEm) to low during the write data wait state. 1: Set address latch enable (ALEm) to high during the write data wait state.
13	ALEWAH	**Address Latch Enable Write Address Hold State.** 0: Set address latch enable (ALEm) to low during the write address hold state. 1: Set address latch enable (ALEm) to high during the write address hold state.
12	ALEWAS	**Address Latch Enable Write Address Setup State.** 0: Set address latch enable (ALEm) to low during the write address setup state. 1: Set address latch enable (ALEm) to high during the write address setup state.
11	WRWDH	**Write Signal Write Data Hold State.** 0: Set write signal ($\overline{WR}$) to low during the write data hold state. 1: Set write signal ($\overline{WR}$) to high during the write data hold state.
10	WRWDW	**Write Signal Write Data Wait State.** 0: Set write signal ($\overline{WR}$) to low during the write data wait state. 1: Set write signal ($\overline{WR}$) to high during the write data wait state.

Table 18.9. EMIFn_IFx_IFWCST Register Bit Descriptions

Bit	Name	Function
9	WRWAH	**Write Signal Write Address Hold State.** 0: Set write signal ($\overline{WR}$) to low during the write address hold state. 1: Set write signal ($\overline{WR}$) to high during the write address hold state.
8	WRWAS	**Write Signal Write Address Setup State.** 0: Set write signal ($\overline{WR}$) to low during the write address setup state. 1: Set write signal ($\overline{WR}$) to high during the write address setup state.
7	OEWDH	**Output Enable Write Data Hold State.** 0: Set output enable ($\overline{OE}$) to low during the write data hold state. 1: Set output enable ($\overline{OE}$) to high during the write data hold state.
6	OEWDW	**Output Enable Write Data Wait State.** 0: Set output enable ($\overline{OE}$) to low during the write data wait state. 1: Set output enable ($\overline{OE}$) to high during the write data wait state.
5	OEWAH	**Output Enable Write Address Hold State.** 0: Set output enable ($\overline{OE}$) to low during the write address hold state. 1: Set output enable ($\overline{OE}$) to high during the write address hold state.
4	OEWAS	**Output Enable Write Address Setup State.** 0: Set output enable ($\overline{OE}$) to low during the write address setup state. 1: Set output enable ($\overline{OE}$) to high during the write address setup state.
3	CSWDH	**Chip Select Write Data Hold State.** 0: Set chip select (CSx) to low during the write data hold state. 1: Set chip select (CSx) to high during the write data hold state.
2	CSWDW	**Chip Select Write Data Wait State.** 0: Set chip select (CSx) to low during the write data wait state. 1: Set chip select (CSx) to high during the write data wait state.
1	CSWAH	**Chip Select Write Address Hold State.** 0: Set chip select (CSx) to low during the write address hold state. 1: Set chip select (CSx) to high during the write address hold state.
0	CSWAS	**Chip Select Write Address Setup State.** 0: Set chip select (CSx) to low during the write address setup state. 1: Set chip select (CSx) to high during the write address setup state.

18.15. EMIFn_IFx Register Memory Map

Table 18.10. EMIFn_IFx Memory Map

Register Name	ALL Offset	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EMIFn_IFx_CONFIG	0x0	ALL | SET | CLR	Reserved															KLREN	Reserved			DELAYOE	Reserved			WDHINH	Reserved			ROEN	ASEN	MUXMD	Reserved	BUSWIDTH
EMIFn_IFx_IFRT	0x10	ALL	Reserved											RDWAIT					Reserved				RDHOLD				RAHOLD				RASET			
EMIFn_IFx_IFWT	0x20	ALL	Reserved											WDWAIT					Reserved				WDHOLD				WAHOLD				WASET			

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: EMIF0_IF0 = 0x4002_6080, EMIF0_IF1 = 0x4002_6100

Table 18.10. EMIFn_IFx Memory Map

Register Name	ALL Offset	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EMIFn_IFx_IFRCST	0x30	ALL	Reserved																ALERDH	ALERDW	ALERAH	ALERAS	WRRDH	WRRDW	WRRAH	WRRAS	OERDH	OERDW	OERAH	OERAS	CSRDH	CSRDW	CSRAH	CSRAS
EMIFn_IFx_IFWCST	0x40	ALL	Reserved																ALEWDH	ALEWDW	ALEWAH	ALEWAS	WRWDH	WRWDW	WRWAH	WRWAS	OEWDH	OEWDW	OEWAH	OEWAS	CSWDH	CSWDW	CSWAH	CSWAS

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: EMIF0_IF0 = 0x4002_6080, EMIF0_IF1 = 0x4002_6100

19. External Oscillator (EXTOSC0)

This section describes the External Oscillator (EXTOSC) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the EXTOSC block, which is used by all device families covered in this document.

19.1. External Oscillator Features

The External Oscillator control has the following features:

- Support for external crystal or ceramic resonator, RC, C, or CMOS oscillators.
- Support external CMOS frequencies from 10 kHz to 50 MHz and external crystal frequencies from 10 kHz to 30 MHz.
- Various drive strengths for flexible crystal oscillator support.
- Internal frequency divide-by-two option available for some oscillator modes.
- Available as an input to the PLL.

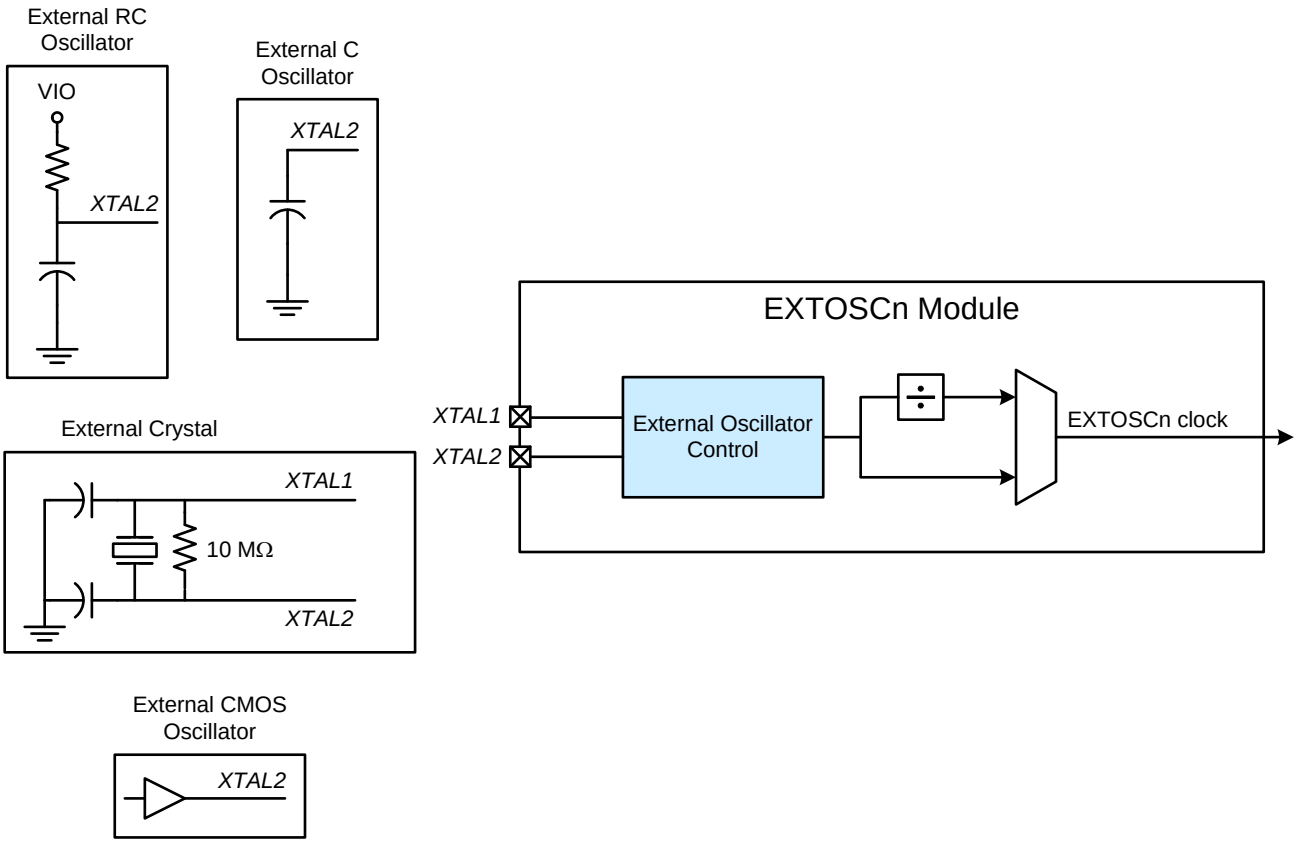


Figure 19.1. External Oscillator Block Diagram

19.2. Introduction

The EXTOSC external oscillator circuit may drive an external crystal, ceramic resonator, capacitor, or RC network. A CMOS clock may also provide a clock input. The external oscillator output may be selected as the AHB clock or used to clock other modules independent of the AHB clock selection.

19.3. External Crystal Oscillator

When OSCMD is 6 or 7, the EXTOSC module is in crystal oscillator mode. The crystal or ceramic resonator and a 10 MΩ resistor must be wired across the XTAL1 and XTAL2 pins as shown in Figure 19.2. Appropriate loading capacitors should be added to XTAL1 and XTAL2, and both pins should be configured for analog mode as described in the port configuration module.

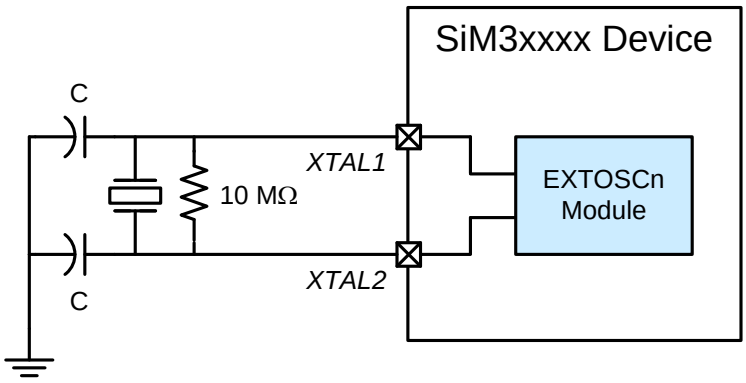


Figure 19.2. External Crystal Oscillator Configuration

The capacitors provide the load capacitance required by the crystal for correct oscillation. These capacitors are “in series” as seen by the crystal and “in parallel” with the stray capacitance of the XTAL1 and XTAL2 pins.

Note: The recommended load capacitance depends upon the crystal and the manufacturer. Refer to the crystal data sheet when completing these calculations.

Equation 19.1 describes the equation for determining the load capacitance for the two capacitors. The C_A and C_B values are the capacitors connected to the crystal leads. The C_S value is the total stray capacitance of the PCB.

$$C_L = \frac{C_A \times C_B}{C_A + C_B} + C_S$$

Equation 19.1. Crystal Load Capacitors

If C_A and C_B are the same (C), the resulting equation is shown in Equation 19.2.

$$C_L = \frac{C}{2} + C_S$$

Equation 19.2. Simplified Crystal Load Capacitors

For example, using Equation 19.2 with a 32.768 kHz tuning-fork crystal with a recommended load capacitance of 12.5 pF placed as close to the pins as possible (assuming 6 pF total stray capacitance) results in crystal load capacitors of 13 pF each.

Note: Crystal oscillator circuits are quite sensitive to PCB layout. The crystal should be placed as close as possible to the XTAL pins on the device. The traces should be as short as possible and shielded with ground plane from any other traces that could introduce noise or interference.

Setting OSCMD to 6 places EXTOSC in crystal oscillator mode, and setting OSCMD to 7 sets the module in crystal oscillator divided by 2 mode. This divide-by-2 stage ensures that the clock derived from the external oscillator has a duty cycle of 50%. When operating in either crystal mode, the frequency control (FREQCN) field must be set to the appropriate value based on the crystal frequency.

Table 19.1. Frequency Control for Crystal Oscillators

Crystal Frequency	FREQCN Value
10 kHz < f ≤ 20 kHz	0
20 kHz < f ≤ 58 kHz	1
58 kHz < f ≤ 155 kHz	2
155 kHz < f ≤ 415 kHz	3
415 kHz < f ≤ 1.1 MHz	4
1.1 MHz < f ≤ 3.1 MHz	5
3.1 MHz < f ≤ 8.2 MHz	6
8.2 MHz < f ≤ 25 MHz	7

19.3.1. Configuring for Crystal Oscillator Mode

The recommended procedure for starting the crystal is as follows:

1. Configure the XTAL1 and XTAL2 pins for analog mode using the device port configuration module.
2. Disable the XTAL1 and XTAL2 digital output drivers by writing 1's to the pin latch in the Port Bank registers.
3. Configure the FREQCN field for the crystal frequency according to Table 19.1.
4. Set the OSCMD field to 6 for crystal mode or 7 for crystal divided by 2 mode.
5. Wait at least 1 ms.
6. Poll on the OSCVLDF flag to determine if the oscillator is running and stable.
7. Set the EXTOSCEN bit in CLKCTRL0_CONFIG to 1, to enable the external oscillator as a clock source.
8. Select the external oscillator as the AHB clock or as the input clock for a module.

19.4. External CMOS Oscillator

If an external CMOS clock is used as the external oscillator, the clock should be directly routed into XTAL2, and the XTAL2 pin should be configured as a digital input using the device port configuration module. XTAL1 is not used in external CMOS clock mode.

The CMOS oscillator mode is available with a divide by 2 stage, which ensures that the clock derived from the external oscillator has a duty cycle of 50%.

The external oscillator valid (OSCVLDF) flag will always return zero when the external oscillator is configured to external CMOS clock mode.

19.4.1. Configuring for CMOS Oscillator Mode

The recommended procedure for enabling the CMOS oscillator is:

1. Configure the XTAL2 pins for digital input mode using the device port configuration module.
2. Set the OSCMD field to 2 for CMOS mode or 3 for CMOS mode with divide by 2 stage.
3. Set the EXTOSCEN bit in CLKCTRL0_CONFIG to 1, to enable the external oscillator as a clock source.
4. Select the external oscillator as the AHB clock or as the input clock for a module.

19.5. External RC Oscillator

When using the EXTOSC module with an external RC network, firmware should set the OSCMD field to 4 for RC divided by 2 mode. The RC network should be added to XTAL2, and XTAL2 should be configured for analog mode with the digital output drivers disabled. XTAL1 is not affected in RC mode. Figure 19.3 shows this hardware configuration.

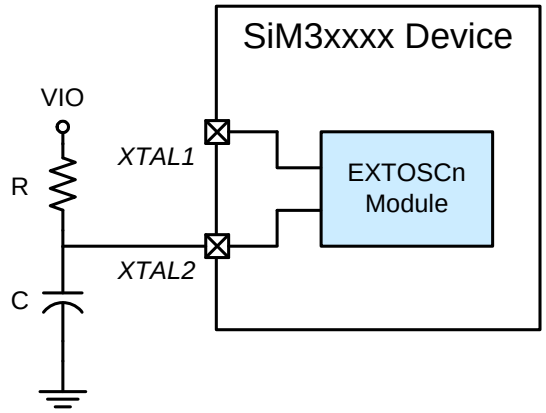


Figure 19.3. External RC Oscillator Configuration

The capacitor used in the RC network should have a value no greater than 100 pF, and the resistor should be no smaller than 10 kΩ. For very small capacitors, the parasitic capacitance in the PCB layout may dominate the total capacitance. The oscillation frequency can be determined by Equation 19.3, where F is the frequency in MHz, R is the pull-up resistor value in kΩ., and C is the capacitor value in the XTAL2 pin in pF.

$$F = \frac{1.23 \times 10^3}{R \times C}$$

Equation 19.3. RC Oscillation Frequency

To determine the required frequency control (FREQCN), first select the RC network value to produce the desired frequency of oscillation. For example, if the desired frequency is 100 kHz, let R = 246 kΩ. and C = 50 pF.

$$F = \frac{1.23 \times 10^3}{R \times C} = \frac{1.23 \times 10^3}{246 \times 50} = 100\ \text{kHz}$$

Table 19.2 shows the frequency ranges for the frequency control (FREQCN) bit in RC oscillator mode. The recommended FREQCN setting for this example is 2.

The RC oscillator mode is only available with a divide by 2 stage, which ensures that the clock derived from the external oscillator has a duty cycle of 50%. The equation for the EXTOSC output frequency is shown in Equation 19.4.

$$F_{OUT} = \frac{F}{2} = \frac{1.23 \times 10^3}{2 \times R \times C}$$

Equation 19.4. EXTOSC Output Frequency in RC Mode

Table 19.2. Frequency Control for RC Oscillators

RC Oscillator Frequency	EXTOSC Output Frequency (f_{OUT})	FREQCN Value
f ≤ 25 kHz	f_{OUT} ≤ 12.5 kHz	0
25 kHz < f ≤ 50 kHz	12.5 kHz < f_{OUT} ≤ 25 kHz	1
50 kHz < f ≤ 100 kHz	25 kHz < f_{OUT} ≤ 50 kHz	2
100 kHz < f ≤ 200 kHz	50 kHz < f_{OUT} ≤ 100 kHz	3
200 kHz < f ≤ 400 kHz	100 kHz < f_{OUT} ≤ 200 kHz	4
400 kHz < f ≤ 800 kHz	200 kHz < f_{OUT} ≤ 400 kHz	5
800 kHz < f ≤ 1.6 MHz	400 kHz < f_{OUT} ≤ 800 kHz	6
1.6 MHz < f ≤ 3.2 MHz	800 kHz < f_{OUT} ≤ 1.6 MHz	7

19.5.1. Configuring for RC Oscillator Mode

The recommended procedure for enabling the RC oscillator is:

1. Configure the XTAL2 pin for analog mode using the device port configuration module.
2. Disable the XTAL2 digital output driver by writing 1 to the pin latch in the Port Bank registers.
3. Configure the FREQCN field for the RC frequency according to Table 19.2.
4. Set the OSCMD field to 4 for RC divided by 2 mode.
5. Wait at least 1 ms.
6. Poll on the OSCVLDF flag to determine if the oscillator is running and stable.
7. Set the EXTOSCEN bit in CLKCTRL0_CONFIG to 1, to enable the external oscillator as a clock source.
8. Select the external oscillator as the AHB clock or as the input clock for a module.

19.6. External C Oscillator

Firmware can enable the external C oscillator by setting OSCMD to 5 for C oscillator divided by 2 mode. The capacitor used should be no greater than 100 pF, and the parasitic capacitance in the PCB layout may dominate very small capacitors. The hardware configuration for the external C oscillator is shown in Figure 19.4.

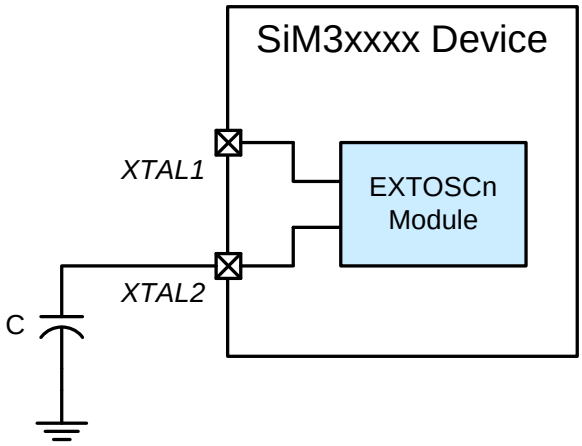


Figure 19.4. External C Oscillator Configuration

To determine the required frequency control (FREQCN) value (XFCN), select the capacitor to be used and find the frequency of oscillation according to Equation 19.5, where F is the frequency of oscillation in MHz, C is the capacitor value in pF, VBAT is the device power supply in Volts, and K is the K Factor.

$$F = \frac{K}{C \times V_{BAT}}$$

Equation 19.5. C Oscillation Frequency

For example, assume VBAT is 3.0 V and the desired C oscillator frequency is 150 kHz. Since the frequency desired is roughly 150 kHz, select the K Factor of 22 from Table 19.3.

$$0.150 = \frac{22}{C \times 3.0}$$

$$C = \frac{22}{0.150 \times 3.0} = 48.8 \text{ pF}$$

Therefore, the FREQCN value to use in this example is 3, and the external capacitor should be 50 pF.

The C oscillator mode is only available with a divide by 2 stage, which ensures that the clock derived from the external oscillator has a duty cycle of 50%. The equation for the EXTOSC output frequency is shown in Equation 19.6.

$$f_{OUT} = \frac{f}{2} = \frac{KF}{2 \times C \times V_{BAT}}$$

Equation 19.6. EXTOSC Output Frequency in C Mode

Table 19.3. Frequency Control for C Oscillators

C Oscillator Frequency	EXTOSC Output Frequency (f_{OUT})	K Factor (KF)	FREQCN Value
f ≤ 25 kHz	f_{OUT} ≤ 12.5 kHz	0.87	0
25 kHz < f ≤ 50 kHz	12.5 kHz < f_{OUT} ≤ 25 kHz	2.6	1
50 kHz < f ≤ 100 kHz	25 kHz < f_{OUT} ≤ 50 kHz	7.7	2
100 kHz < f ≤ 200 kHz	50 kHz < f_{OUT} ≤ 100 kHz	22	3
200 kHz < f ≤ 400 kHz	100 kHz < f_{OUT} ≤ 200 kHz	65	4
400 kHz < f ≤ 800 kHz	200 kHz < f_{OUT} ≤ 400 kHz	180	5
800 kHz < f ≤ 1.6 MHz	400 kHz < f_{OUT} ≤ 800 kHz	664	6
1.6 MHz < f ≤ 3.2 MHz	800 kHz < f_{OUT} ≤ 1.6 MHz	1590	7

19.6.1. Configuring for C Oscillator Mode

The recommended procedure for enabling the C oscillator is:

1. Configure the XTAL2 pin for analog mode using the device port configuration module.
2. Disable the XTAL2 digital output driver by writing 1 to the pin latch in the Port Bank registers.
3. Configure the FREQCN field for the C frequency and K Factor according to Table 19.3.
4. Set the OSCMD field to 5 for C oscillator divided by 2 mode.
5. Wait at least 1 ms.
6. Poll on the OSCVLDF flag to determine if the oscillator is running and stable.
7. Set the EXTOSCEN bit in CLKCTRL0_CONFIG to 1, to enable the external oscillator as a clock source.
8. Select the external oscillator as the AHB clock or as the input clock for a module.

19.7. EXTOSC0 Registers

This section contains the detailed register descriptions for EXTOSC0 registers.

Register 19.1. EXTOSC0_CONTROL: Oscillator Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved									OSCMD			OSCVLDF	FREQCN		
Type	R									RW			R	RW		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

EXTOSC0_CONTROL = 0x4003_C000

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 19.4. EXTOSC0_CONTROL Register Bit Descriptions

Bit	Name	Function
31:7	Reserved	Must write reset value.
6:4	OSCMD	**Oscillator Mode.** 000: External oscillator off. 001: Reserved. 010: External CMOS clock mode. 011: External CMOS with divide by 2 stage. 100: RC oscillator mode with divide by 2 stage. 101: C oscillator mode with divide by 2 stage. 110: Crystal oscillator mode. 111: Crystal oscillator mode with divide by 2 stage.
3	OSCVLDF	**Oscillator Valid Flag.** This bit indicates when the oscillator has stabilized after an initial startup condition. Hardware will clear the bit to 0 when the external oscillator is disabled, and set the bit to 1 once the oscillator is running properly. It will not clear automatically if oscillation fails. This bit is valid for all modes of operation except external CMOS clock and external CMOS clock divide by 2 modes, when OSCVLDF always reads back as 0. 0: The external oscillator is unused or not yet stable. 1: The external oscillator is running and stable.

Table 19.4. EXTOSC0_CONTROL Register Bit Descriptions

Bit	Name	Function
2:0	FREQCN	**Frequency Control.** 000: Set the external oscillator to range 0. 001: Set the external oscillator to range 1. 010: Set the external oscillator to range 2. 011: Set the external oscillator to range 3. 100: Set the external oscillator to range 4. 101: Set the external oscillator to range 5. 110: Set the external oscillator to range 6. 111: Set the external oscillator to range 7.

19.8. EXTOSC0 Register Memory Map

Table 19.5. EXTOSC0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EXTOSC0_CONTROL	0x4003_C000	ALL | SET | CLR	Reserved																									OSCMD			OSCVLDF	FREQCN		

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

20. External Regulator (EXTVREG0)

This section describes the External Regulator (EXTVREG) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the EXTVREG block, which is used by all device families covered in this document.

20.1. External Regulator Features

The External Regulator module has the following features:

- Interfaces with either an NPN or PNP external transistor that serves as the pass device for the high current regulator.
- Automatic current limiting.
- Automatic foldback limiting.
- Sources up to 1 A for use by external circuitry.
- Variable output voltage, adjustable in 100 mV steps.
- The External Regulator provides all the circuitry needed for a high-power regulator except the power transistor (NPN or PNP).

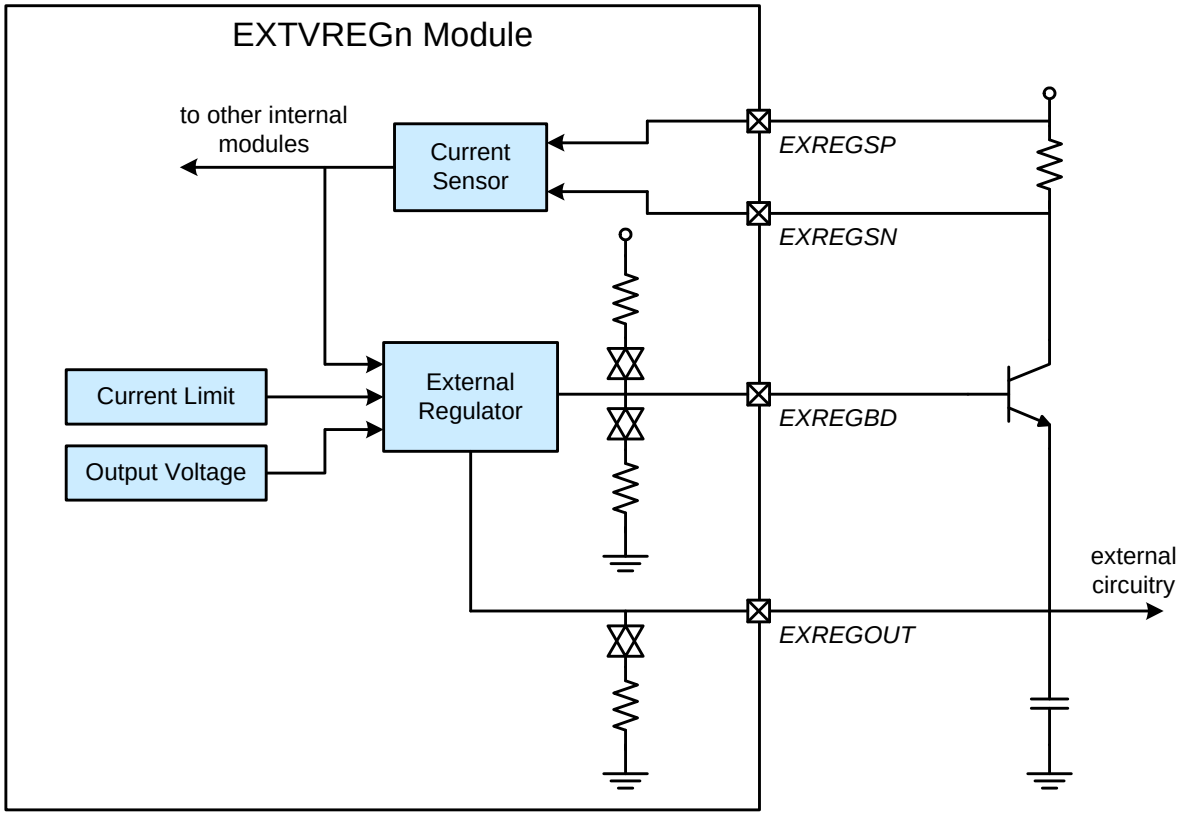


Figure 20.1. External Regulator Block Diagram

20.2. Overview

The External Voltage Regulator (EXTVREGn) module interfaces with an external transistor and decoupling capacitor to create a high current regulator. This external transistor can be either a PNP or an NPN bipolar transistor, and the output voltage is programmable in 100 mV steps. The external decoupling capacitor should be a minimum of 4.7 µF in normal operation and a minimum of 47 nF in stand-alone mode. In both cases, the capacitor should be placed as close to the external regulator pins as possible.

In addition, the EXTVREGn module supports current sensing and limiting, foldback limiting, and internal resistors to ensure regulator stability even with very small loads.

20.3. Operating Modes

In all operating modes, the external regulator is enabled by setting the EVREGEN bit to 1. Firmware can select the output voltage using the VOUTSEL field. The output voltage is equal to VOUTSEL + 1.8 V, where each setting of VOUTSEL represents 100 mV.

Any pins used with the external regulator should be placed in analog mode as described in the port configuration module to disable the general weak pull-ups on the pins.

20.3.1. Normal Mode

When stand-alone mode is disabled (SAEN = 0), the external regulator operates normally and the output of the external transistor must be connected to the EXREGOUT pin. In this mode, the regulator requires at least two pins: one to drive the base of the external bipolar (EXREGBD), and one to sense the output voltage (EXREGOUT).

The PNSEL bit configures whether the external regulator is in PNP or NPN mode. Figure 20.2 shows the external regulator in normal NPN mode, and Figure 20.3 shows the external regulator in normal PNP mode.

The decoupling capacitor on the output of the regulator must be a minimum of 4.7 µF in this mode and placed as close to the output of the bipolar transistor as possible.

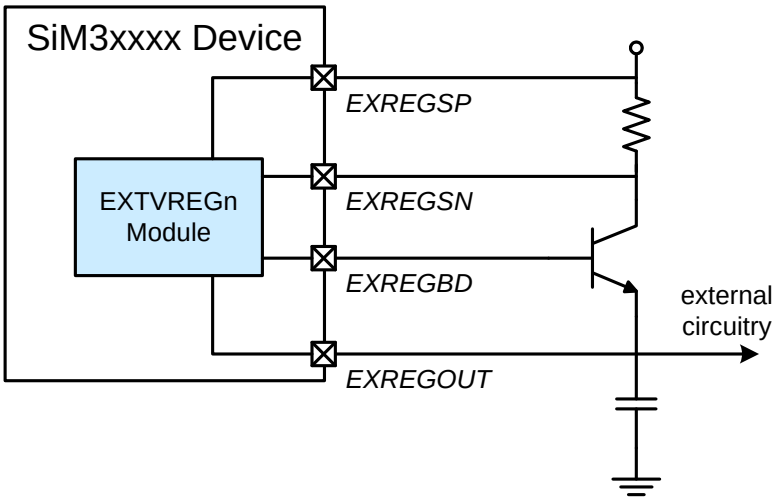


Figure 20.2. External Regulator Normal Mode NPN Configuration

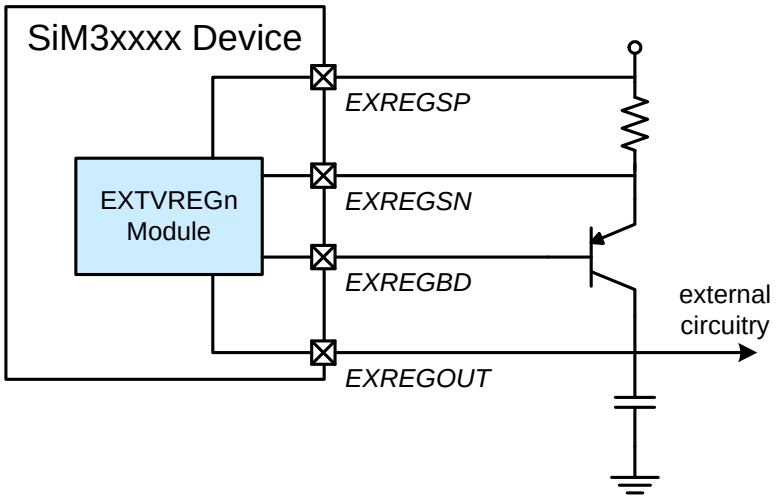


Figure 20.3. External Regulator Normal Mode PNP Configuration

20.3.2. Stand-Alone Mode

When SAEN is set to 1, the regulator transistor base driver (EXREGBD) is the only pin used by the regulator. This allows the regulator to operate in standalone mode, where an external transistor is not used, and the module operates as a voltage regulator with a maximum current output of approximately 12 mA. The supply to the VREGIN pin is used as the supply for the regulator in this mode, and the remaining regulator pins can be used for other functions.

The decoupling capacitor on the output of the regulator must be a minimum of 47 nF in this mode and placed as close to the EXREGBD pin as possible.

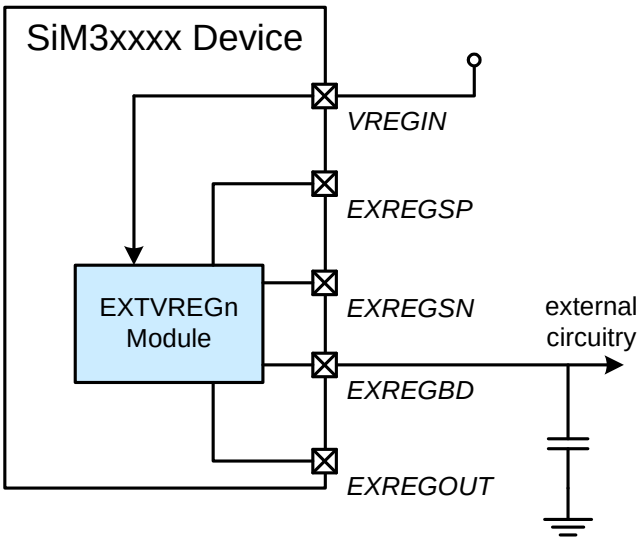


Figure 20.4. External Regulator in Standalone Mode

20.4. Current Sensing

The external regulator includes a current sensing feature that is enabled by placing a resistor in the current path of the external bipolar transistor. When the current limiting feature described in “20.5. Current Limiting” is not in use, the value of the resistor is not restricted, and firmware should set the IMAX field to the maximum value.

Current sensing has two outputs: one for the external regulator for use in voltage regulation and load control, and one as a measurable input for other modules in the device. The current sensing feature is enabled only if one of these outputs is enabled. Note that the current sensing feature relies on the VREG0 bandgap circuit to operate. It is necessary to clear the BGDIS bit in VREG0_CONTROL to 0 before using current sensing. A detailed diagram of the current sensing circuitry is shown in Figure 20.5.

The EXTVREGn module supports a variety of current sensing input configurations.

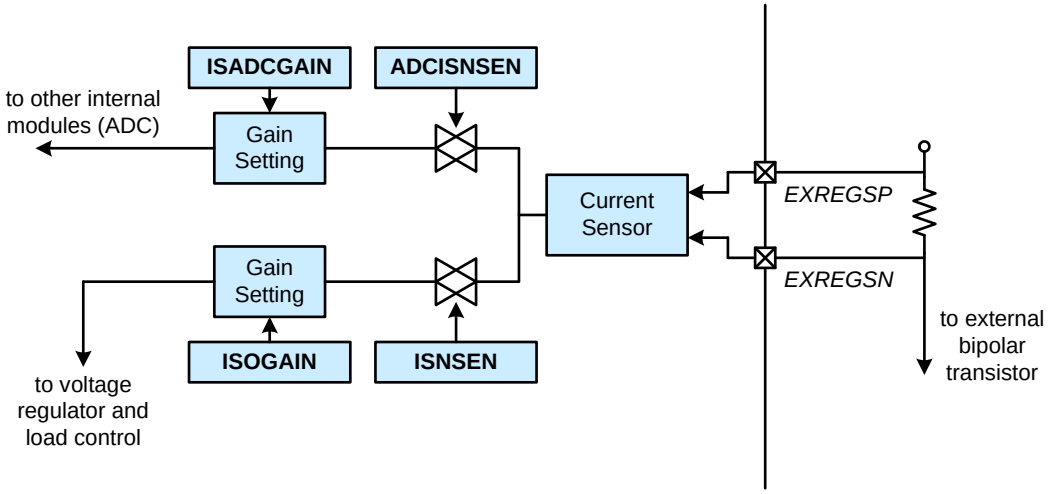


Figure 20.5. Current Sensing

20.4.1. Current Sensing Configurations

The EXTVREGn module supports four different current sensing input configurations selected by the ISINSEL field.

Current sensing mode 0 (ISINSEL = 0) is the standard configuration, using all four of the external regulator pins and providing the most accuracy of the available modes. This mode is shown in Figure 20.6.

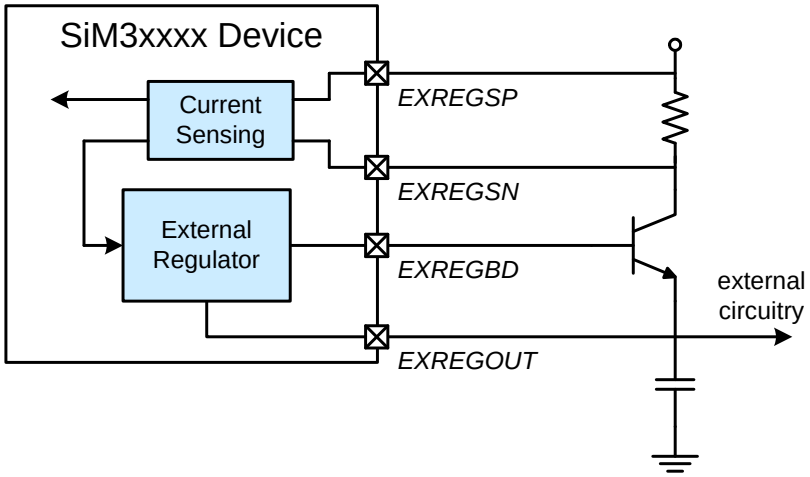


Figure 20.6. Current Sensing Input Configuration Mode 0

In input configuration mode 1 (ISINSEL = 1), the current sensing circuit measures on the low side of the bipolar transistor. In this mode, the external transistor can be connected to higher supply voltages that exceed the maximum ratings for the pins and still allow the regulator to sense the load current. The three pins used in this mode are EXREGBD, EXREGOUT, and EXREGSP. The fourth pin, EXREGSN, is unused, and the second current sensing circuit input is internally connected to the EXREGOUT signal. Figure 20.7 shows the mode 1 configuration.

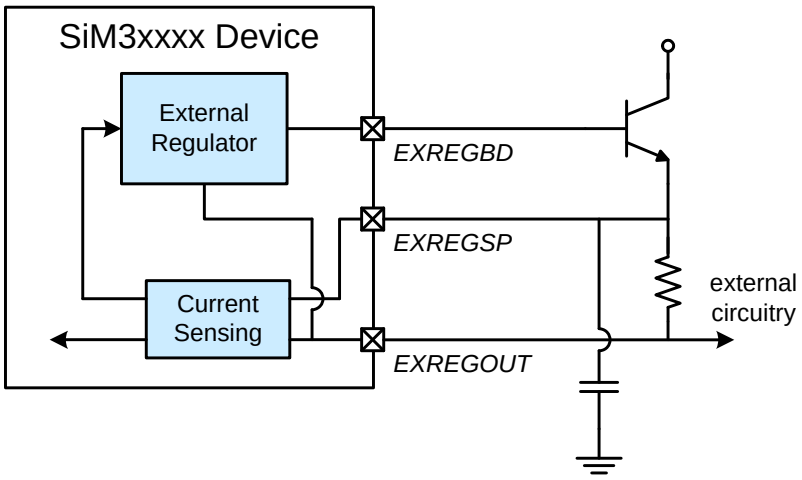


Figure 20.7. Current Sensing Input Configuration Mode 1

Current sensing input mode 2 (ISINSEL = 2) uses the VREGIN pin voltage as one of the inputs to the current sensing circuitry along with the EXREGSN input. This mode requires that the source connected to the external bipolar transistor is the same supply used by the internal voltage regulator (VREGN) module and can be less accurate than input mode 0. The EXREGSP input is not used in this mode. Figure 20.8 shows the input configuration for mode 2.

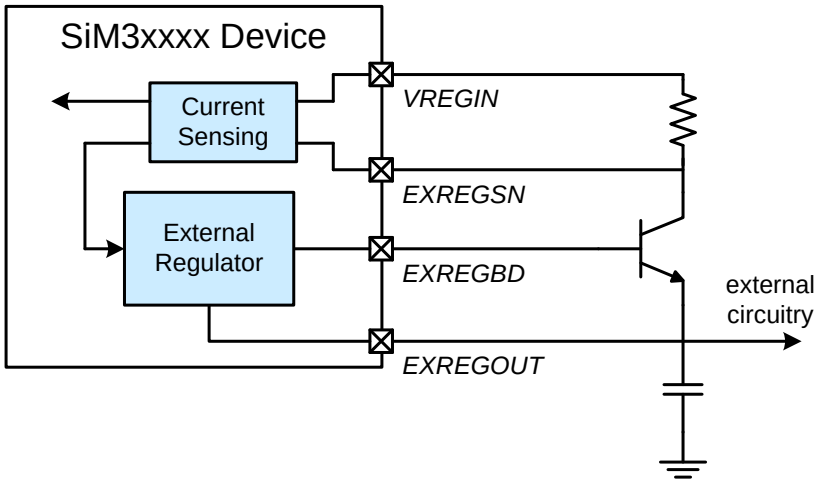


Figure 20.8. Current Sensing Input Configuration Mode 2

20.4.2. External Regulator Sensing

External regulator current sensing is enabled by setting the ISNSEN bit to 1. The regulator current sensing has a programmable gain stage set using the ISOGAIN field with five settings: 1x, 2x, 4x, 8x, and 16x. This gain should be selected based on the resistor value and the expected maximum current output of the regulator.

Equation 20.1 describes how to select the current sensing gain as a function of the maximum current and the sense resistor value. In this equation, I_{BJTMAX} is given in Amps, R_{sense} is given in ohms, and I_{MAX} is given in µA.

$$I_{BJTMAX} = \frac{I_{MAX}}{R_{sense} \times 12.5 \times Gain_{sense}}$$

Equation 20.1. External Regulator Current Sensing

The I_{MAX} value can be set by the IMAX field if using current limiting.

20.4.3. Other Internal Module Sensing (ADC)

The current sensing output for other modules is enabled by setting the ADCISNSEN bit to 1. This output is primarily intended to be measured by an ADC on the device (like the SARADCn module, if available), but can also be an input for other modules.

This output's gain setting is independent form the external regulator gain setting and is selected using the ISADCGAIN field with five settings: 1x, 2x, 4x, 8x, and 16x. If measuring the current sensing output using an ADC, the gain setting should be selected to maximize the ADC's dynamic range: when the external regulator reaches the current limit set by the IMAX field, the ADC should measure a value close to its selected voltage reference.

The current sensing feature of the EXTVREGn module can be used even if the external regulator is disabled (EVREGEN = 0).

20.5. Current Limiting

The current limiting feature of the external regulator utilizes the internal current sensing circuitry and limits the output current of the regulator to the value specified by the IMAX field. Firmware can enable current limiting by enabling current sensing and setting the IMAX field to a value less than or equal to the I_{MAX} value in Equation 20.1.

To use the current limiting feature of the external regulator, the sensing resistor in the current path of the external bipolar transistor must be less than or equal to 1 Ω. Resistor values larger than 1 Ω may cause instability in the regulator.

Figure 20.9 illustrates the current limiting feature of the EXTVREGn module.

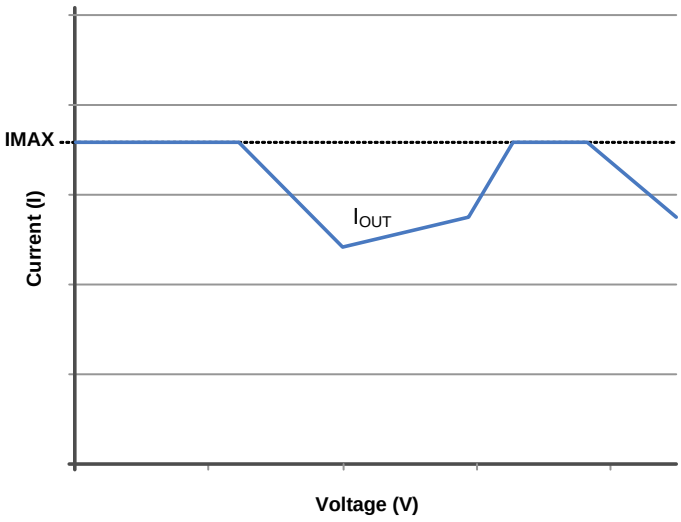


Figure 20.9. Regulator Current Limiting

20.6. Foldback Limiting

The foldback limiting feature is a protection mechanism that allows the EXTVREGn module to limit the output voltage and current to ranges that are within the tolerances of the external bipolar transistor. Bipolar transistors have two specifications: maximum current, and maximum power. Foldback limiting can approximate power limiting. The foldback limiting feature is fully programmable to allow the regulator to match the optimum characteristics of the transistor safe operating area or region.

The maximum current of the external regulator can be specified with the IMAX field. The starting point of the downward safe operating region slope is set by the FBVOSEL field, and the foldback rate or slope is adjusted by the FBRATE field. The transistors then level off to a maximum current at higher voltages. This limit is set in the regulator with the IMIN and IMINFINE fields. Figure 20.10 shows illustrates this voltage and current characteristic curve.

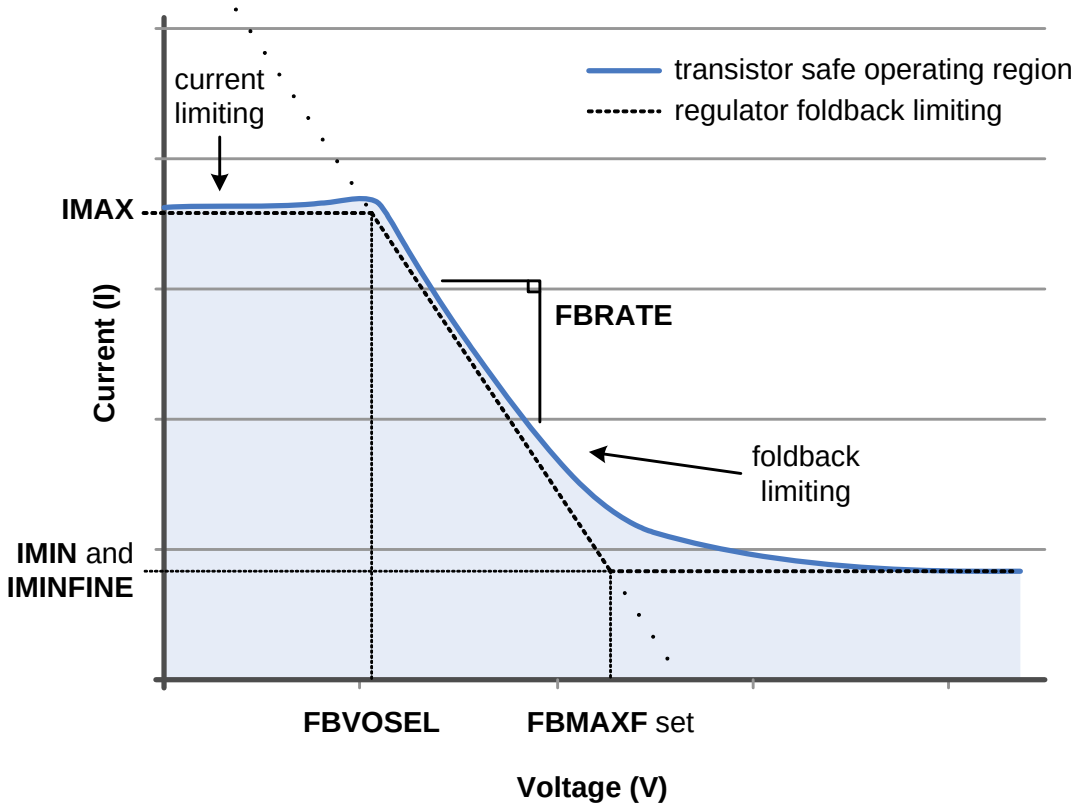


Figure 20.10. Transistor Characterization with Regulator Current and Foldback Limiting

The regulator monitors the voltage across the external transistor using the EXREGSN pin (FBPINSEL = 0) or the VREGIN pin (FBPINSEL = 1).

The FBVOSEL value should be selected using Equation 20.2.

$$FBVOSEL \leq V_{BJT\ Supply} - V_{OUT}$$

Equation 20.2. Selecting the Foldback Offset Voltage

The foldback rate (FBRATE) value can be found using the slope of the safe operating region (maximum power dissipation) of the transistor. Equation 20.3 describes this relationship, where P_{BJTMAX} is given in Watts, R_{sense} is given in ohms, and the FBRATE rate is µA/V.

$$FBRATE = \frac{P_{BJTMAX} \times R_{sense} \times 12.5 \times Gain_{sense}}{(V_{BJT\ Supply} - V_{OUT})^2}$$

Equation 20.3. External Regulator Foldback Rate

Foldback limiting can be used independently of the current sensing or limiting features of the regulator. If current limiting is not used, the IMAX field should be set to the maximum value, and the FBVOSEL field is ignored. The foldback limiting slope defined by FBRATE will then govern the regulator output current at voltages below FBVOSEL.

The lower current limit (IMIN and IMINFINE fields) prevents the regulator from dropping the output current to zero when the output voltage reaches higher levels. If the output of the regulator is connected to ground due to a failure in the external circuitry, the lower current limit determines how fast the output voltage can recover when the short-circuit condition is no longer present. The maximum lower current limit can be found using Equation 20.4.

$$IMIN + IMINFINE \geq I_{MAX} - \left[\left(\frac{P_{BJTMAX}}{V_{BJT\ Supply}}\right) \times R_{sense} \times 12.5 \times Gain_{sense}\right]$$

Equation 20.4. External Regulator Minimum Current

The P_{BJTMAX} divided by $V_{BJT\ Supply}$ term is the actual maximum current the external regulator circuit can allow. Hardware sets the FBMAXF flag to 1 when the regulator reaches maximum foldback at the output current defined by IMIN and IMINFINE.

20.7. Regulator Stability

To ensure regulator stability even if the external load is very small or goes to zero, the EXTVREGn module also has an optional pull-up or pull-down on the EXREGBD pin and an optional pull-down on the EXREGOUT pin. When enabled (WPULLEN = 1), the EXREGBD resistor is connected to ground when in NPN mode (PNSEL = 0) and to the VREGIN pin supply if the regulator is in PNP mode (PNSEL = 1). The EXREGOUT pull-down resistor is also enabled by setting WPULLEN to 1. These resistors can also be used to turn off the bipolar when the regulator is disabled.

These weak pull-up or pull-down resistors are completely independent of the normal weak pull-ups associated with the pins as described in the port configuration section. Any pins used with the external regulator should be placed in analog mode as described in the port configuration module to disable the general weak pull-ups on the pins.

20.8. Configuring the External Regulator

To configure the external regulator:

1. Configure the pins to be used by the external regulator to analog mode.
2. If reset persistence is required, the PBCFG_CONTROL1.EVREGMD bit should be set to 1. This bit will force the regulator pins to retain their state after any reset except for POR.
3. Select the hardware components.
   a. Select the bipolar transistor, typically an NPN device (PNSEL = 0).
   b. Select the decoupling capacitor (minimum of 4.7 µF in normal mode and 47 nF in stand-alone mode).
   c. (Optional) Select the sense resistor (less than or equal to 1 Ω when using current limiting) and current sensing configuration.
4. Select the EXTVREGn output voltage using the VOUTSEL field.
5. (Optional) Configure regulator current sensing.
   a. Set the current sensing gain (ISOGAIN) to an appropriate value for the sense resistor value and the maximum current output of the regulator.
   b. Enable regulator current sensing (ISNSEN = 1).
6. (Optional) Configure current sensing for other modules.
   a. Select the sensing gain (ISADCGAIN). If measured by an ADC, the sensing gain should be set based on the sense resistor value and the reference for the ADC.
   b. Enable the current sensing output for other modules (ADCISNSEN = 1).
7. (Optional) Configure regulator current limiting (regulator current sensing must be enabled).
   a. Set the IMAX field to a value below the maximum current rating for the transistor.
8. (Optional) Configure the foldback limiting.
   a. Set the FBVOSEL, FBRATE, IMIN, and IMINFINE fields to match the bipolar transistor characteristics.
   b. Select the desired foldback limiting sense pin (FBPINSEL).
   c. Enable foldback limiting (FBLEN = 1).
9. Enable the regulator (EVREGNEN = 1).
10. Hardware will set the FBMAXF flag if the external regulator reaches maximum foldback.

20.9. EXTVREG0 Registers

This section contains the detailed register descriptions for EXTVREG0 registers.

Register 20.1. EXTVREG0_CONTROL: Module Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	EVREGEN	Reserved			FBPINSEL	PNSEL	FBLEN	WPULLEN	Reserved							
Type	RW	R			RW	RW	RW	RW	R							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved															SAEN
Type	R															RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

EXTVREG0_CONTROL = 0x4004_2000

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 20.1. EXTVREG0_CONTROL Register Bit Descriptions

Bit	Name	Function
31	EVREGEN	**External Regulator Enable.** 0: Disable the external regulator. 1: Enable the external regulator.
30:28	Reserved	Must write reset value.
27	FBPINSEL	**Foldback Sensing Pin Select.** 0: Use the input to the EXREGSN pin for foldback limiting. 1: Use the input to the VREGIN pin for foldback limiting.
26	PNSEL	**NPN/PNP Type Select.** 0: Select NPN Mode. 1: Select PNP Mode.
25	FBLEN	**Foldback Limiting Enable.** 0: Disable foldback limiting. 1: Enable foldback limiting.

Table 20.1. EXTVREG0_CONTROL Register Bit Descriptions

Bit	Name	Function
24	WPULLEN	**Weak Pull Up/Down Enable.** This bit enables or disables a weak pull-up or -down on the EXREGBD pin. The resistor is connected to ground when in NPN mode (PNSEL = 0) and it is connected to the VREGIN supply if the regulator is in PNP mode (PNSEL = 1). A pull-down resistor connected to the EXREGOUT pin is also turned on if WPULLEN is set to 1 and the external regulator is enabled. This allows for a minimum load that will ensure regulator stability even if the external load is very small or goes to zero. These weak pull-up or pull-down resistors are completely independent of the normal weak pull-ups associated with the pins, and a pin used with the external regulator should be placed in analog mode to disable the general weak pull-up. 0: Disable the external regulator weak pull-up/down resistor on the EXREGBD pin and weak pull-down resistor on the EXREGOUT pin. 1: Enable the external regulator weak pull-up/down resistor on the EXREGBD pin and weak pull-down resistor on the EXREGOUT pin.
23:1	Reserved	Must write reset value.
0	SAEN	**Stand-Alone Mode Enable.** When SAEN is cleared to 0, the external regulator operates normally and the output of the external transistor must be connected to the EXREGOUT pin. In this mode, the regulator requires 2 pins: one to drive the base of the external bipolar (EXREGBD), and one to sense the output voltage (EXREGOUT). When SAEN is set to 1, the voltage supply of the regulator is the VREGIN supply and the transistor base driver (EXREGBD) is the regulator output. This allows the regulator to operate in stand-alone mode, where an external transistor is not used. The maximum current output of the regulator is 12.5 mA in stand-alone mode. A decoupling capacitor is required on the regulator output in both normal and stand-alone modes. 0: Use the external regulator in normal mode. 1: Use the external regulator in stand-alone mode.

Register 20.2. EXTVREG0_CONFIG: Module Configuration

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved		VOUTSEL						Reserved					IMAX		
Type	R		RW						R					RW		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved	FBRATE			Reserved	FBVOSEL			Reserved			IMIN			IMINFINE	
Type	R	RW			R	RW			R			RW			RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

EXTVREG0_CONFIG = 0x4004_2010

Table 20.2. EXTVREG0_CONFIG Register Bit Descriptions

Bit	Name	Function
31:30	Reserved	Must write reset value.
29:24	VOUTSEL	**Regulator Output Voltage Select.** The external regulator output voltage is VOUTSEL + 1.8 V, where V_{OUT} is given in 100 mV steps. For example, a VOUTSEL setting of 0 is equivalent to a 1.8 V output voltage and a setting of 12 is equivalent to a 3 V output voltage.
23:19	Reserved	Must write reset value.
18:16	IMAX	**Maximum Current Select.** Regulator current sensing must be enabled (ISNSEN = 1) for the regulator to limit the output current to the value specified by this field. 000: Maximum current limit is 2 µA. 001: Maximum current limit is 3 µA. 010: Maximum current limit is 4 µA. 011: Maximum current limit is 5 µA. 100: Maximum current limit is 6 µA. 101: Maximum current limit is 7 µA. 110: Maximum current limit is 8 µA. 111: Maximum current limit is 9 µA.
15	Reserved	Must write reset value.

Table 20.2. EXTVREG0_CONFIG Register Bit Descriptions

Bit	Name	Function
14:12	FBRATE	**Voltage Sense Gain Multiplier.** 000: Set the foldback rate to 4 μA/V. 001: Set the foldback rate to 2 μA/V. 010: Set the foldback rate to 1 μA/V. 011: Set the foldback rate to 0.5 μA/V. 100: Reserved. 101: Set the foldback rate to 8 μA/V. 110: Set the foldback rate to 16 μA/V. 111: Set the foldback rate to 32 μA/V.
11	Reserved	Must write reset value.
10:8	FBVOSEL	**Foldback Voltage Offset Select.** 000: Foldback voltage offset is 0 V. 001: Foldback voltage offset is 0.5 V. 010: Foldback voltage offset is 1 V. 011: Foldback voltage offset is 1.5 V. 100: Foldback voltage offset is 2 V. 101: Foldback voltage offset is 2.5 V. 110: Foldback voltage offset is 3 V. 111: Foldback voltage offset is 3.5 V.
7:5	Reserved	Must write reset value.
4:2	IMIN	**Minimum Current Select.** 000: Minimum current limit is 1 μA + IMINFINE current. 001: Minimum current limit is 2 μA + IMINFINE current. 010: Minimum current limit is 3 μA + IMINFINE current. 011: Minimum current limit is 4 μA + IMINFINE current. 100: Minimum current limit is 5 μA + IMINFINE current. 101: Minimum current limit is 6 μA + IMINFINE current. 110: Minimum current limit is 7 μA + IMINFINE current. 111: Minimum current limit is 8 μA + IMINFINE current.
1:0	IMINFINE	**Minimum Current Fine Select.** 00: Minimum current limit is IMIN current + 0 μA. 01: Minimum current limit is IMIN current + 0.25 μA. 10: Minimum current limit is IMIN current + 0.50 μA. 11: Minimum current limit is IMIN current + 0.75 μA.

Register 20.3. EXTVREG0_STATUS: Module Status

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved															FBMAXF
Type	R															R
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address
EXTVREG0_STATUS = 0x4004_2020

Table 20.3. EXTVREG0_STATUS Register Bit Descriptions

Bit	Name	Function
31:1	Reserved	Must write reset value.
0	FBMAXF	**Maximum Foldback Flag.** 0: Maximum foldback has not been reached. 1: Maximum foldback has been reached.

Register 20.4. EXTVREG0_CSCONTROL: Current Sense Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	ADCISNSEN	ISNSEN	Reserved													
Type	RW	RW	R													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

EXTVREG0_CSCONTROL = 0x4004_2040

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 20.4. EXTVREG0_CSCONTROL Register Bit Descriptions

Bit	Name	Function
31	ADCISNSEN	**ADC Current Sense Enable.** 0: Disable ADC current sensing. 1: Enable ADC current sensing.
30	ISNSEN	**External Regulator Current Sense Enable.** 0: Disable external regulator current sensing. 1: Enable external regulator current sensing.
29:0	Reserved	Must write reset value.

Register 20.5. EXTVREG0_CSCONFIG: Current Sense Configuration

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved								ISINSEL		ISOGAIN			ISADCGAIN		
Type	R								RW		RW			RW		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address
EXTVREG0_CSCONFIG = 0x4004_2050

Table 20.5. EXTVREG0_CSCONFIG Register Bit Descriptions

Bit	Name	Function
31:8	Reserved	Must write reset value.
7:6	ISINSEL	**External Regulator Current Sense Input Select.** 00: Select external regulator current sensing mode 0. 01: Select external regulator current sensing mode 1. 10: Select external regulator current sensing mode 2. 11: Reserved.
5:3	ISOGAIN	**External Regulator Current Sense Gain.** 000: External regulator current sensing gain is 16. 001: External regulator current sensing gain is 8. 010: External regulator current sensing gain is 4. 011: External regulator current sensing gain is 2. 100: External regulator current sensing gain is 1. 101-111: Reserved.
2:0	ISADCGAIN	**ADC Current Sense Gain.** 000: ADC current sensing input gain is 16. 001: ADC current sensing input gain is 8. 010: ADC current sensing input gain is 4. 011: ADC current sensing input gain is 2. 100: ADC current sensing input gain is 1. 101-111: Reserved.

20.10. EXTVREG0 Register Memory Map

Table 20.6. EXTVREG0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EXTVREG0_CONTROL	0x4004_2000	ALL | SET | CLR	EVREGEN	Reserved			FBPINSEL	PNSEL	FBLEN	WPULLEN	Reserved																							SAEN
EXTVREG0_CONFIG	0x4004_2010	ALL	Reserved	VOUTSEL							Reserved				IMAX				Reserved	FBRATE			Reserved	FBVOSEL			Reserved			IMIN			IMINFINE	
EXTVREG0_STATUS	0x4004_2020	ALL	Reserved																															FBMAXF

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

Table 20.6. EXTVREG0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EXTVREG0_CSCONTROL	0x4004_2040	ALL | SET | CLR	ADCISNSEN	ISNSEN	Reserved																													
EXTVREG0_CSCONFIG	0x4004_2050	ALL	Reserved																								ISINSEL		ISOGAIN			ISADCGAIN		

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

21. Flash Controller (FLASHCTRL0)

This section describes the flash Controller (FLASHCTRL) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the FLASHCTRL block, which is used by all device families covered in this document.

21.1. Flash Controller Features

The flash Controller module includes the following features:

- Provides control for read timing and prefetch.
- Provides an access port for firmware to write and erase flash in system.
- Buffers multiple writes to the flash and stalls the core if the buffer is full until the flash operations can complete.
- Secures the flash with a lock and key mechanism.
- Allows single writes and erases or multiple writes with a single unlock operation.
- Blocks modifications to flash if the Supply Monitor is not enabled as a reset source.

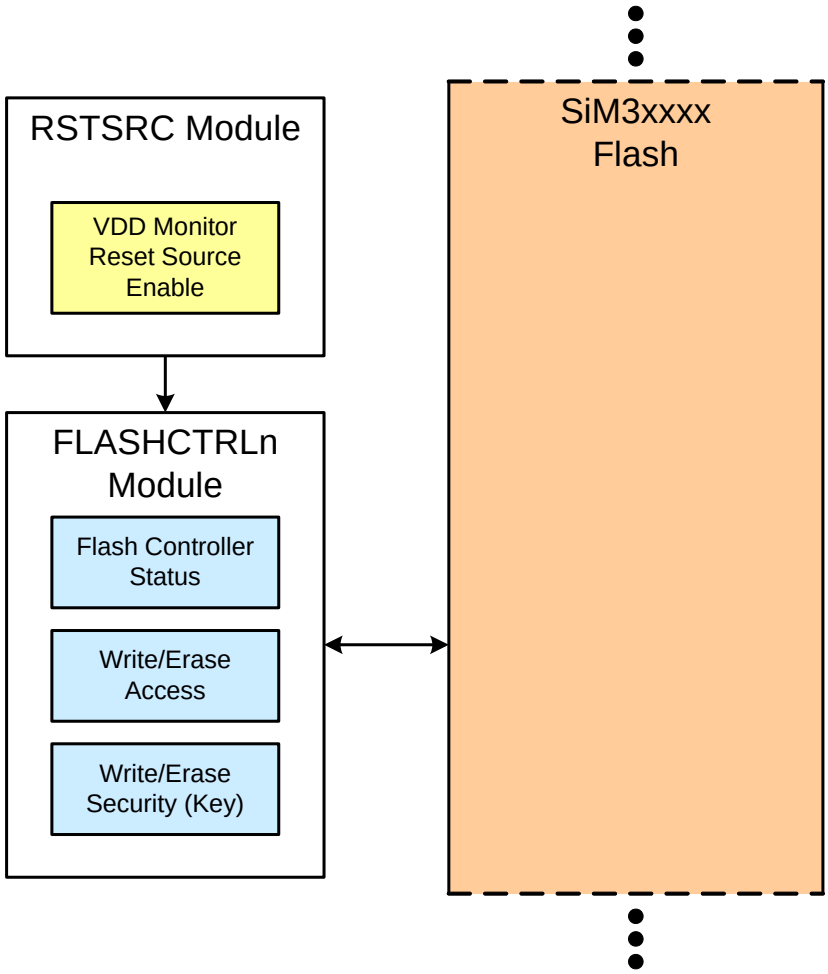


Figure 21.1. Flash Controller Block Diagram

21.2. Overview

The Flash Controller (FLASHCTRL) module provides flash read, write, and erase control. The read controller includes the prefetch engine controls and the flash read timing settings that must be adjusted appropriately for the AHB clock of the device. The write and erase control can be used to modify the flash contents in system. This write and erase interface is protected by a lock and key mechanism to prevent any undesired modification of flash.

21.3. Flash Read Control

The flash read timing is controlled by the speed mode (SPMD) field and the read-store enable (RDSEN) bit. These bits must be set appropriately for the selected AHB frequency for the system.

When read-store enable (RDSEN) is set to 1, the first flash access is stored in the read store pipeline buffer before being passed to the AHB. Otherwise, the first flash access is passed directly to the AHB (read through).

Table 21.1. Read Timing Ranges

SPMD Wait Value	RDSEN Value	Setting	Max AHB Frequency
0	0	Non-pipelined, Latency = 0	21 MHz
0	1	Pipelined, Latency = 0	26 MHz
1	0	Non-pipelined, Latency = 1	43 MHz
1	1	Pipelined, Latency = 1	53 MHz*
2	0	Non-pipelined, Latency = 2	65 MHz*
2	1	Pipelined, Latency = 2	80 MHz*
3	0	Non-pipelined, Latency = 3	87 MHz*
3	1	Pipelined, Latency = 3	107 MHz*

***Note:** Device operation beyond the maximum frequency listed in the data sheet is not guaranteed, and should be avoided.

The flash read timing mode (FLRTMD) can also be configured to save power at slower clock frequencies. It should be set to 1 for AHB clock frequencies above 12 MHz and cleared to 0 for AHB clock frequencies below 12 MHz. The FLRTMD bit should only be set to 1 when RDSEN is disabled and the SPMD Wait value is zero. FLRTMD has no effect on the flash read timing.

The flash read controller also includes prefetch engine controls. Enabling the data prefetch (DPFEN = 1) feature allows data accesses to be stored and queued into the prefetch buffer in addition to instructions, which can help performance if a large amount of data is accessed sequentially in flash. Firmware can also disable the prefetch engine (PFINH = 1), which will reduce performance but improve the device's power consumption.

21.4. Flash Write and Erase Control

The flash write and erase interface allows firmware to modify the flash in system. This interface is protected by a security lock and key interface to prevent inadvertent modifications of memory.

Firmware cannot modify flash through the interface when the supply monitor in the VMONn module is disabled or disabled as a reset source (device reset source module). Any write or erase operations initiated while the supply monitor is disabled or disabled as a reset source will be ignored.

Firmware should disable interrupts when using the interface to modify flash contents. This will ensure the flash interface accesses are sequential in time and take the minimum time possible.

21.4.1. Security Interface

The flash interface is initially locked after a reset. The interface is unlocked by writing the initial unlock key (0xA5) followed by one of the command keys in consecutive writes to the KEY field. Any writes to the WRDATA register while the interface is locked or an incorrect unlock sequence will permanently lock the flash interface until the next device reset.

The single unlock key unlocks the flash interface for one write or erase operation. The flash interface will remain unlocked after a single unlock command if firmware writes an additional value to the KEY field before writing to the WRDATA register.

The multiple unlock key unlocks the flash interface for write or erase operations until the multiple lock key is written to the KEY field. The flash interface will remain unlocked if firmware writes any value other than the multiple write lock to KEY. The multiple lock is not a permanent lock, and the interface can be unlocked again for other operations.

Table 21.2. Flash Interface Keys

Command	KEY Value
Initial Unlock	0xA5
Single Unlock	0xF1
Multiple Unlock	0xF2
Multiple Lock	0x5A

21.4.2. Writing and Erasing Flash

Once the flash security interface is unlocked, write and erase operations occur using the write address (WRADDR) and write data (WRDATA) indirect registers. Writes to WRDATA must be half-word aligned, and each half word may only be written once between erase operations. For a write operation (ERASEEN = 0), the right-justified, half-word value written to WRDATA will be written to the address specified by WRADDR. For an erase operation (ERASEEN = 1), a write to WRDATA will initiate an erase on the flash page specified by the WRADDR field. Flash pages are 1024 bytes, and aligned at 1024-byte boundaries in the device, beginning at address 0x0000.

Firmware should write the data to the WRDATA register 16 bits at a time in right-justified format, and hardware automatically increments the WRADDR field by two after each write operation. The data written to the WRDATA register is first placed in the controller write buffer. When writing multiple bytes and executing from RAM, firmware can poll on the BUFSTS flag to wait until the buffer has room before writing more data to the WRDATA register. This allows firmware to perform other actions while the controller is modifying flash. If the buffer is full and firmware writes another half-word to WRDATA, the flash controller will stall the AHB bus until the write operation completes, and the buffer is no longer full. Using this method, firmware can write to WRDATA in a series of successive writes without having to poll.

Note: For all flash write operations, firmware will stall unless operating from a memory space other than flash.

The flash controller can write multiple sequential half-words to flash faster than using individual accesses if the flash interface is unlocked for multiple byte writes and sequential writes are enabled (SQWEN = 1). Firmware using this feature should run from a memory space other than flash. Otherwise, the flash controller switches out of sequential mode for the flash read and back into sequential mode for the write, which causes a longer delay than individual accesses with SQWEN cleared to 0.

The busy (BUSYF) flag indicates when the flash controller is currently executing a flash write or erase operation.

21.4.3. Writing a Single Half-Word to Flash

To write a single byte to flash:

1. Ensure the supply monitor is enabled and enabled as a reset source in the RSTSRC module.
2. Disable erase operations (ERASEEN = 0).
3. Write the destination address to WRADDR.
4. Disable interrupts.
5. Write the initial unlock value to KEY (0xA5).
6. Write the single unlock value to KEY (0xF1).
7. Write the data half-word to WRDATA in right-justified format.
8. (Optional) If executing code from a memory space other than flash, poll on the BUSYF flag until hardware clears it to 0.
9. Enable interrupts.

21.4.4. Writing Multiple Half-Words to Sequential Flash Addresses

To write a sequential set of bytes to flash, code should execute from a memory space other than flash and complete the following steps:

1. Ensure the supply monitor is enabled and enabled as a reset source in the RSTSRC module.
2. Disable erase operations (ERASEEN = 0).
3. Write the initial destination address to WRADDR.
4. Enable sequential writes (SQWEN = 1).
5. Disable interrupts.
6. Write the initial unlock value to KEY (0xA5).
7. Write the multiple unlock value to KEY (0xF2).
8. Write the data half-word to WRDATA in right-justified format.
9. (Optional) Poll on the BUFSTS flag until the buffer has room for more data. If code is executing from RAM, this allows the core to perform other actions until a write operation completes and the buffer has room. The AHB bus will automatically stall until the operation completes if firmware writes data to WRDATA when the buffer is full.
10. Repeat steps 8 and 9 until all data is written. Hardware automatically increments the WRADDR field by 2 after each write operation.
11. (Optional) If executing code from a memory space other than flash, poll on the BUSYF flag until hardware clears it to 0.
12. Write the multiple lock value to KEY (0x5A).
13. Enable interrupts.

21.4.5. Writing Multiple Half-Words to Non-Sequential Flash Addresses

To write multiple bytes to non-sequential addresses in flash:

1. Ensure the supply monitor is enabled and enabled as a reset source (device reset sources module).
2. Disable erase operations (ERASEEN = 0).
3. Disable interrupts.
4. Write the initial unlock value to KEY (0xA5).

5. Write the multiple unlock value to KEY (0xF2).
6. Write the destination address to WRADDR.
7. Write the data half-word to WRDATA in right-justified format.
8. (Optional) If executing code from a memory space other than flash, poll on the BUSYF flag until hardware clears it to 0.
9. Repeat steps 6, 7, and 8 until all data is written.
10. Write the multiple lock value to KEY (0x5A).
11. Enable interrupts.

21.4.6. Erasing a Page of Flash

To erase a page of flash:

1. Ensure the supply monitor is enabled and enabled as a reset source (device reset sources module).
2. Write the address of a byte in the flash page to WRADDR.
3. Enable erase operations (ERASEEN = 1).
4. Disable interrupts.
5. Write the initial unlock value to KEY (0xA5).
6. Write the single unlock value to KEY (0xF1).
7. Write any value to WRDATA in right-justified format to initiate the page erase.
8. (Optional) If executing code from a memory space other than flash, poll on the BUSYF flag until hardware clears it to 0.
9. Enable interrupts.

21.4.7. Erasing Multiple Flash Pages

To erase multiple pages of flash:

1. Ensure the supply monitor is enabled and enabled as a reset source (device reset sources module).
2. Enable erase operations (ERASEEN = 1).
3. Disable interrupts.
4. Write the initial unlock value to KEY (0xA5).
5. Write the multiple unlock value to KEY (0xF2).
6. Write the address of a byte in the flash page to WRADDR.
7. Write any value to WRDATA in right-justified format to initiate the page erase.
8. (Optional) If executing code from a memory space other than flash, poll on the BUSYF flag until hardware clears it to 0.
9. Repeat steps 6, 7, and 8 for each page.
10. Write the multiple lock value to KEY (0x5A).
11. Enable interrupts.

21.5. FLASHCTRL0 Registers

This section contains the detailed register descriptions for FLASHCTRL0 registers.

Register 21.1. FLASHCTRL0_CONFIG: Controller Configuration

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved											BUSYF	BUFSTS	ERASEEN	Reserved	SQWEN
Type	R											R	R	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved								PFINH	DPFEN	Reserved	RDSEN	Reserved		SPMD	
Type	R					RW			RW	RW	RW	RW	R		RW	
Reset	0	0	0	0	0	1	1	1	0	0	1	0	0	0	0	0

Register ALL Access Address

FLASHCTRL0_CONFIG = 0x4002_E000

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 21.3. FLASHCTRL0_CONFIG Register Bit Descriptions

Bit	Name	Function
31:21	Reserved	Must write reset value.
20	BUSYF	**Flash Operation Busy Flag.** 0: The flash interface is not busy. 1: The flash interface is busy with an operation.
19	BUFSTS	**Flash Buffer Status.** 0: The flash controller write data buffer is empty. 1: The flash controller write data buffer is full.
18	ERASEEN	**Flash Page Erase Enable.** 0: Writes to the WRDATA field will initiate a write to flash at the address in the WRADDR field. 1: Writes to the WRDATA field will initiate an erase of the flash page containing the address in the WRADDR field.
17	Reserved	Must write reset value.

Table 21.3. FLASHCTRL0_CONFIG Register Bit Descriptions

Bit	Name	Function
16	SQWEN	**Flash Write Sequence Enable.** Setting this bit will cause multiple sequential accesses to the flash interface to take less time than individual accesses if the flash interface is unlocked for multiple byte writes. When this bit is used, firmware should be running from a memory space other than flash. 0: Disable sequential write mode. 1: Enable sequential write mode.
15:8	Reserved	Must write reset value.
7	PFINH	**Prefetch Inhibit.** Disabling the prefetch can cause slower performance and better power consumption. 0: Any reads from flash are prefetched until the prefetch buffer is full. 1: Inhibit the prefetch engine.
6	DPFEN	**Data Prefetch Enable.** Setting this bit allows data accesses to be stored and queued into the prefetch buffer, which can help performance if a large amount of data is accessed sequentially in flash. 0: Data accesses are excluded from the prefetch buffer. 1: Data accesses are included in the prefetch buffer.
5	Reserved	Must write reset value.
4	RDSEN	**Read Store Mode Enable.** When set to 1, the first flash access is stored in the prefetch (read store) before being passed to the AHB. Otherwise, the first flash access is passed directly to the AHB (read through). 0: Disable read store mode. 1: Enable read store mode.
3:2	Reserved	Must write reset value.
1:0	SPMD	**Flash Speed Mode.** The flash speed mode must be adjusted appropriately for the system AHB frequency. 00: Read and write the flash at speed mode 0. 01: Read and write the flash at speed mode 1. 10: Read and write the flash at speed mode 2. 11: Read and write the flash at speed mode 3.

Register 21.2. FLASHCTRL0_WRADDR: Flash Write Address

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	WRADDR[31:16]															
Type	RW															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	WRADDR[15:0]															
Type	RW															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Register ALL Access Address

FLASHCTRL0_WRADDR = 0x4002_E0A0

Table 21.4. FLASHCTRL0_WRADDR Register Bit Descriptions

Bit	Name	Function
31:0	WRADDR	**Flash Write Address.** The flash operation will occur at the address (write) or page (erase) specified by this field.

Register 21.3. FLASHCTRL0_WRDATA: Flash Write Data

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	WRDATA[31:16]															
Type	W															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	WRDATA[15:0]															
Type	W															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

FLASHCTRL0_WRDATA = 0x4002_E0B0

Table 21.5. FLASHCTRL0_WRDATA Register Bit Descriptions

Bit	Name	Function
31:0	WRDATA	**Flash Write Data.** When erases are enabled, a write to this field will initiate an erase of the flash page containing the address specified by WRADDR. When erases are disabled, a right-justified, half-word write to this field will write the value to the flash address specified by WRADDR. Any data written to the upper half of WRDATA is ignored.

Register 21.4. FLASHCTRL0_KEY: Flash Modification Key

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved								KEY							
Type	R								RW							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

FLASHCTRL0_KEY = 0x4002_E0C0

Table 21.6. FLASHCTRL0_KEY Register Bit Descriptions

Bit	Name	Function
31:8	Reserved	Must write reset value.
7:0	KEY	**Flash Key.** Writing the initial unlock key (0xA5) followed by the single unlock key (0xF1) to this field will unlock the flash interface for single write or erase operations. The interface will relock after the operation. Writing the initial unlock key (0xA5) followed by the multiple unlock key (0xF2) will unlock the flash interface for multiple write and erase operations. The interface will remain unlocked until the multiple lock key (0x5A) is written to KEY. All other values for the KEY field are reserved.

Register 21.5. FLASHCTRL0_TCONTROL: Flash Timing Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R														RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved									FLRTMD	Reserved					
Type	R									RW	RW					
Reset	0	0	0	0	0	0	0	0	0	1	0	1	1	1	0	0

Register ALL Access Address

FLASHCTRL0_TCONTROL = 0x4002_E0D0

Table 21.7. FLASHCTRL0_TCONTROL Register Bit Descriptions

Bit	Name	Function
31:7	Reserved	Must write reset value.
6	FLRTMD	**Flash Read Timing Mode.** 0: Configure the flash read controller for AHB clocks below 12 MHz. 1: Configure the flash read controller for AHB clocks above 12 MHz.
5:0	Reserved	Must write reset value.

21.6. FLASHCTRL0 Register Memory Map

Table 21.8. FLASHCTRL0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
FLASHCTRL0_CONFIG	0x4002_E000	ALL | SET | CLR	Reserved											BUSYF	BUFSTS	ERASEEN	Reserved	SQWEN	Reserved								PFINH	DPFEN	Reserved	RDSEN	Reserved		SPMD	
FLASHCTRL0_WRADDR	0x4002_E0A0	ALL	WRADDR																															
FLASHCTRL0_WRDATA	0x4002_E0B0	ALL	WRDATA																															

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

Table 21.8. FLASHCTRL0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
FLASHCTRL0_KEY	0x4002_E0C0	ALL	Reserved																								KEY							
FLASHCTRL0_TCONTROL	0x4002_E0D0	ALL	Reserved																									FLRTMD	Reserved					

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

22. Inter-Integrated Circuit Bus (I2C0 and I2C1)

This section describes the I2C module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the I2C block, which is used by I2C0 and I2C1 on all device families covered in this document.

22.1. I2C Features

The I2C module includes the following features:

- Standard (up to 100 kbps) and Fast (400 kbps) transfer speeds.
- Can operate down to APB clock divided by 32768 or up to APB clock divided by 8.
- Support for master, slave, and multi-master modes.
- Hardware synchronization and arbitration for multi-master mode.
- Clock low extending (clock stretching) to interface with faster masters.
- Hardware support for 7-bit slave and general call address recognition.
- Firmware support for 10-bit slave address decoding.
- Ability to disable all slave states.
- Programmable clock high and low period.
- Programmable data setup/hold times.
- Spike suppression up to 2 times the APB period.

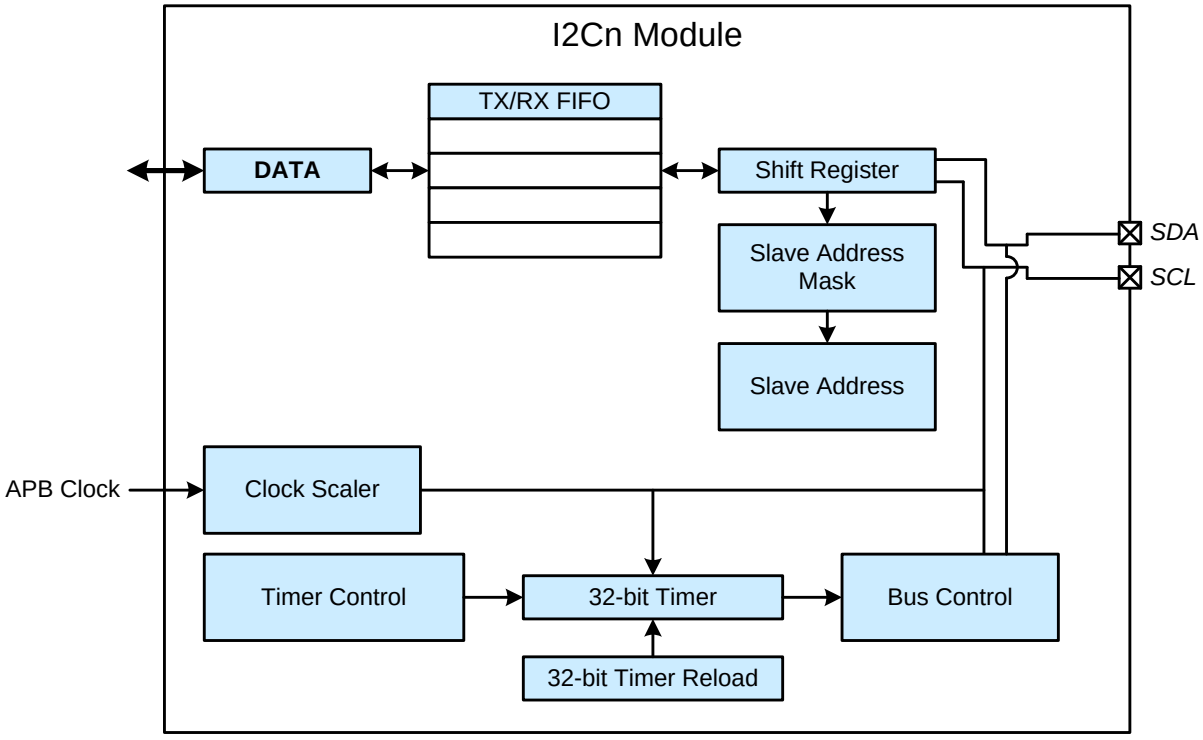


Figure 22.1. I2C Block Diagram

22.2. I2C Protocol

The I2C interface is a two-wire, bi-directional serial bus. Reads and writes to the interface are byte oriented with the I2C interface autonomously controlling the serial transfer of the data. Data can be transferred at up to 1/8th of the APB clock as a master or slave (this can be faster than allowed by the I2C specification, depending on the clock source used). A method of extending the clock-low duration is available to accommodate devices with different speed capabilities on the same bus.

The I2C interface may operate as a master and/or slave, and may function on a bus with multiple masters. The I2C provides control of SDA (serial data), SCL (serial clock) generation and synchronization, arbitration logic, and start/stop control and generation.

22.2.1. Hardware Configuration

Figure 22.2 shows a typical I2C configuration. The I2C specification allows any recessive voltage between 3.0 and 5.0 V; different devices on the bus may operate at different voltage levels. The bi-directional SCL (serial clock) and SDA (serial data) lines must be connected to a positive power supply voltage through a pull-up resistor or similar circuit. Every device connected to the bus must have an open-drain or open-collector output for both the SCL and SDA lines, so that both are pulled high (recessive state) when the bus is free. The maximum number of devices on the bus is limited only by the requirement that the rise and fall times on the bus not exceed 1000 ns (rise) and 300 ns (fall) for Standard mode communication and 300 ns (rise) and 300 ns (fall) for fast mode communication.

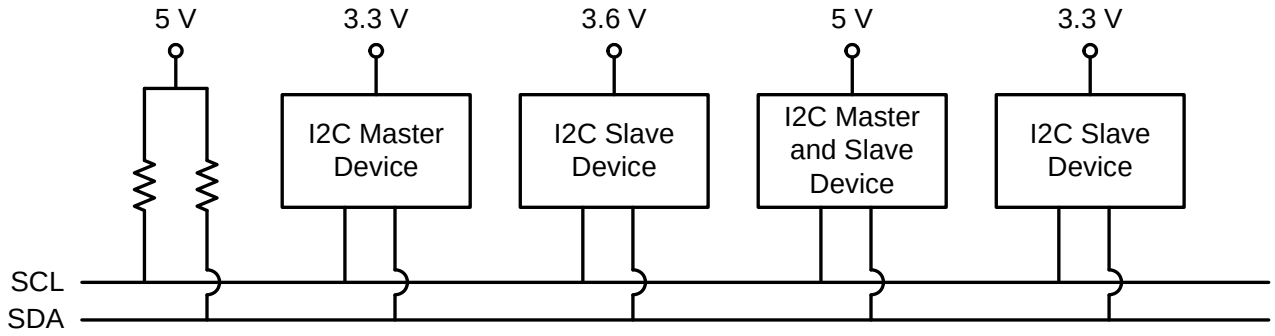


Figure 22.2. I2C Hardware Configuration

A typical I2C transaction consists of a start condition followed by an address (7- or 10-bit), the Read/Write direction bit, one or more bytes of data, and a stop condition. Address and data fields are transferred MSB first. Each byte that is received (by a master or slave) must be acknowledged (ACK) with a low SDA during a high SCL. If the receiving device does not ACK, the transmitting device will read a NACK (not acknowledge), which is a high SDA during a high SCL.

The direction bit (R/W) occupies the least-significant bit position of the address. The direction bit is set to 1 to indicate a read operation and cleared to 0 to indicate a write operation.

A start condition occurs when the SDA signal changes from high to low while SCL is high, and a stop condition occurs when the SDA signal changes from low to high while SCL is high. During normal data bit transitions, the SCL signal is low while SDA changes state.

All transactions are initiated by a master, with one or more addressed slave devices as the target. The master generates the start condition and then transmits the slave address and direction bit. If the transaction is a write operation from the master to the slave, the master transmits the data a byte at a time and waits for an ACK from the slave at the end of each byte. For read operations, the slave transmits the data and waits for an ACK from the master at the end of each byte. At the end of the data transfer, the master generates a stop condition to terminate the transaction and free the bus. Figure 22.3 illustrates a typical I2C 7-bit address transaction.

Both address sizes (7- and 10-bit) can be used on the same I2C bus.

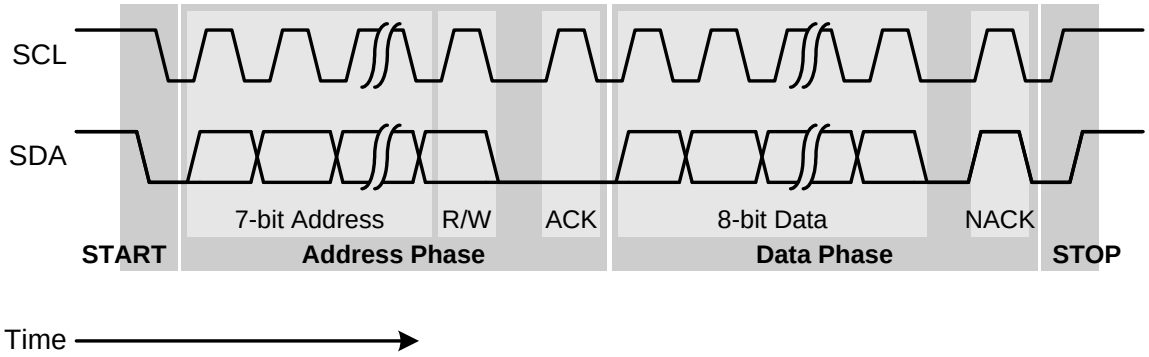


Figure 22.3. Typical I2C 7-bit Address Transaction

22.2.2. Address Phase

The address phase consists of the 7- or 10-bit address, the read/write direction bit (R/W), and the slave's ACK or NACK response.

With 7-bit address transfers, the address phase consists of a single byte operation and one ACK or NACK response from the slave. The R/W bit is the last bit transmitted before the ACK or NACK. Figure 22.4 shows the 7-bit address phase.

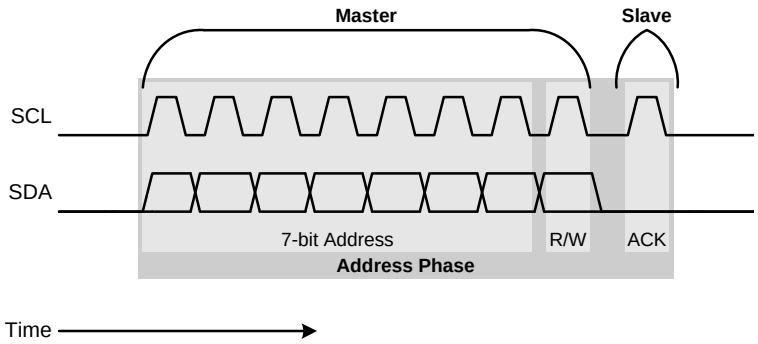


Figure 22.4. I2C 7-bit Address Phase

The 10-bit address phase consists of two bytes and two ACK or NACK responses from the slave. The first address byte contains bits 9 and 8 of the address (fixed value of 1111 0XX, where XX are the two address bits) and the R/W bit, which is the last bit of the byte. The rest of the address (bits 7:0) is transmitted in the second byte. All slaves that match the first part of the address will ACK after the first address byte. The one slave that also matches the second address byte will ACK after the second byte.

The address phase of a write operation using a 10-bit address consists of a single start followed by the two address bytes with R/W set to 0 for a write. The address phase of a read operation using a 10-bit address consists of a start followed by the two address bytes with R/W set to 0 for a write, followed by a repeated start with the first address byte and R/W set to 1 for a read, and the data for the master.

Figure 22.5 shows the 10-bit address phases for both a write and a read operation.

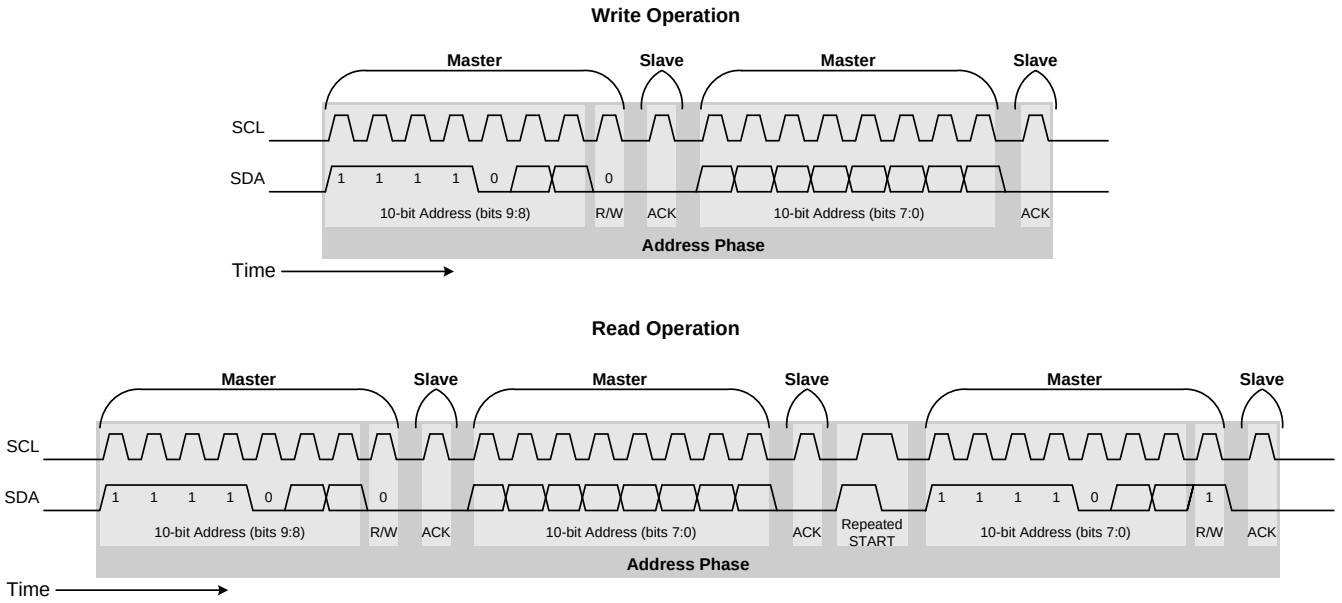


Figure 22.5. I2C 10-bit Address Phase

22.2.3. Data Phase

The data phase follows the address phase and is the same for both 7- and 10-bit address modes. The data phase consists of a byte of data followed back an ACK or NACK. In the case of a read operation, the slave provides the data byte and the master responds as shown in Figure 22.6. In the case of a write operation, the master provides the data byte and the slave responds as shown in Figure 22.7.

The entire data phase for a transaction can consist of one or more bytes. The master determines the total number of bytes sent by deciding when to send the stop condition that ends the transaction.

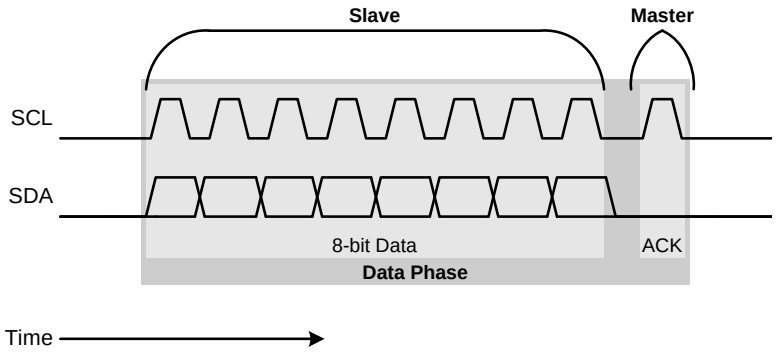


Figure 22.6. Single Byte I2C Read Data Phase

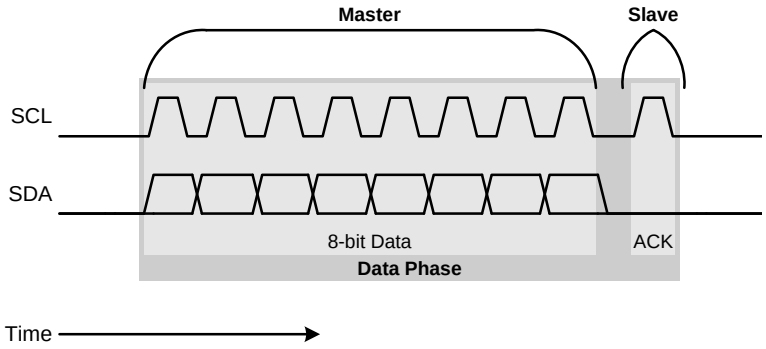


Figure 22.7. Single Byte I2C Write Data Phase

22.3. Clocking

The I2C module has an internal 6-bit APB clock divider that allows the module to support various APB and I2C clock frequencies. The divided I2C clock serves as the timebase for the SCL signal, rise and fall timing, and hardware-supported timeouts. The SCALER field in the CONFIG register sets the I2C module clock frequency (F_{I2C}) according to Equation 22.1.

$$F_{I2C} = \frac{F_{APB}}{(64 - SCALER)}$$

Equation 22.1. I2C Module Clock Frequency

22.3.1. Clock Setup

When operating as an I2C master, the I2C clock high and low times (T_{SCL_HIGH} and T_{SCL_LOW}) are based off the I2C module clock, and can be independently configured. The clock high time is determined by the T1RL field in the TIMERRL register, as shown in Equation 22.2. Likewise, the clock low time is determined by the SCLL field in the SCONFIG register, shown in Equation 22.3.

$$T_{SCL_HIGH} = \frac{256 - T1RL}{F_{I2C}}$$

Equation 22.2. SCL High Time

$$T_{SCL_LOW} = \frac{256 - SCLL}{F_{I2C}}$$

Equation 22.3. SCL Low Time

The I2C bus master controls the clock. However, slave devices have the option to extend the clock low time if needed. When operating as an I2C slave, the SCLL field defines a period for the device to hold SCL low after detecting a falling edge on the bus.

22.4. Operational Modes

The I2C module supports both master and slave states. Each step of the transaction process is controlled both by hardware and firmware to provide flexibility. A typical I2C module interrupt service routine contains several states to determine the module's next actions in response to activity on the bus. Each state must perform actions in the following order:

1. Set and clear bits for the next state.
2. Write or read from the DATA register.
3. Arm the transmit or receive operation.
4. Clear the pending interrupt flag.

The I2C module can send or receive multiple bytes autonomously during the data phase by setting the BC field in the CONFIG register to a value other than 1. If the BC value is set to a value other than 1, the transmit or receive interrupt will occur when all bytes have been sent or received (when BP is equal to BC). If multiple bytes are sent during a transaction using a BC value other than 1, the least-significant byte in the DATA register will be sent or received first.

The I2C module supports 7-bit addresses in hardware and can support 10-bit addresses by using firmware to transmit and decode the second address byte.

The basic software-controlled master and slave transactions are described in detail in the follow section. The additional features of the module are described in Section 22.6 and DMA modes are described in Section 22.8.

22.4.1. Master Write Transaction

In a master write transaction, the I2C module is in master mode and sends one or more bytes of data to an addressed slave on the bus.

The master write operation starts with firmware setting the STA bit to generate a start condition. A start interrupt occurs after the hardware transmits a start condition. The ISR or firmware routine should then clear the start bit (STA), set the targeted slave address and the R/W direction bit in the DATA register, set the byte count, arm the transmission (TXARM = 1), and clear the start interrupt.

After the hardware transmits the address and direction bit, a transmit interrupt occurs. The ISR should check the ACK bit to determine if the addressed slave acknowledged the address and is ready to receive data. If the master received a NACK, no slaves acknowledged the address, and firmware should set the STO bit to generate a stop. If the master received an ACK, the firmware should load the data into the DATA register, arm the transmission (TXARM = 1), and clear the transmit interrupt. This process is identical for each byte or set of bytes in the transmission.

For each set of bytes (determined by the BC field), the hardware generates an acknowledge interrupt if the slave NACKs a transmission, and will not generate an interrupt if the slave ACKs. If the slave NACKs, the master can choose to re-transmit the data or start the process over. To stop the current transmission, firmware should set the STO bit to generate a stop condition. When the stop interrupt occurs, the transmission can be re-started by setting the STA bit to generate a start condition.

When the hardware transmits the last data byte, a transmit interrupt occurs. The firmware can check the ACK or NACK status and should set the STO bit to generate a stop condition and clear the transmit interrupt before exiting the ISR. The transmission does not need to be armed in this case because an address or data isn't being sent.

A stop interrupt occurs after the hardware transmits the stop condition. Firmware should clear the stop interrupt and exit the ISR, completing the transaction.

Figure 22.8 shows a flow diagram of this master write transaction process.

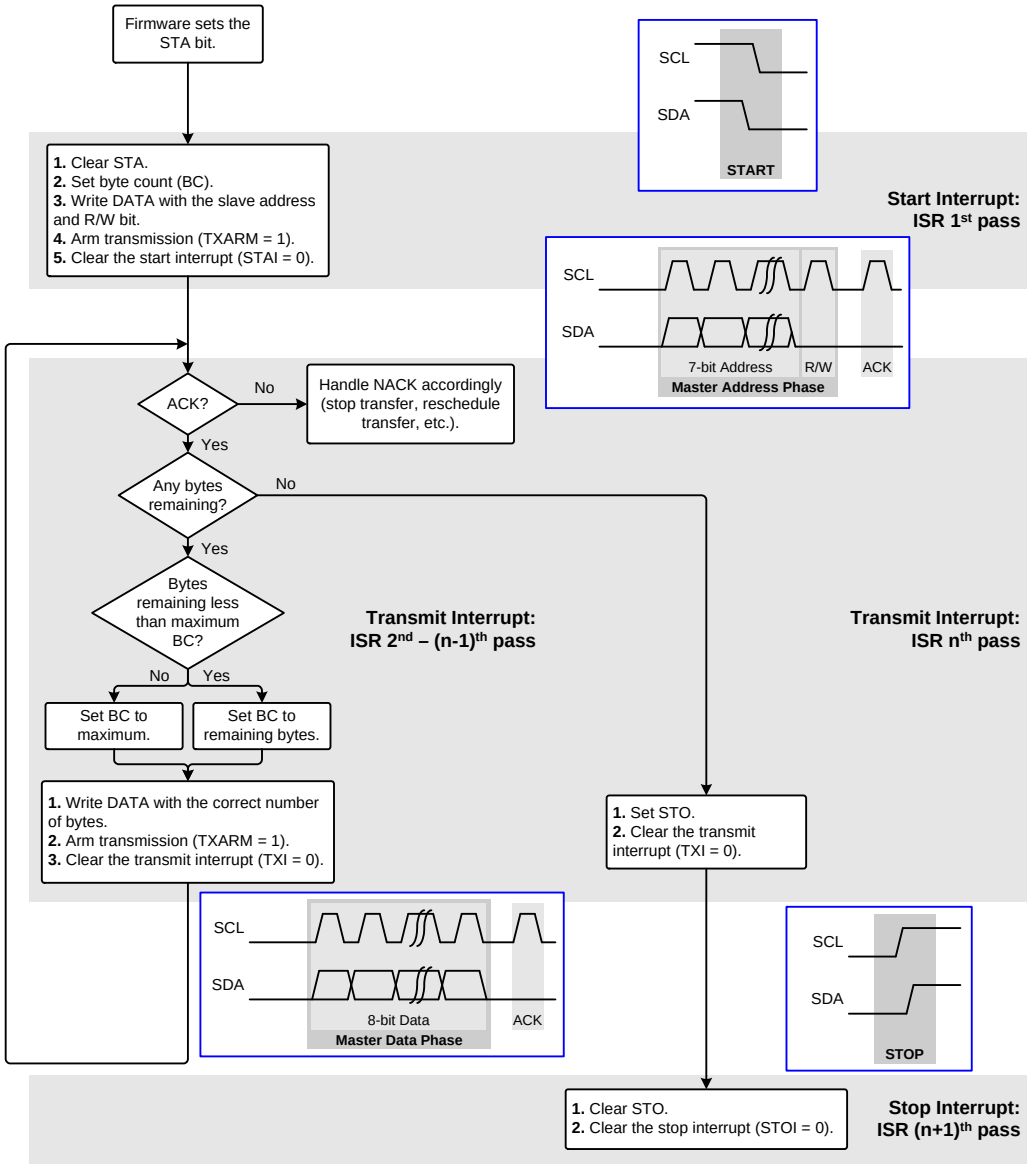


Figure 22.8. Master Write Flow Diagram (7-bit Address)

22.4.2. Master Read Transaction

In a master read transaction, the I2C module is in master mode and receives one or more bytes of data from an addressed slave on the bus.

The master read operation starts the same as the master write operation with firmware setting the STA bit to generate a start condition. A start interrupt occurs after the hardware transmits a start condition. The ISR or firmware routine should then clear the start bit (STA), set the targeted slave address and the R/W direction bit in the DATA register, set the byte count, arm the transmission (TXARM = 1), and clear the start interrupt.

After the hardware transmits the address and direction bit, a transmit interrupt occurs. The ISR should check the ACK bit to determine if the addressed slave acknowledged the address and is ready to receive data. If the master received a NACK, no slaves acknowledged the address, and firmware should set the STO bit to generate a stop. If the master received an ACK, the firmware should arm reception (RXARM = 1), set the byte count, and clear the transmit interrupt.

A master acknowledge interrupt occurs for each byte received from the slave (determined by the BC field). The firmware must set the ACK bit and clear the acknowledge interrupt.

When the master receives all of the bytes of the transfer (BP is equal to BC), a receive interrupt occurs. The firmware must read the data from the FIFO using the DATA register, set the BC field for the next reception, arm reception (RXARM = 1), and clear the receive interrupt. If the master does not need to read any additional data from the slave, the firmware should set the STO bit to generate a stop and clear the receive interrupt.

A stop interrupt occurs after the hardware transmits the stop condition. Firmware should clear the stop interrupt and exit the ISR, completing the transaction.

Figure 22.9 shows a flow diagram of this master read transaction process.

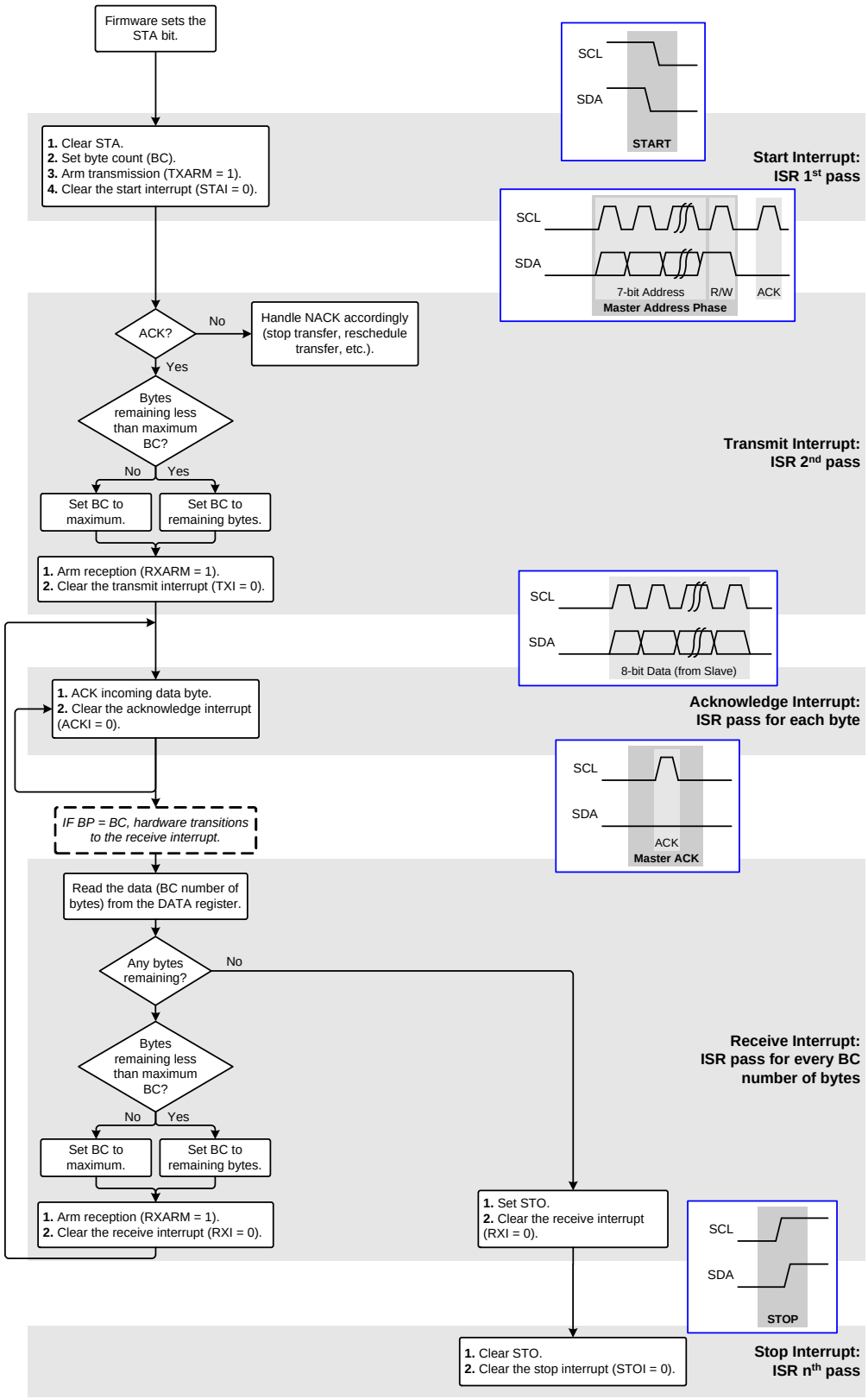


Figure 22.9. Master Read Flow Diagram (7-bit Address)

22.4.3. Repeated Starts

Repeated starts are used in master transactions when slaves require a write before the master can read data. The repeated start allows the master to maintain control of the bus even though two separate transactions take place.

The repeated start is the same as a master write transaction followed by a master read transaction where the write ends with another start instead of a stop condition.

The repeated start operation begins with firmware setting the STA bit to generate a start condition. A start interrupt occurs after the hardware transmits a start condition. The ISR or firmware routine should then clear the start bit (STA), set the targeted slave address and the R/W direction bit (write) in the DATA register, set the byte count, arm the transmission (TXARM = 1), and clear the start interrupt.

After the hardware transmits the address and direction bit, a transmit interrupt occurs. The ISR should check the ACK bit to determine if the addressed slave acknowledged the address and is ready to receive data. If the master received a NACK, no slaves acknowledged the address, and firmware should set the STO bit to generate a stop. If the master received an ACK, the firmware should write the data to the DATA register, set the byte count, arm the transmission (TXARM = 1), and clear the transmit interrupt.

The second transmit interrupt occurs after the master sends the data to the slave. The firmware should set the STA bit again and clear the transmit interrupt to generate the repeated start.

In the second start interrupt pass, the firmware should again write the slave address and R/W direction bit (read) in the DATA register before clearing the start bit, arming the transmission (TXARM = 1), setting the byte count, and clearing the start interrupt.

A master acknowledge interrupt occurs for each byte received from the slave (determined by the BC field). The firmware must set the ACK bit and clear the acknowledge interrupt.

When the master receives all of the bytes of the transfer (BP is equal to BC), a receive interrupt occurs. The firmware must read the data from the DATA register, set the BC field for the next reception, arm reception (RXARM = 1), and clear the receive interrupt. If the master does not need to read any additional data from the slave, the firmware should set the STO bit to generate a stop and clear the receive interrupt.

A stop interrupt occurs after the hardware transmits the stop condition. Firmware should clear the stop interrupt and exit the ISR, completing the transaction.

Figure 22.10 and Figure 22.11 show a flow diagram of the master repeated start process.

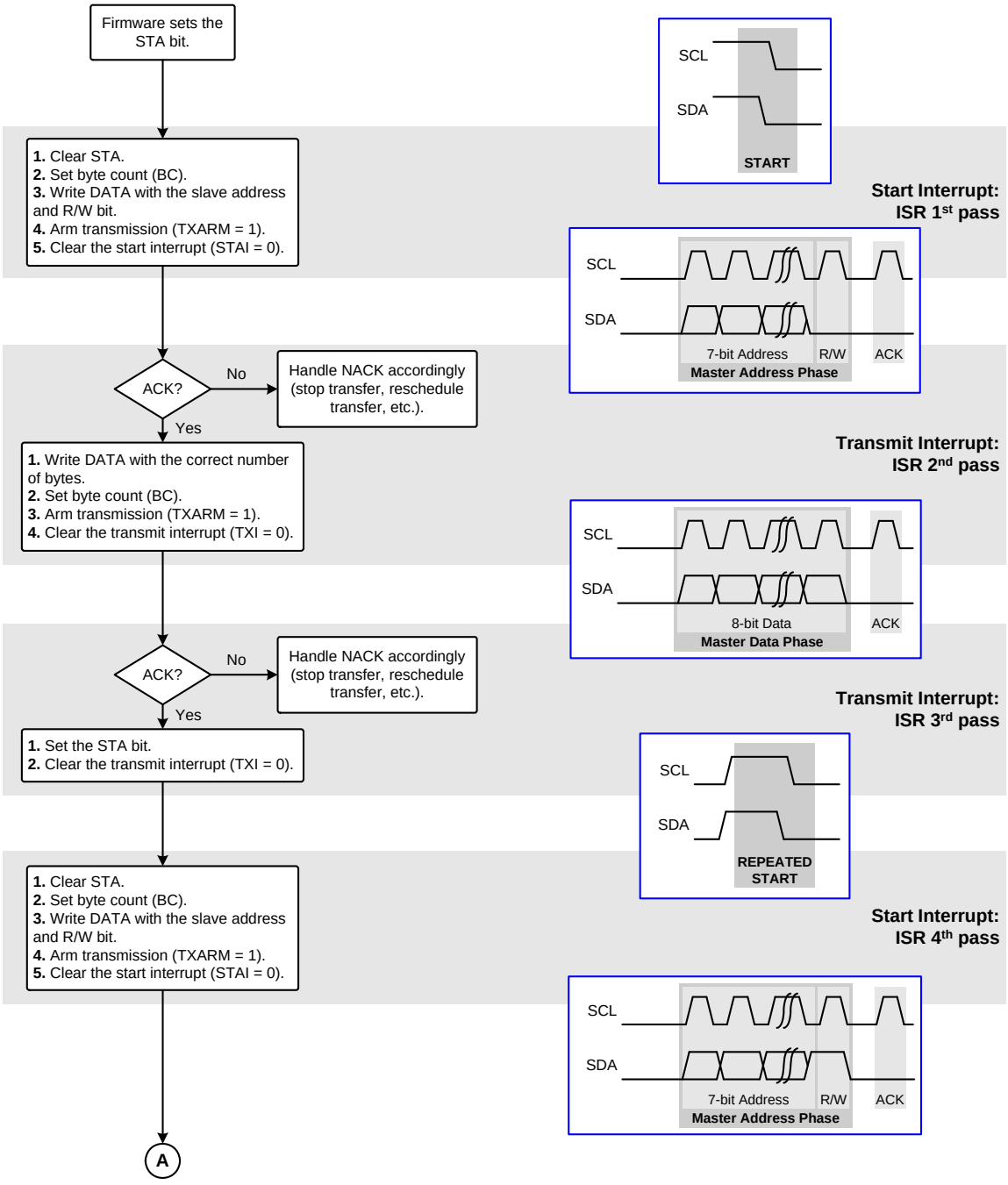


Figure 22.10. Master Repeated Start (Write/Read) Flow Diagram (7-bit Address) (Page 1/2)

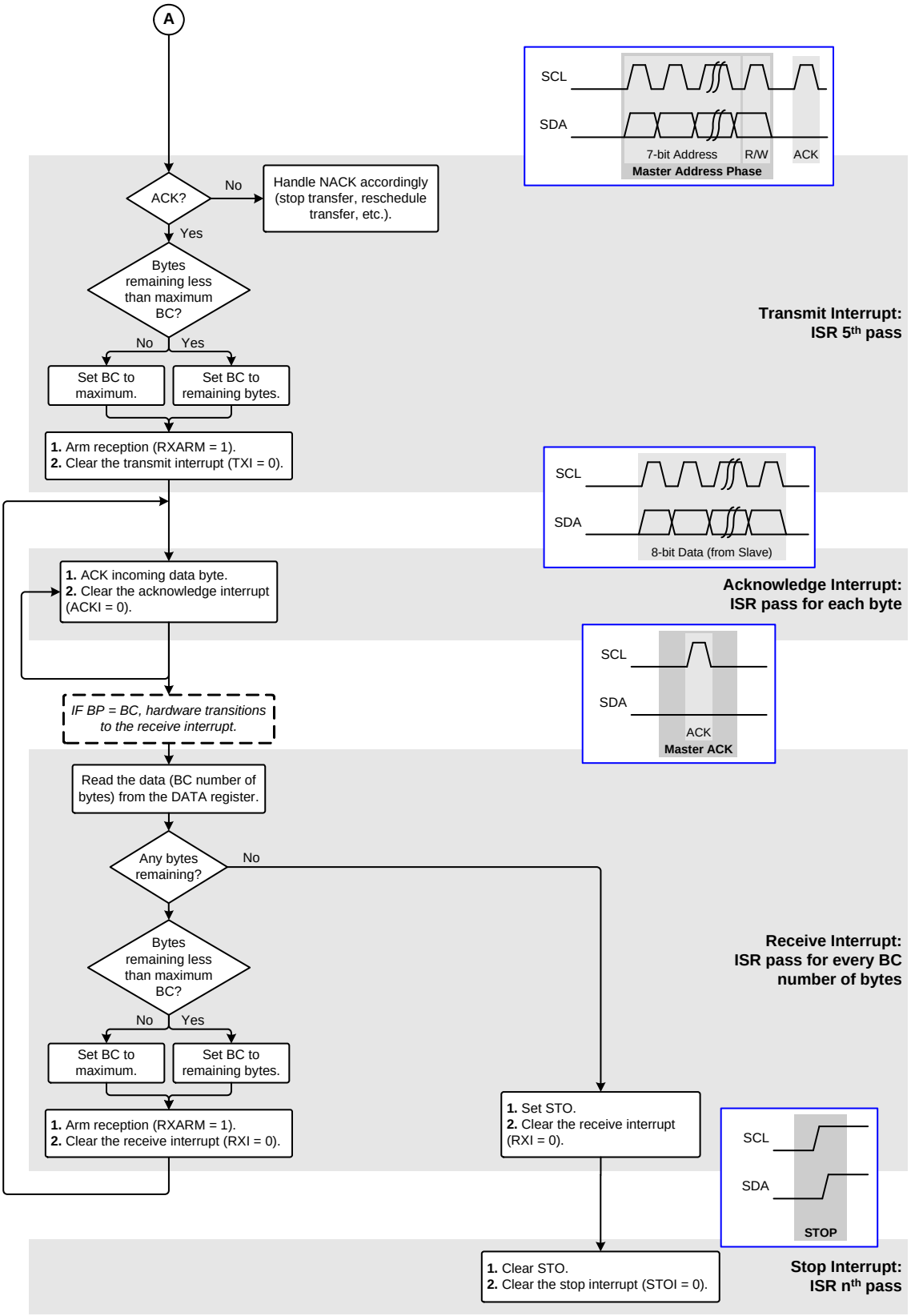


Figure 22.11. Master Repeated Start (Write/Read) Flow Diagram (7-bit Address) (Page 2/2)

22.4.4. Slave Write Transaction

In a slave write transaction, the I2C module is in slave mode and receives one or more bytes of data from a master.

The slave write operation starts with a start interrupt after a master generates a start and sends the address on the bus. The firmware should read the address and R/W direction bit from the DATA register and determine if the address matches the slave address. If not, the slave should send a NACK by clearing the ACK bit to 0 and clear the start interrupt. If the slave address does match, the slave should ACK the address by setting ACK to 1, set the byte count, clear the start bit, arm reception (RXARM = 1), and clear the start interrupt.

An acknowledge interrupt occurs each time the slave receives a byte. The firmware should respond with either an ACK or NACK and clear the acknowledge interrupt.

When the slave receives the last byte of the current packet (BP = BC), a receive interrupt occurs. The slave should read the data from the DATA register, set the byte count for the next set of bytes, arm reception, and clear the receive interrupt. If this is the last set of bytes the slave will receive, the slave should clear the receive interrupt.

A stop interrupt occurs after the master generates a stop condition on the bus. The firmware should clear the stop bit and the stop interrupt.

Figure 22.12 shows a flow diagram of this slave write transaction process.

22.4.5. Slave Read Transaction

In a slave read transaction, the I2C module is in slave mode and sends one or more bytes of data to a master.

The slave read operation starts with a start interrupt after a master generates a start and sends the address on the bus. The firmware should read the address and R/W direction bit from the DATA register and determine if the address matches the slave address. If not, the slave should send a NACK by clearing the ACK bit to 0 and clear the start interrupt. If the slave address does match, the slave should ACK the address by setting ACK to 1, set the byte count, write the data to the DATA register, clear the start bit, arm transmission (TXARM = 1), and clear the start interrupt.

A transmit interrupt occurs each time the slave sends a set of bytes (BP = BC). The firmware should check if there are any bytes remaining for the current transfer. If so, the byte count should be set, the new data written to the DATA register, transmission armed (TXARM = 1), and the transmit interrupt cleared. If no data remains to be sent, the slave should clear the transmit interrupt and wait for the master to generate a stop.

A stop interrupt occurs after the master generates a stop condition on the bus. The firmware should clear the stop bit and clear the stop interrupt.

Figure 22.13 shows a flow diagram of this slave read transaction process.

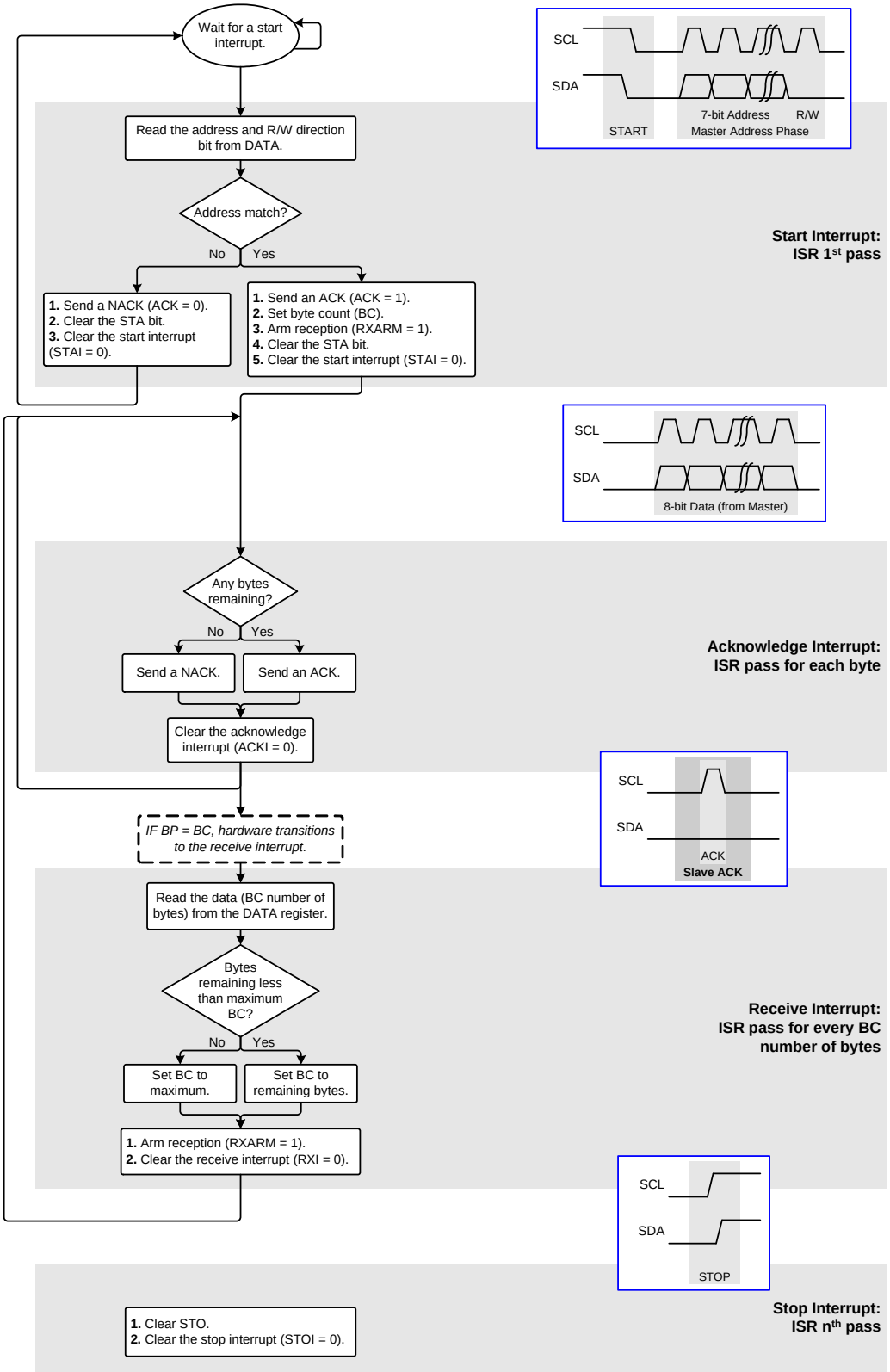


Figure 22.12. Slave Write Flow Diagram (7-bit Address)

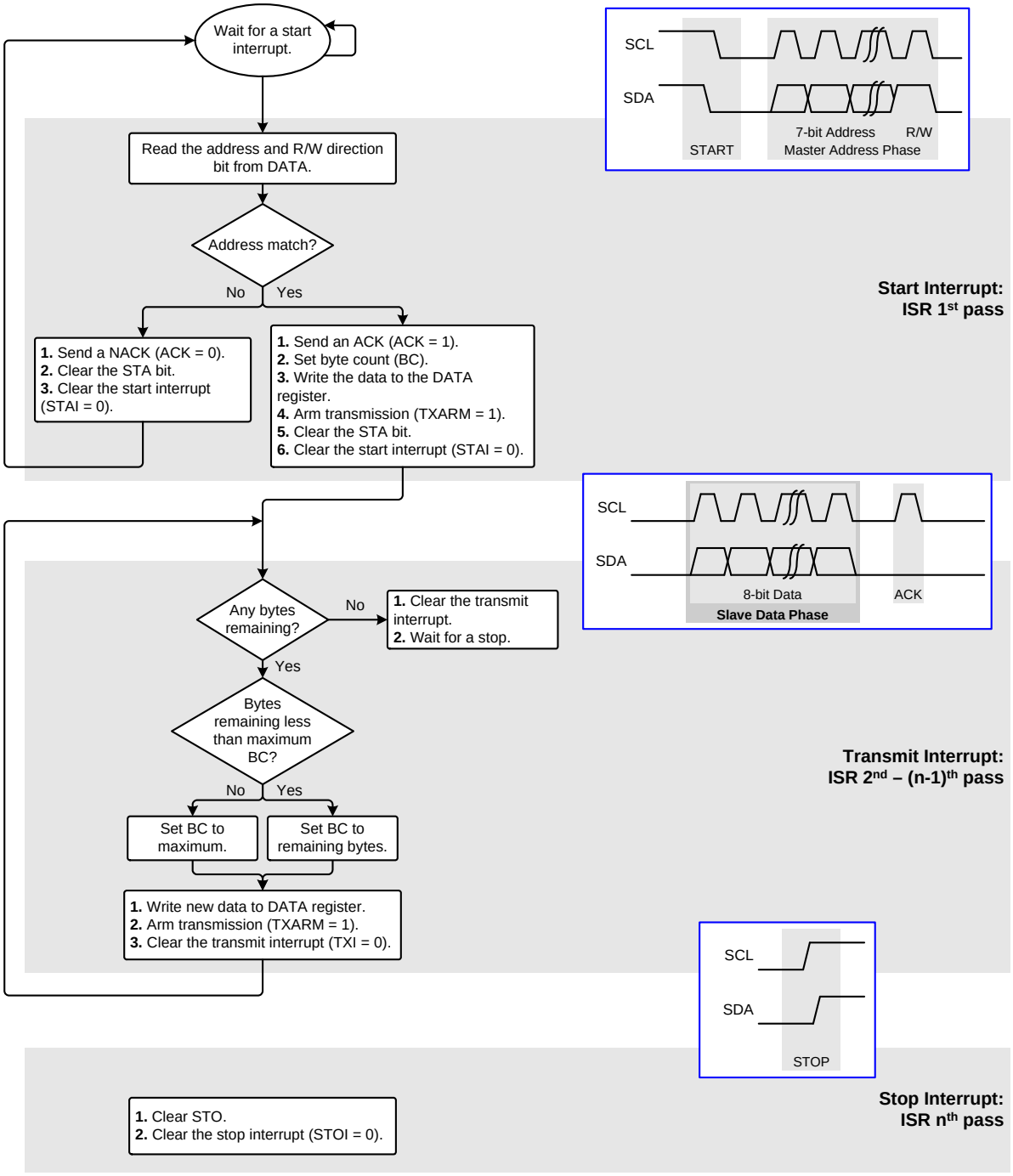


Figure 22.13. Slave Read Flow Diagram (7-bit Address)

22.5. Error Handling

22.5.1. Arbitration

A master may start a transfer only if the bus is free. The bus is free after a stop condition or after the SCL and SDA lines remain high for a specified time. In the event that two or more devices attempt to begin a transfer at the same time, the I2C protocol employs an arbitration scheme to force one master to give up the bus. The master devices continue transmitting until one attempts to transmit a 1 while the other attempts to transmit a 0. Since the bus is open-drain, the bus will be pulled low, and the master attempting to transmit the 1 will detect a low SDA signal and lose the arbitration. The winning master continues its transmission without interruption; the losing master becomes a slave (if slave states are supported) and receives the rest of the transfer, if addressed. This arbitration scheme is non-destructive: one device always wins, and no data is lost.

In the case of an arbitration lost event, the arbitration lost interrupt occurs. The read-only ARBLF flag mirrors the state of the ARBLI interrupt flag. This interrupt can occur in both master or slave mode whenever the I2C module is attempting to transmit data on the bus. When configured as a slave, the ARBLI interrupt flag indicates a protocol violation on the bus. In master mode, the firmware may want to save the aborted slave target and R/W direction bit to allow this attempt to be rescheduled after any pending slave response, if supported.

22.5.2. SMBus SCL Low Timeout

If the SCL line is held low by a device on the bus, no further communication is possible. Furthermore, the master cannot force the SCL line high to correct the error condition. To solve this problem, the SMBus protocol specifies that devices participating in a transfer must detect any clock cycle held low longer than 25 ms as a “timeout” condition. Devices that have detected the timeout condition must reset their communication interfaces no later than 10 ms after detecting the timeout condition.

When the I2C module is enabled (I2CEN = 1), the dedicated I2C 32-bit timer provides a counter for SCL low timeout detection. The SCL low timeout counter is 20 bits, with Timer Byte [3: 2] providing the upper 16 bits of the counter. The least significant bits of the timeout counter are not available to program. The timeout counter can be set by setting the T3:T2 and T3RL:T2RL fields. The hardware automatically forces the timer to reload when SCL is high and allows the timer to count when SCL is low.

In the event that T3:T2 matches the value in T3RL:T2RL, the timeout T3I flag will be set and an interrupt will be generated, if enabled. If this interrupt occurs, firmware can reset the I2C module using the RESET bit.

22.5.3. Bus Free Timeout

The I2C bus is free if all previous transactions ended with a stop condition and both SCL and SDA are high. The I2C module supports a bus free timeout that detects if SCL and SDA are high for the specified timeout period without a stop condition appearing on the bus.

The dedicated I2C Timer Byte 0 serves as the bus free timeout counter when the module is enabled (I2CEN = 1). The T0 field contains the timeout value and T0RL contains the counter reload value. T0 will count up if both SCL and SDA are high, and hardware forces T0 to reload from T0RL if SCL or SDA are low. If the I2C module is waiting for the bus to be free before starting a transfer, the module will generate the start condition after the timeout period expires.

22.6. Additional Features

22.6.1. Hardware Acknowledge and General Call Addressing

When the HACKEN bit in the CONTROL register is cleared to 0, the firmware on the device must detect incoming slave addresses and ACK or NACK the slave address and incoming data bytes. As a receiver, writing the ACK bit defines the outgoing ACK value; as a transmitter, reading the ACK bit indicates the value received during the last ACK cycle. ACKRQF is set each time a byte is received, indicating that an outgoing ACK value is needed. When ACKRQF is set, firmware should write the desired outgoing value to the ACK bit before clearing the appropriate interrupt flag. A NACK will be generated if software does not write the ACK bit before clearing the interrupt. SDA will reflect the defined ACK value immediately following a write to the ACK bit; however, SCL will remain low until firmware clears the interrupt. If a received slave address is not acknowledged, further slave events will be ignored until the next start is detected.

The hardware acknowledge feature (enabled when HACKEN = 1) provides for automatic 7-bit slave address recognition and ACK generation. The ADDRESS and MASK fields define which addresses are automatically recognized by the hardware. A single address or range of addresses (including the General Call Address of all 0's) can be specified using these two registers. A 1 in a bit position of the slave address mask (MASK) enables a comparison between the received slave address and the hardware's slave address (ADDRESS) for those bits. A 0 in the slave address mask means that bit will be treated as a "don't care" for comparison purposes. Additionally, hardware will recognize the General Call Address (all 0's) if the GCEN bit in the CONTROL register is set to 1. Firmware can use the SLVAF bit to determine if the slave recognized the slave address or the General Call. Table 22.1 shows some example parameter settings and the slave addresses that will be recognized by hardware.

Table 22.1. Hardware Address Recognition Examples (HACKEN = 1)

Hardware Slave Address (ADDRESS)	Hardware Slave Address Mask (MASK)	General Call Address (GCEN)	Slave Addresses Recognized by Hardware
0x34	0x7F	Disabled	0x34
0x34	0x7F	Enabled	0x34, 0x00
0x34	0x7E	Disabled	0x34, 0x35
0x34	0x7E	Enabled	0x34, 0x35, 0x00
0x70	0x73	Disabled	0x70, 0x74, 0x78, 0x7C

During the data phases of the transfer, the I2C hardware in receiver mode will use the value currently specified by the ACK bit to automatically respond during the ACK cycle of an incoming data byte. As a transmitter, reading the ACK bit indicates the value received on the last ACK cycle. The ACKRQF bit is not used when hardware ACK generation is enabled. If a received slave address is NACKed by hardware, the hardware will ignore any further slave events until the next start is detected and will not generate a start interrupt.

When hardware acknowledge is enabled in receive mode, the last byte acknowledge enable bit (LBACKEN) can be used to automatically NACK the last byte of a transfer, if desired.

22.6.1.1. Automatic Transmit or Receive Enable

The automatic transmit or receive mode can only be used if hardware acknowledge is enabled. If the ATXRXEN bit is set to 1, the I2C module will automatically ACK the incoming address and switch to either receive or transmit mode depending on the R/W bit. Enabling automatic transmit or receive mode bypasses the start interrupt, so the receive or transmit interrupt is the first interrupt triggered if ATXRXEN is set to 1.

22.6.2. SDA Setup and Hold Time Extensions

The data setup and hold times can be optionally extended using the SETUP and HOLD fields in the SCONFIG register. I2C Timer Byte 0 determines the data setup and hold times if SETUP and HOLD are 0. If SETUP or HOLD is set to a non-zero value, this setting overrides the I2C Timer Byte 0 count.

These extensions are based on the I2C module clock, and the equations for the additional time are provided in the register descriptions for these fields.

22.6.3. General Purpose Timer

When the I2C is not in use (I2CEN = 0), the dedicated I2C 32-bit timer that generates SCL and SDA timing can be used as an additional general purpose count-up timer.

This I2C timer can be configured to the following modes:

- Mode 0: One 32-bit timer with auto-reload (Timer Bytes [3: 0]).
- Mode 1: Two 16-bit timers with auto-reload (high timer: Timer Bytes [3: 2], low timer: Timer Bytes [1: 0]).
- Mode 2: Four 8-bit timers with auto-reload (Timer Byte 3, Timer Byte 2, Timer Byte 1, and Timer Byte 0).
- Mode 3: One 16-bit and two 8-bit timers with auto-reload (Timer Bytes [3: 2], Timer Byte 1, and Timer Byte 0).

The mode of the timers can be controlled using the TMD field. The TxRUN bits control the timers, TxIEN bits enable the timer interrupts, Tx fields contain the current timer value, and TxRL fields contain the reload values. When one of the timers overflows (from all 1's to all 0's), the appropriate TxI interrupt flag will be set and an I2C interrupt will occur, if enabled. The timer run bits (TxRUN) are gated by a global I2C timer enable bit (TIMEREN) that must be set to 1 for the timers to count on the I2C module clock.

22.6.4. Noise Filtering

By default, there is a three-stage filter on both the SDA and SCL lines to help prevent noisy bus situations from affecting the peripheral and causing bus errors. Under most conditions this filter should be left enabled. If the APB clock frequency is relatively slow, this filter could have the side effect of filtering out the desirable bus signals. The filter may be disabled by setting the FMD bit in the CONTROL register to 1.

22.7. Debug Mode

Firmware can set the DBGMD bit to force the I2C module to halt on a debug breakpoint. The I2C block will complete the current byte transfer before halting. Clearing the DBGMD bit forces the module to continue operating while the core halts in debug mode.

22.8. DMA Configuration and Usage

The DMA interface to the I2C block has one channel mapped to transmit and one channel mapped to receive operations. When DMA is enabled (DMAEN = 1), data must be transferred 4 bytes at a time. For a transfer length which is not a multiple of 4, the extra bytes in the MSB of the final DMA transfer are ignored.

The I2C module can be programmed to transfer up to 255 bytes in a single DMA operation by setting the DMALEN field in the I2CDMA register. For transmit operations, the DMA engine requests a word transfer to DATA whenever the DATA register is empty and the DMALEN field is not 0. For receive operations, the DMA engine requests a word transfer from DATA whenever the DATA register is full and the DMALEN field is not 0. The DMALEN field will decrement by 1 when the hardware transmits or receives a byte. If the DMALEN value is less than 4, the DMA transmits or receives the remaining bytes only. A transmit or receive interrupt occurs when DMALEN is zero.

If the hardware receives a NACK during a DMA operation, it will generate an acknowledge interrupt and stop the remaining transaction. If the hardware detects a stop condition before DMALEN is zero, the transfer stops and a stop interrupt occurs.

The I2C Module DMA configuration is shown in Figure 22.14.

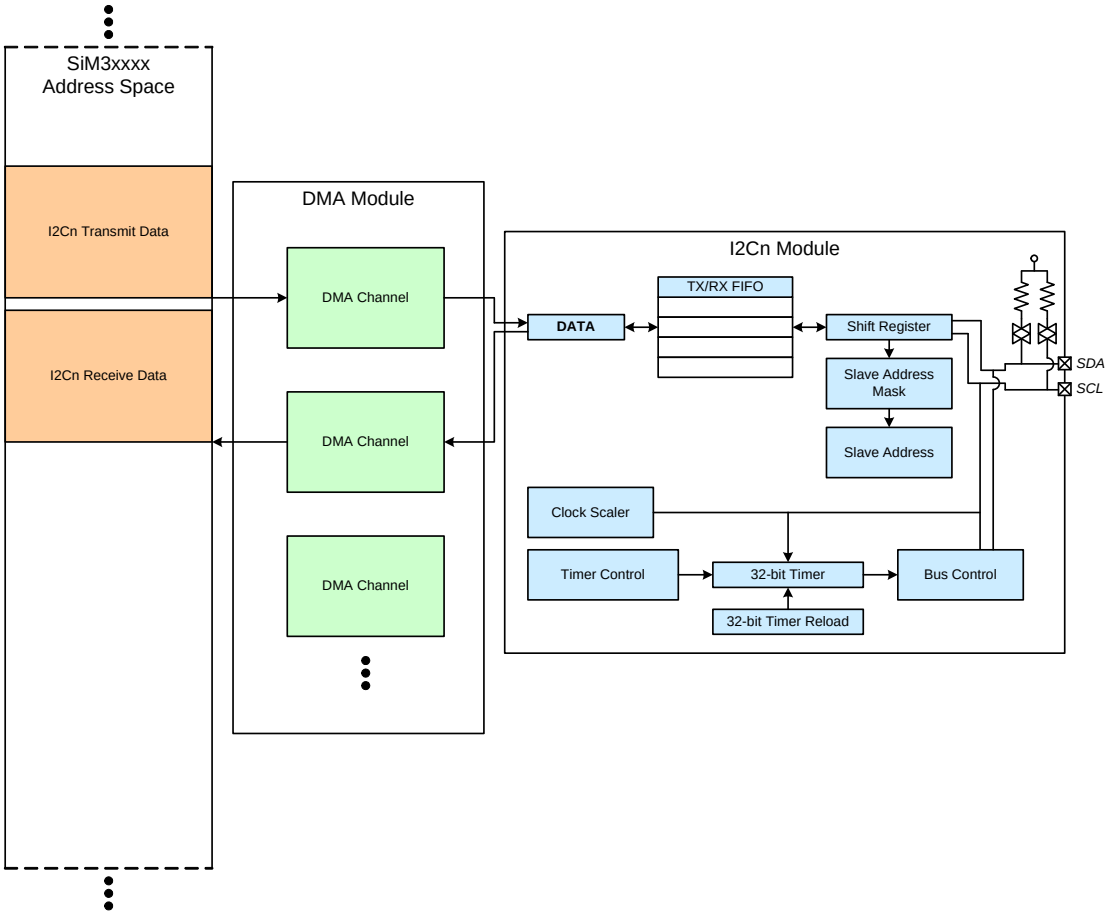


Figure 22.14. I2C Module DMA Configuration

22.8.1. Automatic Hardware DMA Options

To further automate the I2C transfer, firmware can enable Hardware Acknowledge (HACKEN = 1), Automatic Transmit or Receive Enable (ATXRXEN = 1), or Last Byte Acknowledge Enable (LBACKEN = 1) in DMA mode. If all of these modes are enabled, the hardware will automatically acknowledge any received bytes, automatically switch to transmit or receive mode depending on the set direction of the R/W bit, and ACK or NACK the last byte of the transfer, if it's a receive operation.

22.8.2. Master Write with DMA and Automatic Hardware Enabled

For a master write operation with all automatic modes enabled, the firmware should:

1. Set HACKEN and ATXRXEN to 1.
2. Write the address and R/W bit into DATA.
3. Program the DMALEN field to the appropriate value.
4. Set up the DMA transmit channel appropriately.
5. Enable the DMA mode in the I2C module (DMAEN = 1).
6. Issue a start by setting STA to 1.

If the master does not receive a NACK from the slave, the first interrupt received will be the transmit interrupt after all bytes are transferred. The master can continue to send more bytes by setting up another transfer or issue a stop to end the transfer. Once the stop interrupt occurs, the firmware must clear the STO bit to 0 and clear the stop interrupt.

The DMA master write operation is shown in Figure 22.15.

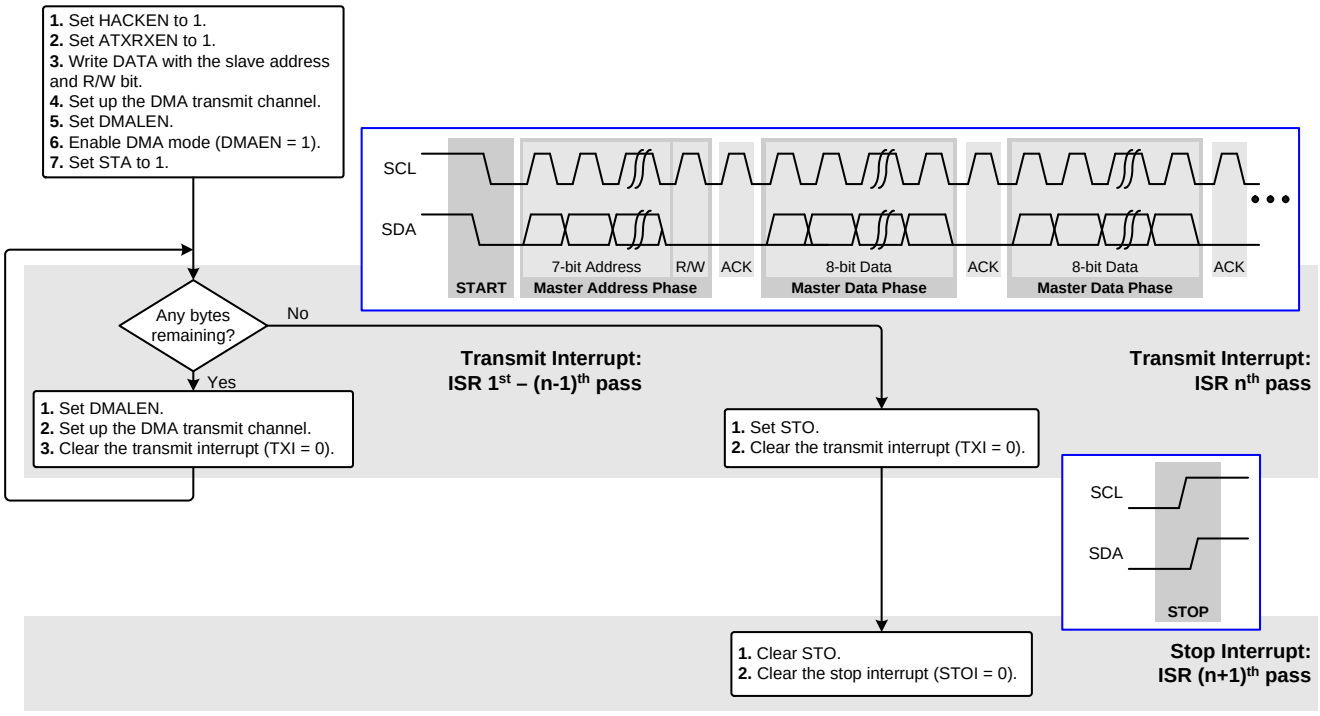


Figure 22.15. Master Write with DMA and Automatic Hardware Flow Diagram (7-bit Address)

22.8.3. Master Read with DMA and Automatic Hardware Enabled

For a master read operation with all automatic modes enabled, the firmware should:

1. Set HACKEN and ATXRXEN and to 1.
2. Write the address and R/W bit into DATA.
3. Program the DMALEN field to the appropriate value.
4. Set up the DMA receive channel appropriately.
5. Set LBACKEN to the appropriate value for the transfer.
6. Enable the DMA mode in the I2C module (DMAEN = 1).
7. Issue a start by setting STA to 1.

If the master does not receive a NACK from the slave, the first interrupt received will be the receive interrupt after all bytes are transferred. The master can continue to receive more bytes by setting up another transfer or issue a stop to end the transfer. Once the stop interrupt occurs, the firmware must clear the STO bit to 0 and clear the stop interrupt.

This DMA master read operation is shown in Figure 22.16.

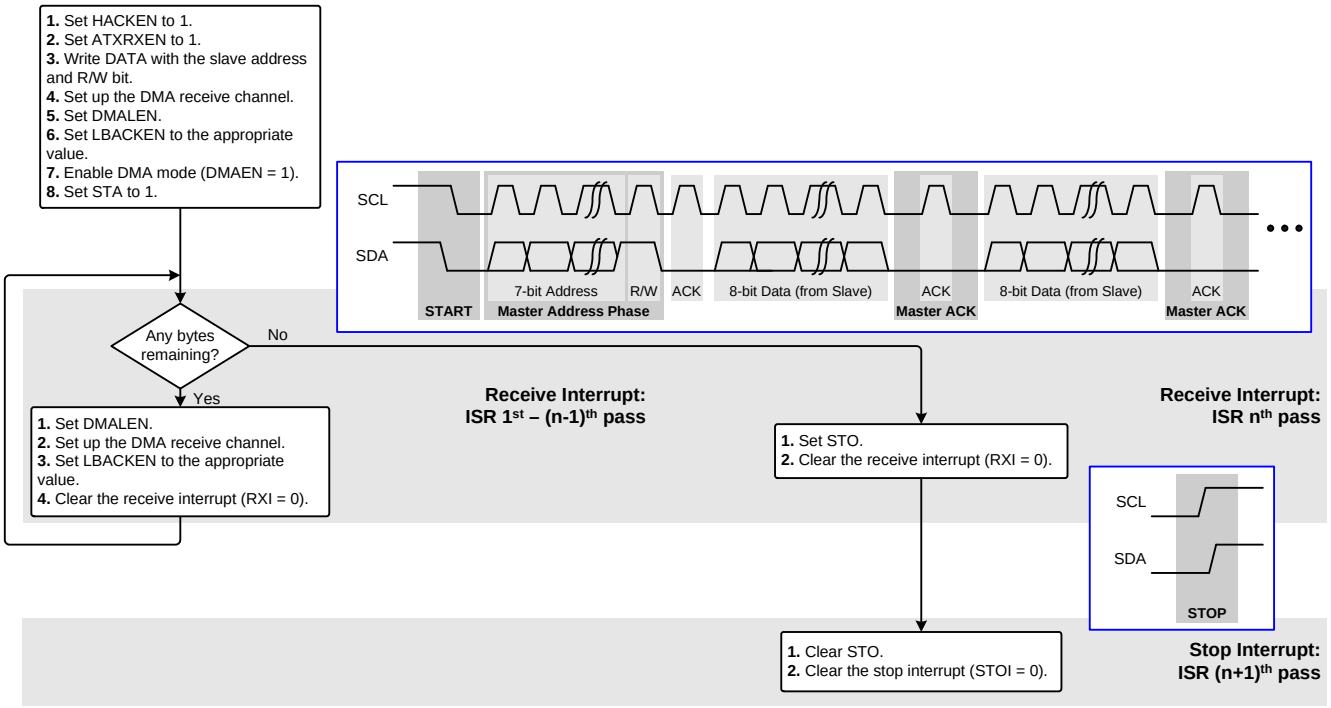


Figure 22.16. Master Read with DMA and Automatic Hardware Flow Diagram (7-bit Address)

22.8.4. Slave Write with DMA and Automatic Hardware Enabled

The DMA slave write firmware procedure with automatic hardware enabled is as follows:

1. Program the slave address and mask in ADDRESS and MASK.
2. Set the DMALEN field.
3. Set up the DMA receive channel appropriately.
4. Enable the DMA mode in the I2C module (DMAEN = 1).

The first interrupt the slave will receive is a receive interrupt, since all the bytes are automatically acknowledged. If more bytes should be received from the master, the firmware can set up another DMA transfer by resetting the DMA channel, reprogramming DMALEN and clearing the receive interrupt.

When the slave receives the stop interrupt, the firmware should clear the STO bit and the stop interrupt to end the transfer.

The DMA slave write operation is shown in Figure 22.17.

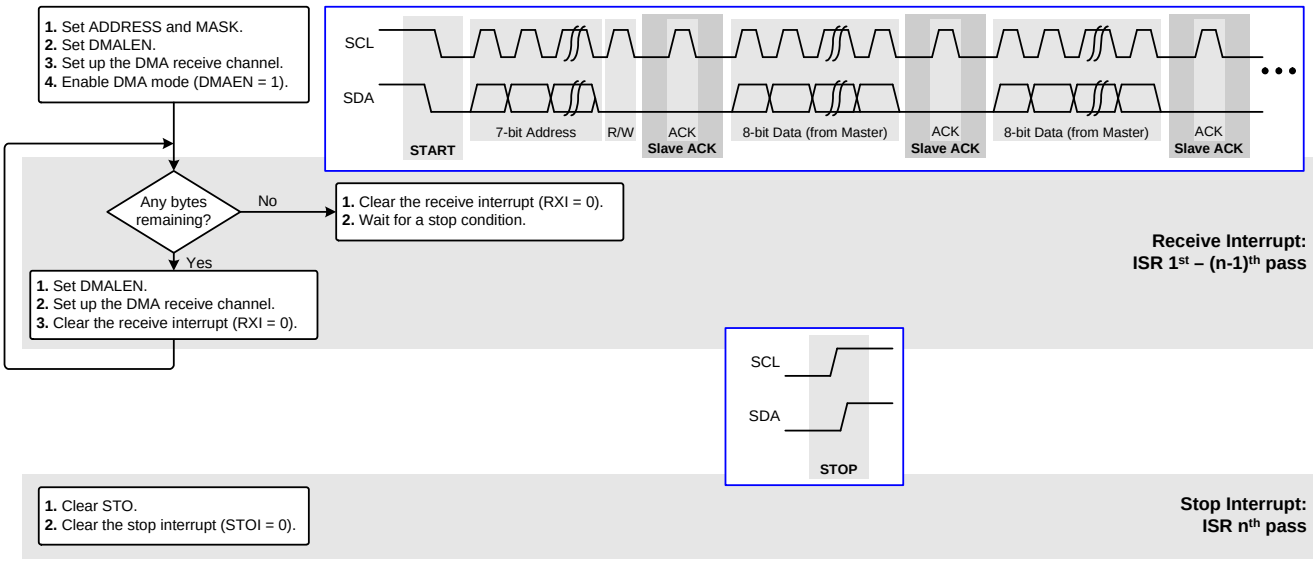


Figure 22.17. Slave Write with DMA and Automatic Hardware Flow Diagram (7-bit Address)

22.8.5. Slave Read with DMA and Automatic Hardware Enabled

The DMA slave read firmware procedure with automatic hardware enabled is as follows:

1. Program the slave address and mask in ADDRESS and MASK.
2. Set the DMALEN field.
3. Set up the DMA transmit channel appropriately.
4. Enable the DMA mode in the I2C module (DMAEN = 1).

The first interrupt the slave will receive is a transmit interrupt after all bytes are sent to the master. If more bytes should be sent to the master, the firmware can set up another DMA transfer by resetting the DMA channel, reprogramming DMALEN, and clearing the transmit interrupt.

If no more data is to be sent, when the slave receives the stop interrupt, the firmware should clear the STO bit and the stop interrupt to end the transfer.

The DMA slave read operation is shown in Figure 22.18.

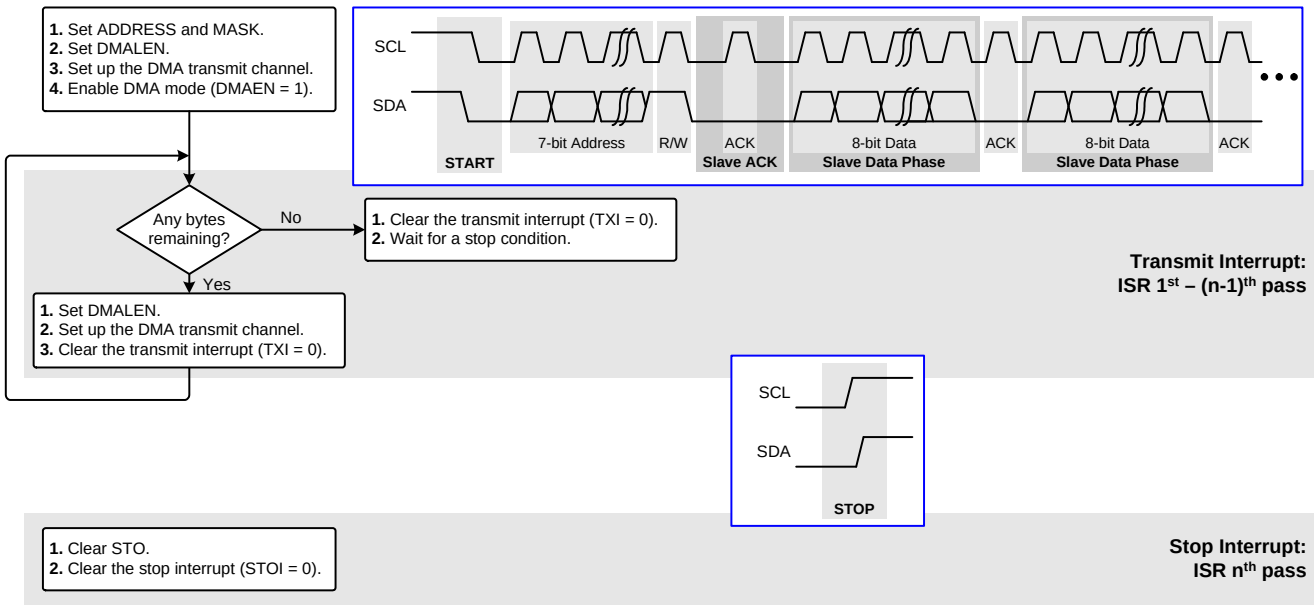


Figure 22.18. Slave Read with DMA and Automatic Hardware Flow Diagram (7-bit Address)

22.9. I2C0 and I2C1 Registers

This section contains the detailed register descriptions for I2C0 and I2C1 registers.

Register 22.1. I2Cn_CONTROL: Module Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	I2CEN	RESET	GCEN	Reserved	LBACKEN	Reserved	HACKEN	SMINH	DBGMD	FMD	ATXRXEN	SLVAF	TXARM	RXARM	T3I	T2I
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	R	RW	RW	RW	RW
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	T1I	T0I	ARBLI	STAI	TXI	RXI	ACKI	STOI	MSMDF	TXMDF	STA	STO	ACKRQF	ARBLF	ACK	BUSYF
Type	RW	RW	RW	RW	RW	RW	RW	RW	R	R	RW	RW	R	R	RW	R
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

I2C0_CONTROL = 0x4000_9000
I2C1_CONTROL = 0x4000_A000
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 22.2. I2Cn_CONTROL Register Bit Descriptions

Bit	Name	Function
31	I2CEN	**I2C Enable.** 0: Disable the I2C module. 1: Enable the I2C module.
30	RESET	**Module Soft Reset.** The following bits and fields are inaccessible while the module is in soft reset (RESET = 1): all interrupt flags (TXI, RXI, STAI, STOI, ACKI, ARBLI, T0I, T1I, T2I, T3I), STA, STO, TXARM, RXARM, ACK, ACKRQF, DMALEN, DATA, TIMER, and SCLLTIMER. 0: I2C module is not in soft reset. 1: I2C module is in soft reset and firmware cannot access all bits in the module.
29	GCEN	**General Call Address Enable.** 0: Disable General Call address decoding. 1: Enable General Call address decoding.

Notes:
1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 22.2. I2Cn_CONTROL Register Bit Descriptions

Bit	Name	Function
28	Reserved	Must write reset value.
27	LBACKEN	**Last Byte Acknowledge Enable.** Automatic hardware acknowledge mode must be enabled (HACKEN = 1) for this bit setting to have an effect. 0: NACK after the last byte is received. 1: ACK after the last byte is received.
26	Reserved	Must write reset value.
25	HACKEN	**Auto Acknowledge Enable .** 0: Disable automatic hardware acknowledge. 1: Enable automatic hardware acknowledge.
24	SMINH	**Slave Mode Inhibit.** 0: Enable Slave modes. 1: Inhibit Slave modes. The module will not respond to a Master on the bus.
23	DBGMD	**I2C Debug Mode.** 0: The I2C module will continue to operate while the core is halted in debug mode. 1: A debug breakpoint will cause the I2C module to halt.
22	FMD	**Filter Mode.** 0: Enable the input filter. 1: Disable the input filter.
21	ATXRXEN	**Auto Transmit or Receive Enable.** 0: Do not automatically switch to transmit or receive mode after a Start. 1: If automatic hardware acknowledge mode is enabled (HACKEN = 1), automatically switch to transmit or receive mode after a Start.
20	SLVAF	**Slave Address Type Flag.** 0: Slave address detected. 1: General Call address detected.
19	TXARM	**Transmit Arm.** 0: Disable data and address transmission. 1: Enable the module to perform a transmit operation.
18	RXARM	**Receive Arm.** 0: Disable data and address reception. 1: Enable the module to perform a receive operation.
17	T3I	**I2C Timer Byte 3 Interrupt Flag.** When the I2C module is enabled (I2CEN = 1), this interrupt flag will be set to 1 if an SCL low timeout occurs. When using the I2C Timer as a stand-alone timer (when I2C is disabled) this interrupt flag will set if an overflow occurs out of the the I2C Timer Byte 3. Writing a 1 to this bit will manually trigger the interrupt. This flag must be cleared by firmware.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 22.2. I2Cn_CONTROL Register Bit Descriptions

Bit	Name	Function
16	T2I	**I2C Timer Byte 2 Interrupt Flag.** When using the I2C Timer as a stand-alone timer (when I2C is disabled) this interrupt flag will set if an overflow occurs out of the the I2C Timer Byte 2. Writing a 1 to this bit will manually trigger the interrupt. This flag must be cleared by firmware.
15	T1I	**I2C Timer Byte 1 Interrupt Flag.** When using the I2C Timer as a stand-alone timer (when I2C is disabled) this interrupt flag will set if an overflow occurs out of the I2C Timer Byte 1. Writing a 1 to this bit will manually trigger the interrupt. This flag must be cleared by firmware.
14	T0I	**I2C Timer Byte 0 Interrupt Flag.** When using the I2C Timer as a stand-alone timer (when I2C is disabled) this interrupt flag will set if an overflow occurs out of the I2C Timer Byte 0. Writing a 1 to this bit will manually trigger the interrupt. This flag must be cleared by firmware.
13	ARBLI	**Arbitration Lost Interrupt Flag.** This bit is set to 1 by hardware when an arbitration lost condition occurs. This bit must be cleared by firmware.
12	STAI	**Start Interrupt Flag.** This bit is set to 1 by hardware when a start or repeated start condition occurs. The STO bit is also set with a repeated start to differentiate from a normal start condition. This bit must be cleared by firmware.
11	TXI	**Transmit Done Interrupt Flag.** This bit is set to 1 by hardware when a the module is transmitting data and BP is equal to BC. This bit must be cleared by firmware.
10	RXI	**Receive Done Interrupt Flag.** This bit is set to 1 by hardware when a the module is receiving data and BP is equal to BC. This bit must be cleared by firmware.
9	ACKI	**Acknowledge Interrupt Flag.** This bit is set to 1 by hardware when an acknowledge phase occurs and requires a response. This bit must be cleared by firmware.
8	STOI	**Stop Interrupt Flag.** This bit is set to 1 by hardware when a stop condition occurred or was generated. This bit must be cleared by firmware.
7	MSMDF	**Master/Slave Mode Flag.** 0: Module is operating in Slave mode. 1: Module is operating in Master mode.
6	TXMDF	**Transmit Mode Flag.** 0: Module is in receiver mode. 1: Module is in transmitter mode.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 22.2. I2Cn_CONTROL Register Bit Descriptions

Bit	Name	Function
5	STA	**Start.** This bit indicates whether hardware generated or detects a start on the bus. This bit should be cleared to 0 by firmware after a start is detected. Setting this bit to 1 generates a start condition in master mode.
4	STO	**Stop.** This bit indicates whether hardware generated or detects a stop on the bus. This bit should be cleared to 0 by firmware after a stop is detected. Setting this bit to 1 generates a stop condition in master mode.
3	ACKRQF	**Acknowledge Request Flag.** 0: ACK has not been requested. 1: ACK requested.
2	ARBLF	**Arbitration Lost Flag.** This read-only flag mirrors the state of the ARBLI interrupt flag. 0: Arbitration lost error has not occurred. 1: Arbitration lost error occurred.
1	ACK	**Acknowledge.** Reading this bit returns the receive status of an ACK. Writing this bit to 1 sets the hardware to transmit an ACK.
0	BUSYF	**Busy Flag.** The BUSYF flag is set to 1 by hardware when a Start is generated or detected. This flag is cleared to 0 when hardware generates or detects a Stop or senses a bus-free timeout condition. 0: A transaction is not currently taking place. 1: A transaction is currently taking place.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Register 22.2. I2Cn_CONFIG: Module Configuration

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	TIMEREN	Reserved	TMD		T3RUN	T2RUN	T1RUN	T0RUN	BP		BC		Reserved		T3IEN	T2IEN
Type	RW	R	RW		RW	RW	RW	RW	R		RW		R		RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	T1IEN	T0IEN	ARBLIEN	STAIEN	TXIEN	RXIEN	ACKIEN	STOIEN	Reserved		SCALER					
Type	RW	RW	RW	RW	RW	RW	RW	RW	R		RW					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

I2C0_CONFIG = 0x4000_9010
I2C1_CONFIG = 0x4000_A010
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 22.3. I2Cn_CONFIG Register Bit Descriptions

Bit	Name	Function
31	TIMEREN	**I2C Timer Enable.** This bit can only be set when using the I2C Timer Bytes 0-3 for general purpose use. If I2CEN is set to 1, this bit should always be cleared to 0. If I2CEN is cleared to 0, this bit can be set to 1 to start the timer running. 0: Disable I2C Timer. 1: Enable I2C Timer for general purpose use. This setting should not be used when the I2C module is enabled (I2CEN = 1).
30	Reserved	Must write reset value.
29:28	TMD	**I2C Timer Mode.** This setting only takes effect when using the I2C timer as a stand-alone clock source and the I2C module is disabled (I2CEN = 0). 00: I2C Timer Mode 0: Operate the I2C timer as a single 32-bit timer : Timer Bytes [3 : 2 : 1 : 0]. 01: I2C Timer Mode 1: Operate the I2C timer as two 16-bit timers : Timer Bytes [3 : 2] and Timer Bytes [1 : 0]. 10: I2C Timer Mode 2: Operate the I2C timer as four independent 8-bit timers : Timer Byte 3, Timer Byte 2, Timer Byte 1, and Timer Byte 0. 11: I2C Timer Mode 3: Operate the I2C timer as one 16-bit and two 8-bit timers : Timer Bytes [3 : 2], Timer Byte 1, and Timer Byte 0.

Table 22.3. I2Cn_CONFIG Register Bit Descriptions

Bit	Name	Function
27	T3RUN	**I2C Timer Byte 3 Run.** When the I2C Timer is configured for Mode 2, this bit enables the clock to the 8-bit timer formed by Timer Byte 3. This bit has no effect if the I2C module is enabled (I2CEN = 1). 0: Stop Timer Byte 3. 1: Start Timer Byte 3 running.
26	T2RUN	**I2C Timer Byte 2 Run.** When the I2C Timer is configured for Mode 1 or 3, this bit enables the clock to the 16-bit timer formed by Timer Bytes [3 : 2]. In Mode 2, this bit enables the clock to the 8-bit timer formed by Timer Byte 2. This bit has no effect if the I2C module is enabled (I2CEN = 1). 0: Stop Timer Byte 2. 1: Start Timer Byte 2 running.
25	T1RUN	**I2C Timer Byte 1 Run.** When the I2C Timer is configured for Mode 2 or 3, this bit enables the clock to the 8-bit timer formed by Timer Byte 1. This bit has no effect if the I2C module is enabled (I2CEN = 1). 0: Stop Timer Byte 1. 1: Start Timer Byte 1 running.
24	T0RUN	**I2C Timer Byte 0 Run.** When the I2C Timer is configured for Mode 1 or 3, this bit enables the clock to the 32-bit timer formed by Timer Bytes [3 : 2 : 1 : 0]. In Mode 1, this bit enables the clock to the 16-bit timer formed by Timer Bytes [1 : 0]. In Mode 2 or 3, this bit enables the clock to the 8-bit timer formed by Timer Byte 0. This bit has no effect if the I2C module is enabled (I2CEN = 1). 0: Stop Timer Byte 0. 1: Start Timer Byte 0 running.
23:22	BP	**Transfer Byte Pointer.** This field indicates the byte of the current transfer being sent or received. This setting has no effect when using the I2C module with the DMA.
21:20	BC	**Transfer Byte Count.** This field is the number of bytes to transmit or receive when using the I2C module in software mode (DMAEN = 0). This field has no effect when DMA Mode is enabled.
19:18	Reserved	Must write reset value.
17	T3IEN	**I2C Timer Byte 3 Interrupt Enable.** 0: Disable the I2C Timer Byte 3 and SCL low timeout interrupt. 1: Enable the I2C Timer Byte 3 and SCL low timeout interrupt (T3I).
16	T2IEN	**I2C Timer Byte 2 Interrupt Enable.** 0: Disable the I2C Timer Byte 2 interrupt. 1: Enable the I2C Timer Byte 2 interrupt (T2I).

Table 22.3. I2Cn_CONFIG Register Bit Descriptions

Bit	Name	Function
15	T1IEN	**I2C Timer Byte 1 Interrupt Enable.** 0: Disable the I2C Timer Byte 1 interrupt. 1: Enable the I2C Timer Byte 1 interrupt (T1I).
14	T0IEN	**I2C Timer Byte 0 Interrupt Enable.** 0: Disable the I2C Timer Byte 0 interrupt. 1: Enable the I2C Timer Byte 0 interrupt (T0I).
13	ARBLIEN	**Arbitration Lost Interrupt Enable.** 0: Disable the arbitration lost interrupt. 1: Enable the arbitration lost interrupt (ARBLI).
12	STAIEN	**Start Interrupt Enable.** 0: Disable the start interrupt. 1: Enable the start interrupt (STAI).
11	TXIEN	**Transmit Done Interrupt Enable.** 0: Disable the transmit done interrupt. 1: Enable the transmit done interrupt (TXI).
10	RXIEN	**Receive Done Interrupt Enable.** 0: Disable the receive done interrupt. 1: Enable the receive done interrupt (RXI).
9	ACKIEN	**Acknowledge Interrupt Enable.** 0: Disable the acknowledge interrupt. 1: Enable the acknowledge interrupt (ACKI).
8	STOIEN	**Stop Interrupt Enable.** 0: Disable the stop interrupt. 1: Enable the stop interrupt (STOI).
7:6	Reserved	Must write reset value.
5:0	SCALER	**I2C Clock Scaler.** The I2C module clock frequency is given by the equation: $F_{I2C} = \frac{F_{APB}}{(64 - SCALER)}$

Register 22.3. I2Cn_SADDRESS: Slave Address

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved								ADDRESS							Reserved
Type	R					RW			RW							R
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

I2C0_SADDRESS = 0x4000_9020
I2C1_SADDRESS = 0x4000_A020

Table 22.4. I2Cn_SADDRESS Register Bit Descriptions

Bit	Name	Function
31:8	Reserved	Must write reset value.
7:1	ADDRESS	**Slave Address.** This field contains the 7-bit Slave Address. If slave modes are enabled, the slave will respond with an ACK to any incoming address that matches ADDRESS after being filtered by MASK.
0	Reserved	Must write reset value.

Register 22.4. I2Cn_SMASK: Slave Address Mask

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved								MASK							Reserved
Type	R					RW			RW							R
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

I2C0_SMASK = 0x4000_9030
I2C1_SMASK = 0x4000_A030

Table 22.5. I2Cn_SMASK Register Bit Descriptions

Bit	Name	Function
31:8	Reserved	Must write reset value.
7:1	MASK	**Slave Address Mask.** This field contains the 7-bit Slave Address Mask. If slave modes are enabled, the slave will respond with an ACK to any incoming address that matches ADDRESS after being filtered by MASK. Any bits set to 1 in MASK will result in the corresponding bit in the incoming address comparing to ADDRESS.
0	Reserved	Must write reset value.

Register 22.5. I2Cn_DATA: Data Buffer Access

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	DATA[31:16]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	DATA[15:0]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

I2C0_DATA = 0x4000_9040

I2C1_DATA = 0x4000_A040

Table 22.6. I2Cn_DATA Register Bit Descriptions

Bit	Name	Function
31:0	DATA	**Data.** This field contains the four byte I2C transmit and receive buffer. For each transaction, the least significant byte will be sent or received first.

Register 22.6. I2Cn_TIMER: Timer Data

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	T3								T2							
Type	RW								RW							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	T1								T0							
Type	RW								RW							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

I2C0_TIMER = 0x4000_9050

I2C1_TIMER = 0x4000_A050

Table 22.7. I2Cn_TIMER Register Bit Descriptions

Bit	Name	Function
31:24	T3	**Timer Byte 3.** If the I2C module is enabled (I2CEN = 1), Timer Byte 3 becomes bits [19:12] of the SCL low timeout counter. If the I2C module is disabled (I2CEN = 0), Timer Byte 3 is available for general purpose use and can be set to different modes using the TMD bits.
23:16	T2	**Timer Byte 2.** If the I2C module is enabled (I2CEN = 1), Timer Byte 2 becomes bits [11:4] of the SCL low timeout counter. If the I2C module is disabled (I2CEN = 0), Timer Byte 2 is available for general purpose use and can be set to different modes using the TMD bits.
15:8	T1	**Timer Byte 1.** If the I2C module is enabled (I2CEN = 1), Timer Byte 1 is used as the SCL clock high or low period timer. If the I2C module is disabled (I2CEN = 0), Timer Byte 1 is available for general purpose use and can be set to different modes using the TMD bits.
7:0	T0	**Timer Byte 0.** If the I2C module is enabled (I2CEN = 1), Timer Byte 0 is used as the SDA data setup or hold time period and the SCL free timeout period. If the I2C module is disabled (I2CEN = 0), Timer Byte 0 is available for general purpose use and can be set to different modes using the TMD bits. When used for I2C operations, T0 will automatically be reloaded by hardware from either the T0RL, HOLD, or SETUP fields as necessary.

Register 22.7. I2Cn_TIMERRL: Timer Reload Values

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	T3RL								T2RL							
Type	RW								RW							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	T1RL								T0RL							
Type	RW								RW							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

I2C0_TIMERRL = 0x4000_9060

I2C1_TIMERRL = 0x4000_A060

Table 22.8. I2Cn_TIMERRL Register Bit Descriptions

Bit	Name	Function
31:24	T3RL	**Timer Byte 3 Reload / SCL Low Timeout Bits [19:12].** These bits contain the reload value for I2C Timer Byte 3. Upon a reload event, this value will be latched into the I2C Timer Byte 3 location. When I2C is enabled, these bits are part of the equation for SCL timeout: $T_{SCL_TO} = \frac{2^{20} - (16 \times [T3RL : T2RL])}{F_{I2C}}$
23:16	T2RL	**Timer Byte 2 Reload / SCL Low Timeout Bits [11:4].** These bits contain the reload value for I2C Timer Byte 2. Upon a reload event, this value will be latched into the I2C Timer Byte 2 location. When I2C is enabled, these bits are part of the equation for SCL timeout: $T_{SCL_TO} = \frac{2^{20} - (16 \times [T3RL : T2RL])}{F_{I2C}}$

Table 22.8. I2Cn_TIMERRL Register Bit Descriptions

Bit	Name	Function
15:8	T1RL	**Timer Byte 1 Reload / SCL High Time.** These bits contain the reload value for I2C Timer Byte 1. Upon a reload event, this value will be latched into the I2C Timer Byte 1 location. When I2C is enabled, these bits dictate the SCL high time, according to the following equation: $T_{SCL_HIGH} = \frac{256 - T1RL}{F_{I2C}}$
7:0	T0RL	**Timer Byte 0 Reload / Bus Free Timeout.** These bits contain the reload value for I2C Timer Byte 0. Upon a reload event, this value will be latched into the I2C Timer Byte 0 location. When I2C is enabled, these bits dictate the bus free timeout, according to the following equation: $T_{BUS_FREE} = \frac{256 - T0RL}{F_{I2C}}$

Register 22.8. I2Cn_SCONFIG: SCL Signal Configuration

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved												SCLLTIMER			
Type	R												R			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	SCLL								HOLD				SETUP			
Type	RW								RW				RW			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

I2C0_SCONFIG = 0x4000_9070

I2C1_SCONFIG = 0x4000_A070

Table 22.9. I2Cn_SCONFIG Register Bit Descriptions

Bit	Name	Function
31:20	Reserved	Must write reset value.
19:16	SCLLTIMER	**SCL Low Timeout Bits [3:0].** When the I2C module is enabled (I2CEN = 1), this read-only field (bits [3:0]) combines with I2C Timer Byte 3 (bits [19:12]) and I2C Timer Byte 2 (bits [11:4]) to create a 20-bit SCL low timeout counter.
15:8	SCLL	**SCL Low Time.** This field provides the I2C Timer Byte 1 reload value used to generate the SCL low time. This is given by the equation: $T_{SCL_LOW} = \frac{256 - SCLL}{F_{I2C}}$
7:4	HOLD	**Data Hold Time Extension.** This field provides an alternate I2C Timer Byte 0 reload value to extend the data hold time. The additional hold time is given by the following equation: $T_{HOLD} = \frac{16 - HOLD}{F_{I2C}}$ Note : When HOLD = 0, no additional hold time is added.

Table 22.9. I2Cn_SCONFIG Register Bit Descriptions

Bit	Name	Function
3:0	SETUP	**Data Setup Time Extension.** This field provides an alternate I2C Timer Byte 0 reload value to extend data setup time. This is given by the equation: $T_{SETUP} = \frac{17 - SETUP}{F_{I2C}}$ Note : When SETUP = 0, the setup time is reduced to a single APB clock.

Register 22.9. I2Cn_I2CDMA: DMA Configuration

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	DMAEN	Reserved														
Type	RW	R														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved								DMALEN							
Type	R								RW							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

I2C0_I2CDMA = 0x4000_9080

Table 22.10. I2Cn_I2CDMA Register Bit Descriptions

Bit	Name	Function
31	DMAEN	**DMA Mode Enable.** 0: Disable I2C DMA data requests. 1: Enable I2C DMA data requests.
30:8	Reserved	Must write reset value.
7:0	DMALEN	**DMA Transfer Length.** If DMAEN is set to 1, this field indicates the number of bytes to transfer with the I2C module. When DMA operations are enabled, DMALEN = 0, and TXARM or RXARM is set to 1, the module will transfer or receive data indefinitely until firmware clears TXARM or RXARM.

22.10. I2Cn Register Memory Map

Table 22.11. I2Cn Memory Map

Register Name	I2Cn_DATA	I2Cn_SMASK	I2Cn_SADDRESS	I2Cn_CONFIG	I2Cn_CONTROL	Register Name
ALL Offset	0x40	0x30	0x20	0x10	0x0	ALL Offset
Access Methods	ALL	ALL	ALL	ALL | SET | CLR	ALL | SET | CLR	Access Methods
Bit 31	DATA	Reserved	Reserved	TIMEREN	I2CEN	Bit 31
Bit 30				Reserved	RESET	Bit 30
Bit 29				TMD	GCEN	Bit 29
Bit 28					Reserved	Bit 28
Bit 27				T3RUN	LBACKEN	Bit 27
Bit 26				T2RUN	Reserved	Bit 26
Bit 25				T1RUN	HACKEN	Bit 25
Bit 24				T0RUN	SMINH	Bit 24
Bit 23				BP	DBGMD	Bit 23
Bit 22					FMD	Bit 22
Bit 21				BC	ATXRXEN	Bit 21
Bit 20					SLVAF	Bit 20
Bit 19				Reserved	TXARM	Bit 19
Bit 18					RXARM	Bit 18
Bit 17				T3IEN	T3I	Bit 17
Bit 16				T2IEN	T2I	Bit 16
Bit 15				T1IEN	T1I	Bit 15
Bit 14				T0IEN	T0I	Bit 14
Bit 13				ARBLIEN	ARBLI	Bit 13
Bit 12				STAIEN	STAI	Bit 12
Bit 11				TXIEN	TXI	Bit 11
Bit 10				RXIEN	RXI	Bit 10
Bit 9				ACKIEN	ACKI	Bit 9
Bit 8				STOIEN	STOI	Bit 8
Bit 7		MASK	ADDRESS	Reserved	MSMDF	Bit 7
Bit 6					TXMDF	Bit 6
Bit 5				SCALER	STA	Bit 5
Bit 4					STO	Bit 4
Bit 3					ACKRQF	Bit 3
Bit 2					ARBLF	Bit 2
Bit 1		Reserved	Reserved		ACK	Bit 1
Bit 0					BUSYF	Bit 0

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: I2C0 = 0x4000_9000, I2C1 = 0x4000_A000

Table 22.11. I2Cn Memory Map

Register Name	ALL Offset	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
I2Cn_TIMER	0x50	ALL	T3								T2								T1								T0							
I2Cn_TIMERRL	0x60	ALL	T3RL								T2RL								T1RL								T0RL							
I2Cn_SCONFIG	0x70	ALL	Reserved												SCLLTIMER				SCLL								HOLD				SETUP			
I2Cn_I2CDMA	0x80	ALL	DMAEN	Reserved																							DMALEN							

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: I2C0 = 0x4000_9000, I2C1 = 0x4000_A000

23. Integrated Interchip Sound (I2S0)

This section describes the I2S module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the I2S block, which is used by all device families covered in this document.

23.1. I2S Features

The I2S module includes the following features:

- Master or slave capability.
- Up to 12.288 MHz clock to support 48 kHz sampling with 2 32-bit channels.
- Support for DMA data transfers.
- Support for various data formats.
- Supports Time Division Multiplexing (TDM).

The I2S module receives digital data from an external source over a data line in the standard I^2S, left-justified, right-justified, or time domain multiplexing format, de-serializes the data, and generates requests to transfer the data using the DMA. The module also reads stereo audio samples from the DMA, serializes the data, and sends it out of the chip on a data line in the same standard serial format for digital audio. The I^2S receive interface consists of 3 signals: SCK (bit clock), WS (word select or frame sync), and SD (data input). The block’s transmit interface consists of 3 signals: SCK (bit clock), WS (word select or frame sync) and SD (data output).

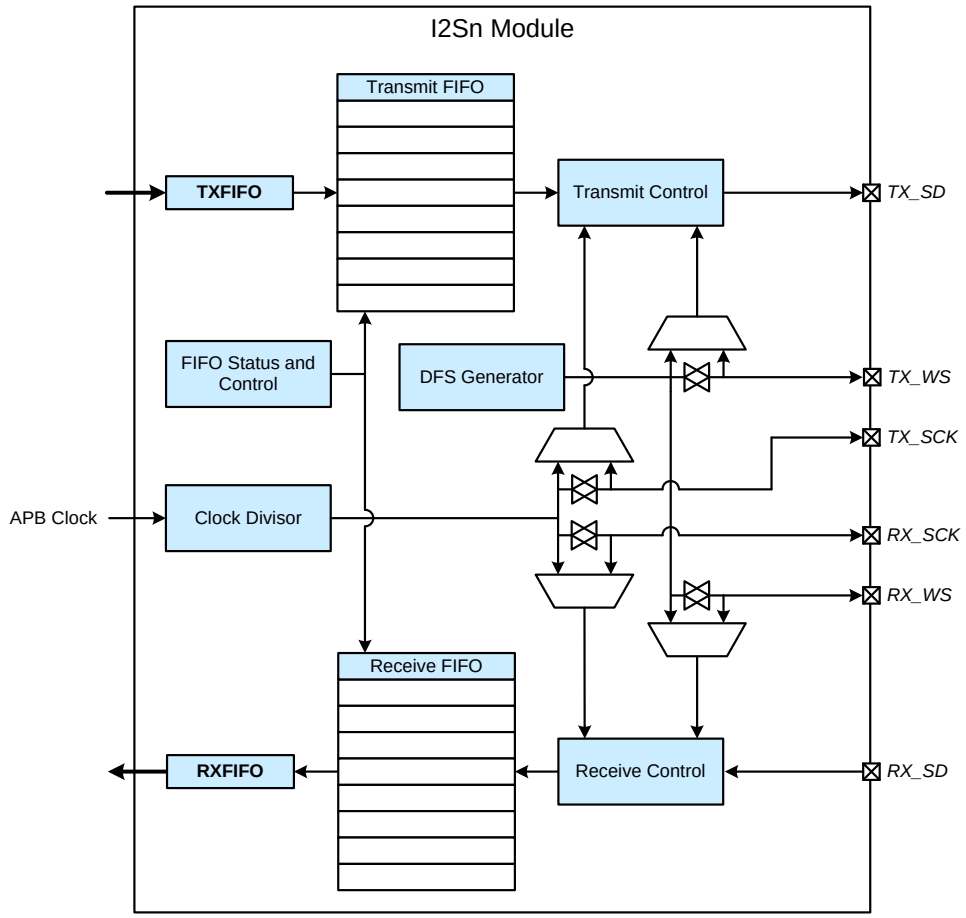


Figure 23.1. I^2S Block Diagram

23.2. Signal Descriptions and Protocol Overview

The I^2S protocol uses three signals: clock (SCK), word select or frame sync (WS), and data (SD). Figure 23.2 shows a typical I^2S transaction.

23.2.1. Clock (SCK)

Each transition of the clock corresponds to a bit of data on the data line. The clock frequency is a multiple of the sampling frequency of the data. For example, the clock rate is 2.8224 MHz for a sampling rate of 44.1 kHz, 32-bits of data per sample, and 2 channels (left and right).

23.2.2. Word Select or Frame Sync (WS)

The word select or frame synchronization signal determines the start of the frame and which channel is active on the bus. The left audio data is sent on the low WS cycle, and the right audio data is sent on the high WS cycle. The WS signal is the same frequency as the sampling rate for non time-division multiplexing systems.

23.2.3. Data (SD)

The data signal is multiplexed between multiple channels and is sent MSB first. The data transfer starts on the second clock cycle after the frame sync transitions. Bits can be sent in varying frame sizes (between 8 and 32), and any unused bits are ignored.

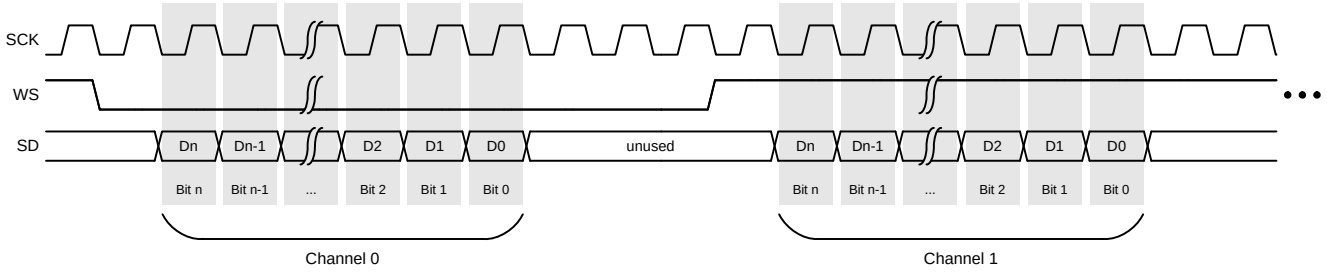


Figure 23.2. I^2S Protocol Transaction

23.3. Clocking

23.3.1. Internal Clock Generator

The I2S module generates a divided version of the APB clock for internal use or to drive on the transmitter or receiver SCK signals. The divider includes a 10-bit integer portion (INTDIV) and an 8-bit fractional portion (FRACDIV). The integer portion creates a 50% duty cycle clock as a direct division of the APB clock. The fractional portion can be used to further divide the input clock and create a non-50% duty cycle average clock frequency. When using the fractional divider, the duty cycle of the clock will have some jitter as some cycles will be stretched by half an APB clock period. Equation 23.1 describes the average I2S clock generator output frequency.

$$F_{I2S} = \frac{F_{APB}}{INTDIV + \frac{FRACDIV}{256}}$$

Equation 23.1. Average Clock Generator Output Frequency

The duty cycle of the clock can be inverted using the DUTYMD bit.

Firmware should set the value of the INTDIV and FRACDIV fields before setting DIVEN to enable the clock divider. Firmware must then write to the CLKUPD bit to update these fields in the clock divider before the changes will take effect. The CDBUSYF flag indicates when the clock divider update completes. Firmware can check the status of the clock divider by checking the CDSTS bit.

23.3.2. Transmitter and Receiver Clocking

The TXCLKSEL and RXCLKSEL bits control whether the transmitter or receiver are clocked from their SCK pins or use the internal clock generator. If changes are made to the TXCLKSEL bit, firmware should poll the transmit clock select ready (TXCLKSELRF) flag to determine when the update completes. Similarly, firmware should poll the receive clock select ready (RXCLKSELRF) flag after changing the RXCLKSEL setting. The SCK frequency will be the same for the transmitter and receiver if both use the internal clock generator. If either the transmitter or receiver uses an incoming SCK signal as its clock, they can operate independently at different frequencies.

The TXCLKEN and RXCLKEN bits enable the clocks to the transmitter and receiver. Firmware should poll the TXCLKENRF or RXCLKENRF flags after changing the corresponding enable bit.

23.4. Clock (SCK) Signal Formatting

The TXSCLKMD and RXSCLKMD bits in the CLKCONTROL register select whether the SCK signals are an input or an output. The transmitter or receiver can invert the SCK signal using the SCLKINVEN bits in the TXCONTROL and RXCONTROL registers regardless of whether the clock originates from the SCK input or the internal clock generator.

To set up the transmitter or receiver to run from an input clock:

1. Set TXSCLKMD or RXSCLKMD to 1 to set SCK as an input.
2. Set SCLKINVEN to the desired value.
3. Set the SCK pin as a digital input in the device port configuration module.
4. Enable the transmitter (TXEN = 1) or receiver (RXEN = 1).

To set up the transmitter or receiver to output a clock:

1. Clear TXSCLKMD or RXSCLKMD to 0 to set SCK as an output.
2. Set SCLKINVEN to the desired value.
3. Set the SCK pin as a digital push-pull output in the device port configuration module.
4. Enable the transmitter (TXEN = 1) or receiver (RXEN = 1).

23.5. Word Select or Frame Sync (WS) Signal Formatting

The FSGEN bit in TXCONTROL enables the internal DFS generator shared between the transmitter and receiver. The DFS generator can be output on the transmitter or receiver WS signals or used to synchronize with an incoming transmitter WS signal using the FSSEN bit. The FSSRCSEL bits select the source of the frame synchronization signal. The FSLOW and FSHIGH fields in the FSDUTY register determine the WS low and high time.

To synchronize the internal DFS generator with an incoming WS signal on the transmitter WS pin, firmware should set FSSEN to 1 before starting the DFS generator (FSGEN = 1). When the generator starts, the hardware waits until a rising edge occurs on the incoming WS signal and will automatically align the internally-generated WS waveform to the incoming signal. The FSLOW and FSHIGH fields must be properly configured so that the generated WS waveform matches the waveform of the incoming WS signal.

The WS signal can be inverted using the FSINVEN bits in TXCONTROL and RXCONTROL regardless of whether the signal is generated internally or input from another device.

23.6. Basic Data (SD) Signal Formatting

When the JSEL, DDIS, and FSDEL bits in TXCONTROL and RXCONTROL and TDMEN in TXMODE and RXMODE are in their default states, the transmitter and receiver are configured to send or receive data in standard I^2S format. The most significant bit of the left channel's sample is sent or captured on the second rising edge of SCK after the falling edge of WS.

23.6.1. Sample Width

The mono bit-width select (MBSEL) fields select the sample width as 8, 9, 16, 24, or 32 bits, and the transmitter and receiver can use different selections. Any unused bit cycles during a frame will be driven to the value that is specified by the FILLSEL field. The transmitter and receiver will always stop sending or capturing the current sample when WS transitions, even if the WS change occurs before all bits of the specified sample size have transferred in the frame.

23.6.2. Right-Justified Format

The JSEL bits select the data justification for the transmitter and receiver. If JSEL is cleared to 0, the data will be sent or received one clock after the WS transition. If JSEL is set to 1, the sample is sent or received after some delay, causing it to be aligned to the right-hand side of the defined slot. The hardware will subtract the sample width (MBSEL) from the slot width (CYCLE) to determine the number of clock cycles to wait after a WS change before sending the first bit of the sample. If the DDIS bit is cleared to 0, then the slot size (CYCLE) must be set as two less than the number of clock cycles per phase of WS to account for the extra cycle of delay which occurs after WS changes. If DDIS is set to 1, then the slot size (CYCLE) must be set as one less than the actual number of clock cycles per phase of WS.

23.6.3. Sample Order

If the ORDER bits are set to 0, the transmitter or receiver expect the left sample first, followed by the right sample. Setting ORDER to 1 swaps the order to the right sample first, followed by the left sample.

23.7. Left-Justified and Longer-Delay Formats

In normal I^2S communications, the first bit of a sample is sent by the transmitter or captured by the receiver on the second rising or falling edge of SCK after a change on WS. By setting the FSDEL fields to a non-zero value, the first bit can be delayed to the cycle specified by FSDEL.

In addition, firmware can set the DDIS bits to have the transmit or receive occur on the first rising or falling edge of SCK after WS changes state. The hardware ignores the FSDEL field when the corresponding DDIS bit is set.

When the transmitter receives WS as an input and is configured to use a left-justified format (DDIS = 1), the transmitter must instantly drive the sample during the same cycle when WS changes state. To do this, the transmitter should be configured to use the internal DFS generator (FSGEN and transmitter FSSRCSEL set to 1). The transmitter hardware is then able to predict when to drive the output sample without relying on the incoming WS signal. The data will be properly aligned if firmware synchronizes the internal DFS generator to the incoming transmitter WS signal by setting FSSEN to 1 before starting the DFS generator. When the generator starts (FSGEN = 1), the hardware waits until a rising edge occurs on the incoming WS signal and will automatically align the generated WS waveform to the incoming signal. Firmware must properly configure the FSLOW and FSHIGH fields so that the generated WS waveform matches the waveform of the incoming WS.

23.8. Time-Division Multiplexing (TDM)

With time-division multiplexing, each WS cycle is split into multiple slots. Different devices on the bus can claim different slots to send or receive samples.

The first slot (slot 0) begins after an optional preamble delay, during which the transmitter will not drive anything onto the data line. The preamble delay is one cycle by default to readily support I^2S format and can be changed up to 256 cycles using the FSDEL fields. The preamble can be decreased to zero cycles by setting DDIS to 1, which may require synchronizing the internal DFS generator with an incoming WS signal as described in “23.5. Word Select or Frame Sync (WS) Signal Formatting” .

Firmware must define the slot width using the CYCLE fields, and the maximum supported slot width is 4096 bits. The START field in the TXMODE register contains the number of the first slot in which the transmitter will drive a sample on SD. Similarly, the START field in the RXMODE register contains the number of the first slot the receiver will capture a sample from the data line. The module can begin driving or capturing samples in any slot from 0 to 255. The SLOTS fields specify the number of consecutive slots in which the module will drive or capture samples. When this field is 0, the module will drive or capture a sample in 1 slot. During the slots when the transmitter is not actively sending a sample, it will either drive 0 (DIMD = 0) or tristate (DIMD = 1) the SD signal.

Since different sources will be driving the shared SD signal, a slot will always end with one device disabling its drivers and another device enabling its drivers at nearly the same time. This can lead to brief conflicts if the drivers overlap for a short time. To prevent this, firmware can define a large enough slot width to accommodate at least 1 cycle of idle time between active slots when no devices are driving the bus. By setting DEDIS to 1, the transmitter will stop driving valid data one cycle before it’s scheduled to release the SD signal. During this cycle, the transmitter will tristate the SD signal if DIMD is set to 1. When using this feature, the mono bit-width (MBSEL) must be defined to be at least 1 bit smaller than the slot width (CYCLE) to accommodate the idle bus cycle.

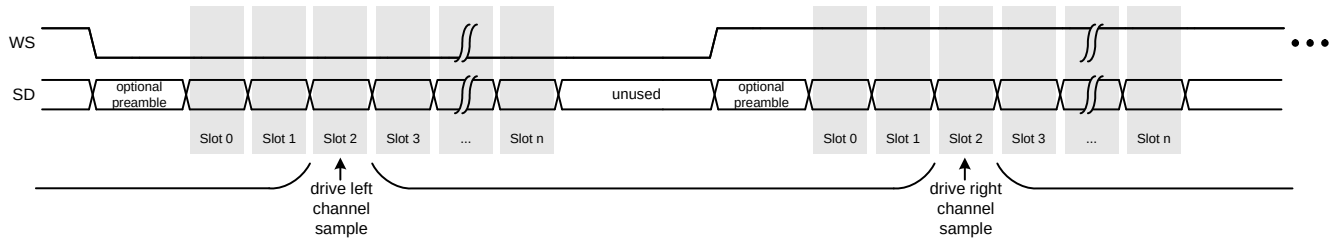


Figure 23.3. Time Division Multiplexing Example (START = 2, SLOTS = 0)

23.9. Transmitter and Receiver

The I2S module transmitter and receiver operate independently. Firmware can enable the I2S module transmitter by setting TXEN to 1. The transmitter has an 8x32-bit word FIFO that can be filled by writing left-justified data words to the TXFIFO register. Multiple samples can be packed into a single word, depending on the enabled data formatting options. Figure 23.4 shows several transmit FIFO formatting examples.

Transmit or Receive FIFO	
L, 24-bit	
R, 24-bit	
L, 24-bit	
R, 24-bit	
L, 24-bit	
R, 24-bit	
L, 24-bit	
R, 24-bit	

Transmit or Receive FIFO	
L, 16-bit	R, 16-bit
L, 16-bit	R, 16-bit
L, 16-bit	R, 16-bit
L, 16-bit	R, 16-bit
L, 16-bit	R, 16-bit
L, 16-bit	R, 16-bit
L, 16-bit	R, 16-bit
L, 16-bit	R, 16-bit

Transmit or Receive FIFO			
L, 9-bit		R, 9-bit	
L, 9-bit		R, 9-bit	
L, 9-bit		R, 9-bit	
L, 9-bit		R, 9-bit	
L, 9-bit		R, 9-bit	
L, 9-bit		R, 9-bit	
L, 9-bit		R, 9-bit	
L, 9-bit		R, 9-bit	

Transmit or Receive FIFO			
L, 8-bit	R, 8-bit	L, 8-bit	R, 8-bit
L, 8-bit	R, 8-bit	L, 8-bit	R, 8-bit
L, 8-bit	R, 8-bit	L, 8-bit	R, 8-bit
L, 8-bit	R, 8-bit	L, 8-bit	R, 8-bit
L, 8-bit	R, 8-bit	L, 8-bit	R, 8-bit
L, 8-bit	R, 8-bit	L, 8-bit	R, 8-bit
L, 8-bit	R, 8-bit	L, 8-bit	R, 8-bit
L, 8-bit	R, 8-bit	L, 8-bit	R, 8-bit

Figure 23.4. FIFO Formatting Examples

The TXFIFONUM field contains the number of words in the transmitter FIFO. The transmit FIFO watermark (TXFIFOWM) allows firmware to receive a notification when more data can be placed in the FIFO. Hardware sets the transmit FIFO low watermark (TXLWMI) flag and can generate an interrupt when the number of entries in the FIFO is less than or equal to the level set by the watermark (TXFIFONUM = TXFIFOWM).

Similar to the transmitter, the receiver is enabled by setting the RXEN bit. The receiver FIFO can also contain up to 8 words, and firmware can read the FIFO by reading data words from the RXFIFO register. The hardware will pack the input data in left-justified bytes, half-words, or words depending on the selected data formatting options.

The RXFIFONUM field describes the number of words in the receiver FIFO. Hardware sets the receive FIFO high watermark (RXHWMI) flag when the number of words is greater than or equal to the receive FIFO watermark field (RXFIFONUM = RXFIFOWM). An interrupt can also be generated when this watermark flag is set.

In the event of an error, the FIFOs can be emptied using the transmit FIFO flush (TXFIFOFL) and receive FIFO flush (RXFIFOFL) bits.

23.10. Interrupts and Flags

The transmit FIFO low watermark (TXLWMI) flag is set by hardware when the number of words in the transmit FIFO is less than or equal to the TXFIFOWM field. This flag can also generate an interrupt, if enabled (TXLWMIEN = 1).

Hardware sets the receive FIFO high watermark (RXHWMI) flag when the number of words in the receive FIFO is greater than or equal to the RXFIFOWM field. Firmware can enable this interrupt by setting RXHWMIEN to 1.

The I2S module also includes two error interrupts. Hardware sets the transmit underrun error (TXUFI) flag when the transmitter attempts to send a sample when the transmit FIFO is empty. The receive overflow error (RXOFI) occurs when the receiver attempts to store more data in the receive FIFO when the FIFO is full. These interrupts can be enabled by firmware by setting TXUFIEN or RXOFIEN to 1, respectively.

23.11. Debug Mode

Firmware can set the TXDBGHEN or RXDBGHEN bits to force the I2S transmitter or receiver to halt on a debug breakpoint. Clearing these bits forces the module to continue operating while the core halts in debug mode.

If TXDBGMD is cleared to 0, the clock to the I2S transmitter is active in debug mode. If TXDBGMD is set to 1, the clock divider keeps running, and the clock to the transmitter will be disabled after two samples are queued and ready to be sent.

If RXDBGMD is cleared to 0, the clock to the I2S receiver is active in debug mode. When RXDBGMD is set to 1, the clock divider keeps running indefinitely, and the hardware disables the receiver clock after the receiver captures two samples.

23.12. Module Reset

To reset the I2S module:

1. Disable the clock divider (DIVEN = 0).
2. Set the CLKUPD bit to update the divider.
3. Poll on CDBUSYF to verify the divider is disabled.
4. Clear the clock divider duty mode (DUTYMD) bit to 0.
5. Set the RESET bit.
6. Poll on RESET to wait for the reset operation to complete.
7. Re-initialize the module.

Following this procedure ensures the transmitter and receiver will not output any glitches on the SCK output signals.

23.13. DMA Usage and Configuration

The I2S transmitter and receiver have separate DMA data requests and can be assigned to two different DMA channels. The I2S module supports burst transfers only, but these burst transfers may be of variable size by setting the transmitter or receiver burst mode.

When DMA is enabled to the transmitter (TXDMAEN = 1), the I2S transmitter will automatically request data from the assigned DMA channel whenever space is available in the transmit FIFO. If burst mode is cleared to 0, (TXDMABMD = 0), the hardware will request data when at least one word is empty in the transmit FIFO. If TXDMABMD is set to 1, the transmitter will request data when at least four words are empty.

When DMA is enabled for the receiver (RXDMAEN = 1), the receiver will automatically request data transfers from the DMA channel when data is present in the receive FIFO. When burst mode is cleared to 0 (RXDMABMD = 0), the receiver will request a data transfer when at least one word is present in the FIFO. If burst mode is set to 1 (RXDMABMD = 1), the hardware will request a data transfer when data occupies at least four words of the FIFO.

Figure 23.5 shows the I2S DMA configuration.

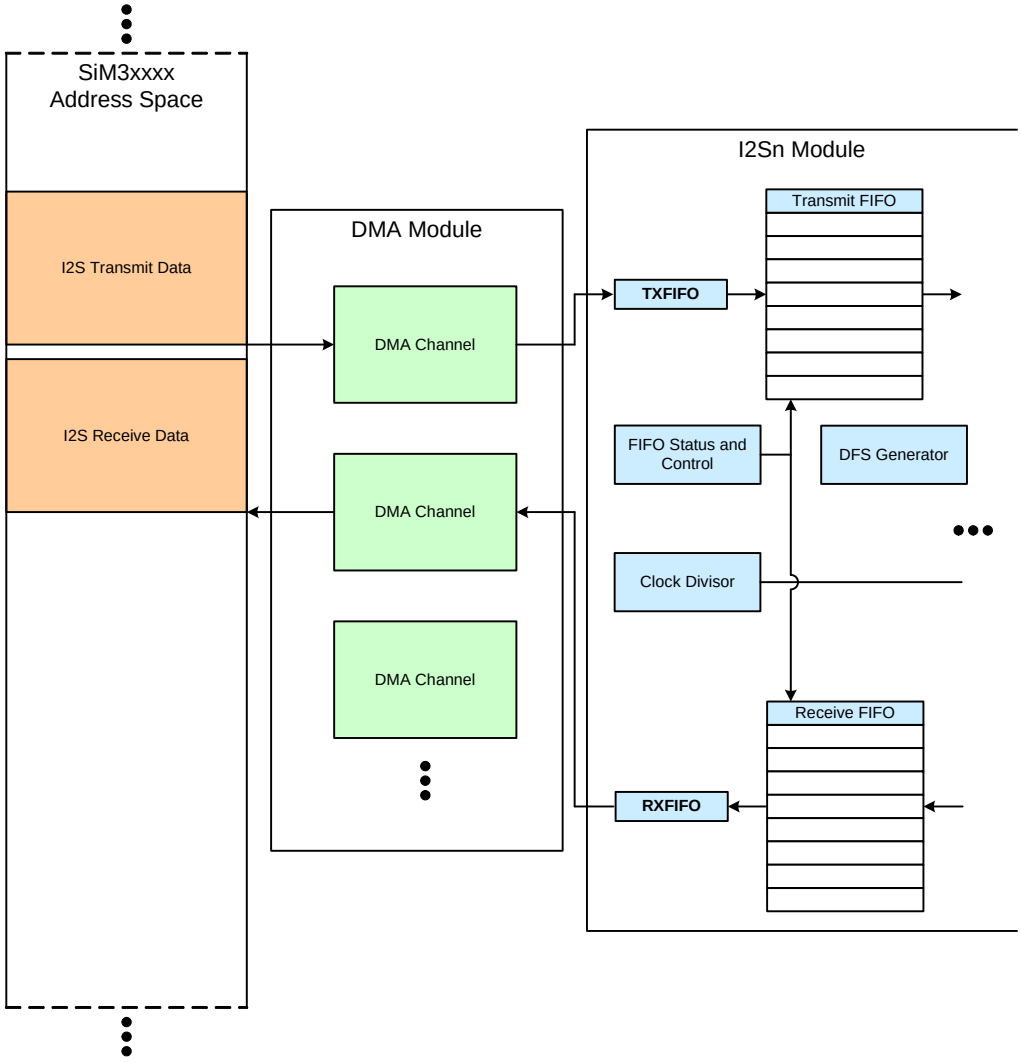


Figure 23.5. I2S DMA Configuration

23.14. I2S0 Registers

This section contains the detailed register descriptions for I2S0 registers.

Register 23.1. I2S0_TXCONTROL: Transmit Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved		TXEN	Reserved		MBSEL			Reserved	ORDER	SCLKINVEN	FSINVEN	Reserved		JSEL	FILLSEL[1]
Type	R		RW	R		RW			R	RW	RW	RW	R		RW	RW
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	FILLSEL[0]	FSSRCSEL	FSDEL								DDIS	Reserved			FSSEN	FSGEN
Type	RW	RW	RW								RW	R			RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

I2S0_TXCONTROL = 0x4003_A000

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 23.1. I2S0_TXCONTROL Register Bit Descriptions

Bit	Name	Function
31:30	Reserved	Must write reset value.
29	TXEN	**Transmitter Enable.** 0: Disable the I2S transmitter. 1: Enable the I2S transmitter.
28:27	Reserved	Must write reset value.
26:24	MBSEL	**Transmit Mono Bit-Width Select.** This field specifies the number of bits per mono sample. 000: 8 bits are sent per mono sample. 001: 9 bits are sent per mono sample. 010: 16 bits are sent per mono sample. 011: 24 bits are sent per mono sample. 100: 32 bits are sent per mono sample. 101-111: Reserved.
23	Reserved	Must write reset value.

Table 23.1. I2S0_TXCONTROL Register Bit Descriptions

Bit	Name	Function
22	ORDER	**Transmit Order.** This bit chooses which audio channel is sent as the first mono sample and second mono sample in various formats. 0: Left sample transmitted first, right sample transmitted second. Use this setting for I2S format. 1: Right sample transmitted first, left sample transmitted second.
21	SCLKINVEN	**Transmit SCK Inversion Enable.** 0: Do not invert the transmitter bit clock. 1: Invert the transmitter bit clock.
20	FSINVEN	**Transmit WS Inversion Enable.** 0: Don't invert the WS signal. Use this setting for I2S format. 1: Invert the WS signal.
19:18	Reserved	Must write reset value.
17	JSEL	**Transmit Data Justification Select.** 0: Use left-justified or I2S-style formats. 1: Use right-justified format.
16:15	FILLSEL	**Transmit Data Fill Select.** Ths field specifies the type of data to send during unused bit cycles. 00: Send zeros during unused bit cycles. 01: Send ones during unused bit cycles. 10: Send the sign bit of the current sample (MSB-first format) or last sample (LSB-first format) during unused bit cycles. 11: Send pseudo-random data generated by an 8-bit LFSR during unused bit cycles.
14	FSSRCSEL	**Transmit Frame Sync Source Select.** Select the source for the word select or frame sync (WS) input to the block. 0: The word select or frame sync is input from the WS pin. 1: The word select or frame sync is input from the internal DFS generator.
13:6	FSDEL	**Transmit Initial Phase Delay.** This field specifies the number of cycles of initial phase delay from the frame sync edge to the first bit of the sample (non-TDM formats) or to start the first slot (TDM formats).
5	DDIS	**Transmit Delay Disable.** Setting this bit to 1 disables the data delay after the frame sync. 0: The first data bit is sent on the second or later rising edge of SCK after WS changes. 1: The first data bit is sent on the first rising edge of SCK after WS changes.
4:2	Reserved	Must write reset value.
1	FSSEN	**DFS Synchronize Enable.** 0: The internal DFS generator starts immediately when FSGEN is set to 1. 1: Synchronize the rising edge of the internally generated WS signal from the DFS generator to the rising edge of the external WS input signal.

Table 23.1. I2S0_TXCONTROL Register Bit Descriptions

Bit	Name	Function
0	FSGEN	**DFS Generator Enable.** 0: Disable the internal DFS generator. 1: Enable the internal DFS generator.

Register 23.2. I2S0_TXMODE: Transmit Mode

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved				TDMEN	DIMD	DEDIS	SLOTS					START[7:4]			
Type	R				RW	RW	RW	RW					RW			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	START[3:0]				CYCLE											
Type	RW				RW											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

I2S0_TXMODE = 0x4003_A010

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 23.2. I2S0_TXMODE Register Bit Descriptions

Bit	Name	Function
31:28	Reserved	Must write reset value.
27	TDMEN	**Transmit Time Division Multiplexing Enable.** 0: Disable the time division multiplexing (TDM) feature. 1: Enable the time division multiplexing (TDM) feature.
26	DIMD	**Transmit Drive Inactive Mode.** Defines the state to drive on the data output pins during non-active slots. 0: Drive zero on the data output pin during non-active slots. 1: Don't drive the data output pin. The data output pin is tristated.
25	DEDIS	**Transmit Drive Early Disable.** 0: Drive the output during every cycle of the transmitter's assigned slot(s), including the last clock cycle. 1: Drive the output for every cycle of the transmitter's assigned slot(s), except for the last clock cycle of the last slot.
24:20	SLOTS	**Transmit Drive Select.** The number of consecutive TDM slots on which to drive data is SLOTS + 1.
19:12	START	**Transmit Start Control.** This field is the first TDM slot number in which digital output will start driving data.
11:0	CYCLE	**Transmit Clock Cycle Select.** If JSEL is set to 1 for the transmitter, this field should be set with the number of slots in the frame. If DDIS is cleared to 0, the number of slots is CYCLE - 1. If DDIS is also set to 1, the number of slots per right-justified field is CYCLE - 2.

Register 23.3. I2S0_FSDUTY: Frame Sync Duty Cycle

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	FSHIGH															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	FSLOW															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

I2S0_FSDUTY = 0x4003_A020

Table 23.3. I2S0_FSDUTY Register Bit Descriptions

Bit	Name	Function
31:16	FSHIGH	**Frame Sync High Time.** The number of bit clock cycles for the high phase of the frame sync generator is equal to FSHIGH + 1.
15:0	FSLOW	**Frame Sync Low Time.** The number of bit clock cycles for the low phase of the frame sync generator is equal to FSLOW + 1.

Register 23.4. I2S0_RXCONTROL: Receive Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved										RXEN	FSSRCSEL	Reserved		MBSEL[2:1]	
Type	R										RW	RW	R		RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	MBSEL[0]	Reserved	ORDER	SCLKINVEN	FSINVEN	Reserved	DDIS	JSEL	FSDEL							
Type	RW	R	RW	RW	RW	R	RW	RW	RW							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

I2S0_RXCONTROL = 0x4003_A030

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 23.4. I2S0_RXCONTROL Register Bit Descriptions

Bit	Name	Function
31:22	Reserved	Must write reset value.
21	RXEN	**Receive Enable.** 0: Disable the I2S receiver. 1: Enable the I2S receiver.
20	FSSRCSEL	**Receive Frame Sync Source Select.** Select the source for the word select or frame sync (WS) input to the block. 0: The word select or frame sync is input from the WS pin. 1: The word select or frame sync is input from the internal DFS generator.
19:18	Reserved	Must write reset value.
17:15	MBSEL	**Receive Mono Bit-Width Select.** This field specifies the number of bits per mono sample. 000: 8 bits are received per mono sample. 001: 9 bits are received per mono sample. 010: 16 bits are received per mono sample. 011: 24 bits are received per mono sample. 100: 32 bits are received per mono sample. 101-111: Reserved.

Table 23.4. I2S0_RXCONTROL Register Bit Descriptions

Bit	Name	Function
14	Reserved	Must write reset value.
13	ORDER	**Receive Order.** This bit chooses which audio channel is received as the first mono sample and second mono sample in various formats. 0: Left sample received first, right sample received second. Use this setting for I2S format. 1: Right sample received first, left sample received second.
12	SCLKINVEN	**Receive SCK Inversion Enable.** 0: Do not invert the receiver bit clock. 1: Invert the receiver bit clock.
11	FSINVEN	**Receive WS Inversion Enable.** 0: Don't invert the WS signal. Use this setting for I2S format. 1: Invert the WS signal.
10	Reserved	Must write reset value.
9	DDIS	**Receive Delay Disable.** Setting this bit to 1 disables the data delay after the frame sync. 0: The first data bit is captured on the second or later rising edge of SCK after WS changes. 1: The first data bit is captured by the receiver on the first rising edge of SCK after WS changes.
8	JSEL	**Receive Data Justification.** 0: Use left-justified or I2S-style formats. 1: Use right-justified format.
7:0	FSDEL	**Receive Initial Phase Delay.** This field specifies the number of cycles of initial phase delay from the frame sync edge to the first bit of the sample (non-TDM formats) or to start the first slot (TDM formats).

Register 23.5. I2S0_RXMODE: Receive Mode

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved				TDMEN	Reserved	SLOTS						START[7:4]			
Type	R				RW	R	RW						RW			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	START[3:0]				CYCLE											
Type	RW				RW											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

I2S0_RXMODE = 0x4003_A040

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 23.5. I2S0_RXMODE Register Bit Descriptions

Bit	Name	Function
31:28	Reserved	Must write reset value.
27	TDMEN	**Receive Time Division Multiplexing Enable.** 0: Disable the time division multiplexing (TDM) feature. 1: Enable the time division multiplexing (TDM) feature.
26	Reserved	Must write reset value.
25:20	SLOTS	**Receive Drive Select.** The number of consecutive TDM slots on which to accept data is equal to SLOTS + 1.
19:12	START	**Receive Start Control.** This field is the first TDM slot number in which digital input will start receiving data.
11:0	CYCLE	**Receive Clock Cycle Select.** If JSEL is set to 1 for the receiver, this field should be set with the number of slots in the frame. If DDIS is cleared to 0, the number of slots is CYCLE - 1. If DDIS is also set to 1, the number of slots per right-justified field is CYCLE - 2.

Register 23.6. I2S0_CLKCONTROL: Clock Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved				TXSCLKMD	RXSCLKMD	TXCLKEN	RXCLKEN	RESET	RXCLKSEL	TXCLKSEL	DIVEN	CLKUPD	DUTYMD	FRACDIV[7:6]	
Type	R				RW	RW	RW	RW	W	RW	RW	RW	W	RW	RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	FRACDIV[5:0]						INTDIV									
Type	RW						RW									
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

I2S0_CLKCONTROL = 0x4003_A050

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 23.6. I2S0_CLKCONTROL Register Bit Descriptions

Bit	Name	Function
31:28	Reserved	Must write reset value.
27	TXSCLKMD	**Transmit SCK Mode.** 0: The I2S transmitter SCK signal is an output. 1: The I2S transmitter SCK signal is an input.
26	RXSCLKMD	**Receive SCK Mode.** 0: The I2S receiver SCK signal is an output. 1: The I2S receiver SCK signal is an input.
25	TXCLKEN	**Transmit Clock Enable.** When changes are made to this bit, software should check the TXCLKENRF bit status to determine when the transmitter is ready to send data. 0: Disable the I2S transmitter clock. 1: Enable the I2S transmitter clock.
24	RXCLKEN	**Receive Clock Enable.** When changes are made to this bit, software should check the RXCLKENRF bit status to determine when the receiver is ready to accept data. 0: Disable the I2S receiver clock. 1: Enable the I2S receiver clock.
23	RESET	**I2S Module Reset.** Writing a 1 to this bit resets the I2S block. This bit is self-clearing when the reset is complete and does not need to be cleared by software.

Table 23.6. I2S0_CLKCONTROL Register Bit Descriptions

Bit	Name	Function
22	RXCLKSEL	**Receive Clock Select.** When changes are made to this bit, software should check the RXCLKSELRF bit status to determine when the receiver is ready to accept data. 0: The I2S receiver is clocked from the internal clock divider. 1: The I2S receiver is clocked from the SCK pin.
21	TXCLKSEL	**Transmit Clock Select.** When changes are made to this bit, software should check the TXCLKSELRF bit status to determine when the transmitter is ready to send data. 0: The I2S transmitter is clocked from the internal clock divider. 1: The I2S transmitter is clocked from the SCK pin.
20	DIVEN	**Clock Divider Enable.** 0: Disable the clock divider. 1: Enable the clock divider.
19	CLKUPD	**Clock Divider Update.** When this bit is set to 1, the clock divider will update with the new values of INTDIV, FRACDIV, and DIVEN.
18	DUTYMD	**Duty Cycle Adjustment Mode.** 0: When the division is fractional, the clock high time will be greater than 50% (by half of the source clock period). 1: When the division is fractional, the clock low time will be greater than 50% (by half of the source clock period).
17:10	FRACDIV	**Clock Divider Fractional Value.** This field is the fractional portion of the clock division ratio.
9:0	INTDIV	**Clock Divider Integer Value.** This field is the integer portion of the clock divider.

Register 23.7. I2S0_TXFIFO: Transmit Data FIFO

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	TXFIFO[31:16]															
Type	W															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	TXFIFO[15:0]															
Type	W															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

I2S0_TXFIFO = 0x4003_A060

Table 23.7. I2S0_TXFIFO Register Bit Descriptions

Bit	Name	Function
31:0	TXFIFO	**Transmit Data FIFO.** A write to this register adds a word to the transmitter FIFO. This field should always be written as a 32-bit word.

Notes:

1. Reads of this register modify the state of hardware. Debug logic should take care when reading this register.
2. The access methods for this register are restricted. Do not use half-word or byte access methods on this register.

Register 23.8. I2S0_RXFIFO: Receive Data FIFO

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	RXFIFO[31:16]															
Type	R															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	RXFIFO[15:0]															
Type	R															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Register ALL Access Address

I2S0_RXFIFO = 0x4003_A070

Table 23.8. I2S0_RXFIFO Register Bit Descriptions

Bit	Name	Function
31:0	RXFIFO	**Receive Data FIFO.** A read from this register removes a word from the receiver FIFO. This field should always be read as a 32-bit word.

Notes:

1. Reads of this register modify the state of hardware. Debug logic should take care when reading this register.
2. The access methods for this register are restricted. Do not use half-word or byte access methods on this register.

Register 23.9. I2S0_FIFOSTATUS: FIFO Status

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved												RXFIFONUM			
Type	R												R			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved												TXFIFONUM			
Type	R												R			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

I2S0_FIFOSTATUS = 0x4003_A080

Table 23.9. I2S0_FIFOSTATUS Register Bit Descriptions

Bit	Name	Function
31:20	Reserved	Must write reset value.
19:16	RXFIFONUM	**Receive FIFO Status.** This field indicates the number of 32-bit words in the I2S receiver FIFO.
15:4	Reserved	Must write reset value.
3:0	TXFIFONUM	**Transmit FIFO Status.** This field indicates the number of 32-bit words in the I2S transmitter FIFO.

Register 23.10. I2S0_FIFOCONTROL: FIFO Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved									RXFIFOFL	TXFIFOFL	Reserved	RXFIFOWM			
Type	R									W	W	R	RW			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved												TXFIFOWM			
Type	R												RW			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

I2S0_FIFOCONTROL = 0x4003_A090

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 23.10. I2S0_FIFOCONTROL Register Bit Descriptions

Bit	Name	Function
31:23	Reserved	Must write reset value.
22	RXFIFOFL	**Receive FIFO Flush.** Writing a 1 to this bit resets the receive FIFO.
21	TXFIFOFL	**Transmit FIFO Flush.** Writing a 1 to this bit resets the transmit FIFO.
20	Reserved	Must write reset value.
19:16	RXFIFOWM	**Receive FIFO High Watermark.** Hardware generates an interrupt when the number of words remaining in the receive FIFO (RXFIFONUM) is greater than or equal to the receive FIFO high watermark (RXFIFOWM).
15:4	Reserved	Must write reset value.
3:0	TXFIFOWM	**Transmit FIFO Low Watermark.** Hardware generates an interrupt when the number of words remaining in the transmit FIFO (TXFIFONUM) is less than or equal to the transmit FIFO low watermark (TXFIFOWM).

Register 23.11. I2S0_INTCONTROL: Interrupt Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved												RXHWMIEN	TXLWMIEN	RXOFIEN	TXUFIEN
Type	R												RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

I2S0_INTCONTROL = 0x4003_A0A0

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 23.11. I2S0_INTCONTROL Register Bit Descriptions

Bit	Name	Function
31:4	Reserved	Must write reset value.
3	RXHWMIEN	**Receive FIFO High Watermark Interrupt Enable.** 0: Disable the receive FIFO high watermark interrupt. 1: Enable the receive FIFO high watermark interrupt.
2	TXLWMIEN	**Transmit FIFO Low Watermark Interrupt Enable.** 0: Disable the transmit FIFO low watermark interrupt. 1: Enable the transmit FIFO low watermark interrupt.
1	RXOFIEN	**Receive Overflow Interrupt Enable.** 0: Disable the receive overflow interrupt. 1: Enable the receive overflow interrupt.
0	TXUFIEN	**Transmit Underflow Interrupt Enable.** 0: Disable the transmit underflow interrupt. 1: Enable the transmit underflow interrupt.

Register 23.12. I2S0_STATUS: Module Status

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved						RXCLKENRF	TXCLKENRF	RXCLKSELRF	TXCLKSELRF	CDSTS	CDBUSYF	RXHWMI	TXLWMI	RXOFI	TXUFI
Type	R						R	R	R	R	R	R	R	R	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	1	0	1	1	0	0

Register ALL Access Address

I2S0_STATUS = 0x4003_A0B0

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 23.12. I2S0_STATUS Register Bit Descriptions

Bit	Name	Function
31:10	Reserved	Must write reset value.
9	RXCLKENRF	**Receive Clock Enable Ready Flag.** When changes are made to the RXCLKEN bit, software should check this bit to determine when the receiver is ready to accept data. 0: The receive clock is not synchronized. 1: The receive clock is synchronized and the receiver is ready to accept data.
8	TXCLKENRF	**Transmit Clock Enable Ready Flag.** When changes are made to the TXCLKEN bit, software should check this bit to determine when the transmitter is ready to send data. 0: The transmit clock is not synchronized. 1: The transmit clock is synchronized and the transmitter is ready to send data.
7	RXCLKSELRF	**Receive Clock Select Ready Flag.** When changes are made to the RXCLKSEL bit, software should check this bit to determine when the receiver is ready to accept data. 0: The receive clock is not synchronized. 1: The receive clock is synchronized and the receiver is ready to accept data.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 23.12. I2S0_STATUS Register Bit Descriptions

Bit	Name	Function
6	TXCLKSELRF	**Transmit Clock Select Ready Flag.** When changes are made to the TXCLKSEL bit, software should check this bit to determine when the transmitter is ready to send data. 0: The transmit clock is not synchronized. 1: The transmit clock is synchronized and the transmitter is ready to send data.
5	CDSTS	**Clock Divider Counter Status.** 0: Divided clock output is running. 1: Divided clock output is halted.
4	CDBUSYF	**Clock Divider Busy Flag.** 0: The divider is not busy and an update is not pending. 1: The divider is busy and an update is pending.
3	RXHWMI	**Receive FIFO High Watermark Interrupt Flag.** 0: Receive FIFO level is below the high watermark. 1: Receive FIFO level is at or above the high watermark.
2	TXLWMI	**Transmit FIFO Low Watermark Interrupt Flag.** 0: Transmit FIFO level is above the low watermark. 1: Transmit FIFO level is at or below the low watermark.
1	RXOFI	**Receive Overflow Interrupt Flag.** 0: A receive overflow has not occurred. 1: A receive overflow occurred.
0	TXUFI	**Transmit Underflow Interrupt Flag.** 0: A transmit underflow has not occurred. 1: A transmit underflow occurred.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Register 23.13. I2S0_DMACONTROL: DMA Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved												RXDMABMD	TXDMABMD	RXDMAEN	TXDMAEN
Type	R												RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

I2S0_DMACONTROL = 0x4003_A0C0

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 23.13. I2S0_DMACONTROL Register Bit Descriptions

Bit	Name	Function
31:4	Reserved	Must write reset value.
3	RXDMABMD	**Receive DMA Burst Mode.** 0: The receiver receives one word at a time. Whenever there is at least one word in the receive FIFO, a single word burst DMA request is generated. 1: The receiver receives four words at a time. Whenever the FIFO depth rises above three, a DMA burst request is generated for four words.
2	TXDMABMD	**Transmit DMA Burst Mode.** 0: The transmitter transmits one word at a time. Whenever there is any room in the transmit FIFO, a single word burst DMA data request is generated. 1: The transmitter transmits four words at a time. Whenever the FIFO depth drops below five, a DMA burst request is generated for four words.
1	RXDMAEN	**Receive DMA Enable.** When this bit is set, the receiver will send a word from the receive FIFO to the assigned DMA channel based on the receive FIFO status. 0: Disable receiver DMA data transfer requests. 1: Enable receiver DMA data transfer requests.
0	TXDMAEN	**Transmit DMA Enable.** When this bit is set, the transmitter will request a word from the assigned DMA channel and place it in the transmitter FIFO based on the transmit FIFO status. 0: Disable transmitter DMA data requests. 1: Enable transmitter DMA data requests.

Register 23.14. I2S0_DBGCONTROL: Debug Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved												RXDBGMD	TXDBGMD	RXDBGHEN	TXDBGHEN
Type	R												RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

I2S0_DBGCONTROL = 0x4003_A0D0

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 23.14. I2S0_DBGCONTROL Register Bit Descriptions

Bit	Name	Function
31:4	Reserved	Must write reset value.
3	RXDBGMD	**I2S Receive Debug Mode.** 0: The clock to the I2S receiver is active in debug mode. 1: The clock to the I2S receiver is not active in debug mode. The clock divider keeps running and the clock will be disabled when two samples are captured in the receiver.
2	TXDBGMD	**I2S Transmit Debug Mode.** 0: The clock to the I2S transmitter is active in debug mode. 1: The clock to the I2S transmitter is not active in debug mode. The clock divider keeps running and the clock will be disabled when two samples are ready to be sent by the transmitter.
1	RXDBGHEN	**I2S Receive DMA Debug Halt Enable.** 0: Receive DMA requests continue while the core is debug mode. 1: Receive DMA requests stop while the core is debug mode.
0	TXDBGHEN	**I2S Transmit DMA Debug Halt Enable.** 0: Transmit DMA requests continue while the core is debug mode. 1: Transmit DMA requests stop while the core is debug mode.

23.15. I2S0 Register Memory Map

Table 23.15. I2S0 Memory Map

Register Name	I2S0_TXCONTROL	I2S0_TXMODE	I2S0_FSDUTY	I2S0_RXCONTROL	I2S0_RXMODE
ALL Address	0x4003_A000	0x4003_A010	0x4003_A020	0x4003_A030	0x4003_A040
Access Methods	ALL | SET | CLR	ALL | SET | CLR	ALL	ALL | SET | CLR	ALL | SET | CLR
Bit 31	Reserved	Reserved	FSHIGH	Reserved	Reserved
Bit 30					
Bit 29	TXEN				TDMEN
Bit 28	Reserved				Reserved
Bit 27		TDMEN			
Bit 26	MBSEL	DIMD			
Bit 25		DEDIS			SLOTS
Bit 24		SLOTS			
Bit 23	Reserved			RXEN	
Bit 22	ORDER			FSSRCSEL	
Bit 21	SCLKINVEN			Reserved	START
Bit 20	FSINVEN			MBSEL	
Bit 19	Reserved	START			
Bit 18				Reserved	
Bit 17	JSEL			ORDER	
Bit 16	FILLSEL			SCLKINVEN	
Bit 15				FSINVEN	
Bit 14	FSSRCSEL			Reserved	CYCLE
Bit 13	FSDEL		FSLOW	DDIS	
Bit 12				JSEL	
Bit 11		CYCLE		FSDEL	
Bit 10					
Bit 9					
Bit 8					
Bit 7					
Bit 6					
Bit 5	DDIS				
Bit 4	Reserved				
Bit 3					
Bit 2					
Bit 1	FSSEN				
Bit 0	FSGEN				

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

Table 23.15. I2S0 Memory Map

Register Name	I2S0_FIFOCONTROL	I2S0_FIFOSTATUS	I2S0_RXFIFO	I2S0_TXFIFO	I2S0_CLKCONTROL
ALL Address	0x4003_A090	0x4003_A080	0x4003_A070	0x4003_A060	0x4003_A050
Access Methods	ALL | SET | CLR	ALL	ALL	ALL	ALL | SET | CLR
Bit 31	Reserved	Reserved	RXFIFO	TXFIFO	Reserved
Bit 30					
Bit 29					
Bit 28					
Bit 27					TXSCLKMD
Bit 26					RXSCLKMD
Bit 25					TXCLKEN
Bit 24					RXCLKEN
Bit 23					RESET
Bit 22	RXFIFOFL				RXCLKSEL
Bit 21	TXFIFOFL				TXCLKSEL
Bit 20	Reserved				DIVEN
Bit 19	RXFIFOWM	RXFIFONUM			CLKUPD
Bit 18					DUTYMD
Bit 17	Reserved	Reserved			FRACDIV
Bit 16					
Bit 15					
Bit 14					
Bit 13					
Bit 12					
Bit 11					
Bit 10					
Bit 9	TXFIFOWM	TXFIFONUM			INTDIV
Bit 8					
Bit 7					
Bit 6					
Bit 5					
Bit 4					
Bit 3					
Bit 2					
Bit 1					
Bit 0					

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

Table 23.15. I2S0 Memory Map

Register Name	I2S0_DBGCONTROL	I2S0_DMACONTROL	I2S0_STATUS	I2S0_INTCONTROL
ALL Address	0x4003_A0D0	0x4003_A0C0	0x4003_A0B0	0x4003_A0A0
Access Methods	ALL | SET | CLR	ALL | SET | CLR	ALL | SET | CLR	ALL | SET | CLR
Bit 31	Reserved	Reserved	Reserved	Reserved
Bit 30				
Bit 29				
Bit 28				
Bit 27				
Bit 26				
Bit 25				
Bit 24				
Bit 23				
Bit 22				
Bit 21				
Bit 20				
Bit 19				
Bit 18				
Bit 17				
Bit 16				
Bit 15				
Bit 14				
Bit 13				
Bit 12				
Bit 11				
Bit 10				
Bit 9			RXCLKENRF	
Bit 8			TXCLKENRF	
Bit 7			RXCLKSELRF	
Bit 6			TXCLKSELRF	
Bit 5			CDSTS	
Bit 4			CDBUSYF	
Bit 3	RXDBGMD	RXDMABMD	RXHWMI	RXHWMIEN
Bit 2	TXDBGMD	TXDMABMD	TXLWMI	TXLWMIEN
Bit 1	RXDBGHEN	RXDMAEN	RXOFI	RXOFIEN
Bit 0	TXDBGHEN	TXDMAEN	TXUFI	TXUFIEN

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

24. Current Mode Digital-to-Analog Converter (IDAC0 and IDAC1)

This section describes the Current Mode DAC (IDAC) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the IDAC block, which is used by both IDAC0 and IDAC1 on all device families covered in this document.

24.1. IDAC Features

The IDAC takes a digital value as an input and outputs a proportional constant current on a pin. The IDAC module includes the following features:

- 10-bit current DAC with output update trigger source options.
- Ability to update on rising, falling, or both edge for any of the external I/O trigger sources (DACnTx).
- Support for three full-scale output modes: 0.5, 1.0, and 2.0 mA.
- Four-word FIFO to aid with high-speed waveform generation or DMA interactions.
- FIFO supports wrapping mode, allowing the four values to be continuously cycled through to achieve 12-bit resolution.
- Individual FIFO overrun, underrun, and went-empty interrupt status sources.
- Support for multiple data packing formats, including: single 10-bit sample per word, dual 10-bit samples per word, or four 8-bit samples per word.
- Support for left- and right-justified data.

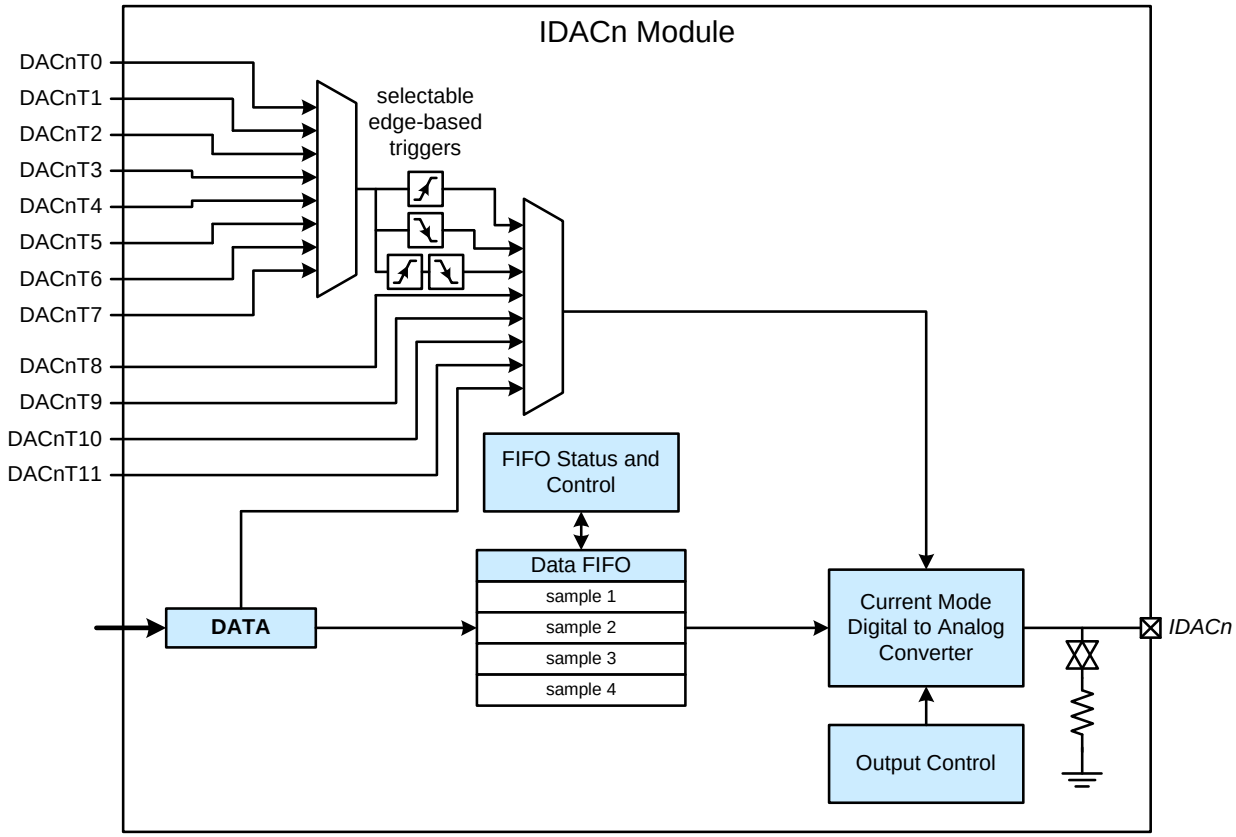


Figure 24.1. IDAC Block Diagram

24.2. IDAC Setup

The various IDAC features and modes are enabled using the CONTROL register. Table 24.3 summarizes the CONTROL register settings for using the IDAC in four different modes: on-demand mode, periodic FIFO wrap mode, periodic FIFO-only mode, and periodic DMA mode. For more detailed bit descriptions, please refer to the IDACn_CONTROL register description.

The entire IDAC block is enabled by setting the IDACEN bit, and can be shut off completely by clearing IDACEN. In most applications, the CONTROL register can be configured with a single write to both configure and enable the IDAC peripheral.

24.2.1. Full Scale Output

The IDAC full scale current output is configured using the OUTMD bit field. Three selectable ranges are available. These are nominally 2.046, 1.023 and 0.5115 mA. The resulting 10-bit LSB sizes of the IDAC in these three modes are 2 µA, 1 µA and 500 nA, respectively.

Optionally, an on-chip load resistor can be enabled by setting the LOADEN bit. This enables an impedance path to ground which effectively produces a voltage at the output pin. The nominal value of the load resistor is given in the device data sheet electrical specification tables. Note that any additional load impedance on the IDAC output pin will affect the output voltage in this mode.

This on-chip load resistor may be useful in some low-accuracy applications. The resistor exhibits a voltage dependence that may contribute some integral non-linearity in addition to the INL of the IDAC itself. The voltage output may also include noise or offset error due to differences in voltage between the internal ground of the resistor and the external board ground. For these reasons, an external resistor is recommended for the generation of an accurate ground-referenced voltage on the IDAC output.

24.2.2. Data Format

The IDAC output data can be packed into 32-bit words in three different ways, selected by the INFMT field: single 10-bit data entry, two 10-bit data entries, and four 8-bit data entries. Only single 10-bit data mode is supported when the trigger source is set to on-demand. Note that in 8-bit mode, only the upper 8 bits of the IDAC are used. When the packing format is configured for one of the two 10-bit options, the justification within each 16-bit half word of the 32-bit words can be specified using the JSEL bit. Data will be interpreted as right-justified when JSEL is 0, and left-justified when JSEL is 1. In the 8-bit packing mode, the JSEL bit has no effect. Figure 24.2 shows the data packing mode and justification options.

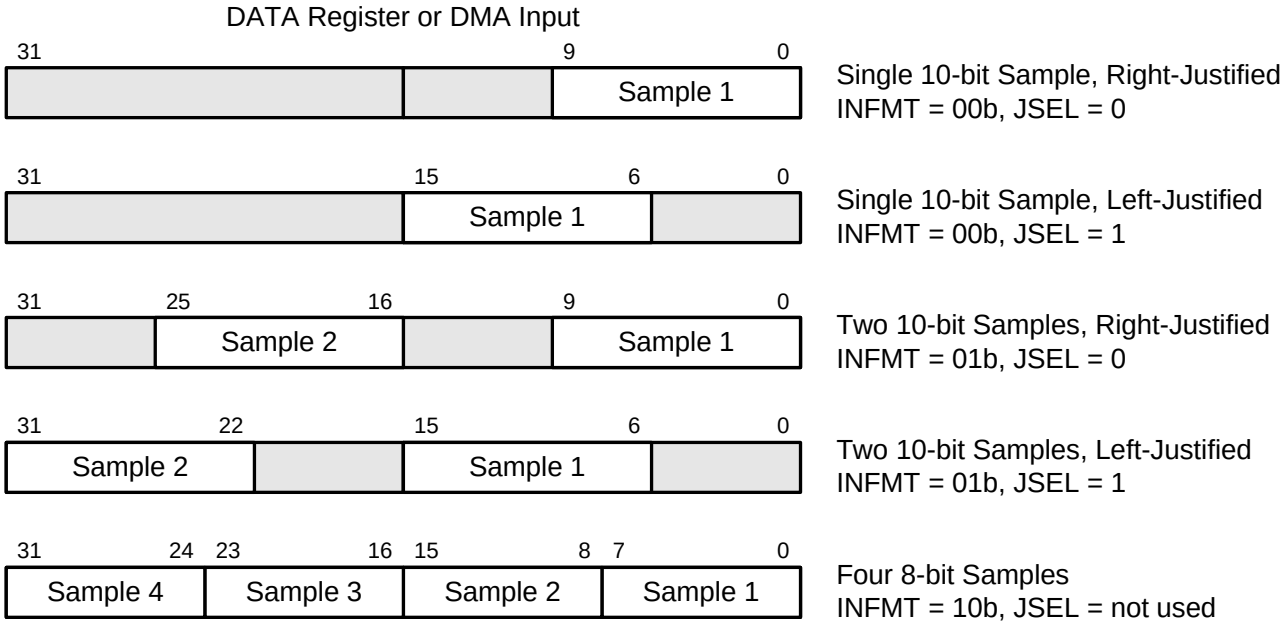


Figure 24.2. Data Packing Modes and Data Justification

24.2.3. Conversion Triggers

IDAC conversions can be triggered “on-demand” with a write to the DATA register, or periodically using one of the internal timer options or external conversion trigger inputs. Up to eight different external pins can be used as external trigger sources. The Sample Sync Generator (SSG) block is also selectable as an external trigger source. Specific trigger input sources for the IDACs vary between package options, and are detailed in Table 24.1 and Table 24.2.

Two fields in the CONTROL register configure the IDAC trigger source: OUPDT and ETRIG. The OUPDT field selects the trigger event from internal trigger sources (DACnT8 through DACnT11), on-demand mode, or one of three external trigger edge options. When set to an external trigger option, the IDAC can be triggered on the rising edge, falling edge, or both edges of a trigger source specified by the ETRIG bits.

Triggering of the IDAC can be inhibited at any time from firmware by writing the TRIGINH bit to 1. Any trigger events are ignored by the IDAC until the TRIGINH bit is cleared to 0 (except in on-demand mode, where TRIGINH is not used).

Table 24.1. IDAC0 and IDAC1 Output Update Triggers

IDAC0 Trigger	IDAC1 Trigger	Trigger Description	Internal Signal
DAC0T8	DAC1T8	Internal Trigger Source	Timer 0 Low overflow
DAC0T9	DAC1T9	Internal Trigger Source	Timer 0 High overflow
DAC0T10	DAC1T10	Internal Trigger Source	Timer 1 Low overflow
DAC0T11	DAC1T11	Internal Trigger Source	Timer 1 High overflow
DAC0T12	DAC1T12	Internal Trigger Source	DACnT0-DACnT7 (selected by ETRIG) rising edge
DAC0T13	DAC1T13	Internal Trigger Source	DACnT0-DACnT7 (selected by ETRIG) falling edge
DAC0T14	DAC1T14	Internal Trigger Source	DACnT0-DACnT7 (selected by ETRIG) any edge
DAC0T15	DAC1T15	Internal Trigger Source	“On Demand” by writing to the DATA field

Table 24.2. IDAC0 and IDAC1 ETRIG Selections

IDAC0 Trigger	IDAC1 Trigger	Trigger Description	SiM3U1x7/C1x7 Pin Name	SiM3U1x6/C1x6 Pin Name	SiM3U1x4/C1x4 Pin Name
DAC0T0	DAC1T0	External Trigger Source	PB3.2	PB3.2	PB3.0
DAC0T1	DAC1T1	External Trigger Source	PB3.3	PB3.3	PB3.1
DAC0T2	DAC1T2	External Trigger Source	PB3.5	PB3.5	PB3.2
DAC0T3	DAC1T3	External Trigger Source	PB3.6	PB3.6	PB3.3
DAC0T4	DAC1T4	External Trigger Source	PB3.7	PB3.7	Reserved
DAC0T5	DAC1T5	External Trigger Source	PB3.8	PB3.8	Reserved
DAC0T6	DAC1T6	External Trigger Source	PB3.9	PB3.9	Reserved
DAC0T7	DAC1T7	Internal Trigger Source	SSG0 (IDAC0 = EX2, IDAC1 = EX3)		

Table 24.3. IDAC Configuration Quick-Reference

Bit Field in CONTROL Register	Operational Mode			
	On-Demand	**Periodic FIFO Wrap (no DMA)**	**Periodic FIFO-Only (no DMA)**	**Periodic with DMA**
IDACEN	1 = Enable IDAC			
DMARUN	0 = Disable DMA			1 = Enable DMA
INFMT	00b = Single, 10-bit Sample	Any Option		
OUPDT	111b = Trigger On-Demand	Any Option Except 111b		
LOADEN	Load resistor enable = Set to 1 if internal load path is desired			
DBGMD	Debug Mode = Set to 1 to let IDAC continue running in debug halt			
JSEL	Data Justification for 10-bit Input Formats: 0 = Right-justify data, 1 = Left-justify data			
OUTMD	Load resistor enable = Set on/off according to application needs			
WEIEN	N/A		Set to enable FIFO Went Empty Interrupt	
URIEN	N/A		Set to enable FIFO Underrun Interrupt	
ORIEN	N/A		Set to enable FIFO Overrun Interrupt	
WRAPEN	N/A	1 = Enable Wrap	0 = Disable Wrap	
TRIGINH	N/A	Set to inhibit IDAC triggering		
BUFRESET	N/A	Set to reset input FIFO		
ETRIG	N/A	Selects external trigger source (if used)		

24.3. Using the IDAC in On-Demand Mode

On-demand mode is useful primarily in DC applications where the IDAC output is changed infrequently. In on-demand mode, the IDAC output is taken directly from the DATA register, and any writes to the DATA register will trigger a corresponding change in current at the IDAC output pin.

The only data input mode supported for on-demand operation is single 10-bit mode. Data writes can be left or right-justified. The FIFO and its associated interrupts are not used in this mode.

24.4. Using the IDAC in Periodic FIFO-Only Mode

Periodic FIFO-only mode is useful in applications where the IDAC output needs to be updated at a specific time interval, and frequent software intervention is desired. In periodic FIFO-only mode, IDAC updates are configured to occur on the selected trigger signal, and the next output value is pulled from a four-sample FIFO buffer associated with the IDAC. Writes to the DATA register from firmware push new data into the FIFO. It is important that the FIFO has enough room for the sample(s) written to the data register. For example, when INFMT is configured for four 8-bit samples, the entire FIFO must be empty before writing to DATA, but when INFMT is configured for single 10-bit samples, only one FIFO entry need be free.

The number of samples currently waiting in the FIFO is reflected in the LEVEL field of the BUFSTATUS register. The contents of the FIFO can be inspected at any time by reading the BUFFER10 and BUFFER32 registers, and the current IDAC output can always be read from the DATA register.

All three interrupt sources can be enabled in periodic FIFO-only mode. Their meanings for this mode are detailed below.

- FIFO Underrun Interrupt (URI): The URI interrupt flag will be set if an IDAC trigger happens and the FIFO buffer level is zero (i.e. when there is no data to pull from the FIFO). The FIFO read pointer and IDAC output will not be updated when an underrun error occurs.
- FIFO Overrun Interrupt (ORI): The ORI interrupt flag will be set if firmware writes to DATA and there is not enough room in the FIFO for the number of samples written. In this case, no data will be written to the FIFO, and the FIFO write pointer will not be updated.
- FIFO Went Empty Interrupt (WEI): The WEI interrupt flag will be set when an IDAC update reads the final byte from the FIFO into the IDAC output latch, and causes the FIFO level to go to zero. This interrupt can be used by firmware to initiate a new write to the FIFO before the next trigger occurs, and avoid an underrun interrupt.

24.5. Using the IDAC in Periodic FIFO Wrap Mode

Periodic FIFO wrap mode is similar to periodic FIFO-only mode in all ways except for the behavior of the FIFO and the IDAC interrupts. In this mode, the IDAC will continuously pull from the four-sample FIFO in a circular fashion, thereby generating a repeating four sample pattern. The underrun and went empty interrupts are masked off in this mode, and will never occur. The overrun interrupt flag is still active in this mode, but is typically not of much use.

This mode can extend the effective resolution of the DAC to 12 bits at one-fourth of the 10-bit sample rate by using the four words in the buffer to interpolate between two adjacent 10-bit values. For example, if the FIFO includes three words of value n and one word of value n+1, then the average output value will be n+0.25, which represents a 12-bit quantity.

To load a new set of four samples into the FIFO, the following sequence should be followed:

1. Wait for an IDAC trigger to occur.
2. Reset the FIFO by writing 1 to the BUFRESET bit in the CONTROL register.
3. Load the FIFO with the next set of four samples. The first of these samples should be written before the next trigger event occurs to avoid any glitches in the IDAC output.

24.6. Using the IDAC in Periodic DMA Mode (on select IDAC peripherals only)

A DMA channel can be used to offload core resources and transfer data into the IDAC FIFO. When used in periodic DMA mode, the configuration and capabilities of the IDAC are largely the same as those described in periodic FIFO-only mode. The difference in DMA mode is how data is written into the FIFO, as well as the meaning of the interrupt sources.

When a DMA channel is used to write the FIFO buffer, the FIFO logic will work to keep the buffer full. The FIFO level is monitored, and when the level falls below the number of samples encoded per data word (as specified by the INFMT field), a DMA request will be generated. The DMA request will remain pending until the FIFO no longer has room for new transfers. When configured for a single sample per data word (INFMT = 00b), DMA requests are generated when the LEVEL field in BUFSTATUS is less than or equal to 3. For two samples per data word (INFMT = 01b), DMA requests are generated when LEVEL is less than or equal to two, and for four samples per data word (INFMT = 10b), LEVEL must be 0 to initiate a DMA transfer.

For the IDAC module, the DMA should be configured as follows:

- Source size (SRCSIZE) and destination size (DSTSIZE) are 2 for a word transfer.
- The source address increment (SRCAIMD) is 2 for word increments.
- The destination address increment (DSTAIMD) is 3 for non-incrementing mode.
- The NCOUNT value is N – 1, where N is the number of 4-byte words.
- RPOWER = 0 (1 word transfer per transaction).

Once the DMA is configured, writing a 1 to DMAEN will enable the DMA request from the IDAC. The FIFO will continue to be serviced by the DMA until the specified transfer operation is complete.

When the DMA transfer is complete, the FIFO went empty interrupt flag will be asserted, and the DMAEN bit will be cleared to 0 by hardware. If firmware requires continuous operation of the IDAC as this occurs, it must handle the went empty interrupt, reconfigure the DMA and enable DMA transfers before the next trigger source occurs.

All three interrupt sources can be enabled in periodic DMA mode, and it is recommended that firmware do so. The meanings of the interrupts for this mode are detailed below.

- FIFO Underrun Interrupt (URI): The URI interrupt flag will be set if an IDAC trigger happens and the FIFO buffer level is zero (i.e. when there is no data to pull from the FIFO). This can occur if the configured DMA transfer has not completed and a new trigger occurs.
- FIFO Overrun Interrupt (ORI): The ORI interrupt flag will be set if a DMA transfer occurs when there is not enough room in the FIFO for the number of samples written. In this case, no data will be written to the FIFO, and the FIFO write pointer will not be updated. An overrun error should not occur when using the DMA unless there is a firmware conflict with the FIFO.
- FIFO Went Empty Interrupt (WEI): In DMA mode, the WEI interrupt flag is only set at the end of a DMA transfer. This enables the WEI interrupt to be used by firmware to initiate the next DMA sequence if needed.

24.7. Adjusting the IDAC Output Current

The output current of the IDAC is factory calibrated to provide the target current for each OUTMD setting. However, the output current can be adjusted slightly in 32 steps using the GAINADJ field. A value of zero in this field represents the minimum current the IDAC can output at the current OUTMD setting, and a maximum value of 31 in this field represents the maximum current. Each step adjusts the output current by about 1.5%.

This GAINADJ field can be used to calibrate small gain errors that result when using either the internal load resistor or a wider tolerance external load resistor. If the device has an ADC module, the IDAC can be connected internally to the ADC if the IDAC output pin is supported on the ADC input mux or connected externally by shorting the IDAC output and ADC input pins together. Firmware can then use the ADC to measure the resulting voltage on the IDAC output and adjust the GAINADJ field until reaching the desired voltage on the IDAC output.

24.8. Debug Mode

Firmware can set the DBGMD bit to force the IDAC module to halt on a debug breakpoint. Clearing the DBGMD bit forces the module to continue operating while the core halts in debug mode.

24.9. IDAC0 and IDAC1 Registers

This section contains the detailed register descriptions for IDAC0 and IDAC1 registers.

Register 24.1. IDACn_CONTROL: Module Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	IDACEN	LOADEN	DBGMD	Reserved						WEIEN	URIEN	ORIEN	Reserved			WRAPEN
Type	RW	RW	RW	R						RW	RW	RW	R			RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved		TRIGINH	BUFRESET	JSEL	DMARUN	INFMT		OUTMD		ETRIG			OUPDT		
Type	R		RW	W	RW	RW	RW		RW		RW			RW		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1

Register ALL Access Addresses

IDAC0_CONTROL = 0x4003_1000
IDAC1_CONTROL = 0x4003_2000
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 24.4. IDACn_CONTROL Register Bit Descriptions

Bit	Name	Function
31	IDACEN	**IDAC Enable.** 0: Disable the IDAC. 1: Enable the IDAC.
30	LOADEN	**Load Resistor Enable.** Enables an on-chip load resistor to ground. 0: Disable the internal load resistor. 1: Enable the internal load resistor.
29	DBGMD	**IDAC Debug Mode.** 0: The IDAC module will continue to operate while the core is halted in debug mode. 1: A debug breakpoint will cause the IDAC module to halt (ignore update triggers).
28:23	Reserved	Must write reset value.
22	WEIEN	**FIFO Went Empty Interrupt Enable.** Enables the FIFO went empty interrupt flag (WEI) to generate an IDAC interrupt when set to 1.

Table 24.4. IDACn_CONTROL Register Bit Descriptions

Bit	Name	Function
21	URIEN	**FIFO Underrun Interrupt Enable.** Enables the FIFO underrun interrupt flag (URI) to generate an IDAC interrupt when set to 1.
20	ORIEN	**FIFO Overrun Interrupt Enable.** Enables the FIFO overrun interrupt flag (ORI) to generate an IDAC interrupt when set to 1.
19:17	Reserved	Must write reset value.
16	WRAPEN	**Wrap Mode Enable.** Enables IDAC to repeatedly cycle over the FIFO contents in a circular fashion. 0: The IDAC will not wrap when it reaches the end of the data buffer. 1: The IDAC will cycle through the data buffer contents.
15:14	Reserved	Must write reset value.
13	TRIGINH	**Trigger Source Inhibit.** Setting this bit to 1 will mask of any periodic trigger sources. No updates to the IDAC will occur when this bit is set, unless the IDAC is configured for on-demand mode. When cleared to 0, IDAC updates will resume on the next trigger source (trigger sources are not queued).
12	BUFRESET	**Data Buffer Reset.** Writing a 1 to this bit resets the data buffer. Writing a 0 has no effect, and this bit always reads back as 0.
11	JSEL	**Data Justification Select.** This bit selects the data justification in 10-bit input modes. 0: Data is right-justified. 1: Data is left-justified.
10	DMARUN	**DMA Run.** Writing a 1 to this bit enables DMA transfers. This bit is automatically cleared when DMA operations are complete.
9:8	INFMT	**Data Input Format.** This field determines the interpretation of data written to the IDAC. Only single, 10-bit samples are supported in on-demand mode. For periodic FIFO-only mode or periodic FIFO wrap mode, FIFO writes occur on a write to the DATA register. In DMA mode, FIFO writes occur on a DMA event. 00: Writes are interpreted as one 10-bit sample. 01: Writes are interpreted as two 10-bit samples. 10: Writes are interpreted as four 8-bit samples. 11: Reserved.
7:6	OUTMD	**Output Mode.** This field selects the IDAC full-scale output current. 00: The full-scale output current is 0.5 mA. 01: The full-scale output current is 1 mA. 10: The full-scale output current is 2 mA. 11: Reserved.

Table 24.4. IDACn_CONTROL Register Bit Descriptions

Bit	Name	Function
5:3	ETRIG	**Edge Trigger Source Select.** When OUPDT is configured for any of the edge-trigger options (100b, 101b, or 110b), this field selects the specific external trigger source. 000: Select DACnT0 as the IDAC external trigger source. 001: Select DACnT1 as the IDAC external trigger source. 010: Select DACnT2 as the IDAC external trigger source. 011: Select DACnT3 as the IDAC external trigger source. 100: Select DACnT4 as the IDAC external trigger source. 101: Select DACnT5 as the IDAC external trigger source. 110: Select DACnT6 as the IDAC external trigger source. 111: Select DACnT7 as the IDAC external trigger source.
2:0	OUPDT	**Output Update Trigger.** This field selects the trigger source for IDAC output updates. 000: The IDAC output updates using the DACnT8 trigger source. 001: The IDAC output updates using the DACnT9 trigger source. 010: The IDAC output updates using the DACnT10 trigger source. 011: The IDAC output updates using the DACnT11 trigger source. 100: The IDAC output updates on the rising edge of the trigger source selected by ETRIG. 101: The IDAC output updates on the falling edge of the trigger source selected by ETRIG. 110: The IDAC output updates on any edge of the trigger source selected by ETRIG. 111: The IDAC output updates on write to DATA register (On Demand).

Register 24.2. IDACn_DATA: Output Data

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	DATA[31:16]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	DATA[15:0]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

IDAC0_DATA = 0x4003_1010

IDAC1_DATA = 0x4003_2010

Table 24.5. IDACn_DATA Register Bit Descriptions

Bit	Name	Function
31:0	DATA	**Output Data.** When the OUPDT field is set to On-Demand mode, writes to this register update the IDAC value immediately, and are assumed to contain a single 10-bit sample. For all other trigger sources, writes to this register are pushed into the data buffer in the format specified by the INFMT field. Reads from this register always return the current output value in the IDAC latch.

Register 24.3. IDACn_BUFSTATUS: FIFO Buffer Status

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved									WEI	URI	ORI	Reserved	LEVEL		
Type	R									RW	RW	RW	R	R		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

IDAC0_BUFSTATUS = 0x4003_1020

IDAC1_BUFSTATUS = 0x4003_2020

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 24.6. IDACn_BUFSTATUS Register Bit Descriptions

Bit	Name	Function
31:7	Reserved	Must write reset value.
6	WEI	**FIFO Went Empty Interrupt Flag.** This bit is set to 1 by hardware when the last sample is transferred from the FIFO into the IDAC output latch. This bit must be cleared by software.
5	URI	**FIFO Underrun Interrupt Flag.** This bit is set to 1 by hardware when a FIFO underrun has occurred. This bit must be cleared by software.
4	ORI	**FIFO Overrun Interrupt Flag.** This bit is set to 1 by hardware when a FIFO overrun has occurred. This bit must be cleared by software.
3	Reserved	Must write reset value.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 24.6. IDACn_BUFSTATUS Register Bit Descriptions

Bit	Name	Function
2:0	LEVEL	**FIFO Level.** Indicates the number of words currently pending in the output data FIFO. 000: The data FIFO is empty. 001: The data FIFO contains one word. 010: The data FIFO contains two words. 011: The data FIFO contains three words. 100: The data FIFO is full and contains four words. 101-111: Reserved.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Register 24.4. IDACn_BUFFER10: FIFO Buffer Entries 0 and 1

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	BUFFER1															
Type	R															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	BUFFER0															
Type	R															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Register ALL Access Addresses

IDAC0_BUFFER10 = 0x4003_1030
IDAC1_BUFFER10 = 0x4003_2030

Table 24.7. IDACn_BUFFER10 Register Bit Descriptions

Bit	Name	Function
31:16	BUFFER1	**FIFO Buffer Entry 1.** This field is the second pending IDAC output. It is justified according to the JSEL selection.
15:0	BUFFER0	**FIFO Buffer Entry 0.** This field is the first pending IDAC output. It is justified according to the JSEL selection.

Register 24.5. IDACn_BUFFER32: FIFO Buffer Entries 2 and 3

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	BUFFER3															
Type	R															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	BUFFER2															
Type	R															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Register ALL Access Addresses

IDAC0_BUFFER32 = 0x4003_1040
IDAC1_BUFFER32 = 0x4003_2040

Table 24.8. IDACn_BUFFER32 Register Bit Descriptions

Bit	Name	Function
31:16	BUFFER3	**FIFO Buffer Entry 3.** This field is the fourth pending IDAC output. It is justified according to the JSEL selection.
15:0	BUFFER2	**FIFO Buffer Entry 2.** This field is the third pending IDAC output. It is justified according to the JSEL selection.

Register 24.6. IDACn_GAINADJ: Output Current Gain Adjust

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved											GAINADJ				
Type	R											RW				
Reset	0	0	0	0	0	0	0	0	0	0	0	X	X	X	X	X

Register ALL Access Addresses

IDAC0_GAINADJ = 0x4003_1050

IDAC1_GAINADJ = 0x4003_2050

Table 24.9. IDACn_GAINADJ Register Bit Descriptions

Bit	Name	Function
31:5	Reserved	Must write reset value.
4:0	GAINADJ	**Output Current Gain Adjust.** This field is factory calibrated to produce the target full-scale output current for all OUTMD settings. However, firmware can modify this field to adjust the output current of the IDAC up or down. A value of 0 represents the minimum current setting, and a value of 31 represents the maximum current setting. Each step adjusts the output current by about 1.5%.

24.10. IDACn Register Memory Map

Table 24.10. IDACn Memory Map

Register Name	IDACn_CONTROL	IDACn_DATA	IDACn_BUFSTATUS	IDACn_BUFFER10
ALL Offset	0x0	0x10	0x20	0x30
Access Methods	ALL | SET | CLR	ALL	ALL | SET | CLR	ALL
Bit 31	IDACEN	DATA	Reserved	BUFFER1
Bit 30	LOADEN			
Bit 29	DBGMD			
Bit 28	Reserved			
Bit 27				
Bit 26				
Bit 25				
Bit 24				
Bit 23				
Bit 22	WEIEN			
Bit 21	URIEN			
Bit 20	ORIEN			
Bit 19	Reserved			
Bit 18				
Bit 17				
Bit 16	WRAPEN			
Bit 15	Reserved			BUFFER0
Bit 14				
Bit 13	TRIGINH			
Bit 12	BUFRESET			
Bit 11	JSEL			
Bit 10	DMARUN			
Bit 9	INFMT			
Bit 8				
Bit 7	OUTMD			
Bit 6				
Bit 5	ETRIG		WEI	
Bit 4			URI	
Bit 3			ORI	
Bit 2	OUPDT		Reserved	
Bit 1				
Bit 0			LEVEL	

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: IDAC0 = 0x4003_1000, IDAC1 = 0x4003_2000

Table 24.10. IDACn Memory Map

Register Name	ALL Offset	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IDACn_BUFFER32	0x40	ALL	BUFFER3																BUFFER2																
IDACn_GAINADJ	0x50	ALL	Reserved																														GAINADJ	

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: IDAC0 = 0x4003_1000, IDAC1 = 0x4003_2000

25. Current-to-Voltage Converter (IVC0)

This section describes the Current-to-Voltage Converter (IVC) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the IVC block, which is used by all device families covered in this document.

25.1. IVC Features

The IVC module allows current sourced from a pin (sunk externally) to be measured by SARADCn modules on the device. The Current-to-Voltage Converter (IVC) module includes the following features:

- Two independent channels.
- Programmable input ranges.

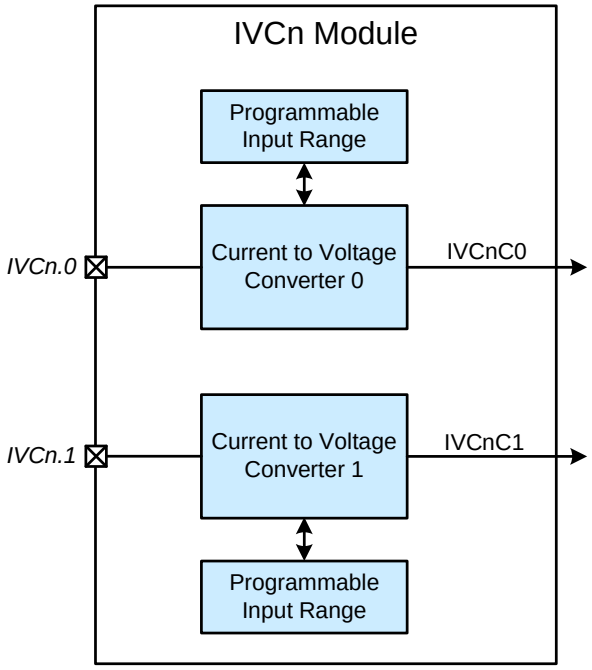


Figure 25.1. IVC Block Diagram

25.2. Functional Description

The IVC module consists of two channels of current-to-voltage conversion circuitry. It allows current sunk by external circuitry to be measured by a SARADCn module on the device. The response time of the IVC circuit is fast enough to settle full-scale current changes at the input within the SARADC's 1 µs sampling rate. Current is sourced from the IVC0.0 or IVC0.1 pins, creating a voltage drop at the pin and the ADC input. A simplified diagram of a single channel of the IVC circuitry is shown in Figure 25.2.

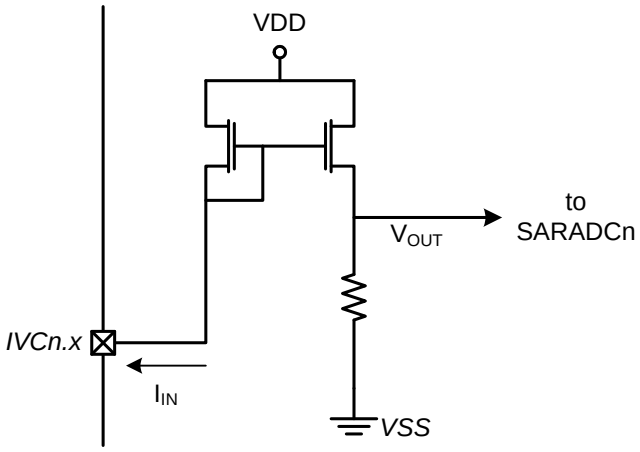


Figure 25.2. Simplified IVC Circuit Diagram

The IVC module has selectable full-scale current ranges of 1 mA to 6 mA, configurable in 1 mA steps. The transfer function of the IVC module is a curve that is 0 V with a 0 mA input, and 1.65 V at full scale. It is important to note that the transfer function of the IVC module has more nonlinearity than the ADC, but the transfer function of the two channels will tend to be reasonably well-matched on a given device. The nonlinearity of the transfer function appears as an INL error in the ADC's output.

Table 25.1. IVC0 Input Channels

IVC0 Input	IVC0 Input Description	SiM3U1x7/C1x7 Pin Name	SiM3U1x6/C1x6 Pin Name	SiM3U1x4/C1x4 Pin Name
IVC0.0	Channel 0	PB0.7	PB0.4	Reserved
IVC0.1	Channel1	PB0.8	PB0.5	Reserved

25.3. Configuration

Each of the two IVC channels are independently enabled using the C0EN and C1EN bits in the CONTROL register. The input range for the channels are also independently selectable, using the IN0RANGE and IN1RANGE fields. The following steps describe how to configure and use an IVC channel:

1. Select the desired IVC channel as an input in the SARADCn module.
2. Configure the input range of the IVC channel using the INxRANGE field in the CONTROL register.
3. Enable the IVC channel using the CxEN bit in the CONTROL register.
4. Begin converting data with the SARADCn module.

25.4. IVC0 Registers

This section contains the detailed register descriptions for IVC0 registers.

Register 25.1. IVC0_CONTROL: Module Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	C1EN	C0EN	Reserved													
Type	RW	RW	R													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved									IN1RANGE			Reserved	IN0RANGE		
Type	R									RW			R	RW		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

IVC0_CONTROL = 0x4004_4000

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 25.2. IVC0_CONTROL Register Bit Descriptions

Bit	Name	Function
31	C1EN	**Converter 1 Enable.** 0: Disable IVC channel 1. 1: Enable IVC channel 1.
30	C0EN	**Converter 0 Enable.** 0: Disable IVC channel 0. 1: Enable IVC channel 0.
29:7	Reserved	Must write reset value.
6:4	IN1RANGE	**Input 1 Range.** These bits dictate the input range for IVC channel 1. The output voltage at the full-scale input current is 1.65 V. 000: Input range is 0-6 mA. 001: Input range is 0-5 mA. 010: Input range is 0-4 mA. 011: Input range is 0-3 mA. 100: Input range is 0-2 mA. 101: Input range is 0-1 mA. 110-111: Reserved.
3	Reserved	Must write reset value.

Table 25.2. IVC0_CONTROL Register Bit Descriptions

Bit	Name	Function
2:0	IN0RANGE	**Input 0 Range.** These bits dictate the input range for IVC channel 0. The output voltage at the full-scale input current is 1.65 V. 000: Input range is 0-6 mA. 001: Input range is 0-5 mA. 010: Input range is 0-4 mA. 011: Input range is 0-3 mA. 100: Input range is 0-2 mA. 101: Input range is 0-1 mA. 110-111: Reserved.

25.5. IVC0 Register Memory Map

Table 25.3. IVC0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IVC0_CONTROL	0x4004_4000	ALL | SET | CLR	C1EN	C0EN	Reserved																							IN1RANGE			Reserved		IN0RANGE	

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

26. Low Power Oscillator (LPOSC0)

This section describes the Low Power Oscillator (LPOSC) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the LPOSC block, which is used by all device families covered in this document.

26.1. Low Power Oscillator Features

The Low Power Oscillator has the following features:

- 20 MHz and divided 2.5 MHz frequencies available for the AHB clock.
- Automatically starts and stops as needed.

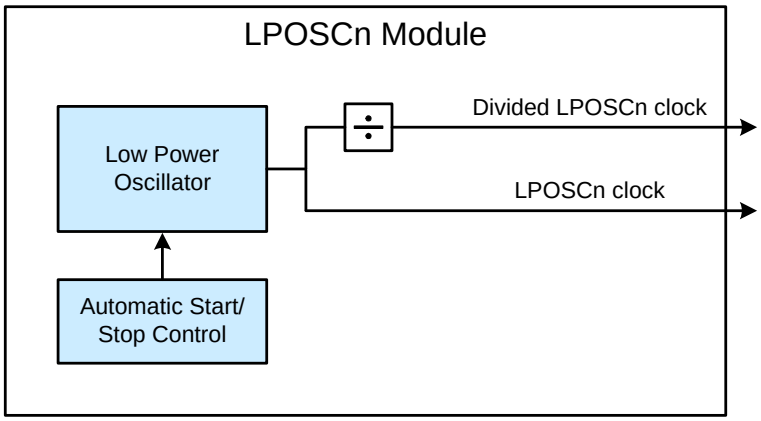


Figure 26.1. Low Power Oscillator Block Diagram

26.2. Operation

The Low Power Oscillator is the default AHB oscillator and enables or disables automatically, as needed. The power consumption of this oscillator is listed in the electrical specifications of the device data sheet.

The default output frequency of this oscillator is factory calibrated to 20 MHz, and a divided 2.5 MHz version is also available to use as an AHB clock source. More information on the clocks available to the device can be found in the clock control description.

26.3. LPOSC0 Registers

This section contains the detailed register descriptions for LPOSC0 registers.

Register 26.1. LPOSC0_OSCVAL: Low Power Oscillator Output Value

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved												OSCVAL			
Type	R												R			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	X	X	X	X

Register ALL Access Address

LPOSC0_OSCVAL = 0x4004_1000

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 26.1. LPOSC0_OSCVAL Register Bit Descriptions

Bit	Name	Function
31:4	Reserved	Must write reset value.
3:0	OSCVAL	**Low Power Oscillator Output Value.** This read-only field is the factory-calibrated output frequency value for the low power oscillator.

26.4. LPOSC0 Register Memory Map

Table 26.2. LPOSC0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
LPOSC0_OSCVAL	0x4004_1000	ALL | SET | CLR	Reserved																												OSCVAL			

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

27. Low Power Timer (LPTIMER0)

This section describes the Low Power Timer (LPTIMER) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the LPTIMER block, which is used by all device families covered in this document.

27.1. Low Power Timer Features

The Low Power Timer includes the following features:

- Runs on RTC0TCLK or an external source (rising or falling edge).
- Overflow and compare threshold-match detection, which can generate an interrupt, reset the timer, or wake the device from low power modes.
- Timer reset on threshold-match allows square-wave generation at variable output frequency.

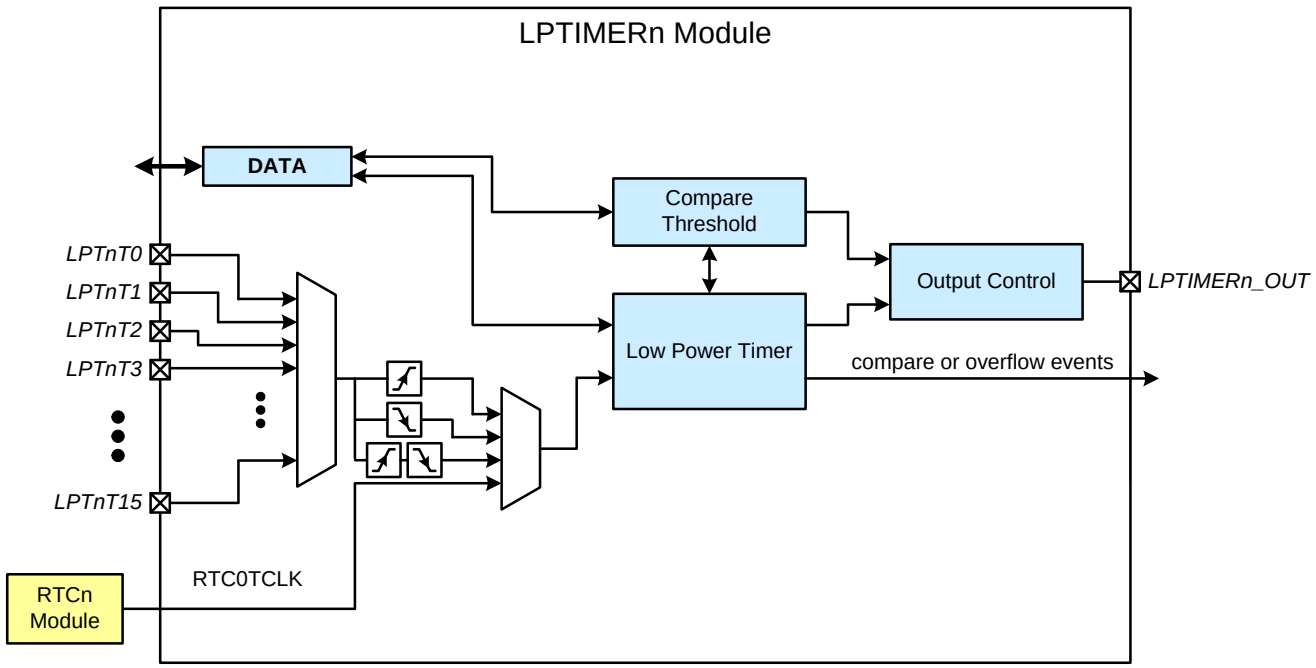


Figure 27.1. Low Power Timer Block Diagram

27.2. Clocking

The Low Power Timer (LPTIMER) module runs from the RTC0 timer clock (RTC0TCLK), allowing the LPTIMER to operate even if the AHB and APB clocks are disabled. When writing and reading registers, a bit or field may take 3 APB and 2 RTC0TCLK clocks to update to a written value.

The LPTIMER counter can increment using one of two clock sources: RTC0TCLK, or rising or falling edges of an external signal.

27.2.1. Clock Input

The LPTIMER timer can use the RTC timer clock (RTC0TCLK) as its input clock. The RTC0TCLK clock is configurable in the RTC0 module as either the RTC0 crystal oscillator (RTC0OSC), an external CMOS clock, or the internal low frequency oscillator (LFOSC0).

27.2.2. External Signal

The LPTIMER module can select one of four external signals as its input clock using the EXTSEL field. The timer can increment on rising, falling, or both edges of the selected external input, controlled by the CMD field. These inputs are synchronized to RTC0TCLK, so they must be high or low for two rising RTC0TCLK edges. Figure 27.2 illustrates the external input signal timing. The external trigger sources available to the LPTIMER vary by package and are defined in Table 27.1.

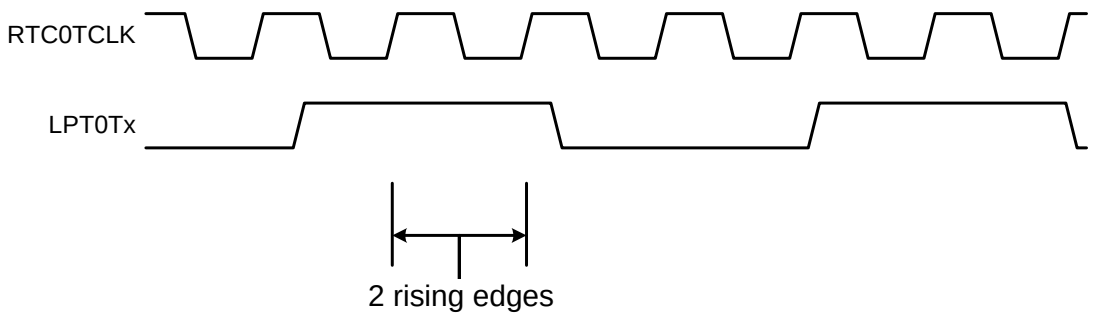


Figure 27.2. External Input Clock Synchronization Timing

Table 27.1. LPTIMER0 Triggers

LPTIMER0 Trigger	LPTIMER0 Trigger Description	SiM3U1x7/C1x7 Pin Name	SiM3U1x6/C1x6 Pin Name	SiM3U1x4/C1x4 Pin Name
LPT0T0	External Trigger Source	PB3.2	PB3.2	PB3.0
LPT0T1	External Trigger Source	PB3.8	PB3.8	PB3.1
LPT0T2	External Trigger Source	PB3.9	PB3.9	PB3.2
LPT0T3	Internal Trigger Source	Comparator 0 output		

27.3. Configuring the Timer

The Low Power Timer is an up-counter that has a compare event and an overflow event. Firmware can access the timer and compare values indirectly through the DATA register. Note that the RUN bit in the CONTROL register must be set to 1 to allow writes of other bit in the LPTIMER block.

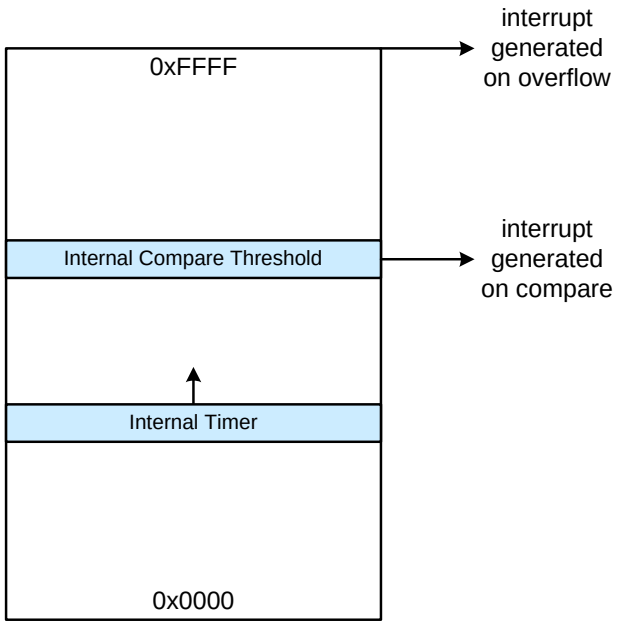


Figure 27.3. Low Power Timer Operation

To write a value to the internal timer or compare thresholds:

1. Write the desired new value to DATA.
2. Set the TMRSET or CMPSET bit to set the timer or compare threshold, respectively.
3. Poll on the TMRSET or CMPSET bit to determine when the operation completes.

To read the current value of the timer or compare threshold:

1. Set the TMRCAP or CMPCAP bit to transfer the value from the internal timer or compare threshold to DATA.
2. Poll on the TMRCAP or CMPCAP bit to determine when the operation completes.
3. Read the DATA register.

Once the LPTIMER timer is configured as desired, firmware can stop and start the timer using the RUN bit. RUN must be set to 1 when changing any register values.

27.4. Interrupts

The LPTIMER module has two sources that can cause an interrupt: overflow and compare events. The overflow interrupt (OVFI) flag indicates when the timer overflows and can generate an interrupt when OVFIEN is set to 1. Hardware sets the compare interrupt (CMPI) flag when the timer value matches the value in the internal compare threshold and can generate an interrupt if CMPIEN is set to 1.

Whenever hardware sets the OVFI and CMPI flags, firmware should clear the appropriate flag and poll until it reads back as zero.

27.5. Output

The LPTIMER module has an output (OUT) that can be toggled on an overflow or compare event. This output can be connected to a physical pin using the device port configuration module. Table 27.2 shows a summary of the output behavior.

Table 27.2. Low Power Timer Output Modes

OVFTMD Bit Value	CMPTMD Bit Value	Output Mode
0	0	never toggles
0	1	toggles on compare events only
1	0	toggles on overflow events only
1	1	toggles on both overflow and compare events

27.6. Automatic Reset

In addition to potentially generating an interrupt or toggling the LPTIMER output, the compare event can also automatically reset the timer to zero. This automatic reset mode is enabled by setting CMPRSTEN to 1.

27.7. Debug Mode

Firmware can set the DBGMD bit to force the LPTIMER module to halt on a debug breakpoint. Clearing the DBGMD bit forces the module to continue operating while the core halts in debug mode.

27.8. LPTIMER0 Registers

This section contains the detailed register descriptions for LPTIMER0 registers.

Register 27.1. LPTIMER0_CONTROL: Module Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	RUN	DBGMD	Reserved					CMPRSTEN	Reserved				CMPTMD	OVFTMD	CMPIEN	OVFIEN
Type	RW	RW	R					RW	R				RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved				CMPCAP	CMPSET	TMRCAP	TMRSET	EXTSEL				Reserved		CMD	
Type	R				RW	RW	RW	RW	RW				R		RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

LPTIMER0_CONTROL = 0x4003_8000

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 27.3. LPTIMER0_CONTROL Register Bit Descriptions

Bit	Name	Function
31	RUN	**Timer Run Control and Compare Threshold Enable.** This bit controls the timer operation and compare threshold. Note that this bit must be set to 1 in order to write any LPTIMER registers. 0: Stop the timer and disable the compare threshold. 1: Start the timer running and enable the compare threshold.
30	DBGMD	**Low Power Timer Debug Mode.** 0: The Low Power Timer module will continue to operate while the core is halted in debug mode. 1: A debug breakpoint will cause the Low Power Timer module to halt.
29:25	Reserved	Must write reset value.
24	CMPRSTEN	**Timer Compare Event Reset Enable.** 0: Timer compare events do not reset the timer. 1: Timer compare events reset the timer.

Notes:

1. When writing to any of the LPTIMER registers, the CONTROL.RUN bit must be set to 1.
2. When writing and reading registers, a bit or field may take 3 APB and 2 RTC0TCLK cycles to update to a written value.

Table 27.3. LPTIMER0_CONTROL Register Bit Descriptions

Bit	Name	Function
23:20	Reserved	Must write reset value.
19	CMPTMD	**Timer Compare Event Toggle Mode .** 0: Timer compare events do not toggle the Low Power Timer output. 1: Timer compare events toggle the Low Power Timer output.
18	OVFTMD	**Timer Overflow Toggle Mode.** 0: Timer overflows do not toggle the Low Power Timer output. 1: Timer overflows toggle the Low Power Timer output.
17	CMPIEN	**Timer Compare Event Interrupt Enable.** 0: Disable the timer compare event interrupt. 1: Enable the timer compare event interrupt.
16	OVFIEN	**Timer Overflow Interrupt Enable.** 0: Disable the timer overflow interrupt. 1: Enable the timer overflow interrupt.
15:12	Reserved	Must write reset value.
11	CMPCAP	**Timer Comparator Capture.** Writing a 1 to CMPCAP initiates a read of the internal comparator register into the DATA register. This field is automatically cleared by hardware when the operation completes and does not need to be cleared by software.
10	CMPSET	**Timer Comparator Set.** Writing a 1 to CMPSET initiates a copy of the value in DATA into the internal timer comparator register. This field is automatically cleared by hardware when the copy is complete and does not need to be cleared by software.
9	TMRCAP	**Timer Capture.** Writing a 1 to TMRCAP initiates a read of internal timer register into the DATA register. This field is automatically cleared by hardware when the operation completes and does not need to be cleared by software.
8	TMRSET	**Timer Set.** Writing a 1 to TMRSET initiates a copy of the value from the DATA register into the internal timer register. This field is automatically cleared by hardware when the copy is complete and does not need to be cleared by software.

Notes:

1. When writing to any of the LPTIMER registers, the CONTROL.RUN bit must be set to 1.
2. When writing and reading registers, a bit or field may take 3 APB and 2 RTC0TCLK cycles to update to a written value.

Table 27.3. LPTIMER0_CONTROL Register Bit Descriptions

Bit	Name	Function
7:4	EXTSEL	**External Trigger Source Select.** 0000: Select external trigger LPTnT0. 0001: Select external trigger LPTnT1. 0010: Select external trigger LPTnT2. 0011: Select external trigger LPTnT3. 0100: Select external trigger LPTnT4. 0101: Select external trigger LPTnT5. 0110: Select external trigger LPTnT6. 0111: Select external trigger LPTnT7. 1000: Select external trigger LPTnT8. 1001: Select external trigger LPTnT9. 1010: Select external trigger LPTnT10. 1011: Select external trigger LPTnT11. 1100: Select external trigger LPTnT12. 1101: Select external trigger LPTnT13. 1110: Select external trigger LPTnT14. 1111: Select external trigger LPTnT15.
3:2	Reserved	Must write reset value.
1:0	CMD	**Count Mode.** 00: The timer is free running mode on the RTC0 timer clock (RTC0TCLK). 01: The timer is incremented on the rising edges of the selected external trigger (LPTnTx). 10: The timer is incremented on the falling edges of the selected external trigger (LPTnTx). 11: The timer is incremented on both edges of the selected external trigger (LPTnTx).

Notes:
1. When writing to any of the LPTIMER registers, the CONTROL.RUN bit must be set to 1.
2. When writing and reading registers, a bit or field may take 3 APB and 2 RTC0TCLK cycles to update to a written value.

Register 27.2. LPTIMER0_DATA: Timer and Comparator Data

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	DATA															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

LPTIMER0_DATA = 0x4003_8010

Table 27.4. LPTIMER0_DATA Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15:0	DATA	**Timer and Comparator Data.** This field provides read and write access to both the internal timer and compare registers.

Notes:

1. When writing to any of the LPTIMER registers, the CONTROL.RUN bit must be set to 1.
2. When writing and reading registers, a bit or field may take 3 APB and 2 RTC0TCLK cycles to update to a written value.

Register 27.3. LPTIMER0_STATUS: Module Status

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved														CMPI	OVFI
Type	R														RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

LPTIMER0_STATUS = 0x4003_8020

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 27.5. LPTIMER0_STATUS Register Bit Descriptions

Bit	Name	Function
31:2	Reserved	Must write reset value.
1	CMPI	**Timer Compare Event Interrupt Flag.** Hardware sets this flag to 1 when the timer equals the compare threshold. Firmware must clear this flag and poll until it reads back as zero.
0	OVFI	**Timer Overflow Interrupt Flag.** Hardware sets this flag to 1 when a timer overflow occurs. Firmware must clear this flag and poll until it reads back as zero.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.
2. When writing to any of the LPTIMER registers, the CONTROL.RUN bit must be set to 1.
3. When writing and reading registers, a bit or field may take 3 APB and 2 RTC0TCLK cycles to update to a written value.

27.9. LPTIMER0 Register Memory Map

Table 27.6. LPTIMER0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
LPTIMER0_CONTROL	0x4003_8000	ALL | SET | CLR	RUN	DBGMD	Reserved					CMPRSTEN	Reserved				CMPTMD	OVFTMD	CMPIEN	OVFIEN	Reserved				CMPCAP	CMPSET	TMRCAP	TMRSET	EXTSEL				Reserved		CMD	
LPTIMER0_DATA	0x4003_8010	ALL	Reserved																DATA															
LPTIMER0_STATUS	0x4003_8020	ALL | SET | CLR	Reserved																														CMPI	OVFI

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

28. Enhanced Programmable Counter Array (EPCA0)

This section describes the Enhanced Programmable Counter Array (EPCA) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the EPCA block, which is used by all device families covered in this document.

28.1. Enhanced Programmable Counter Array Features

The Enhanced PCA module is a multi-purpose counter array with features designed for motor control applications. The EPCA module includes:

- Three sets of channel pairs (six channels total) capable of generating complementary waveforms.
- Center- and edge-aligned waveform generation.
- Programmable dead times that ensure channel pairs are never both active at the same time.
- Programmable clock divisor and multiple options for clock source selection.
- Waveform update scheduling.
- Option to function while the core is inactive.
- Multiple synchronization triggers and outputs to synchronize with other blocks in the device, such as the SARADCs.
- Pulse-Width Modulation (PWM) waveform generation.
- High-speed square wave generation.
- Input capture mode.
- DMA capability for both input capture and waveform generation.
- PWM generation halt input.

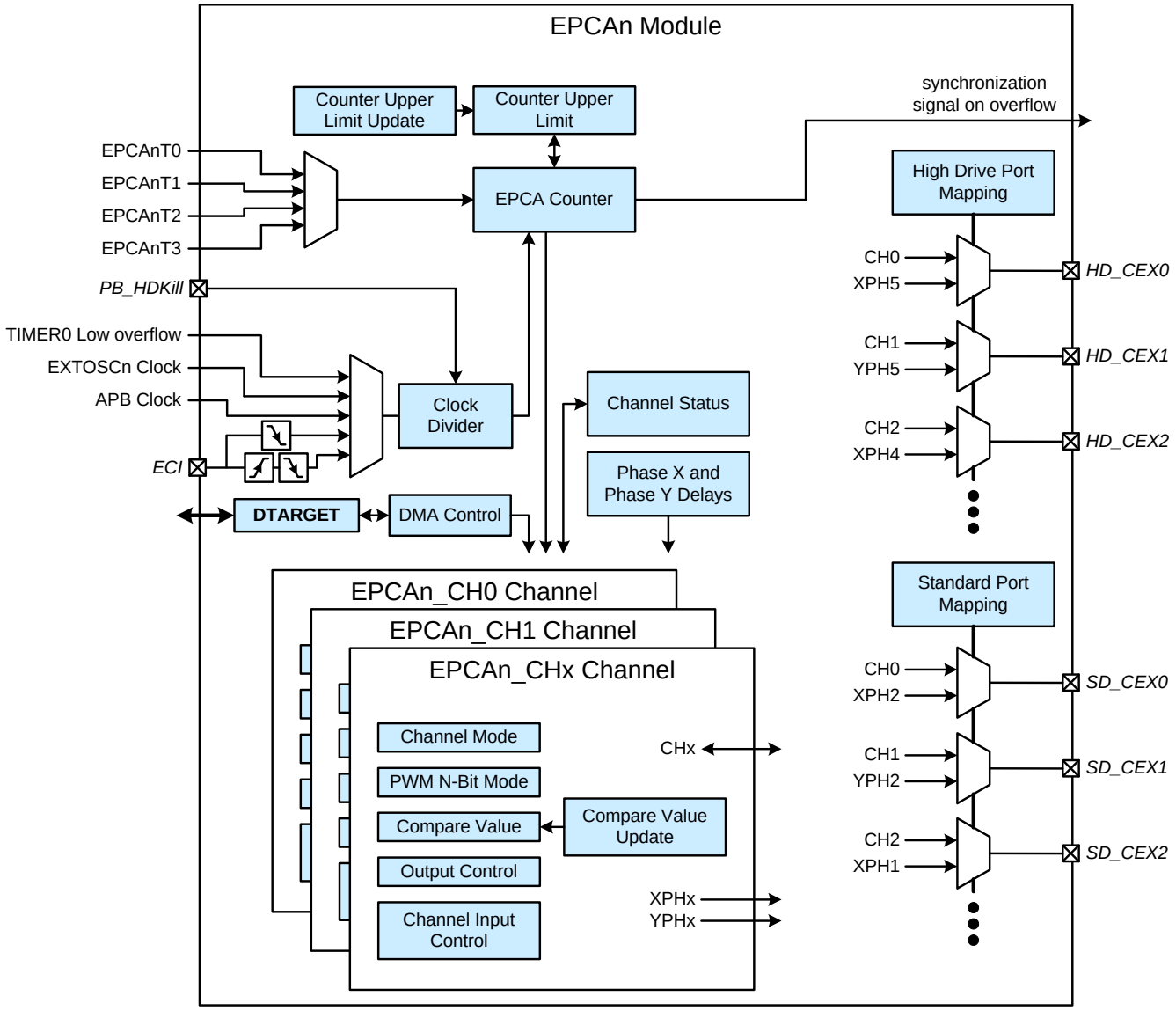


Figure 28.1. EPCA Block Diagram

28.2. Module Overview

The Enhanced Programmable Counter Array (EPCA) provides flexible timer/counter functionality. The EPCA consists of a dedicated 16-bit counter/timer and multiple (up to six) 16-bit capture/compare channels (EPCA0_CHx). All channels trigger or capture based on the shared 16-bit counter/timer, so the channels are inherently synchronized. The counter/timer is a 16-bit up counter with a programmable upper count limit. The counter starts at 0 and counts continuously from zero to the upper limit. Each channel includes a data register that can compare against the counter or capture the state of the counter based on a set of programmable conditions.

Each channel has its own associated output lines and may be configured to operate independently of the others in one of six modes: Edge-Aligned PWM, Center-Aligned PWM, High-Frequency / Square Wave, Timer / Capture, n-bit Edge-Aligned PWM, or Software Timer. The output lines may be single (one per channel, CHx) or differential (a complimentary pair per channel, XPHx and YPHx). These signals are then mapped to the HD_CEXx and STD_CEXx EPCA signals based on the STDOSEL and HDOSEL fields.

The RUN bit starts or stops the EPCA counter, but there are also additional controls that can halt the counter. The counter clock is normally suspended when the core halts to save power. The DBGMD bit in the CONTROL register controls whether the EPCA counter runs while the core is halted in debug mode. The EPCA also has an active-low external signal input, PB_HDKill, which can be configured to enter the “safe state” on PB4 pins by setting the SSMDEN bit in the PBHD0_PBSS register. In turn, when SSMDEN is set, the EPCA counter will halt and an interrupt can be generated. To resume operation, the PB_HDKill signal must be de-asserted and the SSMDEN bit must be cleared.

The EPCA additionally has up to four external trigger signals that can start the counter. The STSEL field in the MODE register determines which of the trigger sources can start the counter. STESEL selects the active polarity of the external signal, and setting STEN to 1 enables the external signal to start the counter. Once the counter is running, the external trigger will not stop the counter.

Three different events can change a channel output or cause a capture. An overflow/limit event occurs when the EPCA counter (COUNTER) is equal to the upper limit (LIMIT). A capture/compare event occurs whenever a channel’s capture/compare register (CCAPV) matches the current EPCA counter value. An intermediate overflow event can only occur in n-bit PWM mode and occurs when the channel’s n-bit capture/compare range matches the EPCA counter.

To facilitate counter and channel updates while the counter continues to run, the counter upper limit register (LIMIT) and channel compare/capture register (CCAPV) have update registers (LIMITUPD and CCAPVUPD, respectively). Firmware can write new values to the LIMITUPD and CCAPVUPD registers without modifying the current EPCA cycle. The hardware will load these values into the LIMIT and CCAPV registers when the next counter overflow/limit event occurs, and the module UPDCF and channel CUPDCF flags indicate when an update operation completes. Firmware can set the NOUPD bit to 1 to prevent updates from occurring.

28.3. Clocking

The clock input for the counter/timer is selected by the CLKSEL field in the MODE register as shown in Figure 28.2.

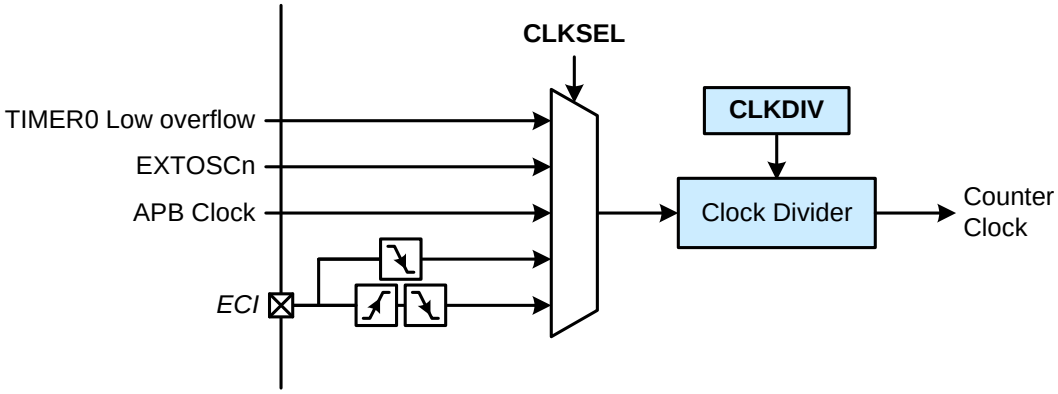


Figure 28.2. Clock Source Selection

The selected clock is passed through a divide-by-n clock divider where n can be between 1 and 1024. The CLKDIV field sets the clock divider for the selected clock before it drives the 16-bit counter/timer.

All non-APB clock sources must be synchronized with the APB clock, so the maximum possible operating speed using these sources is one-half the APB frequency, if not slower.

28.3.1. APB Clock

When the APB clock signal is selected as the counter clock source, the EPCA counter clock divider will use the APB clock source defined by the device clock control module. Selecting the APB clock provides the fastest clock source for the module.

28.3.2. External Clock (EXTOSCn)

When the external clock is selected as the counter clock source, the counter will run from the external clock source (EXTOSCn), regardless of the clock selection of the core. The external clock source is synchronized to the selected core clock in this mode. In order to guarantee that the external clock transitions are recognized by the device, the external clock signal must be high or low for at least one APB clock period. This limits the maximum frequency of an external clock in this mode to one-half the APB clock.

28.3.3. TIMER0 Low Overflow

The EPCA module can select the TIMER0 module low timer overflows as its clock source. TIMER0 can operate in either 32-bit or 16-bit mode and still provide the clock for the EPCA clock divider. The maximum speed for the TIMER0 low overflows as an EPCA counter clock source is one-half the APB clock.

28.3.4. External Clock Input (ECI)

When the external clock input (ECI) is selected as the EPCA clock source, the clock divider decrements on falling edges or both rising and falling edges of the pin. The ECI pin is synchronized to the selected AHB clock in this mode. The maximum clock rate for the ECI external clock input is the APB divided by 4.

28.3.5. Clock Divider

The clock divider provides a flexible time base for the EPCA counter. The divider starts at one-half the CLKDIV value and decrements to 0. Using this method, the divider counts the number of input clocks until the next counter clock edge (either rising or falling) rather than whole counter clock periods.

The current value of the divider can be read and written using the DIV field in the MODE register. This allows access during debugging to observe module events at the various EPCA clock edges rather than stepping through a large number of APB clocks. Firmware should always write to the CLKDIV field to modify the divider value.

The DIVST bit displays the current output phase of the clock divider.

28.4. Interrupts

The EPCA module has one interrupt vector and multiple interrupt sources within the module.

The module has two interrupt sources: a counter overflow/limit and the halt input. The OVFIEN bit enables the counter overflow/limit interrupt, which occurs when counter (COUNTER) equals the upper limit (LIMIT) and resets to 0. The OVFI interrupt flag indicates when an counter overflow/limit event occurs. The halt input, PB_HDKill, must be enabled (HALTEN = 1) and enabled as an interrupt source (HALTIEN = 1) in order to generate an interrupt. The HALTI interrupt flag indicates that an external signal input interrupt has occurred. The flag is cleared after a reset and set following the assertion of the halt input, if enabled.

Each channel (CHx) has a compare/capture interrupt flag (CxCCI) and an intermediate overflow flag (CxIOVFI). These interrupts can be enabled in the channel registers (CCIEN for capture/compare, CIOVFIEN for intermediate overflow).

Firmware can check the source of the EPCA interrupt by checking the appropriate flags in the interrupt service routine.

28.5. Outputs

The EPCA module has up to six module outputs (CEXx) that can be routed to physical pins by configuring the device port configuration module.

Each EPCA capture/compare channel has two independent output modes: single and differential. In single output mode, the channel has one CHx output. In differential mode, the channel has two XPHx and YPHx outputs. The mapping of these channel outputs to the EPCA module outputs (HD_CEXx and STD_CEXx) is determined by the HDOSEL and STDOSEL fields.

The HDOSEL field determines the output mapping for high drive ports, and the STDOSEL field determines the mapping for standard port banks. The mappings are selected such that all six potential channels can operate in differential mode at the same time: three (0, 1, and 2) are available for the standard port banks, and three (3, 4, and 5) are available for the high drive ports.

Table 28.1 shows the output mapping for the high drive ports, and shows the output mapping for standard ports.

Table 28.1. High Drive Output Mapping

EPCA Output	HDOSEL Value			
	0	1	2	3
HD_CEX0	XPH3	CH2	CH1	CH0
HD_CEX1	YPH3	CH3	CH2	CH1
HD_CEX2	XPH4	XPH4	CH3	CH2
HD_CEX3	YPH4	YPH4	CH4	CH3
HD_CEX4	XPH5	XPH5	XPH5	CH4
HD_CEX5	YPH5	YPH5	YPH5	CH5

Table 28.2. Standard Output Mapping

EPCA Output	STDOSEL Value			
	0	1	2	3
STD_CEX0	CH0	CH0	CH0	XPH0
STD_CEX1	CH1	CH1	CH1	YPH0
STD_CEX2	CH2	CH2	XPH1	XPH1
STD_CEX3	CH3	CH3	YPH1	YPH1
STD_CEX4	CH4	XPH2	XPH2	XPH2
STD_CEX5	CH5	YPH2	YPH2	YPH2

Note: The COUTST (and optionally XPHST, YPHST, and ACTIVEPH) bit determines the starting state of the channel output. If the starting state is active, this means the channel could be sitting in an active state for some time while firmware finishes initializing the module and starts the counter. If the output is connected to a transistor where this behavior is undesireable, firmware can initialize the counter to a mid-range value and the outputs with an inactive value. This will ensure that any sensitive external circuits will not be damaged.

28.5.1. Single Output Mode

When a channel operates in single output mode, the COUTST bit in the EPCA channels determines the polarity of the output. This value should be set when the counter is not running to ensure predictable operation. Hardware sets the COUTST bit on the rising edges of the APB clock to reflect the current output state of the channel.

Figure 28.3 shows an EPCA channel single output timing diagram.

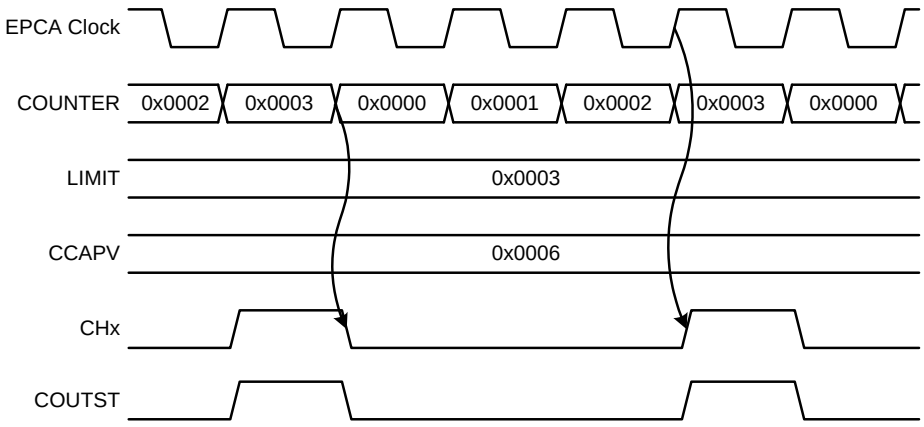


Figure 28.3. Example Channel Single Output Timing Diagram (Toggle Mode)

The CHx output can toggle, set, or clear on an overflow/limit, compare, or intermediate overflow event. In addition, the output can ignore these events and stay at its previous value. The COSEL field in the channel MODE register determines the CHx output behavior. This field allows firmware to modify the behavior of the CHx output without changing the configuration of the channel, interrupting the counter, or changing the port configuration.

28.5.2. Differential Output Mode

A channel additionally generates two synchronous outputs when operating in differential mode (DIFGEN = 1).

The YPHST and XPHST bits in the EPCA channels determine the polarity of the outputs. These values should be set when the counter is not running to ensure predictable operation. These bits can be read at any time to determine the current states of the outputs.

In addition, the differential outputs have programmable dead-time delays (AHB clocks) that prevent both channels from being active at the same time. The X delay time (DTIMEX) starts when the XPHx output switches to its inactive state. The YPHx output then switches to its active state when the time expires, and the Y delay time (DTIMEY) behaves similarly. These delay times are global for all channels.

The channel ACTIVEPH bit indicates which output is currently active. The active channel can be asserted or deasserted pending the dead time delay timeout.

Figure 28.4 shows an EPCA channel differential output timing diagram.

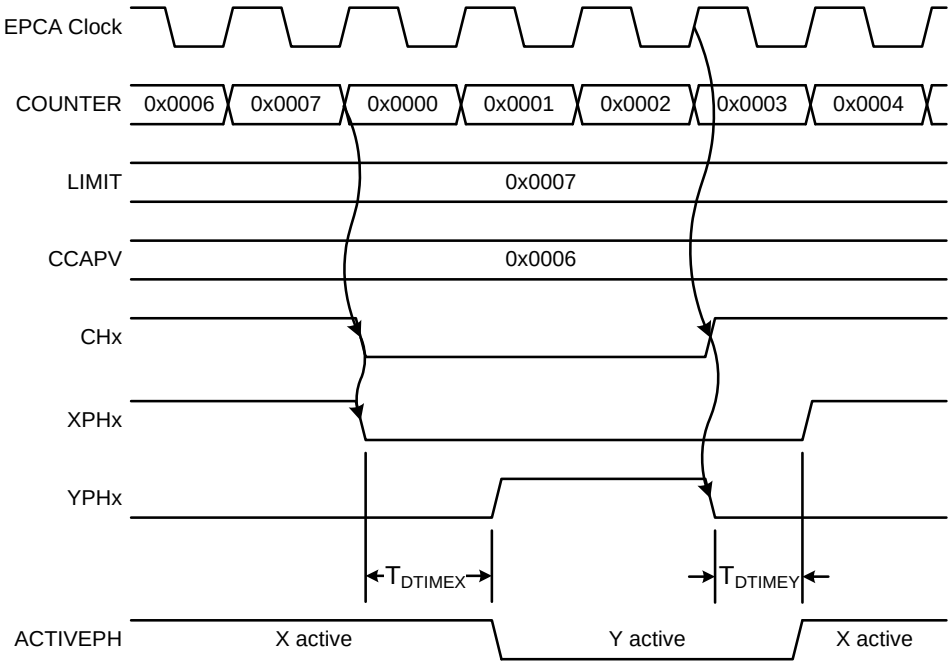


Figure 28.4. Example Channel Differential Output Timing Diagram

The output behavior of the XPHx and YPHx outputs depends on the behavior of the CHx output. These phase outputs will toggle as long as CHx is toggling. If the COSEL field is set such that the CHx output is no longer toggling (set, clear, or ignore), the XPHx and YPHx outputs will not change state. As long as the hardware control of the CHx output remains static, firmware can manually stimulate the XPHx and YPHx outputs by writing COUTST to different states and directly controlling the CHx output.

28.5.3. Synchronization Signal

In addition to the CHx, XPHx, and YPHx outputs, the EPCA module can generate a synchronization signal for use by other modules on the device (SARADC0, SARADC1, TIMER0, and TIMER1). This signal can pulse when a counter overflow/limit (OVFSEN = 1), channel intermediate overflow (CIOVFSEN = 1), or channel capture/compare (CCSEN = 1) event occurs.

28.6. Triggers

The EPCA supports four trigger sources. The selections for the trigger are made in the STSEL, STESEL and STEN fields of the CONTROL register, The trigger sources for SiM3U1xx/SiM3C1xx devices are defined in Table 28.3.

Table 28.3. EPCA0 Triggers

EPCA0 Trigger	EPCA0 Trigger Description	Internal Signal
EPCA0T0	Internal Trigger Source	Comparator 0 output
EPCA0T1	Internal Trigger Source	Comparator 1 output
EPCA0T2	Internal Trigger Source	Timer 0 High Overflow output
EPCA0T3	Internal Trigger Source	Timer 1 High Overflow output

To use a trigger:

1. Set up the desired trigger source module or channel.
2. Select the trigger source using the STSEL field.
3. Set the polarity of the trigger using the STESEL bit.
4. Enable the trigger (STEN = 1).
5. Set up the desired EPCA counter and channel settings.
6. Start the EPCA counter by setting RUN to 1.

The EPCA counter will start running when the selected trigger source meets the criteria set by the STESEL polarity. The counter will then continue to run regardless of the state of the trigger source.

28.7. Operational Modes

The EPCA module has six operational modes that each channel can independently select: Edge-Aligned PWM, Center-Aligned PWM, High-Frequency/Square Wave, Timer/Capture, n-bit Edge-Aligned PWM, and Software Timer modes. The CMD bits select the channel operational mode.

The EPCA counter is 16 bits and counts in full EPCA clock cycles. The channel CCAPV registers are 18 bits and represent half EPCA clock cycles. To match an counter value, the CCAPV field should be written with double the counter value. The CCAPV LSB provides timing resolution to one half-clock counter cycle. The additional 18th bit of the CCAPV register allows the channel outputs to create 0-100% duty cycles (0x00000 is 0% and 0x20000 is 100% duty cycle).

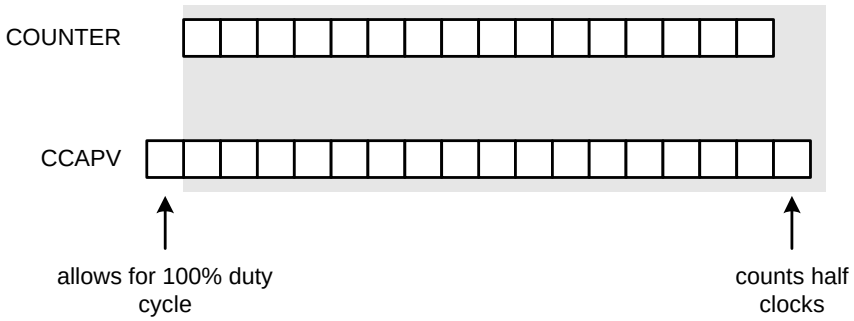


Figure 28.5. EPCA Counter and Channel Compare/Capture Registers

This section discusses the channel behavior in each of these modes in detail.

28.7.1. Edge-Aligned Pulse Width Modulation (PWM) Mode

In edge-aligned PWM mode (CMD = 0), the 18-bit capture/compare register (CCAPV) defines the number of EPCA half clocks for the inactive time of the PWM signal. A capture/compare event occurs when the counter matches the register contents, and the output will change from the initial state set by COUTST depending on the selected COSEL value (typically 00b for toggle). An overflow/limit event occurs when the counter reaches the LIMIT value and resets to 0, and CHx will again change depending on the COSEL value. To output a varying duty cycle, firmware can write to the CCAPVUPD register, which hardware will automatically load into the channel's CCAPV register on the next counter overflow/limit if the module register update inhibit (NOUPD) is cleared to 0.

Assuming that the inactive and initial state of the output is low and COSEL is set to toggle, the CHx output duty cycle in edge-aligned PWM mode is shown in Equation 28.1. No output pulse is generated if CCAPV is 0, and 100% duty cycle results from CCAPV set to 0x20000. The resulting XPHx and YPHx timing also depends on the dead-time delay values.

Figure 28.6 shows an example edge-aligned PWM timing diagram.

$$\text{Duty Cycle} = \frac{((\text{LIMIT} + 1) \times 2) - \text{CCAPV}}{(\text{LIMIT} + 1) \times 2}$$

Equation 28.1. Edge-Aligned PWM Duty Cycle

Because CCAPV is given in half clocks, an odd CCAPV value results in the CHx output changing state at the mid-cycle edge of the EPCA clock.

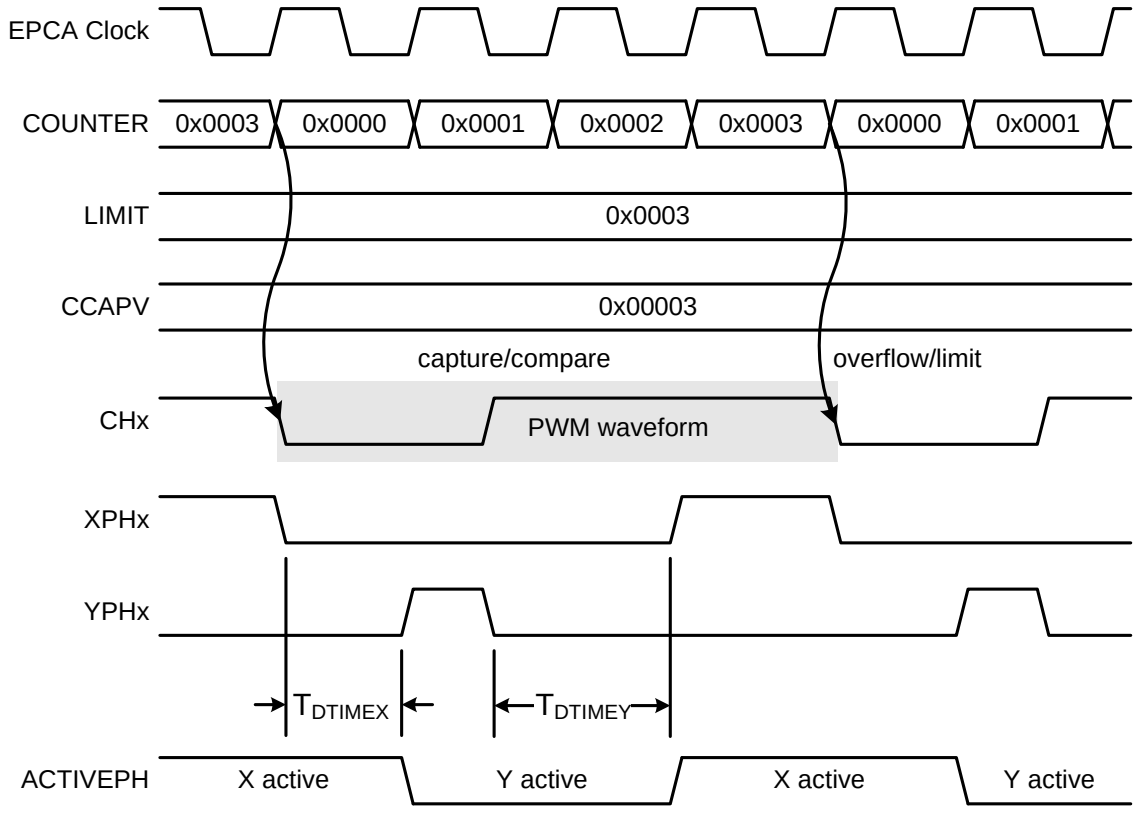


Figure 28.6. Example Edge-Aligned PWM Timing Diagram

28.7.2. Center-Aligned Pulse Width Modulation (PWM) Mode

In center-aligned PWM mode (CMD = 1), a channel generates a PWM waveform symmetric about the counter's overflow/limit event defined by the LIMIT field. Multiple channels in an array configured in this mode will each have PWM waveforms which are all symmetric about the same center-pulse points (counter equal to zero).

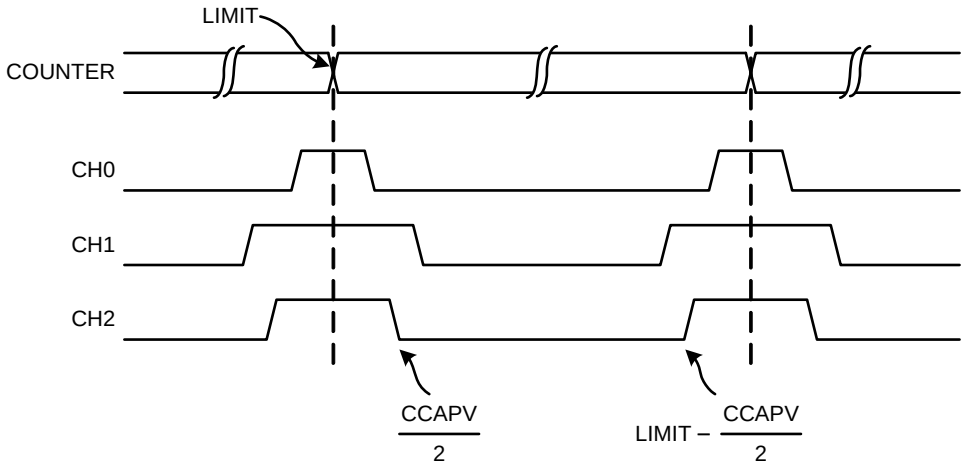


Figure 28.7. Multiple Center-Aligned PWM Channels

The channel 18-bit capture/compare register (CCAPV) is used to generate two capture/compare events in one counter cycle (0 to LIMIT). The first event occurs when the counter is equal to CCAPV divided by 2. The second event occurs when the counter is equal to LIMIT minus CCAPV divided by 2. In the event of an odd value in CCAPV, the hardware adds the extra half cycle to the capture/compare event following the counter overflow/limit event.

Firmware can write to the channel's CCAPVUPD register to update the waveform. Hardware will update the CCAPV field with the CCAPVUPD value when the counter overflows from the upper limit to zero as long as the update inhibit bit (NOUPD) is cleared to 0.

The COUTST (and optionally XPHST, YPHST, and ACTIVEPH) bit determines the starting state of the channel output and should be set before starting the counter. If the starting state is active, this means the channel could be sitting in an active state for some time while firmware finishes initializing the module and starts the counter. If the output is connected to a transistor where this behavior is undesireable, firmware can initialize the counter to a mid-range value and the outputs with an inactive value. This will ensure that any sensitive external circuits will not be damaged.

Assuming that the active and initial state of the output is high and COSEL is set to toggle, the CHx output duty cycle in center-aligned PWM mode is shown in Equation 28.2. No output pulse is generated if CCAPV is 0, and 100% duty cycle results when CCAPV (in half clocks) is set to a number of full clocks equal to or greater than LIMIT. The resulting XPHx and YPHx timing also depends on the dead-time delay values.

Figure 28.8 shows an example center-aligned PWM timing diagram.

$$\text{Duty Cycle} = \frac{(\text{LIMIT} \times 2) - \text{CCAPV}}{\text{LIMIT} \times 2}$$

Equation 28.2. Center-Aligned PWM Duty Cycle

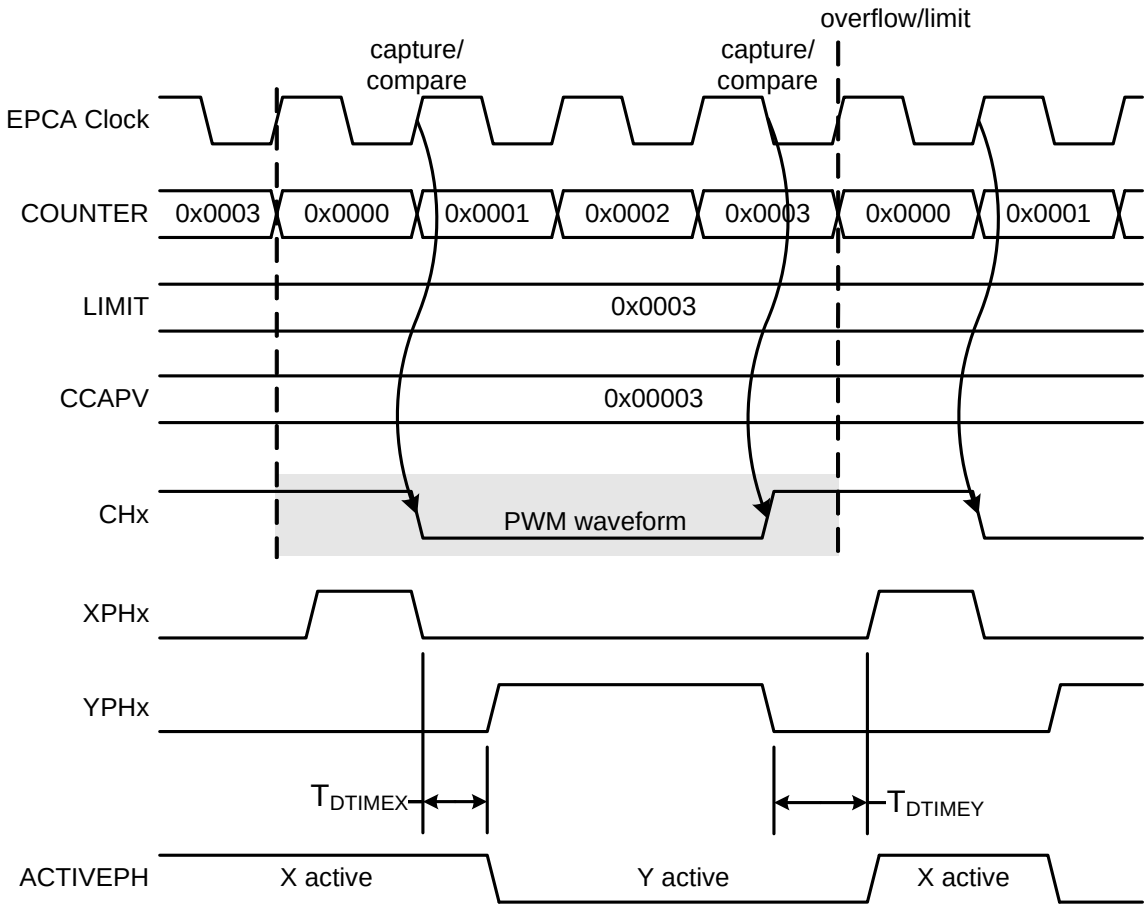


Figure 28.8. Example Center-Aligned PWM Timing Diagram

28.7.3. High-Frequency / Square Wave Mode

High-frequency square-wave mode (CMD = 2) produces a 50% duty cycle waveform of programmable period on the channel's CHx output. This mode provides a flexible way of creating square waves with a fast period. Each half-period of the generated output clock can be as short as one counter clock period (two CCAPV half clocks) and as long as 128 counter clock periods, programmable in half clock steps.

Bits [15:8] of the channel capture/compare register (CCAPV) contain the waveform half period in the number of half EPCA clocks. The lower byte (bits [7:0]) is the calculated value compared to the counter. When a match occurs between the lower byte and the counter, the hardware triggers a capture/compare event and automatically adds the match value of the counter to the CCAPV[15:8] value to create a new compare value for the lower byte. An overflow/limit event occurs when the counter reaches the upper limit defined by the LIMIT field. Firmware should program the upper limit register to reset the counter at a (multiple of 128) - 1 to avoid undesired waveform edges. Figure 28.9 shows how the hardware creates the waveform.

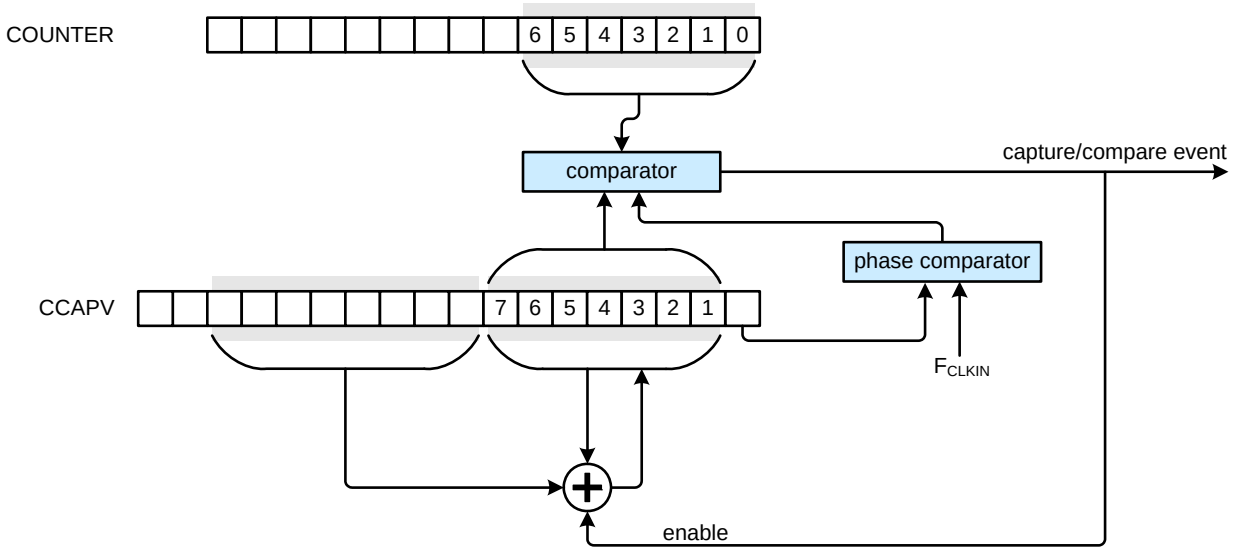


Figure 28.9. High-Frequency Square Wave Waveform Generation

Firmware can update the frequency of the output by writing to the CCAPVUPD register. Hardware will automatically load bits [15:8] of this register to the CCAPV register at the next counter overflow/limit event, if possible (NOUPD = 0). The rest of the CCAPVUPD register (bits [17:16] and [7:0]) are ignored.

Assuming that the COSEL field for the channel is set to toggle, the resulting CHx output frequency in high-frequency square-wave mode is shown in Equation 28.3. A CCAPV[15:8] value of 1 is not valid, and a CCAPV[15:8] value of 0 results in an output waveform half period of 128 counter clocks. The resulting XPHx and YPHx timing also depends on the dead-time delay values.

Figure 28.10 shows an example high-frequency square wave mode timing diagram.

$$F_{CHx} = \frac{F_{EPCA}}{CCAPV[15:8]}$$

Equation 28.3. High-Frequency Square Wave Output Frequency

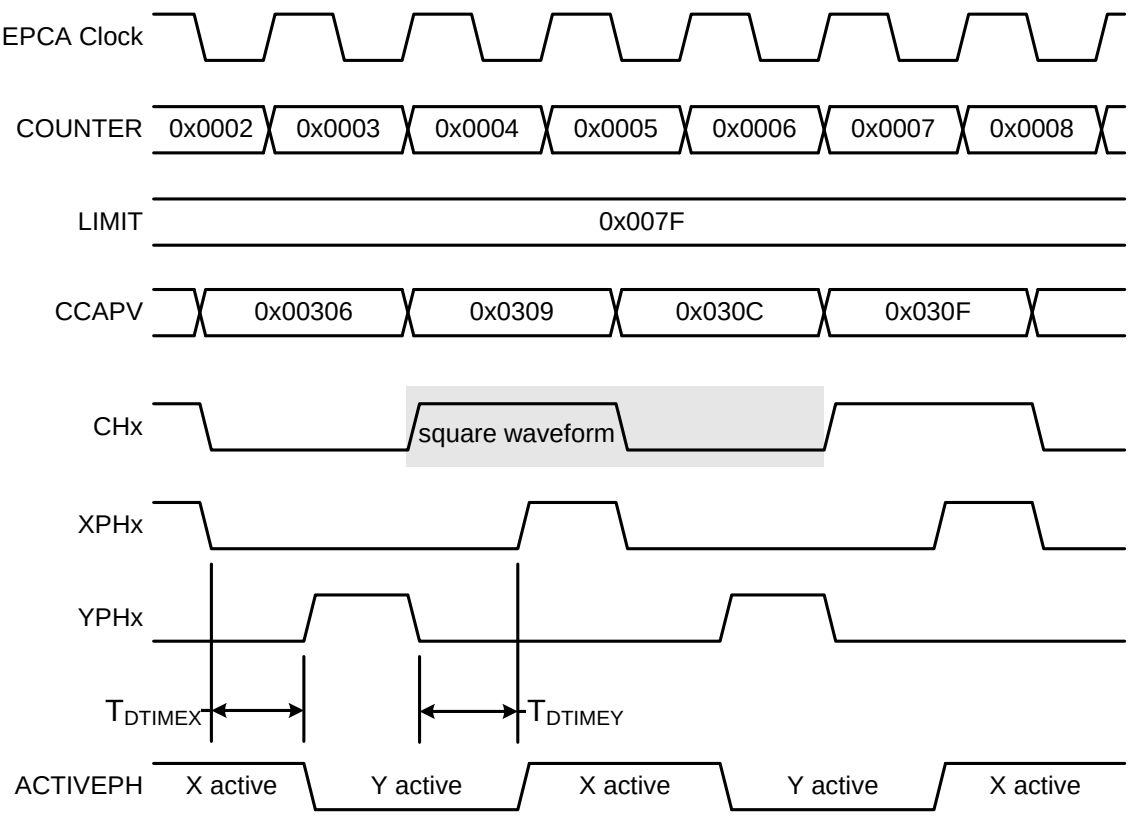


Figure 28.10. Example High-Frequency Square Wave Mode Timing Diagram

28.7.4. Timer / Capture Mode

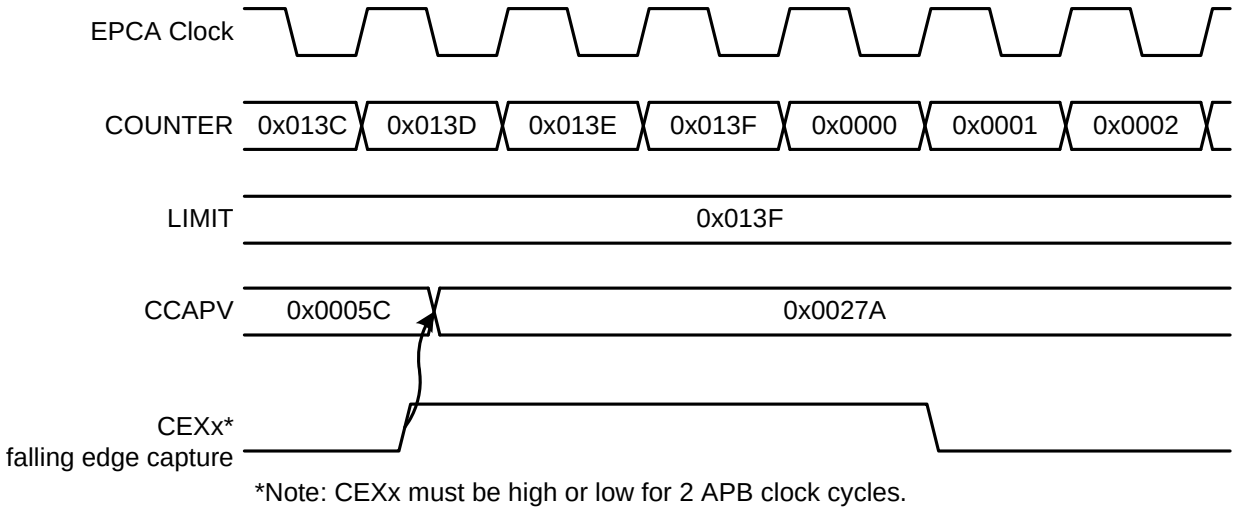
In timer/capture mode (CMD = 3), the channel uses the CEXn line as an input signal that triggers a capture/compare event and stores the counter state in the channel capture/compare register (CCAPV) in half clocks. Firmware can configure the event to trigger on rising, falling, or both edges using the channel CPCAPEN and CNCAPEN bits. Table 28.4 shows the capture edge configuration options.

Table 28.4. Capture Edge Configuration Options

CPCAPEN	CNCAPEN	Selected Edge
0	0	Capture disabled.
0	1	Capture on the falling edge.
1	0	Capture on the rising edge.
1	1	Capture on both edges.

The input CEXn signal must remain high or low for at least two APB clocks to be recognized by the hardware. The capture occurs at the next EPCA clock edge.

Figure 28.11 shows an example timer/capture mode timing diagram.



*Note: CEXx must be high or low for 2 APB clock cycles.

Figure 28.11. Example Timer / Capture Mode Timing Diagram

28.7.5. N-bit Edge-Aligned Pulse Width Modulation (PWM) Mode

The n-bit edge-aligned PWM modes allow each channel to be independently configured to generate edge-aligned PWM waveforms with a faster duty cycle than the counter. In n-bit edge-aligned PWM mode (CMD = 4), the least-significant n bits (set by PWMMD) of the counter define the number of full clocks of the PWM waveform. The channel's capture/compare register (CCAPV) defines the number of EPCA half clocks for the inactive time of the PWM signal, and the higher order unused bits of CCAPV are ignored. A capture/compare event occurs when the n-bit counter is equal to the number of half clocks defined by the CCAPV field. An intermediate overflow event occurs when the counter overflows the n-bit range. A counter overflow/limit event occurs when the counter reaches the upper limit (LIMIT) within the full 16-bit range. If one of the channels operates in n-bit mode, firmware should set the counter's upper limit as an even multiple of the n-bit boundary to ensure the channel's output does not have any irregular edges from the overflow/limit events.

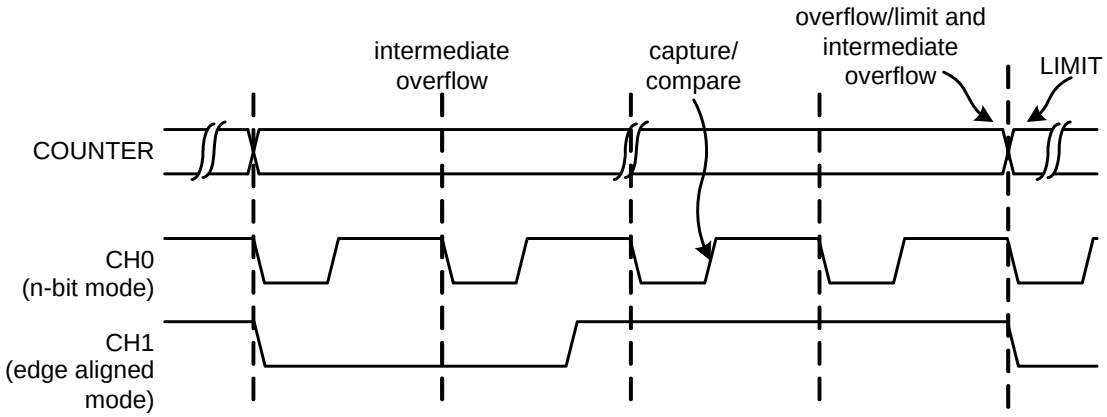


Figure 28.12. Multiple Edge-Aligned PWM Channels (N-bit and Edge Aligned)

Table 28.5 provides a list of the intermediate overflow, recommended upper limit values, and 100% duty cycle values for each n-bit setting.

Table 28.5. N-bit Intermediate Overflow Values

PWMMD Value	N-bit PWM Mode	Counter Intermediate Overflow Value (full clocks)	Recommended Counter Upper Limit (full clocks)	CCAPV 100% Duty Cycle Value (half clocks)
0	0-bit	0	any value	2
1	1-bit	1	(multiple of 2) - 1	4
2	2-bit	3	(multiple of 4) - 1	8
3	3-bit	7	(multiple of 8) - 1	16
4	4-bit	15	(multiple of 16) - 1	32
5	5-bit	31	(multiple of 32) - 1	64
6	6-bit	63	(multiple of 64) - 1	128
7	7-bit	127	(multiple of 128) - 1	256
8	8-bit	255	(multiple of 256) - 1	512
9	9-bit	511	(multiple of 512) - 1	1024
10	10-bit	1023	(multiple of 1024) - 1	2048
11	11-bit	2047	(multiple of 2048) - 1	4096
12	12-bit	4095	(multiple of 4096) - 1	8192
13	13-bit	8191	(multiple of 8192) - 1	16334
14	14-bit	16333	(multiple of 16334) - 1	32768
15	15-bit	32767	(multiple of 32768) - 1	65536

Firmware can write to the CCAPVUPD register to update the duty cycle, which hardware will automatically load into the channel's CCAPV register on the next counter overflow/limit event if the module register update inhibit (NOUPD) is cleared to 0.

Assuming that the inactive and initial state of the output is low, COSEL is set to toggle, and the upper limit is set to an even multiple, the CHx output duty cycle in n-bit edge-aligned PWM mode is shown in Equation 28.4. No output pulse is generated if CCAPV is 0. The resulting XPHx and YPHx timing also depends on the dead-time delay values.

Figure 28.13 shows an example n-bit edge-aligned PWM timing diagram.

$$\text{Duty Cycle} = \frac{(2^n \times 2) - \text{CCAPV}}{2^n \times 2}$$

Equation 28.4. N-bit Edge-Aligned PWM Duty Cycle

Because CCAPV is given in half clocks, an odd CCAPV value results in the CHx output changing state at the mid-cycle edge of the EPCA clock.

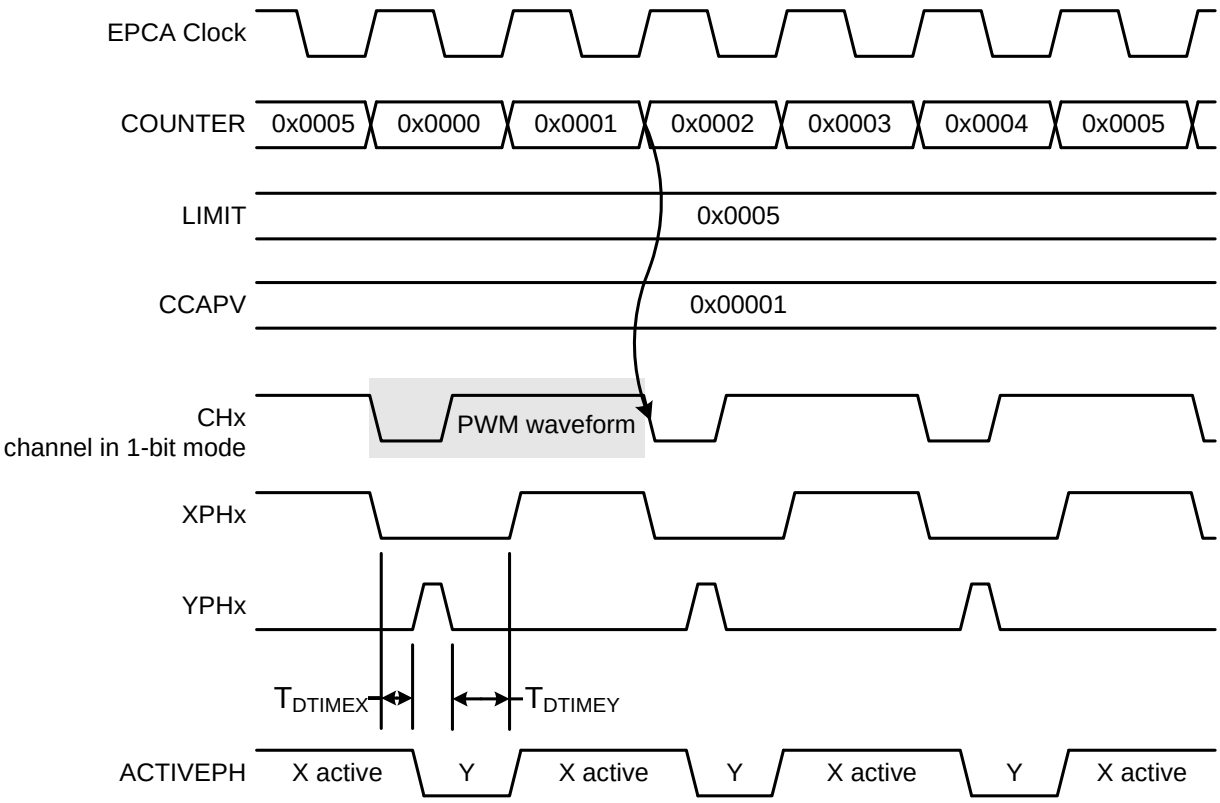


Figure 28.13. Example N-bit Edge-Aligned PWM Timing Diagram

28.7.6. Software Timer Mode

Software timer mode is a free-running mode with the CHx output unaffected by the counter or channel state. Setting the COSEL field to 3 (output ignore events) enters software timer mode, regardless of the CMD field setting. The counter overflow/limit, intermediate overflow, and capture/compare events can still be used to generate interrupts, even though the channel output is disabled.

Figure 28.14 shows an example software timer mode timing diagram.

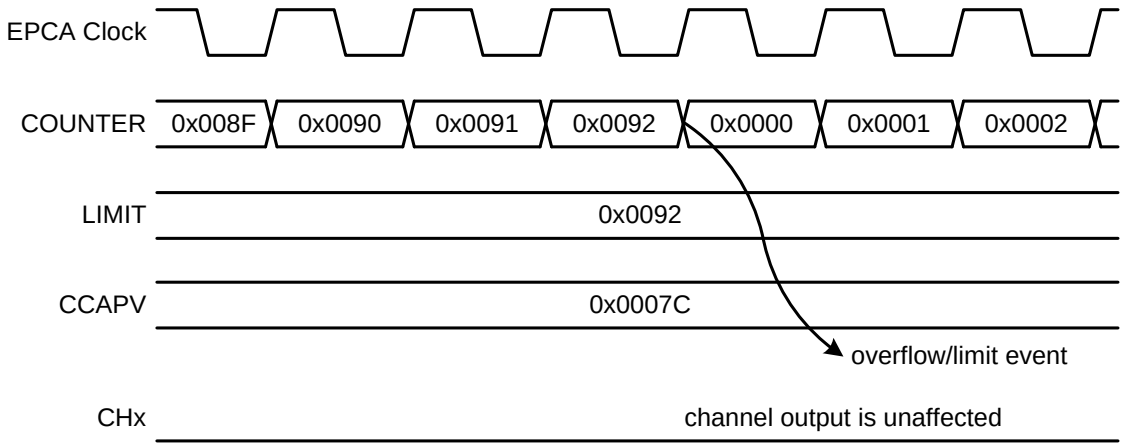


Figure 28.14. Example Software Timer Mode Timing Diagram

28.8. DMA Configuration and Usage

The EPCA supports two DMA channels: control and capture. The control requests move data from memory into the counter and channel update registers to autonomously set up new waveform generation. Capture requests move the counter capture data from the channel CCAPV register to memory.

The hardware can generate a DMA transfer request with a counter overflow/limit (OVFDEN = 1), channel intermediate overflow (CIOVFDEN = 1), or channel capture event (CCDEN = 1). When these events are enabled as a source for a DMA transfer, hardware automatically clears the flags associated with the request after the transfer completes. A DMA transfer to service a control request will not clear a flag for a pending capture service request.

The EPCA Module DMA configuration is shown in Figure 28.15.

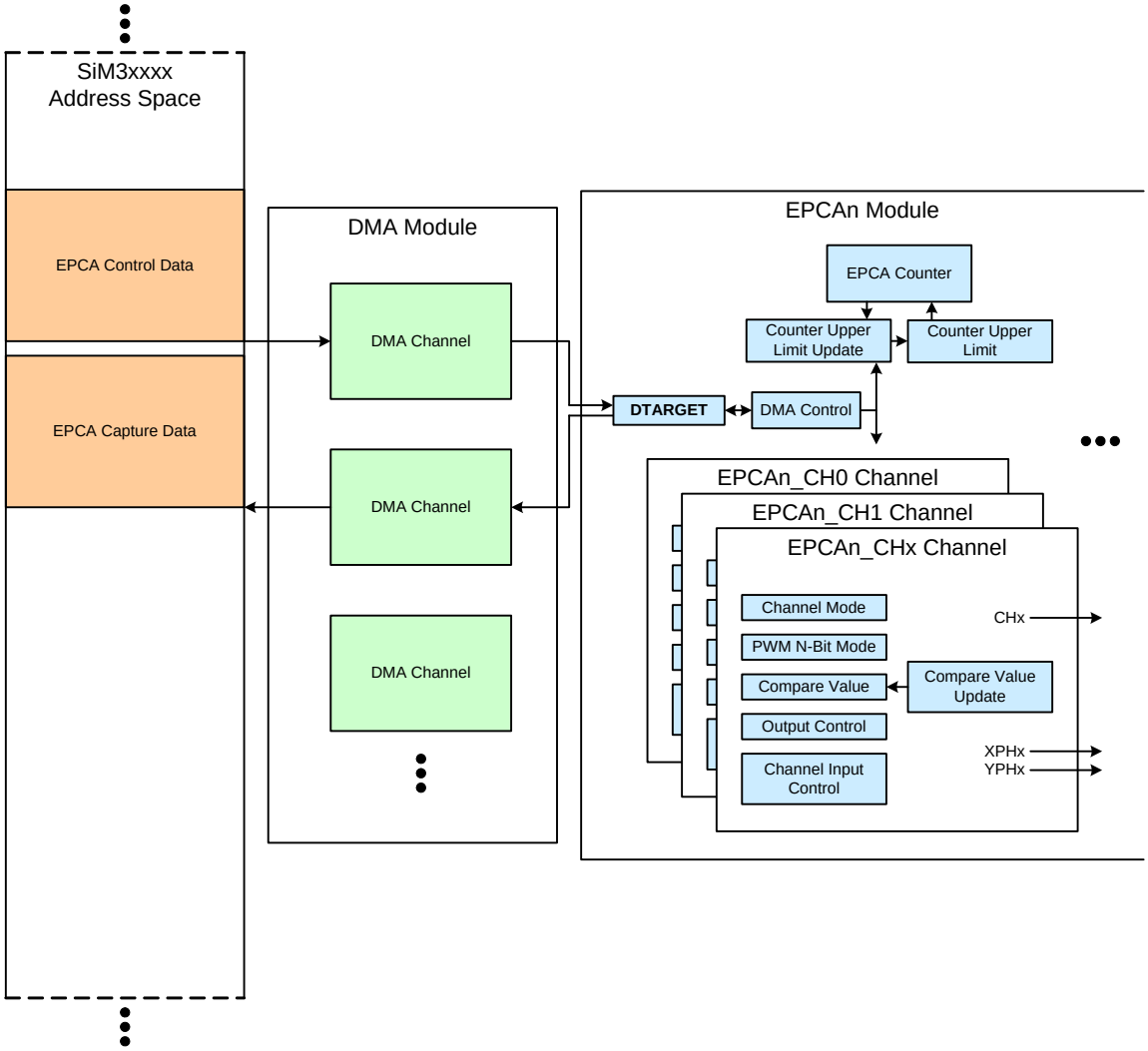


Figure 28.15. EPCA Module DMA Configuration

28.8.1. Control DMA Transfers

In a control transfer, the DMA should be configured to transfer one word at a time to the DTARGET location in non-incrementing mode. The module MODE register contains three pointers for use with a control transfer: DSTART, DEND, and DPTR. These pointers control the transfer of data from the DTARGET register to the module and channel update registers (LIMITUPD and CCAPVUPD). The DSTART pointer indicates the starting register in a new DMA write transfer, DEND indicates the ending register in the transfer, and DPTR is a circular pointer to the next register a DMA write to DTARGET will update.

Table 28.6. DMA Control Transfer Pointer Slots

DSTART, DEND, or DPTR Value	Register
0	LIMITUPD
1	Channel 0 CCAPVUPD
2	Channel 1 CCAPVUPD
3	Channel 2 CCAPVUPD
4	Channel 3 CCAPVUPD
5	Channel 4 CCAPVUPD
6	Channel 5 CCAPVUPD
7	empty

Slot 7 is not normally written unless DEND is set to 7, DSTART is set to 0, and the DMA uses 8-word transfers. In this case, the 8th data write is discarded.

A transfer starting at slot 5 (Channel 4 CCAPVUPD) can wrap around to end at an earlier slot. Any wraps ignore slot 7, so the sequence for a 4 DMA word transfers would be: slot 5, slot 6, slot 0, slot 1.

Firmware should not modify these fields while a DMA transfer is in progress, indicated by the DBUSYF flag.

28.8.2. Software DMA Transfers

Software DMA requests can be directly generated by two different software mechanisms:

1. Configure the EPCA for a DMA control request without enabling the DMA channel in the controller. Software can then write to the update registers by writing directly the DTARGET field.
2. Completely configure the DMA transfer (including the controller channel) and trigger DMA transfers by writing to the associated event interrupt flag.

28.9. EPCA0 Registers

This section contains the detailed register descriptions for EPCA0 registers.

Register 28.1. EPCA0_MODE: Module Operating Mode

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved			STDOSEL		Reserved	DBUSYF	DSTART			DPTR			DEND		
Type	R			RW		R	RW	RW			RW			RW		
Reset	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	HDOSEL		Reserved	CLKSEL			CLKDIV									
Type	RW		R	RW			RW									
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

EPCA0_MODE = 0x4000_E180

Table 28.7. EPCA0_MODE Register Bit Descriptions

Bit	Name	Function
31:29	Reserved	Must write reset value.
28:27	STDOSEL	**Standard Port Bank Output Select.** 00: Select the non-differential channel outputs (Channels 0-5) for the standard PB pins. 01: Select the differential output from Channel 2 and non-differential outputs from Channels 0, 1, 3, and 4 for the standard PB pins. 10: Select the differential outputs from Channels 1 and 2 and non-differential outputs from Channels 0 and 3 for the standard PB pins. 11: Select three differential outputs from Channels 0, 1, and 2 for the standard PB pins.
26	Reserved	Must write reset value.
25	DBUSYF	**DMA Busy Flag.** 0: The DMA channel is not servicing an EPCA control transfer. 1: The DMA channel is busy servicing an EPCA control transfer.

Table 28.7. EPCA0_MODE Register Bit Descriptions

Bit	Name	Function
24:22	DSTART	**DMA Target Start Index.** This field is the first register to be accessed in a new DMA write transfer set to DTARGET. This field should be written by software before the start of the DMA transfer. 000: Set the first register in a DMA write transfer to LIMITUPD. 001: Set the first register in a DMA write transfer to Channel 0 CCAPVUPD. 010: Set the first register in a DMA write transfer to Channel 1 CCAPVUPD. 011: Set the first register in a DMA write transfer to Channel 2 CCAPVUPD. 100: Set the first register in a DMA write transfer to Channel 3 CCAPVUPD. 101: Set the first register in a DMA write transfer to Channel 4 CCAPVUPD. 110: Set the first register in a DMA write transfer to Channel 5 CCAPVUPD. 111: Empty slot.
21:19	DPTR	**DMA Write Transfer Pointer.** This field is the current target of the DMA. The next word written to DTARGET will be transferred to the register selected by DPTR. This field is set by hardware and should not be modified by software. 000: The DMA channel will write to LIMITUPD next. 001: The DMA channel will write to Channel 0 CCAPVUPD next. 010: The DMA channel will write to Channel 1 CCAPVUPD next. 011: The DMA channel will write to Channel 2 CCAPVUPD next. 100: The DMA channel will write to Channel 3 CCAPVUPD next. 101: The DMA channel will write to Channel 4 CCAPVUPD next. 110: The DMA channel will write to Channel 5 CCAPVUPD next. 111: Empty slot.
18:16	DEND	**DMA Write End Index.** This field is the last register to be accessed in a DMA write transfer set. This field should be written by software before the start of the DMA transfer. 000: Set the last register in a DMA write transfer to LIMITUPD. 001: Set the last register in a DMA write transfer to Channel 0 CCAPVUPD. 010: Set the last register in a DMA write transfer to Channel 1 CCAPVUPD. 011: Set the last register in a DMA write transfer to Channel 2 CCAPVUPD. 100: Set the last register in a DMA write transfer to Channel 3 CCAPVUPD. 101: Set the last register in a DMA write transfer to Channel 4 CCAPVUPD. 110: Set the last register in a DMA write transfer to Channel 5 CCAPVUPD. 111: Empty slot.
15:14	HDOSEL	**High Drive Port Bank Output Select.** 00: Select three differential outputs from Channels 3, 4, and 5 for the High Drive pins. 01: Select the differential outputs from Channels 4 and 5 and non-differential outputs from Channels 2 and 3 for the High Drive pins. 10: Select the differential output from Channel 5 and non-differential outputs from Channels 1-4 for the High Drive pins. 11: Select the non-differential channel outputs (Channels 0-5) for the High Drive pins.
13	Reserved	Must write reset value.

Table 28.7. EPCA0_MODE Register Bit Descriptions

Bit	Name	Function
12:10	CLKSEL	**Input Clock (F_{CLKIN}) Select.** 000: Set the APB as the input clock (F_{CLKIN}). 001: Set Timer 0 low overflows divided by 2 as the input clock (F_{CLKIN}). 010: Set high-to-low transitions on ECI divided by 2 as the input clock (F_{CLKIN}). 011: Set the external oscillator module output (EXTOSCn) divided by 2 as the input clock (F_{CLKIN}). 100: Set ECI transitions divided by 2 as the input clock (F_{CLKIN}). 101-111: Reserved.
9:0	CLKDIV	**Input Clock Divider.** The EPCA module clock is given by the equation: $F_{EPCA} = \frac{F_{CLKIN}}{CLKDIV + 1}$ Where the input clock (CLKIN) is determined by the CLKSEL bits.

Register 28.2. EPCA0_CONTROL: Module Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	DIV										DIVST	Reserved				
Type	RW										RW	R				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved	STEN	STESEL	STSEL		Reserved	HALTEN	Reserved		DBGMD	Reserved	NOUPD	HALTIEN	OVFSEN	OVFDEN	OVFIEN
Type	R	RW	RW	RW		R	RW	R		RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

EPCA0_CONTROL = 0x4000_E190

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 28.8. EPCA0_CONTROL Register Bit Descriptions

Bit	Name	Function
31:22	DIV	**Current Clock Divider Count.** This field is the current value of the internal EPCA clock divider. The clock divider is a counter that starts at CLKDIV / 2 and counts down to zero.
21	DIVST	**Clock Divider Output State.** 0: The clock divider is currently in the first half-cycle. 1: The clock divider is currently in the second half-cycle.
20:15	Reserved	Must write reset value.
14	STEN	**Synchronous Input Trigger Enable.** 0: Disable the input trigger (EPCAnTx). The EPCA counter/timer will continue to run if the RUN bit is set regardless of the value on the input trigger. 1: Enable the input trigger (EPCAnTx). If RUN is set to 1, the EPCA counter/timer will start running when the selected input trigger (STSEL) meets the criteria set by STESEL. It will not stop running if the criteria is no longer met.
13	STESEL	**Synchronous Input Trigger Edge Select.** 0: A high-to-low transition (falling edge) on EPCAnTx will start the counter/timer. 1: A low-to-high transition (rising edge) on EPCAnTx will start the counter/timer.

Notes:

1. Because hardware updates the DIV and DIVST fields, the SET and CLR addresses are the only safe way to access the other fields in this register while the EPCA is running.

Table 28.8. EPCA0_CONTROL Register Bit Descriptions

Bit	Name	Function
12:11	STSEL	**Synchronous Input Trigger Select.** 00: Select input trigger 0, Comparator0 output. 01: Select input trigger 1, Comparator1 output. 10: Select input trigger 2, Timer 0 high overflow. 11: Select input trigger 3, Timer 1 high overflow.
10	Reserved	Must write reset value.
9	HALTEN	**Halt Input Enable.** 0: The Halt input (PB_HDKill) does not affect the EPCA counter/timer. 1: An assertion of the Halt input (PB_HDKill) will stop the EPCA counter/timer.
8:7	Reserved	Must write reset value.
6	DBGMD	**EPCA Debug Mode.** 0: A debug breakpoint will stop the EPCA counter/timer. 1: The EPCA will continue to operate while the core is halted in debug mode.
5	Reserved	Must write reset value.
4	NOUPD	**Internal Register Update Inhibit.** 0: The EPCA registers will automatically load any new update values after an overflow/limit event occurs. 1: The EPCA registers will not load any new update values after an overflow/limit event occurs.
3	HALTIEN	**EPCA Halt Input Interrupt Enable.** 0: Do not generate an interrupt if the EPCA halt input is high. 1: Generate an interrupt if the EPCA halt input is high.
2	OVFSEN	**EPCA Counter Overflow/Limit Synchronization Signal Enable.** The sychronization signal generated by the EPCA module can be used as an input by other modules to synchronize with a particular EPCA event or state. 0: Do not send a synchronization signal when a EPCA counter overflow/limit event occurs. 1: Send a synchronization signal when a EPCA counter overflow/limit event occurs.
1	OVFDEN	**EPCA Counter Overflow/Limit DMA Request Enable.** 0: Do not request DMA data when a EPCA counter overflow/limit event occurs. 1: Request DMA data when a EPCA counter overflow/limit event occurs.
0	OVFIEN	**EPCA Counter Overflow/Limit Interrupt Enable.** 0: Disable the EPCA counter overflow/limit event interrupt. 1: Enable the EPCA counter overflow/limit event interrupt.

Notes:

1. Because hardware updates the DIV and DIVST fields, the SET and CLR addresses are the only safe way to access the other fields in this register while the EPCA is running.

Register 28.3. EPCA0_STATUS: Module Status

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	C5IOVFI	C4IOVFI	C3IOVFI	C2IOVFI	C1IOVFI	C0IOVFI	HALTI	UPDCF	OVFI	RUN	C5CCI	C4CCI	C3CCI	C2CCI	C1CCI	C0CCI
Type	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

EPCA0_STATUS = 0x4000_E1A0

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 28.9. EPCA0_STATUS Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15	C5IOVFI	**Channel 5 Intermediate Overflow Interrupt Flag.** This bit is set by hardware when a counter overflows the n-bit range in Channel 5 n-bit PWM mode. This bit must be cleared by firmware. 0: Channel 5 did not count past the channel n-bit mode limit. 1: Channel 5 counted past the channel n-bit mode limit.
14	C4IOVFI	**Channel 4 Intermediate Overflow Interrupt Flag.** This bit is set by hardware when a counter overflows the n-bit range in Channel 4 n-bit PWM mode. This bit must be cleared by firmware. 0: Channel 4 did not count past the channel n-bit mode limit. 1: Channel 4 counted past the channel n-bit mode limit.
13	C3IOVFI	**Channel 3 Intermediate Overflow Interrupt Flag.** This bit is set by hardware when a counter overflows the n-bit range in Channel 3 n-bit PWM mode. This bit must be cleared by firmware. 0: Channel 3 did not count past the channel n-bit mode limit. 1: Channel 3 counted past the channel n-bit mode limit.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 28.9. EPCA0_STATUS Register Bit Descriptions

Bit	Name	Function
12	C2IOVFI	**Channel 2 Intermediate Overflow Interrupt Flag.** This bit is set by hardware when a counter overflows the n-bit range in Channel 2 n-bit PWM mode. This bit must be cleared by firmware. 0: Channel 2 did not count past the channel n-bit mode limit. 1: Channel 2 counted past the channel n-bit mode limit.
11	C1IOVFI	**Channel 1 Intermediate Overflow Interrupt Flag.** This bit is set by hardware when a counter overflows the n-bit range in Channel 1 n-bit PWM mode. This bit must be cleared by firmware. 0: Channel 1 did not count past the channel n-bit mode limit. 1: Channel 1 counted past the channel n-bit mode limit.
10	C0IOVFI	**Channel 0 Intermediate Overflow Interrupt Flag.** This bit is set by hardware when a counter overflows the n-bit range in Channel 0 n-bit PWM mode. This bit must be cleared by firmware. 0: Channel 0 did not count past the channel n-bit mode limit. 1: Channel 0 counted past the channel n-bit mode limit.
9	HALTI	**Halt Input Interrupt Flag.** This bit is set by hardware when the halt input is asserted. This bit must be cleared by firmware. 0: The Halt input (PB_HDKill) was not asserted. 1: The Halt input (PB_HDKill) was asserted.
8	UPDCF	**Register Update Complete Flag.** 0: An EPCA register update completed or is not pending. 1: An EPCA register update has not completed and is still pending.
7	OVFI	**Counter/Timer Overflow/Limit Interrupt Flag.** This bit is set by hardware when the counter reaches the value in LIMIT and overflows to zero. This bit must be cleared by firmware. 0: An EPCA Counter/Timer overflow/limit event did not occur. 1: An EPCA Counter/Timer overflow/limit event occurred.
6	RUN	**Counter/Timer Run.** 0: Stop the EPCA Counter/Timer. 1: Start the EPCA Counter/Timer.
5	C5CCI	**Channel 5 Capture/Compare Interrupt Flag.** This bit is set by hardware when a match or capture occurs in Channel 5. This bit must be cleared by firmware. 0: A Channel 5 match or capture event did not occur. 1: A Channel 5 match or capture event occurred.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 28.9. EPCA0_STATUS Register Bit Descriptions

Bit	Name	Function
4	C4CCI	**Channel 4 Capture/Compare Interrupt Flag.** This bit is set by hardware when a match or capture occurs in Channel 4. This bit must be cleared by firmware. 0: A Channel 4 match or capture event did not occur. 1: A Channel 4 match or capture event occurred.
3	C3CCI	**Channel 3 Capture/Compare Interrupt Flag.** This bit is set by hardware when a match or capture occurs in Channel 3. This bit must be cleared by firmware. 0: A Channel 3 match or capture event did not occur. 1: A Channel 3 match or capture event occurred.
2	C2CCI	**Channel 2 Capture/Compare Interrupt Flag.** This bit is set by hardware when a match or capture occurs in Channel 2. This bit must be cleared by firmware. 0: A Channel 2 match or capture event did not occur. 1: A Channel 2 match or capture event occurred.
1	C1CCI	**Channel 1 Capture/Compare Interrupt Flag.** This bit is set by hardware when a match or capture occurs in Channel 1. This bit must be cleared by firmware. 0: A Channel 1 match or capture event did not occur. 1: A Channel 1 match or capture event occurred.
0	C0CCI	**Channel 0 Capture/Compare Interrupt Flag.** This bit is set by hardware when a match or capture occurs in Channel 0. This bit must be cleared by firmware. 0: A Channel 0 match or capture event did not occur. 1: A Channel 0 match or capture event occurred.

Notes:
1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Register 28.4. EPCA0_COUNTER: Module Counter/Timer

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	COUNTER															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address
EPCA0_COUNTER = 0x4000_E1B0

Table 28.10. EPCA0_COUNTER Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15:0	COUNTER	**Counter/Timer.** This field is the current value of EPCA counter/timer.

Register 28.5. EPCA0_LIMIT: Module Upper Limit

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	LIMIT															
Type	RW															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Register ALL Access Address
EPCA0_LIMIT = 0x4000_E1C0

Table 28.11. EPCA0_LIMIT Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15:0	LIMIT	**Upper Limit.** The EPCA Counter/Timer counts from 0 to the upper limit represented by this field.

Register 28.6. EPCA0_LIMITUPD: Module Upper Limit Update Value

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	LIMITUPD															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

EPCA0_LIMITUPD = 0x4000_E1D0

Table 28.12. EPCA0_LIMITUPD Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15:0	LIMITUPD	**Module Upper Limit Update Value.** This field will be transferred to the LIMIT field when a EPCA counter/timer overflow/limit event occurs if updates are allowed (NOUPD = 0). The UPDCF bit will be set to 1 by hardware when the transfer operation is complete.

Register 28.7. EPCA0_DTIME: Phase Delay Time

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	DTIMEY								DTIMEX							
Type	RW								RW							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

EPCA0_DTIME = 0x4000_E1E0

Table 28.13. EPCA0_DTIME Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15:8	DTIMEY	**Y Phase Delay Time.** This field is the amount of time in AHB clock cycles after the differential Y Phase output de-asserts before the X Phase output can assert. This is a global setting for all channels in the EPCA module.
7:0	DTIMEX	**X Phase Delay Time.** This field is the amount of time in AHB clock cycles after the differential X Phase output de-asserts before the Y Phase output can assert. This is a global setting for all channels in the EPCA module.

Register 28.8. EPCA0_DTARGET: DMA Transfer Target

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	DTARGET[31:16]															
Type	W															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	DTARGET[15:0]															
Type	W															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

EPCA0_DTARGET = 0x4000_E200

Table 28.14. EPCA0_DTARGET Register Bit Descriptions

Bit	Name	Function
31:0	DTARGET	**DMA Transfer Target.** Writes to this field will be written to the EPCA register selected by the DPTR field.

Notes:

1. The access methods for this register are restricted. Do not use half-word or byte access methods on this register.

28.10. EPCA0 Register Memory Map

Table 28.15. EPCA0 Memory Map

Register Name	EPCA0_LIMIT	EPCA0_COUNTER	EPCA0_STATUS	EPCA0_CONTROL	EPCA0_MODE	Register Name
ALL Address	0x4000_E1C0	0x4000_E1B0	0x4000_E1A0	0x4000_E190	0x4000_E180	ALL Address
Access Methods	ALL	ALL	ALL | SET | CLR	ALL | SET | CLR	ALL	Access Methods
Bit 31	Reserved	Reserved	Reserved	DIV	Reserved	Bit 31
Bit 30						Bit 30
Bit 29						Bit 29
Bit 28					STDOSEL	Bit 28
Bit 27						Bit 27
Bit 26					Reserved	Bit 26
Bit 25					DBUSYF	Bit 25
Bit 24					DSTART	Bit 24
Bit 23						Bit 23
Bit 22						Bit 22
Bit 21				DIVST	DPTR	Bit 21
Bit 20				Reserved		Bit 20
Bit 19						Bit 19
Bit 18					DEND	Bit 18
Bit 17						Bit 17
Bit 16						Bit 16
Bit 15			C5IOVFI	STEN	HDOSEL	Bit 15
Bit 14			C4IOVFI			Bit 14
Bit 13			C3IOVFI	STESEL	Reserved	Bit 13
Bit 12			C2IOVFI	STSEL	CLKSEL	Bit 12
Bit 11			C1IOVFI			Bit 11
Bit 10			C0IOVFI	Reserved		Bit 10
Bit 9	LIMIT	COUNTER	HALTI	HALTEN	CLKDIV	Bit 9
Bit 8			UPDCF	Reserved		Bit 8
Bit 7			OVFI			Bit 7
Bit 6			RUN	DBGMD		Bit 6
Bit 5			C5CCI	Reserved		Bit 5
Bit 4			C4CCI	NOUPD		Bit 4
Bit 3			C3CCI	HALTIEN		Bit 3
Bit 2			C2CCI	OVFSEN		Bit 2
Bit 1			C1CCI	OVFDEN		Bit 1
Bit 0			C0CCI	OVFIEN		Bit 0

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

Table 28.15. EPCA0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EPCA0_LIMITUPD	0x4000_E1D0	ALL	Reserved																LIMITUPD															
EPCA0_DTIME	0x4000_E1E0	ALL	Reserved																DTIMEY								DTIMEX						
EPCA0_DTARGET	0x4000_E200	ALL	DTARGET																															

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

28.11. EPCA0_CH0-5 Registers

This section contains the detailed register descriptions for EPCA0_CH0, EPCA0_CH1, EPCA0_CH2, EPCA0_CH3, EPCA0_CH4 and EPCA0_CH5 registers.

Register 28.9. EPCAn_CHx_MODE: Channel Capture/Compare Mode

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved					CMD			Reserved	DIFGEN	PWMMD				COSEL	
Type	R					RW			R	RW	RW				RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

EPCA0_CH0_MODE = 0x4000_E000
EPCA0_CH1_MODE = 0x4000_E040
EPCA0_CH2_MODE = 0x4000_E080
EPCA0_CH3_MODE = 0x4000_E0C0
EPCA0_CH4_MODE = 0x4000_E100
EPCA0_CH5_MODE = 0x4000_E140

Table 28.16. EPCAn_CHx_MODE Register Bit Descriptions

Bit	Name	Function
31:11	Reserved	Must write reset value.
10:8	CMD	**Channel Operating Mode.** 000: Configure the channel for edge-aligned PWM mode. 001: Configure the channel for center-aligned PWM mode. 010: Configure the channel for high-frequency/square-wave mode. 011: Configure the channel for timer/capture mode. 100: Configure the channel for n-bit edge-aligned PWM mode. 101-111: Reserved.
7	Reserved	Must write reset value.
6	DIFGEN	**Differential Signal Generator Enable.** 0: Disable the differential signal generator. The channel will output a single non-differential output. 1: Enable the differential signal generator. The channel will output two differential outputs: X Phase (XPH) and Y Phase (YPH).

Table 28.16. EPCAn_CHx_MODE Register Bit Descriptions

Bit	Name	Function
5:2	PWMMD	**PWM N-Bit Mode.** This field represents the n-bit PWM for this channel. When in n-bit PWM mode, the channel will behave as if the EPCA Counter/Timer is only n bits wide.
1:0	COSEL	**Channel Output Function Select.** 00: Toggle the channel output at the next capture/compare, overflow, or intermediate event. 01: Set the channel output at the next capture/compare, overflow, or intermediate event. 10: Clear the output at the next capture/compare, overflow, or intermediate event. 11: Capture/Compare, overflow, or intermediate events do not control the output state.

Register 28.10. EPCAn_CHx_CONTROL: Channel Capture/Compare Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved		CIOVFSEN	CIOVFDEN	CIOVFIEN	CCSEN	CCDEN	CCIEN	XPHST	ACTIVEPH	YPHST	Reserved	CUPDCF	CNCAPEN	CPCAPEN	COUTST
Type	R		RW	RW	RW	RW	RW	RW	RW	RW	RW	R	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

EPCA0_CH0_CONTROL = 0x4000_E010
EPCA0_CH1_CONTROL = 0x4000_E050
EPCA0_CH2_CONTROL = 0x4000_E090
EPCA0_CH3_CONTROL = 0x4000_E0D0
EPCA0_CH4_CONTROL = 0x4000_E110
EPCA0_CH5_CONTROL = 0x4000_E150
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 28.17. EPCAn_CHx_CONTROL Register Bit Descriptions

Bit	Name	Function
31:14	Reserved	Must write reset value.
13	CIOVFSEN	**Intermediate Overflow Synchronization Signal Enable.** 0: Do not send a synchronization signal when a channel intermediate overflow event occurs. 1: Send a synchronization signal when a channel intermediate overflow occurs.
12	CIOVFDEN	**Intermediate Overflow DMA Request Enable.** 0: Do not request DMA data when a channel intermediate overflow event occurs. 1: Request DMA data when a channel intermediate overflow event occurs.
11	CIOVFIEN	**Intermediate Overflow Interrupt Enable.** 0: Disable the channel intermediate overflow interrupt. 1: Enable the channel intermediate overflow interrupt.
10	CCSEN	**Capture/Compare Synchronization Signal Enable.** 0: Do not send a synchronization signal when a channel capture/compare event occurs. 1: Send a synchronization signal when a channel capture/compare event occurs.

Table 28.17. EPCAn_CHx_CONTROL Register Bit Descriptions

Bit	Name	Function
9	CCDEN	**Capture/Compare DMA Request Enable.** 0: Do not request DMA data when a channel capture/compare event occurs. 1: Request DMA data when a channel capture/compare event occurs.
8	CCIEN	**Capture/Compare Interrupt Enable.** 0: Disable the channel capture/compare interrupt. 1: Enable the channel capture/compare interrupt.
7	XPHST	**Differential X Phase State.** 0: Set the X Phase output state to low. 1: Set the X Phase output state to high.
6	ACTIVEPH	**Active Channel Select.** This bit indicates which phase logic is currently controlling the differential outputs. 0: The Y Phase is active and X Phase is inactive. 1: The X Phase is active and Y Phase is inactive.
5	YPHST	**Differential Y Phase State.** 0: Set the Y Phase output state to low. 1: Set the Y Phase output state to high.
4	Reserved	Must write reset value.
3	CUPDCF	**Channel Register Update Complete Flag.** 0: A EPCA channel register update completed or is not pending. 1: A EPCA channel register update has not completed and is still pending.
2	CNCAPEN	**Negative Edge Input Capture Enable.** 0: Disable negative-edge input capture. 1: Enable negative-edge input capture.
1	CPCAPEN	**Positive Edge Input Capture Enable.** 0: Disable positive-edge input capture. 1: Enable positive-edge input capture.
0	COUTST	**Channel Output State.** 0: The channel output state is low. 1: The channel output state is high.

Register 28.11. EPCAn_CHx_CCAPV: Channel Compare Value

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved														CCAPV[17:16]	
Type	R														RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	CCAPV[15:0]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

EPCA0_CH0_CCAPV = 0x4000_E020
EPCA0_CH1_CCAPV = 0x4000_E060
EPCA0_CH2_CCAPV = 0x4000_E0A0
EPCA0_CH3_CCAPV = 0x4000_E0E0
EPCA0_CH4_CCAPV = 0x4000_E120
EPCA0_CH5_CCAPV = 0x4000_E160

Table 28.18. EPCAn_CHx_CCAPV Register Bit Descriptions

Bit	Name	Function
31:18	Reserved	Must write reset value.
17:0	CCAPV	**Channel Compare Value.** This field holds the channel compare value for comparator functions or the channel capture data from timer functions. The LSB represents 1/2 PCA clock period.

Register 28.12. EPCAn_CHx_CCAPVUPD: Channel Compare Update Value

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved														CCAPVUPD[17:16]	
Type	R														RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	CCAPVUPD[15:0]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

EPCA0_CH0_CCAPVUPD = 0x4000_E030
EPCA0_CH1_CCAPVUPD = 0x4000_E070
EPCA0_CH2_CCAPVUPD = 0x4000_E0B0
EPCA0_CH3_CCAPVUPD = 0x4000_E0F0
EPCA0_CH4_CCAPVUPD = 0x4000_E130
EPCA0_CH5_CCAPVUPD = 0x4000_E170

Table 28.19. EPCAn_CHx_CCAPVUPD Register Bit Descriptions

Bit	Name	Function
31:18	Reserved	Must write reset value.
17:0	CCAPVUPD	**Channel Compare Update Value.** This field will be transferred to the CCAPV field when a EPCA counter/timer overflow occurs if updates are allowed (NOUPD = 0). The CUPDCF bit will be set to 1 by hardware when the transfer operation is complete.

28.12. EPCAn_CHx Register Memory Map

Table 28.20. EPCAn_CHx Memory Map

Register Name	ALL Offset	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EPCAn_CHx_MODE	0x0	ALL	Reserved																					CMD			Reserved	DIFGEN	PWMMD				COSEL	
EPCAn_CHx_CONTROL	0x10	ALL | SET | CLR	Reserved																		CIOVFSEN	CIOVFDEN	CIOVFIEN	CCSEN	CCDEN	CCIEN	XPHST	ACTIVEPH	YPHST	Reserved	CUPDCF	CNCAPEN	CPCAPEN	COUTST

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: EPCA0_CH0 = 0x4000_E000, EPCA0_CH1 = 0x4000_E040, EPCA0_CH2 = 0x4000_E080, EPCA0_CH3 = 0x4000_E0C0, EPCA0_CH4 = 0x4000_E100, EPCA0_CH5 = 0x4000_E140

Table 28.20. EPCAn_CHx Memory Map

Register Name	ALL Offset	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
EPCAn_CHx_CCAPV	0x20	ALL	Reserved														CCAPV																		
EPCAn_CHx_CCAPVUPD	0x30	ALL	Reserved														CCAPVUPD																		

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: EPCA0_CH0 = 0x4000_E000, EPCA0_CH1 = 0x4000_E040, EPCA0_CH2 = 0x4000_E080, EPCA0_CH3 = 0x4000_E0C0, EPCA0_CH4 = 0x4000_E100, EPCA0_CH5 = 0x4000_E140

29. Programmable Counter Array (PCA0 and PCA1)

This section describes the Programmable Counter Array (PCA) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the PCA block, which is used by both PCA0 and PCA1 on all device families covered in this document.

29.1. Programmable Counter Array Features

The PCA module includes the following features:

- Independent channels.
- Center- and edge-aligned waveform generation.
- Programmable clock divisor and multiple options for clock source selection.
- Pulse-Width Modulation waveform generation.

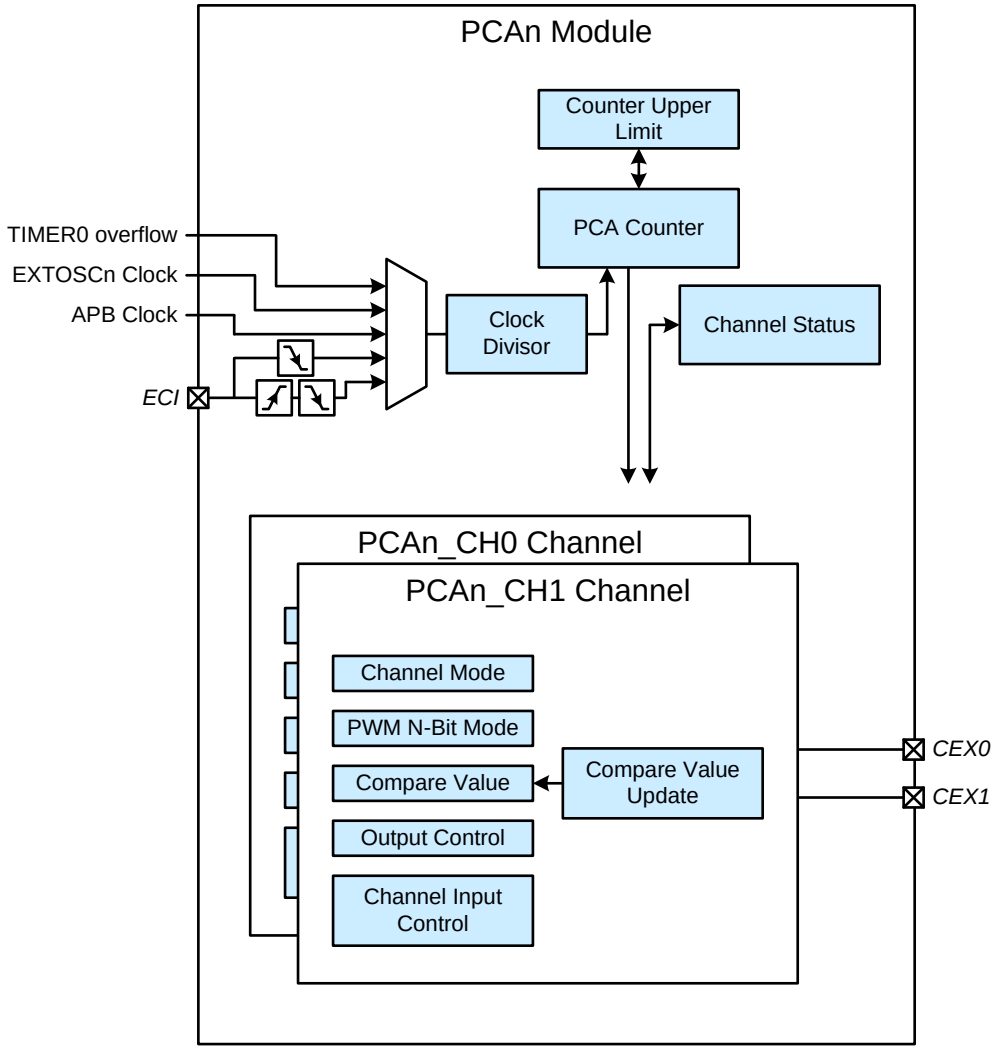


Figure 29.1. PCA Block Diagram

29.2. Module Overview

The Programmable Counter Array (PCA) provides flexible timer/counter functionality. The PCA consists of a dedicated 16-bit counter/timer and multiple (up to two) 16-bit capture/compare channels (PCAn_CHx). All channels trigger or capture based on the shared 16-bit counter/timer, so the channels are inherently synchronized. The counter/timer is a 16-bit up counter with a programmable upper count limit. The counter starts at 0 and counts continuously from zero to the upper limit. Each channel includes a data register that can compare against the counter or capture the state of the counter based on a set of programmable conditions.

Each channel has its own associated output lines and may be configured to operate independently of the others in one of six modes: Edge-Aligned PWM, Center-Aligned PWM, High-Frequency / Square Wave, Timer / Capture, n-bit Edge-Aligned PWM, or Software Timer. Each channel has a single output (CHx).

The RUN bit starts or stops the PCA counter. The DBGMD bit in the CONTROL register controls whether the PCA counter runs while the core is halted in debug mode.

Three different events can change a channel output or cause a capture. An overflow/limit event occurs when the PCA counter (COUNTER) is equal to the upper limit (LIMIT). A capture/compare event occurs whenever a channel's capture/compare register (CCAPV) matches the current PCA counter value. An intermediate overflow event can only occur in n-bit PWM mode and occurs when the channel's n-bit capture/compare range matches the PCA counter.

To facilitate counter and channel updates while the counter continues to run, the channel compare/capture register (CCAPV) has an update register (CCAPVUPD). Firmware can write a new value to the CCAPVUPD register without modifying the current PCA cycle. The hardware will load these values into the CCAPV register when the next counter overflow/limit event occurs, and the channel CUPDCF flag indicates when an update operation completes.

29.3. Clocking

The clock input for the counter/timer is selected by the CLKSEL field in the MODE register as shown in Figure 29.2.

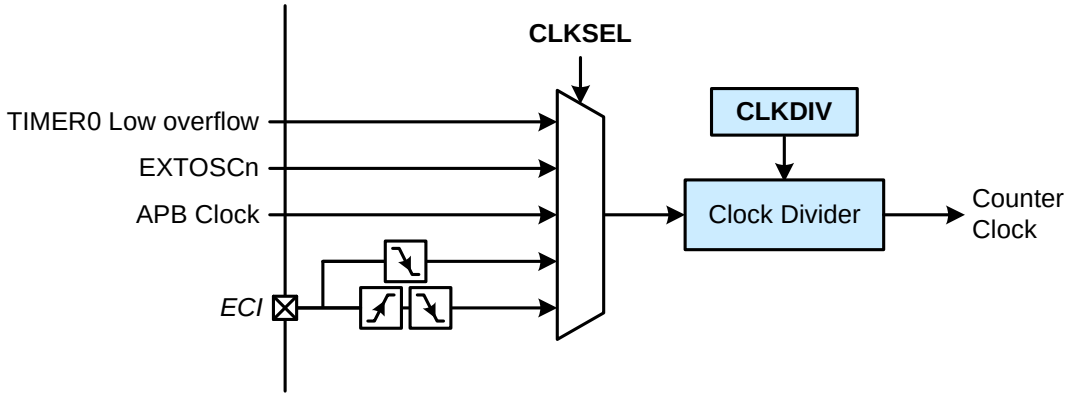


Figure 29.2. Clock Source Selection

The selected clock is passed through a divide-by-n clock divider where n can be between 1 and 1024. The CLKDIV field sets the clock divider for the selected clock before it drives the 16-bit counter/timer.

All non-APB clock sources must be synchronized with the APB clock, so the maximum possible operating speed using these sources is one-half the APB frequency, if not slower.

29.3.1. APB Clock

When the APB clock signal is selected as the counter clock source, the PCA counter clock divider will use the APB clock source defined by the device clock control module. Selecting the APB clock provides the fastest clock source for the module.

29.3.2. External Clock (EXTOSCn)

When the external clock is selected as the counter clock source, the counter will run from the external clock source (EXTOSCn), regardless of the clock selection of the core. The external clock source is synchronized to the selected core clock in this mode. In order to guarantee that the external clock transitions are recognized by the device, the external clock signal must be high or low for at least one APB clock period. This limits the maximum frequency of an external clock in this mode to one-half the APB clock.

29.3.3. TIMER0 Low Overflow

The PCA module can select the TIMER0 module low timer overflows as its clock source. TIMER0 can operate in either 32-bit or 16-bit mode and still provide the clock for the PCA clock divider. The maximum speed for the TIMER0 low overflows as an PCA counter clock source is one-half the APB clock.

29.3.4. External Clock Input (ECI)

When the external clock input (ECI) is selected as the PCA clock source, the clock divider decrements on falling edges or both rising and falling edges of the pin. The ECI pin is synchronized to the selected AHB clock in this mode. The maximum clock rate for the ECI external clock input is the APB divided by 4.

29.3.5. Clock Divider

The clock divider provides a flexible time base for the PCA counter. The divider starts at one-half the CLKDIV value and decrements to 0. Using this method, the divider counts the number of input clocks until the next counter clock edge (either rising or falling) rather than whole counter clock periods.

The current value of the divider can be read and written using the DIV field in the MODE register. This allows access during debugging to observe module events at the various PCA clock edges rather than stepping through a large number of APB clocks. Firmware should always write to the CLKDIV field to modify the divider value.

The DIVST bit displays the current output phase of the clock divider.

29.4. Interrupts

The PCA module has one interrupt vector and multiple interrupt sources within the module.

The module has one counter overflow/limit interrupt source. The OVFIEN bit enables the counter overflow/limit interrupt, which occurs when counter (COUNTER) equals the upper limit (LIMIT) and resets to 0. The OVFI interrupt flag indicates when an counter overflow/limit event occurs.

Each channel (CHx) has a compare/capture interrupt flag (CxCCI) and an intermediate overflow flag (CxIOVFI). These interrupts can be enabled in the channel registers (CCIEN for capture/compare, CIOVFIEN for intermediate overflow).

Firmware can check the source of the PCA interrupt by checking the appropriate flags in the interrupt service routine.

29.5. Outputs

The PCA module has up to two module outputs (CEXx) that can be routed to physical pins by configuring the device port configuration module. Each PCA capture/compare channel has a single CHx output.

The COUTST bit in the PCA channels determines the polarity of the output. This value should be set when the counter is not running to ensure predictable operation. Hardware sets the COUTST bit on the rising edges of the APB clock to reflect the current output state of the channel.

Note: The COUTST bit determines the starting state of the channel output. If the starting state is active, this means the channel could be sitting in an active state for some time while firmware finishes initializing the module and starts the counter. If the output is connected to a transistor where this behavior is undesirable, firmware can initialize the counter to a mid-range value and the outputs with an inactive value. This will ensure that any sensitive external circuits will not be damaged.

Figure 29.3 shows a PCA channel single output timing diagram.

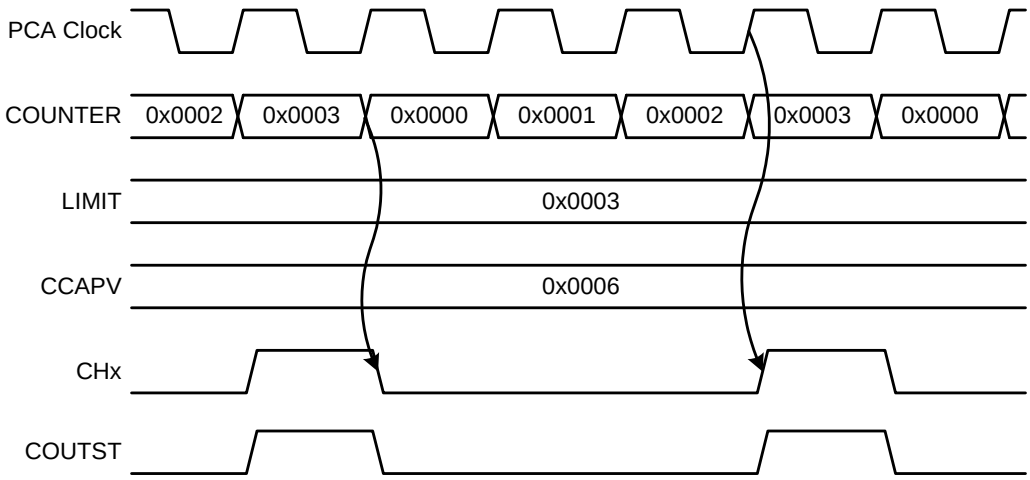


Figure 29.3. Example Channel Single Output Timing Diagram (Toggle Mode)

The CHx output can toggle, set, or clear on an overflow/limit, compare, or intermediate overflow event. In addition, the output can ignore these events and stay at its previous value. The COSEL field in the channel MODE register determines the CHx output behavior. This field allows firmware to modify the behavior of the CHx output without changing the configuration of the channel, interrupting the counter, or changing the port configuration.

29.6. Operational Modes

The PCA module has six operational modes that each channel can independently select: Edge-Aligned PWM, Center-Aligned PWM, High-Frequency/Square Wave, Timer/Capture, n-bit Edge-Aligned PWM, and Software Timer modes. The CMD bits select the channel operational mode.

The PCA counter is 16 bits and counts in full PCA clock cycles. The channel CCAPV registers are 18 bits and represent half PCA clock cycles. To match an counter value, the CCAPV field should be written with double the counter value. The CCAPV LSB provides timing resolution to one half-clock counter cycle. The additional 18th bit of the CCAPV register allows the channel outputs to create 0-100% duty cycles (0x00000 is 0% and 0x20000 is 100% duty cycle).

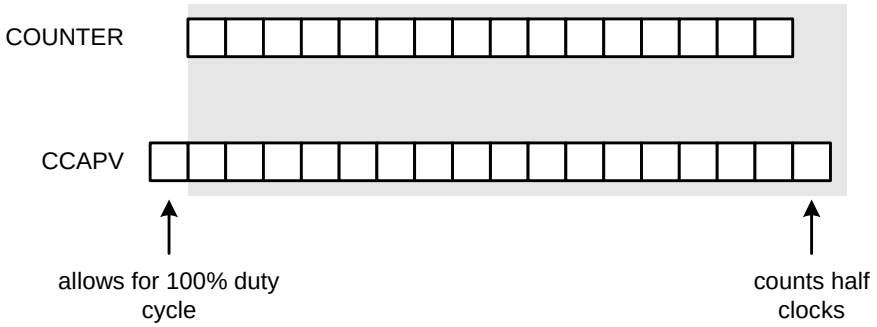


Figure 29.4. PCA Counter and Channel Compare/Capture Registers

This section discusses the channel behavior in each of these modes in detail.

29.6.1. Edge-Aligned Pulse Width Modulation (PWM) Mode

In edge-aligned PWM mode (CMD = 0), the 18-bit capture/compare register (CCAPV) defines the number of PCA half clocks for the inactive time of the PWM signal. A capture/compare event occurs when the counter matches the register contents, and the output will change from the initial state set by COUTST depending on the selected COSEL value (typically 00b for toggle). An overflow/limit event occurs when the counter reaches the LIMIT value and resets to 0, and CHx will again change depending on the COSEL value. To output a varying duty cycle, firmware can write to the CCAPVUPD register, which hardware will automatically load into the channel's CCAPV register on the next counter overflow/limit if the module register update inhibit (NOUPD) is cleared to 0.

Assuming that the inactive and initial state of the output is low and COSEL is set to toggle, the CHx output duty cycle in edge-aligned PWM mode is shown in Equation 29.1. No output pulse is generated if CCAPV is 0, and 100% duty cycle results from CCAPV set to 0x20000.

Figure 29.5 shows an example edge-aligned PWM timing diagram.

$$\text{Duty Cycle} = \frac{(\text{LIMIT} \times 2) - \text{CCAPV}}{\text{LIMIT} \times 2}$$

Equation 29.1. Edge-Aligned PWM Duty Cycle

Because CCAPV is given in half clocks, an odd CCAPV value results in the CHx output changing state at the mid-cycle edge of the PCA clock.

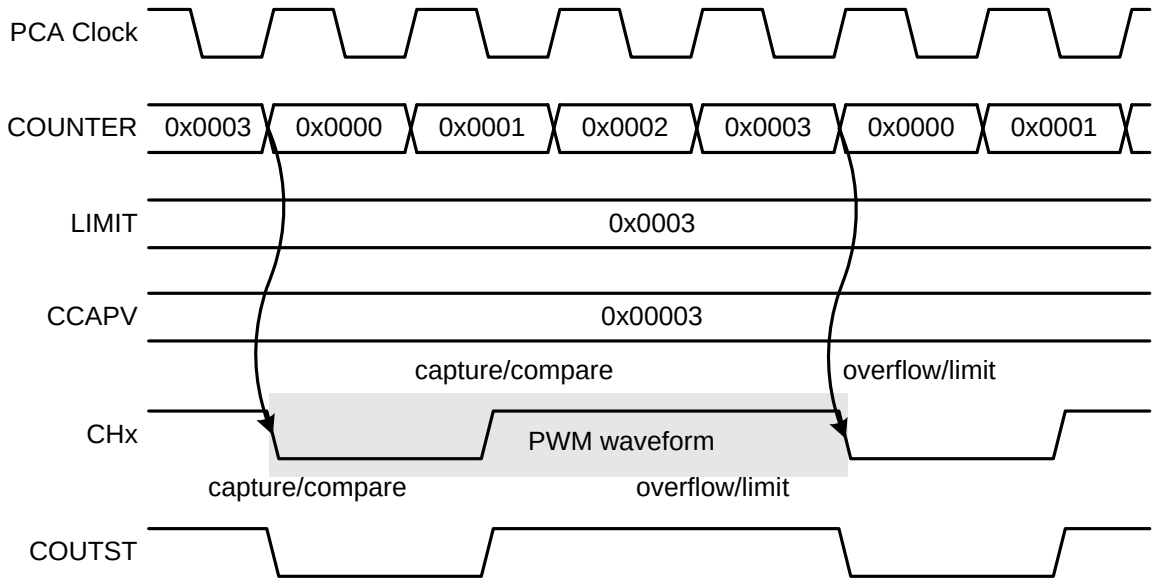


Figure 29.5. Example Edge-Aligned PWM Timing Diagram

29.6.2. Center-Aligned Pulse Width Modulation (PWM) Mode

In center-aligned PWM mode (CMD = 1), a channel generates a PWM waveform symmetric about the counter's overflow/limit event defined by the LIMIT field. Multiple channels in an array configured in this mode will each have PWM waveforms which are all symmetric about the same center-pulse points (counter equal to zero).

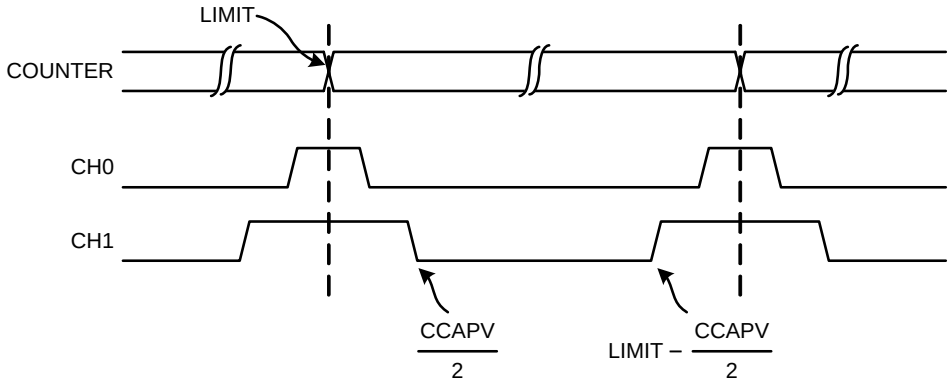


Figure 29.6. Multiple Center-Aligned PWM Channels

The channel 18-bit capture/compare register (CCAPV) is used to generate two capture/compare events in one counter cycle (0 to LIMIT). The first event occurs when the counter is equal to CCAPV divided by 2. The second event occurs when the counter is equal to LIMIT minus CCAPV divided by 2. In the event of an odd value in CCAPV, the hardware adds the extra half cycle to the capture/compare event following the counter overflow/limit event.

Firmware can write to the channel's CCAPVUPD register to update the waveform. Hardware will update the CCAPV field with the CCAPVUPD value when the counter overflows from the upper limit to zero as long as the update inhibit bit (NOUPD) is cleared to 0.

The COUTST bit determines the starting state of the channel output and should be set before starting the counter. If the starting state is active, this means the channel could be sitting in an active state for some time while firmware finishes initializing the module and starts the counter. If the output is connected to a transistor where this behavior is undesirable, firmware can initialize the counter to a mid-range value and the outputs with an inactive value. This will ensure that any sensitive external circuits will not be damaged.

Assuming that the active and initial state of the output is high and COSEL is set to toggle, the CHx output duty cycle in center-aligned PWM mode is shown in Equation 29.2. No output pulse is generated if CCAPV is 0, and 100% duty cycle results when CCAPV (in half clocks) is set to a number of full clocks equal to or greater than LIMIT.

Figure 29.7 shows an example center-aligned PWM timing diagram.

$$\text{Duty Cycle} = \frac{(\text{LIMIT} \times 2) - \text{CCAPV}}{\text{LIMIT} \times 2}$$

Equation 29.2. Center-Aligned PWM Duty Cycle

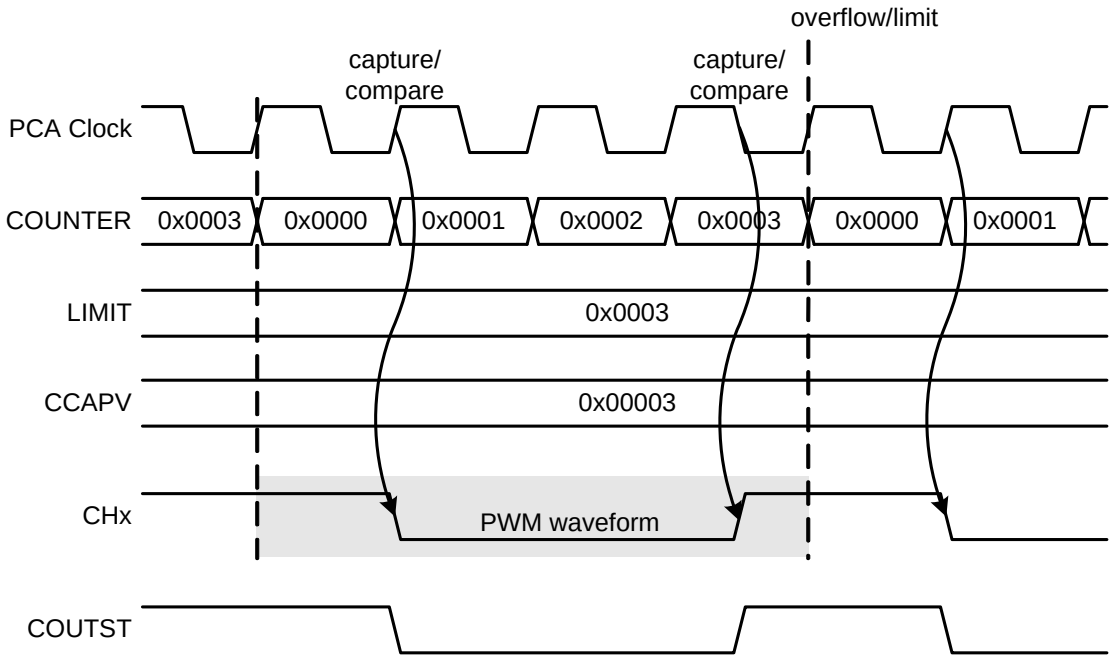


Figure 29.7. Example Center-Aligned PWM Timing Diagram

29.6.3. High-Frequency / Square Wave Mode

High-frequency square-wave mode (CMD = 2) produces a 50% duty cycle waveform of programmable period on the channel's CHx output. This mode provides a flexible way of creating square waves with a fast period. Each half-period of the generated output clock can be as short as one counter clock period (two CCAPV half clocks) and as long as 128 counter clock periods, programmable in half clock steps.

Bits [15:8] of the channel capture/compare register (CCAPV) contain the waveform half period in the number of half PCA clocks. The lower byte (bits [7:0]) is the calculated value compared to the counter. When a match occurs between the lower byte and the counter, the hardware triggers a capture/compare event and automatically adds the match value of the counter to the CCAPV[15:8] value to create a new compare value for the lower byte. An overflow/limit event occurs when the counter reaches the upper limit defined by the LIMIT field. Firmware should program the upper limit register to reset the counter at a (multiple of 128) - 1 to avoid undesired waveform edges. Figure 29.8 shows how the hardware creates the waveform.

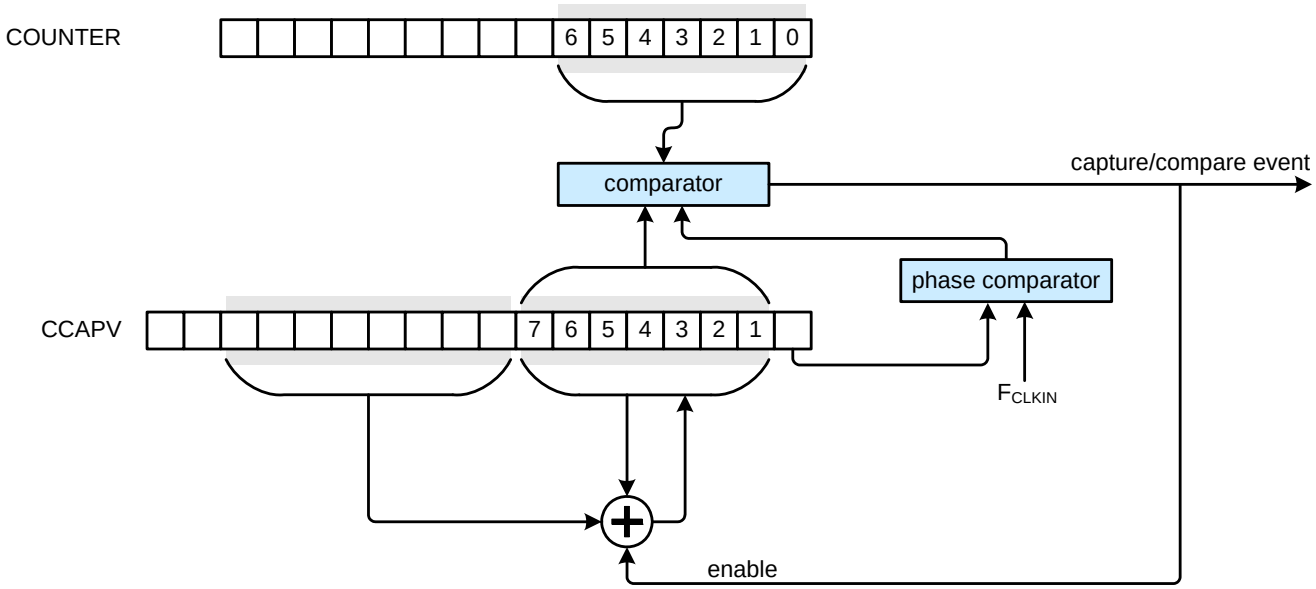


Figure 29.8. High-Frequency Square Wave Waveform Generation

Firmware can update the frequency of the output by writing to the CCAPVUPD register. Hardware will automatically load bits [15:8] of this register to the CCAPV register at the next counter overflow/limit event, if possible (NOUPD = 0). The rest of the CCAPVUPD register (bits [17:16] and [7:0]) are ignored.

Assuming that the COSEL field for the channel is set to toggle, the resulting CHx output frequency in high-frequency square-wave mode is shown in Equation 29.3. A CCAPV[15:8] value of 1 is not valid, and a CCAPV[15:8] value of 0 results in an output waveform half period of 128 counter clocks.

Figure 29.9 shows an example high-frequency square wave mode timing diagram.

$$F_{CHx} = \frac{F_{PCA}}{CCAPV[15:8]}$$

Equation 29.3. High-Frequency Square Wave Output Frequency

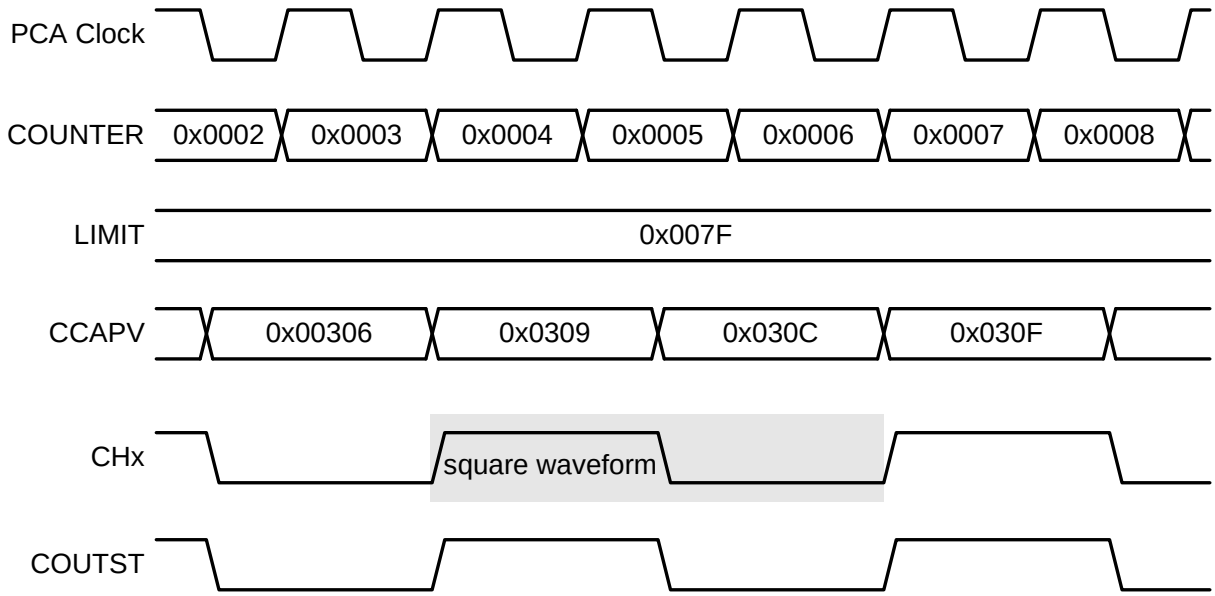


Figure 29.9. Example High-Frequency Square Wave Mode Timing Diagram

29.6.4. Timer/Capture Mode

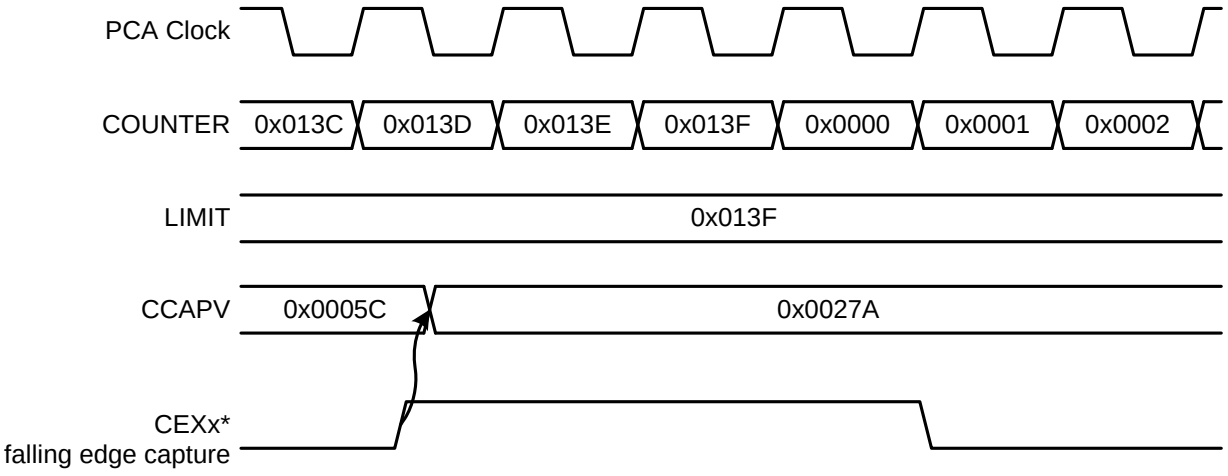
In timer/capture mode (CMD = 3), the channel uses the CEXn line as an input signal that triggers a capture/compare event and stores the counter state in the channel capture/compare register (CCAPV) in half clocks. Firmware can configure the event to trigger on rising, falling, or both edges using the channel CPCAPEN and CNCAPEN bits. Table 29.1 shows the capture edge configuration options.

Table 29.1. Capture Edge Configuration Options

CPCAPEN	CNCAPEN	Selected Edge
0	0	Capture disabled.
0	1	Capture on the falling edge.
1	0	Capture on the rising edge.
1	1	Capture on both edges.

The input CEXn signal must remain high or low for at least two APB clocks to be recognized by the hardware. The capture occurs at the next PCA clock edge.

Figure 29.10 shows an example timer/capture mode timing diagram.



*Note: CEXx must be high or low for 2 APB clock cycles.

Figure 29.10. Example Timer / Capture Mode Timing Diagram

29.6.5. N-bit Edge-Aligned Pulse Width Modulation (PWM) Mode

The n-bit edge-aligned PWM modes allow each channel to be independently configured to generate edge-aligned PWM waveforms with a faster duty cycle than the counter. In n-bit edge-aligned PWM mode (CMD = 4), the least-significant n bits (set by PWMMD) of the counter define the number of full clocks of the PWM waveform. The channel's capture/compare register (CCAPV) defines the number of PCA half clocks for the inactive time of the PWM signal, and the higher order unused bits of CCAPV are ignored. A capture/compare event occurs when the n-bit counter is equal to the number of half clocks defined by the CCAPV field. An intermediate overflow event occurs when the counter overflows the n-bit range. A counter overflow/limit event occurs when the counter reaches the upper limit (LIMIT) within the full 16-bit range. If one of the channels operates in n-bit mode, firmware should set the counter's upper limit as an even multiple of the n-bit boundary to ensure the channel's output does not have any irregular edges from the overflow/limit events.

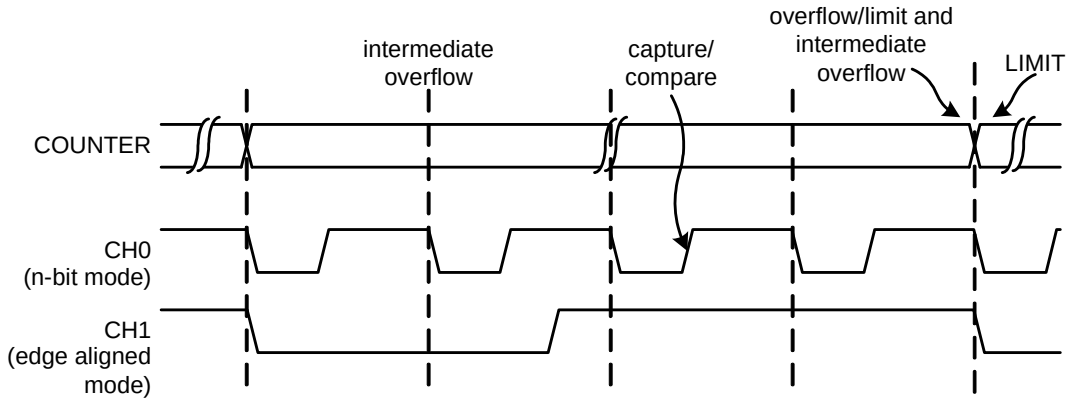


Figure 29.11. Multiple Edge-Aligned PWM Channels (N-bit and Edge Aligned)

Table 29.2 provides a list of the intermediate overflow, recommended upper limit values, and 100% duty cycle values for each n-bit setting.

Table 29.2. N-bit Intermediate Overflow Values

PWMMD Value	N-bit PWM Mode	Counter Intermediate Overflow Value (full clocks)	Recommended Counter Upper Limit (full clocks)	CCAPV 100% Duty Cycle Value (half clocks)
0	0-bit	0	any value	2
1	1-bit	1	(multiple of 2) - 1	4
2	2-bit	3	(multiple of 4) - 1	8
3	3-bit	7	(multiple of 8) - 1	16
4	4-bit	15	(multiple of 16) - 1	32
5	5-bit	31	(multiple of 32) - 1	64
6	6-bit	63	(multiple of 64) - 1	128
7	7-bit	127	(multiple of 128) - 1	256
8	8-bit	255	(multiple of 256) - 1	512
9	9-bit	511	(multiple of 512) - 1	1024
10	10-bit	1023	(multiple of 1024) - 1	2048
11	11-bit	2047	(multiple of 2048) - 1	4096
12	12-bit	4095	(multiple of 4096) - 1	8192
13	13-bit	8191	(multiple of 8192) - 1	16334
14	14-bit	16333	(multiple of 16334) - 1	32768
15	15-bit	32767	(multiple of 32768) - 1	65536

Firmware can write to the CCAPVUPD register to update the duty cycle, which hardware will automatically load into the channel's CCAPV register on the next counter overflow/limit event if the module register update inhibit (NOUPD) is cleared to 0.

Assuming that the inactive and initial state of the output is low, COSEL is set to toggle, and the upper limit is set to an even multiple, the CHx output duty cycle in n-bit edge-aligned PWM mode is shown in Equation 29.4. No output pulse is generated if CCAPV is 0.

Figure 29.12 shows an example n-bit edge-aligned PWM timing diagram.

$$\text{Duty Cycle} = \frac{(2^n \times 2) - \text{CCAPV}}{2^n \times 2}$$

Equation 29.4. N-bit Edge-Aligned PWM Duty Cycle

Because CCAPV is given in half clocks, an odd CCAPV value results in the CHx output changing state at the mid-cycle edge of the PCA clock.

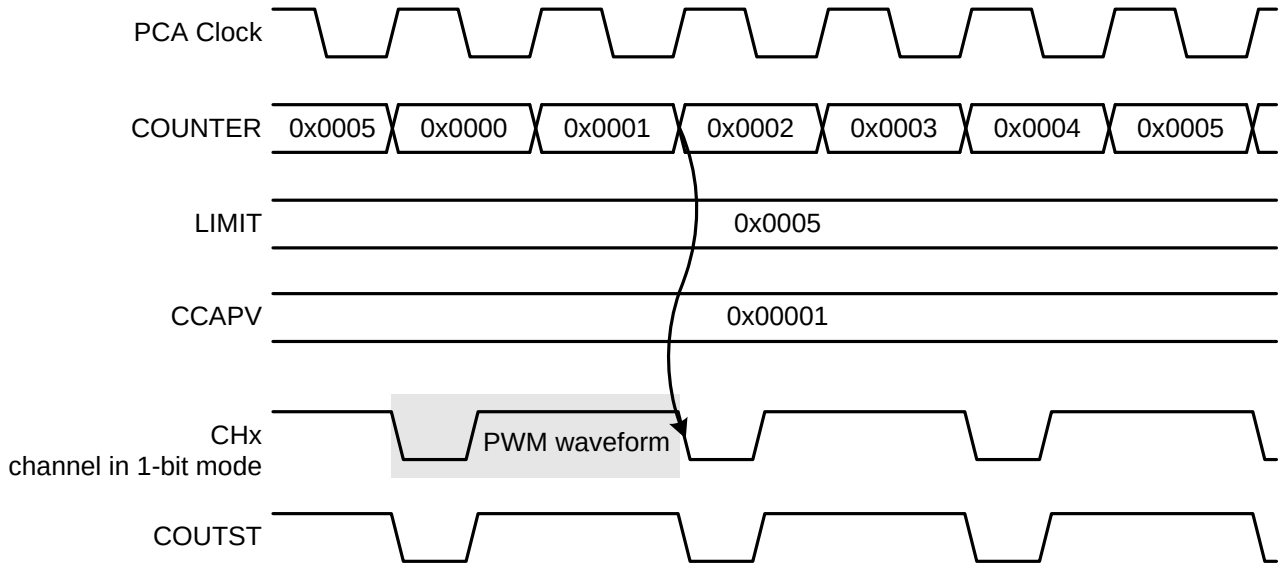


Figure 29.12. Example N-bit Edge-Aligned PWM Timing Diagram

29.6.6. Software Timer Mode

Software timer mode is a free-running mode with the CHx output unaffected by the counter or channel state. Setting the COSEL field to 3 (output ignore events) enters software timer mode, regardless of the CMD field setting. The counter overflow/limit, intermediate overflow, and capture/compare events can still be used to generate interrupts, even though the channel output is disabled.

Figure 29.13 shows an example software timer mode timing diagram.

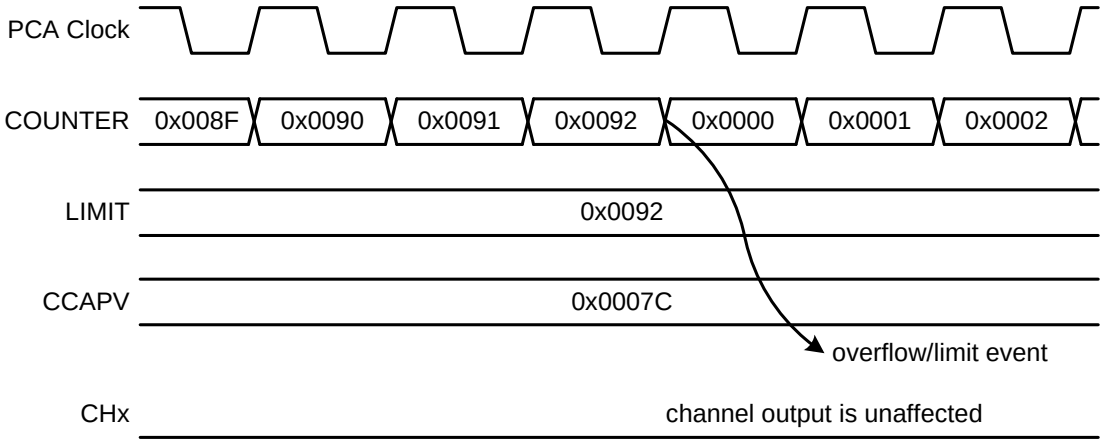


Figure 29.13. Example Software Timer Mode Timing Diagram

29.7. PCA0 and PCA1 Registers

This section contains the detailed register descriptions for PCA0 and PCA1 registers.

Register 29.1. PCAn_MODE: Module Operating Mode

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved			CLKSEL			CLKDIV									
Type	R			RW			RW									
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

PCA0_MODE = 0x4000_F180
PCA1_MODE = 0x4001_0180

Table 29.3. PCAn_MODE Register Bit Descriptions

Bit	Name	Function
31:13	Reserved	Must write reset value.
12:10	CLKSEL	**Input Clock (F_{CLKIN}) Select.** 000: Set the APB as the input clock (F_{CLKIN}). 001: Set Timer 0 low overflows divided by 2 as the input clock (F_{CLKIN}). 010: Set high-to-low transitions on ECI divided by 2 as the input clock (F_{CLKIN}). 011: Set the external oscillator module output (EXTOSCn) divided by 2 as the input clock (F_{CLKIN}). 100: Set ECI transitions divided by 2 as the input clock (F_{CLKIN}). 101-111: Reserved.
9:0	CLKDIV	**Input Clock Divisor.** The PCA module clock is given by the equation: $F_{PCA} = \frac{F_{CLKIN}}{CLKDIV + 1}$ Where the input clock (CLKIN) is determined by the CLKSEL bits.

Register 29.2. PCAn_CONTROL: Module Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	DIV										DIVST	Reserved				
Type	RW										RW	R				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved									DBGMD	Reserved					OVFIEN
Type	R									RW	R					RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

PCA0_CONTROL = 0x4000_F190
PCA1_CONTROL = 0x4001_0190
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 29.4. PCAn_CONTROL Register Bit Descriptions

Bit	Name	Function
31:22	DIV	**Current Clock Divider Count.** This field is the current value of the internal PCA clock divider. The clock divider is a counter that starts at CLKDIV / 2 and counts down to zero.
21	DIVST	**Clock Divider Output State.** 0: The clock divider is currently in the first half-cycle. 1: The clock divider is currently in the second half-cycle.
20:7	Reserved	Must write reset value.
6	DBGMD	**PCA Debug Mode.** 0: A debug breakpoint will cause the PCA to halt. 1: The PCA will continue to operate while the core is halted in debug mode.
5:1	Reserved	Must write reset value.
0	OVFIEN	**PCA Counter Overflow/Limit Interrupt Enable.** 0: Disable the PCA counter overflow/limit event interrupt. 1: Enable the PCA counter overflow/limit event interrupt.

Notes:
1. Because hardware updates the DIV and DIVST fields, the SET and CLR addresses are the only safe way to access the other fields in this register while the PCA is running.

Register 29.3. PCAn_STATUS: Module Status

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved				C1IOVFI	C0IOVFI	Reserved		OVFI	RUN	Reserved				C1CCI	C0CCI
Type	R				RW	RW	R		RW	RW	R				RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

PCA0_STATUS = 0x4000_F1A0
PCA1_STATUS = 0x4001_01A0
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 29.5. PCAn_STATUS Register Bit Descriptions

Bit	Name	Function
31:12	Reserved	Must write reset value.
11	C1IOVFI	**Channel 1 Intermediate Overflow Interrupt Flag.** This bit is set by hardware when a counter overflows the n-bit range in Channel 1 n-bit PWM mode. This bit must be cleared by firmware. 0: Channel 1 did not count past the channel n-bit mode limit. 1: Channel 1 counted past the channel n-bit mode limit.
10	C0IOVFI	**Channel 0 Intermediate Overflow Interrupt Flag.** This bit is set by hardware when a counter overflows the n-bit range in Channel 0 n-bit PWM mode. This bit must be cleared by firmware. 0: Channel 0 did not count past the channel n-bit mode limit. 1: Channel 0 counted past the channel n-bit mode limit.
9:8	Reserved	Must write reset value.
7	OVFI	**Counter/Timer Overflow/Limit Interrupt Flag.** This bit is set by hardware when the counter reaches the value in LIMIT and overflows to zero. This bit must be cleared by firmware. 0: A PCA Counter/Timer overflow/limit event did not occur. 1: A PCA Counter/Timer overflow/limit event occurred.

Notes:
1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 29.5. PCAn_STATUS Register Bit Descriptions

Bit	Name	Function
6	RUN	**Counter/Timer Run.** 0: Stop the PCA Counter/Timer. 1: Start the PCA Counter/Timer.
5:2	Reserved	Must write reset value.
1	C1CCI	**Channel 1 Capture/Compare Interrupt Flag.** This bit is set by hardware when a match or capture occurs in Channel 1. This bit must be cleared by firmware. 0: A Channel 1 match or capture event did not occur. 1: A Channel 1 match or capture event occurred.
0	C0CCI	**Channel 0 Capture/Compare Interrupt Flag.** This bit is set by hardware when a match or capture occurs in Channel 0. This bit must be cleared by firmware. 0: A Channel 0 match or capture event did not occur. 1: A Channel 0 match or capture event occurred.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Register 29.4. PCAn_COUNTER: Module Counter/Timer

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	COUNTER															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

PCA0_COUNTER = 0x4000_F1B0

PCA1_COUNTER = 0x4001_01B0

Table 29.6. PCAn_COUNTER Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15:0	COUNTER	**Counter/Timer.** This field is the current value of PCA counter/timer.

Register 29.5. PCAn_LIMIT: Module Counter/Timer Upper Limit

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	LIMIT															
Type	RW															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Register ALL Access Addresses

PCA0_LIMIT = 0x4000_F1C0
PCA1_LIMIT = 0x4001_01C0

Table 29.7. PCAn_LIMIT Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15:0	LIMIT	**Upper Limit.** The PCA Counter/Timer counts from 0 to the upper limit represented by this field.

29.8. PCAn Register Memory Map

Table 29.8. PCAn Memory Map

Register Name	ALL Offset	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCAn_MODE	0x0	ALL	Reserved																			CLKSEL			CLKDIV									
PCAn_CONTROL	0x10	ALL | SET | CLR	DIV										DIVST	Reserved													DBGMD	Reserved						OVFIEN
PCAn_STATUS	0x20	ALL | SET | CLR	Reserved																				C1IOVFI	C0IOVFI	Reserved		OVFI	RUN	Reserved				C1CCI	C0CCI
PCAn_COUNTER	0x30	ALL	Reserved																			COUNTER												
PCAn_LIMIT	0x40	ALL	Reserved																			LIMIT												

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: PCA0 = 0x4000_F180, PCA1 = 0x4001_0180

29.9. PCA0_CH0-1 and PCA1_CH0-1 Registers

This section contains the detailed register descriptions for PCA0_CH0, PCA0_CH1, PCA1_CH0 and PCA1_CH1 registers.

Register 29.6. PCAn_CHx_MODE: Channel Capture/Compare Mode

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved					CMD			Reserved		PWMMD				COSEL	
Type	R					RW			R		RW				RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

PCA0_CH0_MODE = 0x4000_F000
PCA0_CH1_MODE = 0x4000_F040
PCA1_CH0_MODE = 0x4001_0000
PCA1_CH1_MODE = 0x4001_0040

Table 29.9. PCAn_CHx_MODE Register Bit Descriptions

Bit	Name	Function
31:11	Reserved	Must write reset value.
10:8	CMD	**Channel Operating Mode.** 000: Configure the channel for edge-aligned PWM mode. 001: Configure the channel for center-aligned PWM mode. 010: Configure the channel for high-frequency/square-wave mode. 011: Configure the channel for timer/capture mode. 100: Configure the channel for n-bit edge-aligned PWM mode. 101-111: Reserved.
7:6	Reserved	Must write reset value.
5:2	PWMMD	**PWM N-Bit Mode.** This field represents the n-bit PWM for this channel. When in n-bit PWM mode, the channel will behave as if the PCA Counter/Timer is only n bits wide.

Table 29.9. PCAn_CHx_MODE Register Bit Descriptions

Bit	Name	Function
1:0	COSEL	**Channel Output Function Select.** 00: Toggle the channel output at the next capture/compare, overflow, or intermediate event. 01: Set the channel output at the next capture/compare, overflow, or intermediate event. 10: Clear the output at the next capture/compare, overflow, or intermediate event. 11: Capture/Compare, overflow, or intermediate events do not control the output state.

Register 29.7. PCAn_CHx_CONTROL: Channel Capture/Compare Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved				CIOVFIEN	Reserved		CCIEN	Reserved				CUPDCF	CNCAPEN	CPCAPEN	COUTST
Type	R				RW	R		RW	R				RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

PCA0_CH0_CONTROL = 0x4000_F010
PCA0_CH1_CONTROL = 0x4000_F050
PCA1_CH0_CONTROL = 0x4001_0010
PCA1_CH1_CONTROL = 0x4001_0050
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 29.10. PCAn_CHx_CONTROL Register Bit Descriptions

Bit	Name	Function
31:12	Reserved	Must write reset value.
11	CIOVFIEN	**Intermediate Overflow Interrupt Enable.** 0: Disable the channel intermediate overflow interrupt. 1: Enable the channel intermediate overflow interrupt.
10:9	Reserved	Must write reset value.
8	CCIEN	**Capture/Compare Interrupt Enable.** 0: Disable the channel capture/compare interrupt. 1: Enable the channel capture/compare interrupt.
7:4	Reserved	Must write reset value.
3	CUPDCF	**Channel Register Update Complete Flag.** 0: A PCA channel register update completed or is not pending. 1: A PCA channel register update has not completed and is still pending.
2	CNCAPEN	**Negative Edge Input Capture Enable.** 0: Disable negative-edge input capture. 1: Enable negative-edge input capture.

Table 29.10. PCAn_CHx_CONTROL Register Bit Descriptions

Bit	Name	Function
1	CPCAPEN	**Positive Edge Input Capture Enable.** 0: Disable positive-edge input capture. 1: Enable positive-edge input capture.
0	COUTST	**Channel Output State.** 0: The channel output state is low. 1: The channel output state is high.

Register 29.8. PCAn_CHx_CCAPV: Channel Compare Value

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved														CCAPV[17:16]	
Type	R														RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	CCAPV[15:0]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

PCA0_CH0_CCAPV = 0x4000_F020
PCA0_CH1_CCAPV = 0x4000_F060
PCA1_CH0_CCAPV = 0x4001_0020
PCA1_CH1_CCAPV = 0x4001_0060

Table 29.11. PCAn_CHx_CCAPV Register Bit Descriptions

Bit	Name	Function
31:18	Reserved	Must write reset value.
17:0	CCAPV	**Channel Compare Value.** This field holds the channel compare value for comparator functions or the channel capture data from timer functions.

Register 29.9. PCAn_CHx_CCAPVUPD: Channel Compare Update Value

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved														CCAPVUPD[17:16]	
Type	R														RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	CCAPVUPD[15:0]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

PCA0_CH0_CCAPVUPD = 0x4000_F030
PCA0_CH1_CCAPVUPD = 0x4000_F070
PCA1_CH0_CCAPVUPD = 0x4001_0030
PCA1_CH1_CCAPVUPD = 0x4001_0070

Table 29.12. PCAn_CHx_CCAPVUPD Register Bit Descriptions

Bit	Name	Function
31:18	Reserved	Must write reset value.
17:0	CCAPVUPD	**Channel Compare Update Value.** This field will be transferred to the CCAPV field when a PCA counter/timer overflow occurs if updates are allowed (NOUPD = 0). The CUPDCF bit will be set to 1 by hardware when the transfer operation is complete.

29.10. PCAn_CHx Register Memory Map

Table 29.13. PCAn_CHx Memory Map

Register Name	ALL Offset	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCAn_CHx_MODE	0x0	ALL	Reserved																					CMD			Reserved		PWMMD				COSEL	
PCAn_CHx_CONTROL	0x10	ALL | SET | CLR	Reserved																				CIOVFIEN	Reserved		CCIEN	Reserved				CUPDCF	CNCAPEN	CPCAPEN	COUTST

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: PCA0_CH0 = 0x4000_F000, PCA0_CH1 = 0x4000_F040, PCA1_CH0 = 0x4001_0000, PCA1_CH1 = 0x4001_0040

Table 29.13. PCAn_CHx Memory Map

Register Name	ALL Offset	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PCAn_CHx_CCAPV	0x20	ALL	Reserved														CCAPV																		
PCAn_CHx_CCAPVUPD	0x30	ALL	Reserved														CCAPVUPD																		

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: PCA0_CH0 = 0x4000_F000, PCA0_CH1 = 0x4000_F040, PCA1_CH0 = 0x4001_0000, PCA1_CH1 = 0x4001_0040

30. Phase-Locked Loop (PLL0)

This section describes the phase-locked loop (PLL) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the PLL block, which is used by all device families covered in this document. Note that features related to the USB oscillator are only available on the SiM3U1xx device family.

30.1. PLL Features

The PLL module includes the following features:

- Five output ranges with output frequencies ranging from 23 to 80 MHz (can also be divided by up to 128 to provide lower system clock speeds).
- Multiple reference frequency inputs.
- Three output modes: free-running DCO, frequency-locked, and phase-locked.
- Ability to sense the rising edge or falling edge of the reference source.
- DCO frequency LSB dithering to provide finer average output frequencies.
- Spectrum spreading to reduce generated system noise.
- Low jitter and fast lock times.
- Ability to suspend all output frequency updates (including dithering and spectrum spreading) using the STALL bit during jitter-sensitive measurements.

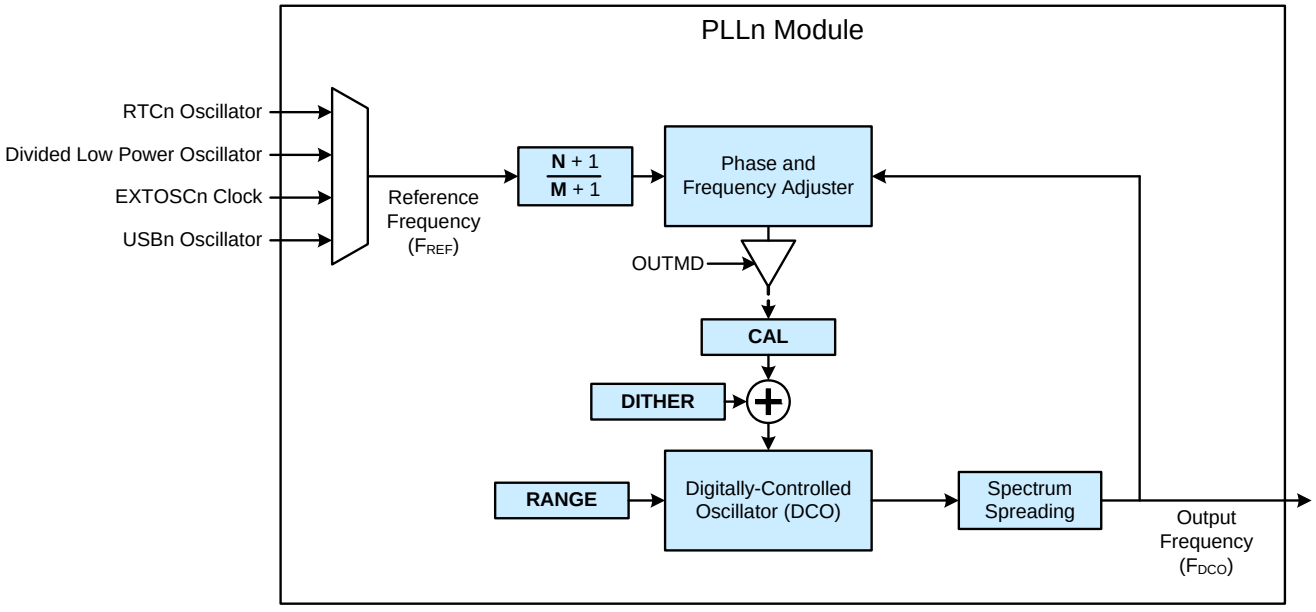


Figure 30.1. PLL Block Diagram

30.2. Overview

The PLL module consists of a dedicated Digitally-Controlled Oscillator (DCO) that can be used in free-running DCO mode without a reference frequency, frequency-locked mode with a reference frequency, or phase-locked mode with a reference frequency. The reference frequency for frequency-lock and phase-lock modes can be sourced from multiple sources (including a dedicated USB oscillator or external oscillator) to provide maximum flexibility for different application needs. Because the PLL module generates its own clock, the DCO can be locked to a particular reference frequency and then moved to free-running mode to reduce system power or noise.

30.3. Interrupts

A PLL interrupt can be generated whenever the PLL module locks to or unlocks from the reference frequency in either frequency-lock or phase-lock mode. The LCKI flag indicates the locked state of the module, and the LCKSEN bit configures whether the hardware setting (“is locked”) or clearing (“is unlocked”) the LCKI bit causes the interrupt, if enabled (LCKIEN = 1). A PLL interrupt can also be generated whenever the module is trying to lock but saturates (high or low) the DCO period, which indicates that the range should be adjusted. This “is saturated” interrupt is indicated by the HLMT and LLMT flags and is automatically disabled when the module is locked (LCKI = 1). It will automatically become active again if locked status is lost for any reason (LCKI = 0). The LCKI interrupt is level sensitive, and the HLMT and LLMT interrupts are edge sensitive. All three flags (LCKI, HLMT, and LLMT) can be cleared by setting OUTMD to 00b or 01b.

30.4. Output Modes

The PLL module has three available output modes: free-running DCO, frequency-lock, and phase-lock.

30.4.1. Free-Running DCO Mode

The PLL module includes its own Digitally-Controlled Oscillator (DCO) and does not need a reference frequency to generate a clock output. Using the module in this matter is called free-running DCO mode and is enabled by setting OUTMD to 01b.

When in free-running DCO mode, the output frequency of the DCO is determined by the RANGE, CAL, and DITHER settings. The spectrum spreading feature is also available in free-running DCO mode.

The output frequency of the DCO in free-running DCO mode is given by Equation 30.1:

$$F_{DCO_{avg}} = \frac{1}{T(RANGE) + T(CAL + \frac{DITHER}{16})}$$

Equation 30.1. Average DCO Output Frequency

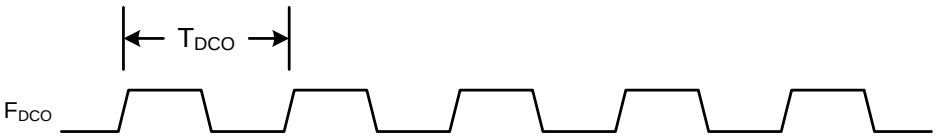


Figure 30.2. Free-Running DCO Mode

More information on dithering and spectrum spreading can be found in Section 30.5.1 and Section 30.5.2.

30.4.1.1. Free-Running DCO PLL Setup

To set up the PLL for free-running DCO mode:

1. If the output of the DCO will be used as the AHB clock and the AHB clock will be increasing in frequency, program the flash read timing bits to the appropriate value for the new clock rate.
2. Program the CAL field to the maximum value.

3. Program the desired output range by setting the RANGE field.
4. Program the desired output frequency by setting the CAL field.
5. (Optional) Enable dithering by setting DITHER to a non-zero value.
6. (Optional) Enable spectrum spreading by writing a non-zero value to SSAMP.
7. Set OUTMD to 01b.
8. Switch the AHB clock source to the DCO output using the device's clock control module.
9. If the output of the DCO will be used as the AHB clock and the AHB clock will be decreasing in frequency, program the flash read timing bits to the appropriate value for the new clock rate.

If the CAL, DITHER, or SSAMP DCO settings need to be changed when the output frequency running, the settings can be modified directly even when using the DCO output as the AHB clock. Any changes to RANGE should be made while the DCO output is off (OUTMD = 00b).

30.4.2. Frequency-Lock Mode

In frequency-lock mode, the PLL module will ensure the frequency of the DCO output (F_{DCO}) is derived from the reference frequency (F_{REF}) correctly based on the values of N and M, but the phase of the output frequency may not necessarily match the reference source. Any changes in frequency on the reference will propagate through to the DCO output frequency. frequency-lock mode is enabled by setting OUTMD to 10b.

$$F_{DCO} = F_{REF} \times \frac{N + 1}{M + 1}$$

Equation 30.2. PLL Output Frequency

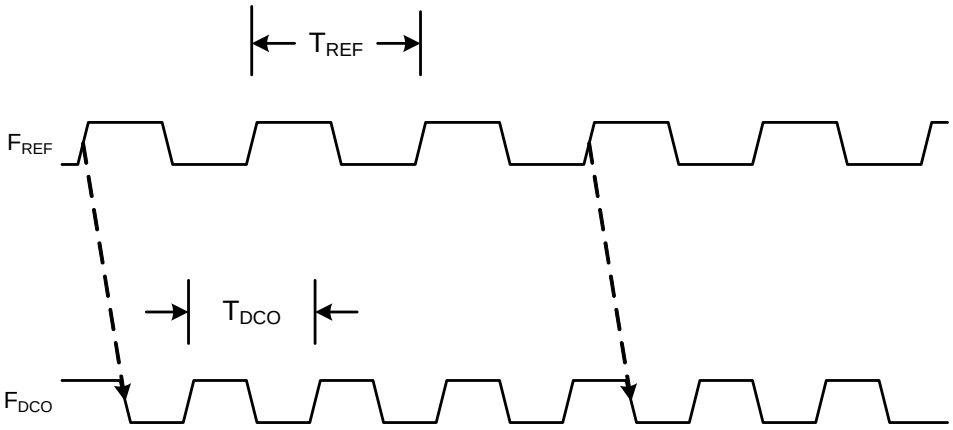


Figure 30.3. Frequency-Lock Mode

The hardware will take the firmware-set RANGE, N, and M values and automatically adjust CAL to match the reference frequency. Additionally, finer frequency matching can be achieved by enabling automatic dithering (DITHEN = 1), and generated noise can be reduced by enabling spectrum spreading.

30.4.3. Phase-Lock Mode

Phase-Lock mode matches the phase and frequency of the DCO output (F_{DCO}) to the reference frequency (F_{REF}) based on the values of N and M. Any drift or changes in phase and frequency on the reference will propagate through to the DCO output phase and frequency. Phase-Lock mode is enabled by setting OUTMD to 11b.

$$F_{DCO} = F_{REF} \times \frac{N + 1}{M + 1}$$

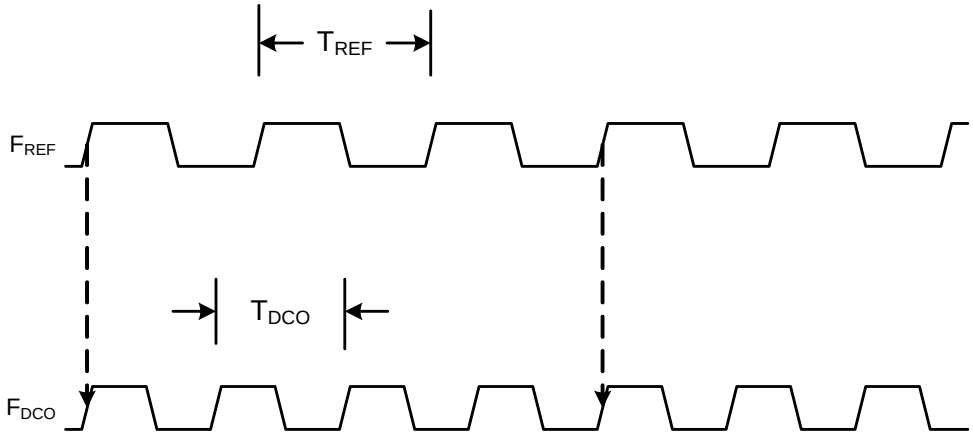


Figure 30.4. Phase-Lock Mode

The hardware will take the firmware-set RANGE, N, and M values and automatically adjust CAL to match the reference phase and frequency. Additionally, finer frequency matching can be achieved by enabling automatic dithering (DITHEN = 1), and generated noise can be reduced by enabling spectrum spreading.

30.4.4. Frequency-Lock and Phase-Lock Differences

In frequency-lock mode, the frequency error is bounded by definition, but this error will not necessarily approach zero over time. phase-lock mode guarantees the average frequency error approaches zero over time. As a result, the PLL module may need to make DCO output period corrections more often in phase-lock mode, resulting in higher output jitter. Additionally, frequency-lock mode generates less overshoot and undershoot compared to phase-lock mode, resulting in a better transient response.

frequency-lock mode is sufficient for most applications that require a specific output frequency.

30.4.5. Selecting N and M Values

In frequency-lock and phase-lock modes, the output jitter and lock time of the PLL module are dependent upon the N and M factors and the DCO output frequency. Larger values of N and M increase lock time but decrease output jitter. This relationship is shown in Figure 30.5. Longer DCO output periods (slower frequencies) increase lock time.

Equation 30.3 gives the approximate equation for DCO output jitter in frequency-lock or phase-lock modes.

$$\Delta t \sim \pm t_{DCO} \times \left(\frac{1}{N + 1} + \frac{1}{2500} \right)$$

Equation 30.3. Output Jitter

Equation 30.4 and Equation 30.5 approximate the maximum DCO lock time in frequency-lock or phase-lock modes (the PLL may lock faster). In the equations, t_{REF} is the period of the reference clock, t_{DCO} is the period of the DCO output, and t_{INIT} is the period of the DCO before enabling the PLL.

$$t_{lock} \approx t_{REF} \times (M + 1) + 2 \times t_{REF} + 32 \times t_{INIT}$$

Equation 30.4. DCO Lock Time with LOCKTH = 0

$$t_{lock} \approx t_{REF} \times (M + 1) \times (LOCKTH + 1) + 2 \times t_{REF} + 32 \times t_{DCO}$$

Equation 30.5. DCO Lock Time with LOCKTH > 0

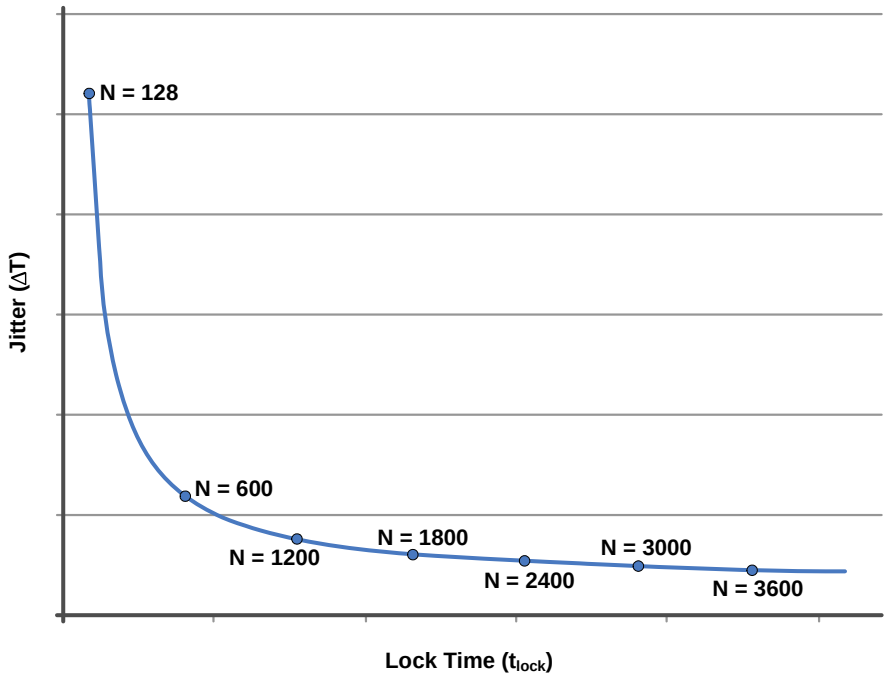


Figure 30.5. Jitter vs. Lock Time for Values of N

The value of M and N should be selected such that the lock time and jitter requirements of the system are met. Alternatively, a small value of N can be used initially to speed the locking process, and then a larger value of N (and consequently M) can be re-written to the registers to reduce output jitter once the module is locked to the reference. Once N is selected, the desired output frequency dictates the corresponding value of M. This method is discussed in detail in Section 30.6.2. The process for updating the PLL register values when the DCO is running is discussed in Section 30.4.5.1.

30.4.5.1. Initial Frequency-Lock and Phase-Lock PLL Setup

To set up the PLL for frequency-lock or phase-lock mode:

1. Ensure that the reference clock to be used (internal or external) is running and stable.
2. If the output of the PLL will be used as the AHB clock and the AHB clock will be increasing in frequency, program the flash read timing bits to the appropriate value for the new clock rate.
3. Set the REFSEL bits to select the desired reference clock for the PLL DCO.
4. Program the CAL field to the maximum value.
5. Program the desired output range by setting the RANGE field.
6. Program the appropriate N and M values to achieve the desired output frequency.
7. Select the edge (rising or falling) of the reference frequency the DCO output should lock to using the EDGSEL bit.
8. (Recommended) Enable dithering by setting DITHEN.
9. (Optional) Enable spectrum spreading by writing a non-zero value to SSAMP.
10. (Optional) Enable CAL saturation interrupts by setting LMTIEN to 1.
11. (Optional) Enable PLL interrupts and set LCKSEN to 1 to enable the “is locked” interrupt when the hardware sets the LCKI flag to 1.
12. Set OUTMD to 10b for frequency-lock mode or 11b for phase-lock mode.
13. Wait for the LCKI interrupt (if interrupts are enabled) or poll on LCKI until it changes from 0 to 1. If RANGE is set to the proper value, the HLMT and LLMT saturation interrupts should not occur. If these interrupts are enabled and trigger, this indicates that the RANGE setting is too low or high for the desired output frequency. The RANGE setting can be adjusted by writing OUTMD to 00b, writing RANGE with the new value, and setting OUTMD back to 10b for frequency-lock mode or 11b for phase-lock mode.
14. (Optional) Switch the value of LCKSEN to 0 to enable interrupts if the PLL module loses lock on the reference frequency.BBREN
15. Switch the AHB clock source to the DCO output using the CONTROL register in the CLKCTRL module.
16. If the output of the PLL will be used as the AHB clock and the AHB clock will be decreasing in frequency, program the flash read timing bits to the appropriate value for the new clock rate.

30.4.5.2. Modifying the RANGE Setting while the PLL is Locked and Running

If the PLL RANGE setting needs to be changed when the output frequency is locked and running, implement the following procedure:

1. Switch the AHB clock to another clock source that is running and stable.
2. Set the OUTMD bits to 00b.
3. Program the CAL field to the maximum value.
4. Modify the RANGE setting as desired.
5. If interrupts are enabled, set LCKSEN to 1 to enable the “is locked” interrupt when the LCKI flag is set to 1.
6. Set the OUTMD bits to 10b for frequency-lock mode or 11b for phase-lock mode. The PLL module will relock to the reference frequency using the new settings.
7. Wait for the LCKI interrupt (if interrupts are enabled) or poll on LCKI until it changes from 0 to 1. If RANGE is set to the proper value, the HLMT and LLMT saturation interrupts should not occur.
8. (Optional) Switch the value of LCKSEN to 0 to enable interrupts if the PLL module loses lock.

9. Switch the AHB clock source to the DCO output using the CONTROL register in the CLKCTRL module.

30.4.5.3. Modifying PLL Settings (other than RANGE) while the PLL is Locked and Running

If the other PLL settings need to be changed when the output frequency is locked and running, the following procedure should be implemented:

1. Set the OUTMD bits to 01.
2. Modify the settings as desired.
3. If interrupts are enabled, set LCKSEN to 1 to enable the “is locked” interrupt when the LCKI flag is set to 1.
4. Set the OUTMD bits to 10b for frequency-lock mode or 11b for phase-lock mode. The PLL module will relock to the reference frequency using the new settings.
5. Wait for the LCKI interrupt (if interrupts are enabled) or poll on LCKI until it changes from 0 to 1. If RANGE is set to the proper value, the HLMT and LLMT saturation interrupts should not occur.
6. (Optional) Switch the value of LCKSEN to 0 to enable interrupts if the PLL module loses lock.

If a sensitive analog measurement needs to be taken, the automatic DCO output frequency adjustment (including dithering and spectrum spreading) can be temporarily halted by setting the STALL bit. The module will not resume the output updates until STALL has been cleared to 0 by firmware.

30.5. Additional Features

In addition to three output modes, the PLL module has the additional features of output frequency dithering and spectrum spreading.

30.5.1. Output Frequency Dithering

The CAL field provides 12 bits of frequency steps within a particular output range, determined by RANGE. The output frequency dithering feature allows an average frequency of finer resolution than CAL alone can provide, though the instantaneous frequency will not be equal to the average frequency at a particular moment in time. The dithering setting controls how often a 1 is added to CAL to achieve the desired average frequency. The DITHER value acts like fractional bits of CAL.

In free-running DCO mode (OUTMD set to 01b), firmware can set the dithering to a particular setting by writing the desired value to DITHER. Dithering can be disabled by writing 0’s to DITHER.

In frequency-lock or phase-lock modes (OUTMD set to 10b or 11b), automatic dithering is enabled by setting DITHEN to 1 and will cause the hardware to automatically set these dithering bits to achieve average frequencies the CAL setting would normally not be able to achieve. Disabling automatic dithering causes the hardware to always set DITHER to 0 when in frequency-lock or phase-lock mode.

Setting the STALL bit to 1 will disable dithering until STALL is written with a 0.

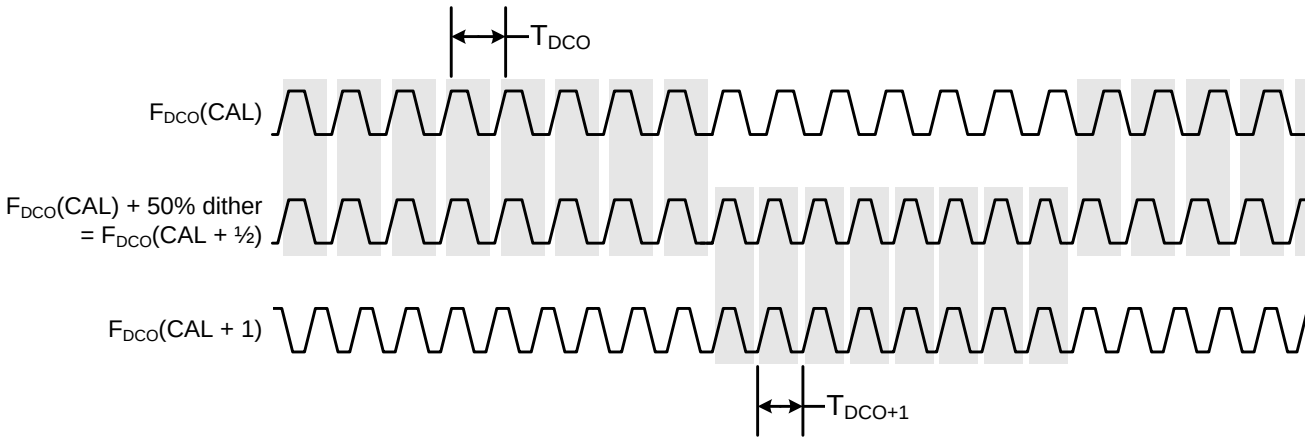


Figure 30.6. Dithering Timing Example

The average DCO output frequency when dithering is enabled is given by Equation 30.1. Dithering is available in any output mode.

30.5.2. Output Frequency Spectrum Spreading

Spectrum spreading adds intentional noise to the DCO output period to improve system noise performance. Spectrum spreading is a signed random number of uniform distribution added to the DCO output period to slightly vary the resulting clock output. This random number has an adjustable amplitude based on the SSAMP field and an adjustable update rate based on the SSUINV field. Setting the STALL bit to 1 will disable spectrum spreading until STALL is written with a 0.

Note: The added noise from spectrum spreading is centered about the DCO output period. As a result, the peak frequencies may exceed the maximum allowable frequency for the AHB clock. As a result, the desired output frequency created by RANGE, N, and M should be programmed below the maximum allowable frequency to allow some headroom for spectrum spreading.

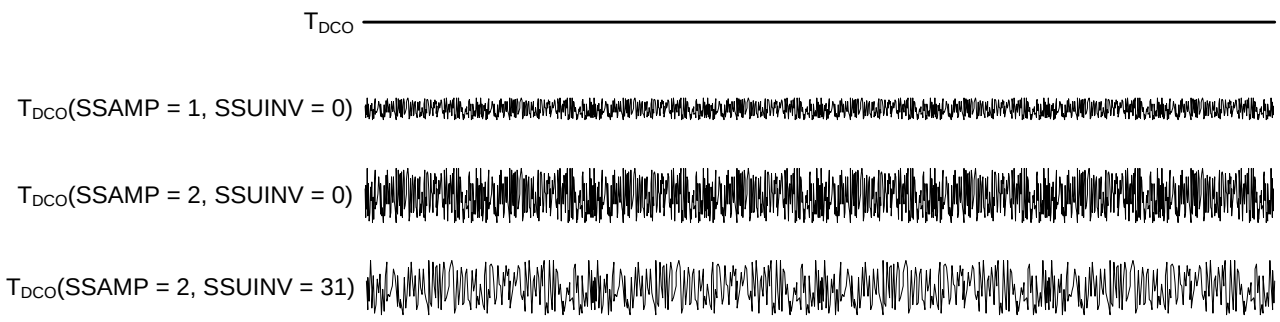


Figure 30.7. Spectrum Spreading Example

The spectrum spreading amplitude has fixed settings of approximately ± 0.1%, ± 0.2%, ± 0.4%, ± 0.8%, or ± 1.6% of the DCO output period. Equation 30.6 describes the spectrum spreading update rate.

$$UpdateRate = 4 \times T_{DCO} \times (SSUINV + 1)$$

Equation 30.6. Spectrum Spreading Update Rate

30.6. Advanced Setup Examples

This section discusses advanced uses of the PLL module to save power or reduce the external components required by a system.

30.6.1. Temporarily Using a Reference

The PLL module can temporarily use a reference to tune the DCO to a frequency. Once the DCO is locked to the reference, the PLL can be switched to free-running DCO mode and the reference source can be disabled. This enables the system to operate at the desired frequency without providing power to the reference source. The DCO output created in this manner will remain stable at the correct frequency, but may not have tolerances as tight as the original reference source. The reference can periodically be enabled and the DCO allowed to relock to improve tolerances, if desired.

To use the PLL module in this manner, follow the procedure in Section 30.4.5.1 for frequency-lock mode and then set OUTMD to 01b for free-running DCO mode. The hardware-tuned CAL and DITHER settings will remain until overwritten by either firmware or hardware (OUTMD set to 10b or 11b).

30.6.2. Obtaining Fast Lock Times and Low Output Jitter

As discussed in Section 30.4.4, the value of N affects both the lock time and the output jitter of the DCO output frequency. To obtain both a fast lock time and low output jitter, a small value of N can be used initially to speed the locking process, and then a larger value of N can be re-written to the registers to reduce output jitter once the module is locked to the reference.

For example, a DCO output frequency of approximately 70 MHz is desired using a reference clock of 48 MHz. Using Equation 30.2, if N is 500, the corresponding value of M is 342 for a desired output frequency of 70 MHz. If N is 3000, the corresponding value of M is 2056.

To obtain a fast lock time and a long-term low output jitter:

1. Follow the procedure in Section 30.4.5.1 for frequency-lock or phase-lock mode, setting N to 128 and M to 87.
2. Once the DCO output is LCKI, set OUTMD to 01b for free-running DCO mode.
3. Change N to 3000 and M to 2056.
4. Set OUTMD back to 10b for frequency-lock mode or 11b for phase-lock mode.

30.7. PLL0 Registers

This section contains the detailed register descriptions for PLL0 registers.

Register 30.1. PLL0_DIVIDER: Reference Divider Setting

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved				N											
Type	R				RW											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved				M											
Type	R				RW											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

PLL0_DIVIDER = 0x4003_B000

Table 30.1. PLL0_DIVIDER Register Bit Descriptions

Bit	Name	Function
31:28	Reserved	Must write reset value.
27:16	N	**N Divider Value.** Selects the reference clock multiplier. Valid values include 32 to 4095 (values below 31 are invalid and may cause incorrect behavior). The frequency-lock or phase-lock output frequency is the result of the equation: $F_{DCO} = F_{REF} \times \frac{N + 1}{M + 1}$
15:12	Reserved	Must write reset value.

Notes:

1. All PLL register fields except LCKPOL, LMTIEN, and STALL must remain static while OUTMD is set to frequency-lock or phase-lock modes. If RANGE setting changes are required, turn off the DCO output, write the maximum value to CAL, write to RANGE, and re-enable frequency-lock or phase-lock mode. All other settings can be changed by putting the module in Free-Running Mode, writing to the registers, and re-enabling frequency-lock or phase-lock mode.

Table 30.1. PLL0_DIVIDER Register Bit Descriptions

Bit	Name	Function
11:0	M	**M Divider Value.** Selects the reference clock divider. Valid values include the entire 12-bit range of M (0 to 4095). The frequency-lock or phase-lock output frequency is the result of the equation: $F_{DCO} = F_{REF} \times \frac{N + 1}{M + 1}$

Notes:

1. All PLL register fields except LCKPOL, LMTIEN, and STALL must remain static while OUTMD is set to frequency-lock or phase-lock modes. If RANGE setting changes are required, turn off the DCO output, write the maximum value to CAL, write to RANGE, and re-enable frequency-lock or phase-lock mode. All other settings can be changed by putting the module in Free-Running Mode, writing to the registers, and re-enabling frequency-lock or phase-lock mode.

Register 30.2. PLL0_CONTROL: Module Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	OUTMD		EDGSEL	DITHEN	Reserved	STALL	Reserved				LOCKTH		Reserved		REFSEL	
Type	RW		RW	RW	RW	RW	R				RW		R		RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved				LCKPOL	LCKIEN	LMTIEN	Reserved						LCKI	HLMTF	LLMTF
Type	R				RW	RW	RW	R						R	R	R
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

PLL0_CONTROL = 0x4003_B010

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 30.2. PLL0_CONTROL Register Bit Descriptions

Bit	Name	Function
31:30	OUTMD	**PLL Output Mode.** Sets the DCO output mode. All PLL register fields except LCKPOL, LMTIEN, and STALL must remain static while OUTMD is set to frequency-lock or phase-lock modes. If RANGE setting changes are required, turn off the DCO output, write the maximum value to CAL, write to RANGE, and re-enable frequency-lock or phase-lock mode. All other settings can be changed by putting the module in Free-Running Mode, writing to the registers, and re-enabling frequency-lock or phase-lock mode. 00: DCO output is off. 01: DCO output is in Free-Running DCO mode. 10: DCO output is in frequency-lock mode (reference source required). 11: DCO output is in phase-lock mode (reference source required).

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.
2. All PLL register fields except LCKPOL, LMTIEN, and STALL must remain static while OUTMD is set to frequency-lock or phase-lock modes. If RANGE setting changes are required, turn off the DCO output, write the maximum value to CAL, write to RANGE, and re-enable frequency-lock or phase-lock mode. All other settings can be changed by putting the module in Free-Running Mode, writing to the registers, and re-enabling frequency-lock or phase-lock mode.

Table 30.2. PLL0_CONTROL Register Bit Descriptions

Bit	Name	Function
29	EDGSEL	**Edge Lock Select.** Selects between locking on the rising edge or falling edge of the reference frequency in frequency-lock and phase-lock modes. The setting of this bit has no effect in Free-Running DCO mode. 0: Lock DCO output frequency to the falling edge of the reference frequency. 1: Lock DCO output frequency to the rising edge of the reference frequency.
28	DITHEN	**Dithering Enable.** Enables automatic dithering on the DCO output period in frequency-lock and phase-lock modes. The DITHER field is used to generate a 1-bit dither of the DCO output frequency. This provides better performance and generally reduces the overall jitter in frequency-lock and phase-lock modes. Automatic dithering can be disabled by clearing DITHEN to 0, which forces the hardware to always generate DITHER values of 0. When OUTMD is set to 01b, firmware can write a non-zero value to the DITHER bits to enable dithering, even if DITHEN is 0. Setting DITHEN to 1 in Free-Running DCO mode will have no effect. 0: Automatic DCO output dithering disabled. 1: Automatic DCO output dithering enabled.
27	Reserved	Must write reset value.
26	STALL	**DCO Output Updates Stall.** When STALL is set to 1, the phase-lock and frequency-lock modes are temporarily disabled. Spectrum spreading and dithering are also disabled. This is useful for providing the quietest environment possible for sensitive analog measurements. The phase-lock and frequency-lock modes, spectrum spreading, and dithering, are re-enabled when STALL is cleared to 0. 0: In phase-lock and frequency-lock modes, spectrum spreading, and dithering operate normally, if enabled. 1: In phase-lock and frequency-lock modes, spectrum spreading, and dithering are prevented from updating the output of the DCO.
25:22	Reserved	Must write reset value.
21:20	LOCKTH	**Lock Threshold Control.** Lock is deterministic based on the number of (M+1)xTREF cycles over which the algorithm has operated. Assuming lock is possible given the present RANGE setting, phase lock is guaranteed after one (M+1)xTREF cycle. The value of LOCKTH sets the number of extra (M+1)xTREF cycles to wait before declaring lock. It is recommended to set the LOCKTH field to 1 if the PLL will be switched to free-running mode immediately after lock.
19:18	Reserved	Must write reset value.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.
2. All PLL register fields except LCKPOL, LMTIEN, and STALL must remain static while OUTMD is set to frequency-lock or phase-lock modes. If RANGE setting changes are required, turn off the DCO output, write the maximum value to CAL, write to RANGE, and re-enable frequency-lock or phase-lock mode. All other settings can be changed by putting the module in Free-Running Mode, writing to the registers, and re-enabling frequency-lock or phase-lock mode.

Table 30.2. PLL0_CONTROL Register Bit Descriptions

Bit	Name	Function
17:16	REFSEL	**Reference Clock Selection Control.** 00: PLL reference clock (FREF) is the RTC0 oscillator (RTC0OSC). 01: PLL reference clock (FREF) is the divided Low Power Oscillator (LPOSC0). 10: PLL reference clock (FREF) is the external oscillator output (EXTOSC0). 11: PLL reference clock (FREF) is the USB0 oscillator (USB0OSC).
15:12	Reserved	Must write reset value.
11	LCKPOL	**Lock Interrupt Polarity.** Sets the state of LCKI that causes the PLL lock state interrupt to occur, if enabled. 0: The lock state PLL interrupt will occur when LCKI is 0. 1: The lock state PLL interrupt will occur when LCKI is 1.
10	LCKIEN	**Locked Interrupt Enable.** 0: The PLL locking does not cause an interrupt 1: An interrupt is generated if LCKI matches the state selected by LCKPOL.
9	LMTIEN	**Limit Interrupt Enable.** Enables interrupt generation if the DCO output frequency saturates high or low, preventing the DCO from reliably locking to the reference frequency or phase. This interrupt will not be generated while the DCO is locked (LCKI = 1). 0: Saturation (high and low) interrupt disabled. 1: Saturation (high and low) interrupt enabled.
8:3	Reserved	Must write reset value.
2	LCKI	**Phase-Lock and Frequency-Lock Locked Interrupt Flag.** Indicates when the PLL module DCO is locked to the reference clock. In phase-lock mode, this indicates that the DCO is phase-locked with the reference clock. In frequency-lock mode, this indicates that the DCO is frequency-locked (not necessarily phase-locked) with the reference clock. This bit will also assert the level-sensitive lock state interrupt, if enabled. This bit is automatically set and cleared by hardware, but can also be manually cleared by writing 00b or 01b to OUTMD. 0: DCO is disabled or not locked. 1: DCO is enabled and locked.
1	HLMTF	**CAL Saturation (High) Flag.** Indicates when the DCO output period is saturated high and the DCO cannot lock reliably, so the RANGE value should be decreased. This flag is not automatically cleared by hardware and must be cleared by software by writing 00b or 01b to OUTMD. 0: DCO period is not saturated high. 1: DCO period is saturated high.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.
2. All PLL register fields except LCKPOL, LMTIEN, and STALL must remain static while OUTMD is set to frequency-lock or phase-lock modes. If RANGE setting changes are required, turn off the DCO output, write the maximum value to CAL, write to RANGE, and re-enable frequency-lock or phase-lock mode. All other settings can be changed by putting the module in Free-Running Mode, writing to the registers, and re-enabling frequency-lock or phase-lock mode.

Table 30.2. PLL0_CONTROL Register Bit Descriptions

Bit	Name	Function
0	LLMTF	**CAL Saturation (Low) Flag.** Indicates when the DCO output period is saturated low and the DCO cannot lock reliably, so the RANGE value should be increased. This flag is not automatically cleared by hardware and must be cleared by software by writing 00b or 01b to OUTMD. 0: DCO period is not saturated low. 1: DCO period is saturated low.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.
2. All PLL register fields except LCKPOL, LMTIEN, and STALL must remain static while OUTMD is set to frequency-lock or phase-lock modes. If RANGE setting changes are required, turn off the DCO output, write the maximum value to CAL, write to RANGE, and re-enable frequency-lock or phase-lock mode. All other settings can be changed by putting the module in Free-Running Mode, writing to the registers, and re-enabling frequency-lock or phase-lock mode.

Register 30.3. PLL0_SSPR: Spectrum Spreading Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved			SSUINV					Reserved					SSAMP		
Type	R			RW					R					RW		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address
PLL0_SSPR = 0x4003_B020

Table 30.3. PLL0_SSPR Register Bit Descriptions

Bit	Name	Function
31:13	Reserved	Must write reset value.
12:8	SSUINV	**Spectrum Spreading Update Interval.** The update interval is given by the following equation: $UpdateInterval = 4 \times T_{DCO} \times (SSUINV + 1)$
7:3	Reserved	Must write reset value.
2:0	SSAMP	**Spectrum Spreading Amplitude.** 000: Disable Spectrum Spreading. 001: Spectrum Spreading set to approximately ± 0.1% of TDCO. 010: Spectrum Spreading set to approximately ± 0.2% of TDCO. 011: Spectrum Spreading set to approximately ± 0.4% of TDCO. 100: Spectrum Spreading set to approximately ± 0.8% of TDCO. 101: Spectrum Spreading set to approximately ± 1.6% of TDCO. 110-111: Reserved.

Notes:
1. All PLL register fields except LCKPOL, LMTIEN, and STALL must remain static while OUTMD is set to frequency-lock or phase-lock modes. If RANGE setting changes are required, turn off the DCO output, write the maximum value to CAL, write to RANGE, and re-enable frequency-lock or phase-lock mode. All other settings can be changed by putting the module in Free-Running Mode, writing to the registers, and re-enabling frequency-lock or phase-lock mode.

Register 30.4. PLL0_CALCONFIG: Calibration Configuration

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved													RANGE		
Type	R													RW		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	X	X	X
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	CAL												DITHER			
Type	RW												RW			
Reset	X	X	X	X	X	X	X	X	X	X	X	X	0	0	0	0

Register ALL Access Address

PLL0_CALCONFIG = 0x4003_B030

Table 30.4. PLL0_CALCONFIG Register Bit Descriptions

Bit	Name	Function
31:19	Reserved	Must write reset value.
18:16	RANGE	**DCO Range.** Determines the output frequency range of the DCO. 000: DCO operates from 23 to 37 MHz. 001: DCO operates from 33 to 54 MHz. 010: DCO operates from 45 to 71 MHz. 011: DCO operates from 53 to 80 MHz. 100: DCO operates from 73 to 80 MHz. 101-111: Reserved.
15:4	CAL	**DCO Calibration Value.** This value adjusts the output period of the PLL module DCO in fine steps. Increasing CAL increases the DCO output period and decreases the DCO output frequency.

Notes:

1. All PLL register fields except LCKPOL, LMTIEN, and STALL must remain static while OUTMD is set to frequency-lock or phase-lock modes. If RANGE setting changes are required, turn off the DCO output, write the maximum value to CAL, write to RANGE, and re-enable frequency-lock or phase-lock mode. All other settings can be changed by putting the module in Free-Running Mode, writing to the registers, and re-enabling frequency-lock or phase-lock mode.

Table 30.4. PLL0_CALCONFIG Register Bit Descriptions

Bit	Name	Function
3:0	DITHER	**DCO Dither Setting.** The DITHER bits control how often a 1 is added to CAL to create an average frequency in between the two CAL settings (CAL and CAL + 1). The DITHER value acts like fractional bits of CAL. In Free-Running DCO mode, firmware can write these bits to select a specific dithering setting. Writing 0's to this field disables dithering. The value of DITHEN has no effect in this mode. When DITHEN is set to 1 in frequency-lock or phase-lock modes, the hardware will automatically adjust the DITHER bits. If DITHEN is set to 0 in these modes, the hardware will force the DITHER bits to 0.

Notes:

1. All PLL register fields except LCKPOL, LMTIEN, and STALL must remain static while OUTMD is set to frequency-lock or phase-lock modes. If RANGE setting changes are required, turn off the DCO output, write the maximum value to CAL, write to RANGE, and re-enable frequency-lock or phase-lock mode. All other settings can be changed by putting the module in Free-Running Mode, writing to the registers, and re-enabling frequency-lock or phase-lock mode.

30.8. PLL0 Register Memory Map

Table 30.5. PLL0 Memory Map

Register Name	PLL0_CALCONFIG	PLL0_SSPR	PLL0_CONTROL	PLL0_DIVIDER
ALL Address	0x4003_B030	0x4003_B020	0x4003_B010	0x4003_B000
Access Methods	ALL	ALL	ALL | SET | CLR	ALL
Bit 31	Reserved	Reserved	OUTMD	Reserved
Bit 30				
Bit 29			EDGSEL	
Bit 28			DITHEN	
Bit 27			Reserved	N
Bit 26			STALL	
Bit 25			Reserved	
Bit 24				
Bit 23				
Bit 22				
Bit 21			LOCKTH	
Bit 20				
Bit 19			Reserved	
Bit 18				
Bit 17			REFSEL	
Bit 16				
Bit 15	RANGE		Reserved	Reserved
Bit 14				
Bit 13				
Bit 12				
Bit 11	CAL	SSUINV	LCKPOL	M
Bit 10			LCKIEN	
Bit 9			LMTIEN	
Bit 8			Reserved	
Bit 7		Reserved		
Bit 6				
Bit 5				
Bit 4				
Bit 3				
Bit 2	DITHER	SSAMP	LCKI	
Bit 1			HLMTF	
Bit 0			LLMTF	

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

31. Real Time Clock and Low Frequency Oscillator (RTC0)

This section describes the Real Time Clock and Low Frequency Oscillator (RTC) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the RTC block, which is used by all device families covered in this document.

31.1. RTC Features

The RTC module includes the following features:

- 32-bit timer (supports up to 36 hours) with three separate alarms.
- Option for one alarm to automatically reset the RTC timer.
- Missing clock detector.
- Can be used with the internal low frequency oscillator (LFOSC0), an external 32.768 kHz crystal (no additional resistors or capacitors necessary), or with an external CMOS clock.
- Programmable internal loading capacitors support a wide range of external 32.768 kHz crystals.
- Operates directly from VDD and remains operational even when the device goes into its lowest power down mode.
- Two buffered outputs provide an accurate, low frequency clock to other devices:
  - RTC0TCLK_OUT: This output connects via the crossbar to an IO pin. Because the crossbar is powered down in the sleep mode, this output is not suitable for applications requiring the RTC clock output in sleep mode.
  - RTC0OSC_OUT; This output connects directly to an IO pin (consult data sheet pin definition) and does not pass through the crossbar. This output should be used for applications requiring the RTC clock output in sleep mode."

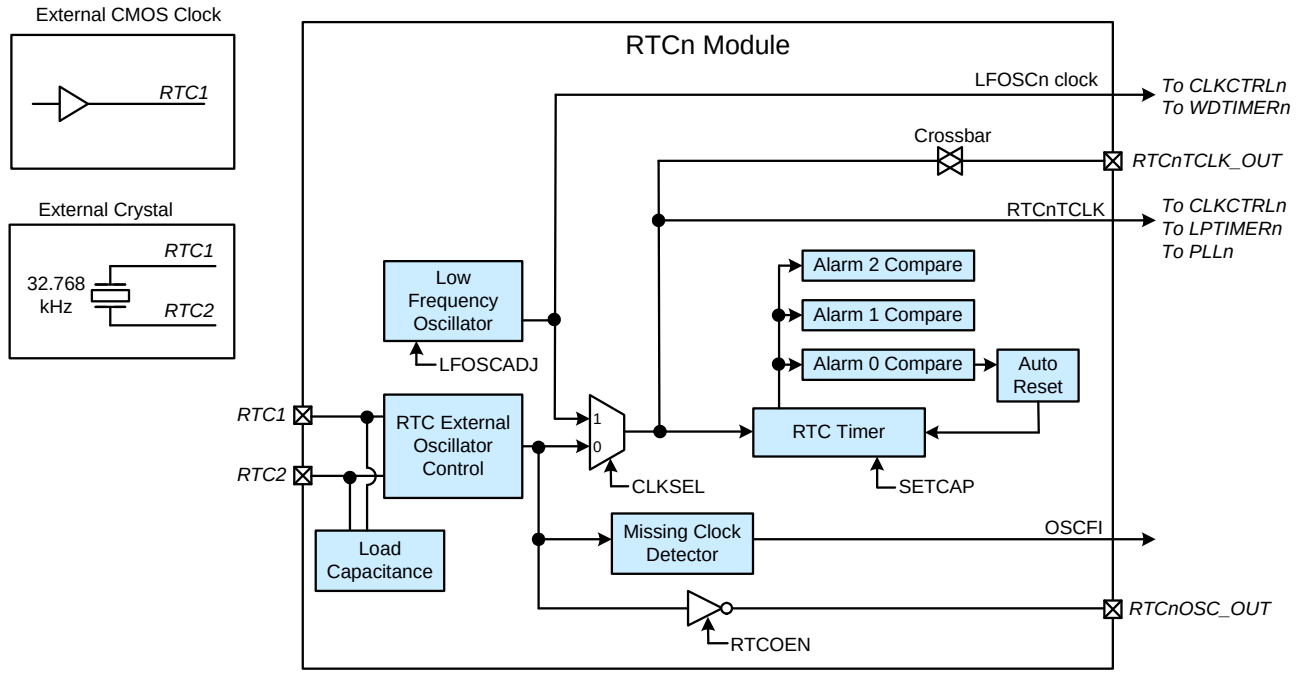


Figure 31.1. RTC Block Diagram

31.2. Overview

The RTC module allows a maximum of 36 hour 32-bit independent time-keeping when used with a 32.768 kHz watch crystal. The RTC provides three alarm events in addition to a missing clock event, which can also function as an interrupt, reset or wakeup source.

The RTC module includes internal loading capacitors that are programmable to 16 discrete levels, allowing compatibility with a wide range of crystals.

The RTCnOSC output can be buffered and routed to a port bank pin to provide an accurate, low frequency clock to other devices while the core is in its lowest power down mode. The module also includes a low power internal low frequency oscillator that reduces sleep mode current and is available for other modules to use as a clock source.

31.3. Clocking

The RTC module clocks from its own timebase independent of the AHB clock. This timebase can be derived from an external CMOS clock, an external 32.768 kHz crystal, or an internal low frequency oscillator.

31.3.1. Crystal Mode

When using crystal mode, a 32.768 kHz crystal should be connected between RTC1 and RTC2. No other external components are required. The following steps show how to start the RTC crystal oscillator in firmware:

1. Configure the RTC1 and RTC2 pins as analog inputs using the device port configuration module.
2. Disable automatic gain control (AGCEN = 0) and enable bias doubling (BDEN = 1).
3. Program the RTCLC field.
4. Enable automatic load capacitance stepping (ASEN = 1).
5. Select the RTC oscillator (CLKSEL = 0).
6. Enable the crystal oscillator (CRYSEN = 1).
7. Wait 20 ms.
8. Poll the clock valid (CLKVF) until the crystal oscillator stabilizes.
9. Poll the load capacitance ready (LRDYF) flag until the load capacitance reaches its programmed value.
10. Enable automatic gain control (AGCEN = 1) and disable bias doubling (BDEN = 0) for maximum power savings.
11. Enable the missing clock detector (MCLKEN = 1).
12. Wait 2 ms.
13. Clear OSCFI.
14. If serving as a wake up source from a low-power mode, clear the wake-up source flags in the device power management module.

Figure 31.2 shows the hardware configuration for crystal mode.

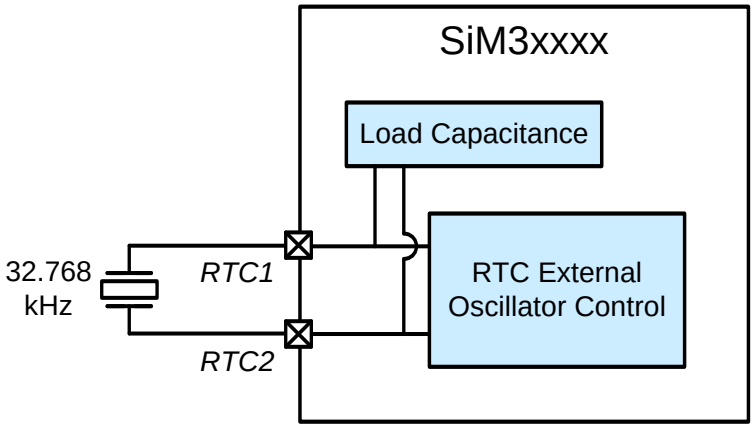


Figure 31.2. Crystal Mode Hardware Configuration

31.3.2. External CMOS Clock Mode

The RTC oscillator may also be driven by an external CMOS clock. The CMOS clock should be applied to RTC1, and RTC2 should be left floating. In this mode, the external CMOS clock should have a minimum voltage swing of 400 mV without exceeding VDD or dropping below VSS. Bias levels closer to VDD will result in lower I/O power consumption because the RTC1 pin has a built-in weak pull-up.

To use the module with an external CMOS clock:

1. Configure the RTC1 as an analog input using the device port configuration module. RTC2 may be left floating or used for other functions.
2. Disable automatic gain control (AGCEN = 0) and disable bias doubling (BDEN = 0).
3. Select the lowest bias setting (RTCLC = 0).
4. Select the RTC oscillator (CLKSEL = 0).
5. Enable the crystal oscillator (CRYSEN = 1).
6. Wait 2 ms.
7. Enable the missing clock detector (MCLKEN = 1).
8. Wait 2 ms.
9. Check the OSCFI flag to ensure a valid clock is present on RTC1.
10. If serving as a wake up source from a low-power mode, clear the wake-up source flags in the device power management module.

The CLKVF bit is indeterminate when using a CMOS clock with the RTC module.

Figure 31.3 shows the hardware configuration for external CMOS clock mode.

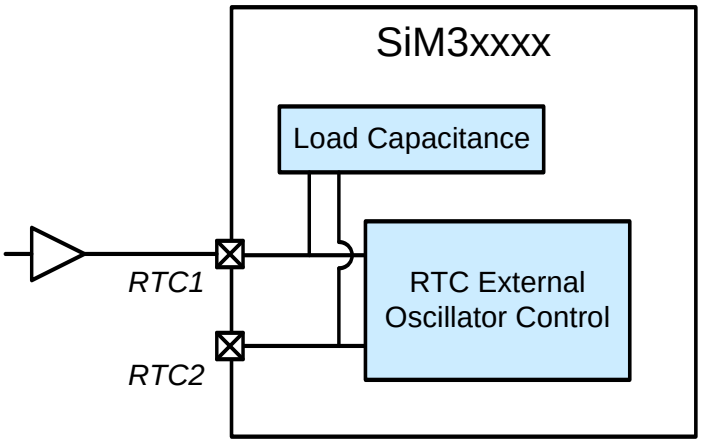


Figure 31.3. External CMOS Clock Mode Hardware Configuration

31.3.3. Low Frequency Oscillator

The low frequency oscillator (LFOSC0) provides a low power internal clock source for the RTC timer. No external components are required to use the low frequency oscillator and the RTC1 and RTC2 pins do not need to be shorted together. The typical oscillation frequency of the oscillator is 16.4 kHz, but may vary depending on supply voltage, temperature and process. Consult the electrical characteristics tables in the device data sheet for more information.

To use the low frequency oscillator with the RTC module:

1. Enable the low frequency oscillator (LFOSCEN = 1). The oscillator starts oscillating instantaneously.
2. Select the low frequency oscillator as the RTC timer clock source (CLKSEL = 1).
3. Disable the crystal oscillator (CRYSEN = 0).

4. If serving as a wake up source from a low-power mode, clear the wake-up source flags in the device power management module.

When using the low frequency oscillator as its clock source, the RTC module increments bit 1 of the 32-bit timer instead of bit 0, effectively multiplying the oscillator frequency by 2.

31.3.4. RTC Timer Clock Selection

The RTC timer clock (RTCnTCLK) is configured by the Clock Select (CLKSEL) bits to be either the RTC Crystal Oscillator (RTCnOSC), an external CMOS clock, or the low frequency oscillator (LFOSCn).

31.3.5. RTC Clock Outputs

31.3.5.1. RTC0OSC_OUT

If the RTC oscillator output is enabled (RTCOEN = 1), the RTCnOSC clock is buffered and output to the RTCnOSC_OUT pin for use by external modules. This output may still be enabled in sleep mode. Consult data sheet pin definition for location of the RTCnOSC_OUT function

31.3.5.2. RTC0TCLK_OUT

If the RTC0TCLK_OUT is enabled in the crossbar, the RTC timer clock is output onto the RTC0TCLK_OUT pin. Depending on the CLKSEL configuration, the RTC timer clock can be either the Low Frequency Oscillator (LFOOSC) or the RTC External Oscillator (RTCOSC). This output cannot be enabled in sleep mode.

31.3.6. Programmable Load Capacitance

The programmable load capacitance has 16 values to support a wide range of crystal oscillators. If automatic load capacitance stepping is enabled (ASEN = 1), the crystal load capacitors start at the smallest setting to allow a fast startup time, then increase the capacitance until reaching the final programmed value in the RTCLC field. The RTCLC setting specifies the amount of internal load capacitance and does not include any stray PCB capacitance. Once the final programmed loading capacitance value is reached, the hardware will set the load ready (LRDYF) flag to 1.

Table 31.1 shows the equivalent crystal load capacitance for RTCLC settings.

Table 31.1. Load Capacitance Settings

RTCLC Value	Crystal Load Capacitance	Equivalent Capacitance seen on RTC1 and RTC2
0	4.0 pF	8.0 pF
1	4.5 pF	9.0 pF
2	5.0 pF	10.0 pF
3	5.5 pF	11.0 pF
4	6.0 pF	12.0 pF
5	6.5 pF	13.0 pF
6	7.0 pF	14.0 pF
7	7.5 pF	15.0 pF
8	8.0 pF	16.0 pF
9	8.5 pF	17.0 pF
10	9.0 pF	18.0 pF
11	9.5 pF	19.0 pF

Table 31.1. Load Capacitance Settings

RTCLC Value	Crystal Load Capacitance	Equivalent Capacitance seen on RTC1 and RTC2
12	10.5 pF	21.0 pF
13	11.5 pF	23.0 pF
14	12.5 pF	25.0 pF
15	13.5 pF	27.0 pF

31.3.7. Automatic Gain Control (Crystal Mode Only) and Bias Doubling

Automatic gain control (AGC) allows the RTC oscillator to trim the oscillation amplitude of a crystal in order to achieve the lowest possible power consumption. Automatic gain control automatically detects when the oscillation amplitude has reached a point where it safe to reduce the drive current, and it may be enabled during crystal startup. It is recommended to enable AGC in most systems that use the RTC oscillator in crystal mode. The following are recommended crystal specifications and operating conditions when enabling AGC:

- ESR < 50 kΩ
- Load capacitance < 10 pF
- Supply voltage < 3.0 V
- Temperature > –20 °C

The chosen crystal should undergo an oscillation robustness test to ensure it will oscillate under the worst case condition to which the system will be exposed. This worst case condition will occur at the following system conditions: lowest temperature, highest supply voltage, highest ESR, highest load capacitance, and lowest bias current (AGC enabled, bias doubling disabled).

To perform the oscillation robustness test, the RTC oscillator output should be routed to a port pin configured as a push-pull digital output using the device port configuration module. The positive duty cycle of the output clock can be used as an indicator of oscillation robustness.

As shown in Figure 31.4, duty cycles less than the low threshold indicate a robust oscillation. As the duty cycle approaches the high threshold, oscillation becomes less reliable and the risk of clock failure increases. Increasing the bias current by disabling AGC will always improve oscillation robustness and will reduce the output clock's duty cycle. This test should be performed at the worst case system conditions, as results at very low temperatures or high supply voltage will vary from results taken at room temperature or low supply voltage. Consult the device data sheet for information on the robust duty cycle range specifications.

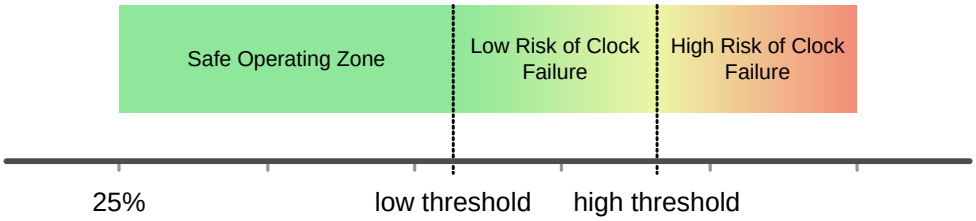


Figure 31.4. Interpreting Oscillation Robustness (Duty Cycle) Test Results

AGC may be disabled at the cost of increased power consumption. Disabling Automatic Gain Control will provide the crystal oscillator with higher immunity against the external factors that may cause clock failure.

The bias doubling feature can increase the self-oscillation frequency and allow a higher crystal drive strength in crystal mode. High crystal drive strength is recommended when the crystal is exposed to poor environmental conditions, including excessive moisture. The bias doubler should always be enabled during oscillator startup. Note that when the bias doubler is disabled, the RTC External Oscillator Valid Flag (CLKVF) is disabled and will always read 0.

Table 31.2 shows a summary of the oscillator AGC and bias doubling settings.

Table 31.2. RTC Bias Settings

Bias Doubling (BDEN) Setting	AGC (AGCEN) Setting	Power Consumption
off (0)	on (1)	lowest
off (0)	off (0)	low
on (1)	on (1)	high
on (1)	off (0)	highest

31.3.8. Missing Clock Detector

The missing clock detector (MCD) is a one-shot circuit enabled by setting MCLKEN to 1. When the MCD is enabled, hardware sets the oscillator fail (OSCFI) flag if the RTC oscillator (RTCnOSC) frequency drops below the missing clock detector trigger frequency given in the device data sheet.

The missing clock detector can only be used for the external crystal oscillator or external CMOS clock modes, and should be disabled if using the low frequency oscillator clock as the RTC timer clock.

An MCD timeout can trigger an interrupt, wake the device from a low power mode, or reset the device. This feature should be disabled when making changes to the oscillator settings to avoid undesired interrupts or resets.

31.3.9. Oscillator Crystal Valid Detector

The RTC oscillator crystal valid detector is an oscillation amplitude detector circuit used during crystal startup to determine when oscillation is nearly stable. Firmware can read the output of this detector using the clock valid (CLKVF) flag in the CONTROL register. The output of CLKVF is not valid for the first 2 ms after turning on the crystal oscillator. The CLKVF bit is always low when bias doubling is disabled (BDEN = 0).Therefore, the bias doubler should be enabled during crystal startup.

The crystal valid detector is not intended for detecting an oscillator failure - once set, the CLKVF will not be reset unless the external oscillator is disabled. To determine if the oscillator has failed, firmware should use the missing clock detector and OSCFI flag.

31.4. Accessing the Timer

The RTC timer is a 32-bit counter that increments every RTC oscillator cycle.

Firmware can set the value of the timer by writing a 32-bit value to the SETCAP register and setting the TMRSET. Hardware will automatically clear TMRSET when the set operation completes.

Firmware can read the current value of the timer by setting the TMRCAP bit. Hardware will automatically clear TMRCAP when the capture operation completes, and firmware can then read the SETCAP register.

If the AHB clock is greater than 4X the RTC clock, the RTC High Speed Mode Enable bit (HSMDEN) must be set to allow the timer value to be written or captured.

31.5. Alarms

The RTC timer has three alarm functions that can be set to generate an interrupt, wake the device from a low power mode, or reset the device at a specific time.

The alarms can be set using the ALARM0, ALARM1, and ALARM2 registers. These 32-bit fields are compared directly to the 32-bit timer value. Hardware sets the ALM0I, ALM1I, and ALM2I when the corresponding ALARMx value matches the timer, generating an interrupt, if enabled.

The alarms *and* the alarm interrupts are enabled setting the ALM0EN, ALM1EN, and ALM2EN bits. Note that there is not a separate interrupt enable bit for the alarms.

31.5.1. Automatic Timer Reset

The RTC timer includes an automatic reset feature that resets the timer to zero when alarm 0 triggers.

When using this auto-reset feature, the alarm match value should always be set to 2 counts less than the desired match value to account for delays. When using the low frequency oscillator in combination with auto-reset, the right-justified alarm 0 value should be set to 4 counts less than the desired match value.

The auto-reset feature can be enabled by writing a 1 to ALM0AREN and writing a 1 to ALRM0EN.

31.6. Interrupts

The RTC module has interrupts for each of the alarms and the missing clock detector. The ALM0I, ALM1I, and ALM2I flags can cause an interrupt if the corresponding alarm is enabled (ALM0EN, ALM1EN, or ALM2EN set to 1). Hardware sets the oscillator fail (OSCFI) flag to 1 when a missing clock detector event triggers, causing an interrupt.

31.7. Usage Models

The RTC timer and alarms have two operating modes to suit varying applications.

31.7.1. Usage Mode 1

The first mode uses the RTC timer as a perpetual timebase that is never reset to zero. Every 36 hours, the timer is allowed to overflow without being stopped or disrupted. Firmware manages the alarm intervals and adds the intervals to the expired value in the ALARMx registers after each alarm. This allows the alarm match value to always stay ahead of the timer by one firmware-managed interval. If firmware uses 32-bit unsigned addition to increment the alarm match values, then it does not need to handle overflows since both the timer and the alarm match values will overflow in the same manner.

This mode is ideal for applications using a long alarm interval (24 or 36 hours) or have a need for a perpetual timebase, which is useful in situations where the wake-up interval is constantly changing. For these applications, firmware can keep track of the number of timer overflows in a 16-bit variable, extending the 32-bit (36 hour) timer to a 48-bit (272 year) perpetual timebase.

31.7.2. Usage Mode 2

The second mode uses the RTC timer as a general purpose up-counter that is auto-reset to zero by hardware after each alarm 0 event. Hardware manages the alarm intervals in the ALARMx registers, and firmware only needs to set the alarm intervals once during device initialization. After each alarm 0 event, firmware should keep a count of the number of alarms that have occurred in order to keep track of time. Alarm 1 and alarm 2 events do not trigger the timer auto-reset.

This mode is ideal for applications that require minimal firmware intervention or have a fixed alarm interval. This mode is the most power-efficient since it requires less core processing time per alarm.

31.8. RTC0 Registers

This section contains the detailed register descriptions for RTC0 registers.

Register 31.1. RTC0_CONFIG: RTC Configuration

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	RTCEN	CLKSEL	RTCOEN	Reserved		ALM2EN	ALM1EN	ALM0EN	Reserved					AGCEN	CRYSEN	BDEN
Type	RW	RW	RW	R		RW	RW	RW	R					RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved								RTCLC				ASEN	MCLKEN	RUN	ALM0AREN
Type	R								RW				RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

RTC0_CONFIG = 0x4002_9000

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 31.3. RTC0_CONFIG Register Bit Descriptions

Bit	Name	Function
31	RTCEN	**RTC Timer Enable.** 0: Disable the RTC timer. 1: Enable the RTC timer.
30	CLKSEL	**RTC Timer Clock Select.** 0: Select the External Crystal or External CMOS Clock as the RTC Timer clock (RTCnTCLK) source. 1: Select the Low Frequency Oscillator as the RTC Timer clock (RTCnTCLK) source.
29	RTCOEN	**RTC0 External Output Enable.** Setting this bit to 1 allows the RTC module to drive the RTCnOSC on an external pin. 0: Disable the external RTCnOSC output. 1: Enable the external RTCnOSC output.
28:27	Reserved	Must write reset value.

Table 31.3. RTC0_CONFIG Register Bit Descriptions

Bit	Name	Function
26	ALM2EN	**Alarm 2 Enable.** 0: Disable RTC Alarm 2. 1: Enable RTC Alarm 2 and Alarm 2 Interrupt.
25	ALM1EN	**Alarm 1 Enable.** 0: Disable RTC Alarm 1. 1: Enable RTC Alarm 1 and Alarm 1 Interrupt.
24	ALM0EN	**Alarm 0 Enable.** 0: Disable RTC Alarm 0. 1: Enable RTC Alarm 0 and Alarm 0 Interrupt.
23:19	Reserved	Must write reset value.
18	AGCEN	**Automatic Gain Control Enable.** 0: Disable automatic gain control. 1: Enable automatic gain control, saving power.
17	CRYSEN	**Crystal Oscillator Enable.** 0: Disable the crystal oscillator circuitry. 1: Enable the crystal oscillator circuitry.
16	BDEN	**Bias Doubler Enable.** The bias doubler should always be enabled at startup in External Crystal Mode. When the bias doubler is disabled (BDEN=0), the RTC External Oscillator Valid Flag (CLKVF) is also disabled and will always read 0. 0: Disable the bias doubler, saving power. 1: Enable the bias doubler.
15:8	Reserved	Must write reset value.
7:4	RTCLC	**Load Capacitance Value.** This field is the load capacitance value. This field will be automatically set by hardware if automatic load capacitance stepping is enabled (ASEN = 1).
3	ASEN	**Automatic Crystal Load Capacitance Stepping Enable.** 0: Disable automatic load capacitance stepping. 1: Enable automatic load capacitance stepping.
2	MCLKEN	**Missing Clock Detector Enable.** 0: Disable the missing clock detector. 1: Enable the missing clock detector. If the missing clock detector triggers, it will generate an RTC Fail event.
1	RUN	**RTC Timer Run Control.** 0: Stop the RTC timer. 1: Start the RTC timer running.

Table 31.3. RTC0_CONFIG Register Bit Descriptions

Bit	Name	Function
0	ALM0AREN	**Alarm 0 Automatic Reset Enable.** Setting this bit to 1 will automatically reset the RTC timer when a Alarm 0 event occurs. Note that Alarm 0 must be enabled (ALM0EN=1) to use the Automatic Reset feature. 0: Disable the Alarm 0 automatic reset. 1: Enable the Alarm 0 automatic reset.

Register 31.2. RTC0_CONTROL: RTC Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved							LRDYF	HSMDEN	OSCFI	CLKVF	TMRSET	TMRCAP	ALM2I	ALM1I	ALM0I
Type	R							R	RW	RW	R	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	X	0	0	0	0	0	0	0	0

Register ALL Access Address

RTC0_CONTROL = 0x4002_9010

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 31.4. RTC0_CONTROL Register Bit Descriptions

Bit	Name	Function
31:9	Reserved	Must write reset value.
8	LRDYF	**RTC Load Capacitance Ready Flag.** This bit is set by hardware when the load capacitance matches the programmed value. 0: The load capacitance is currently stepping. 1: The load capacitance has reached its programmed value.
7	HSMDEN	**RTC High Speed Mode Enable.** This bit should be set to 1 by firmware if the AHB clock is greater than or equal to 4x the RTC Timer Clock (RTCnTCLK) frequency. 0: Disable high speed mode. (AHBCLK < 4x RTCnTCLK) 1: Enable high speed mode. (AHBCLK >= 4x RTCnTCLK)
6	OSCFI	**RTC Oscillator Fail Interrupt Flag.** This bit is set by hardware when a missing clock detector timeout occurs. This bit must be cleared by software. 0: Oscillator is running. 1: Oscillator has failed.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 31.4. RTC0_CONTROL Register Bit Descriptions

Bit	Name	Function
5	CLKVF	**RTC External Oscillator Valid Flag.** Note that the Bias Double must be enabled(BDEN=1) to use this flag. When the BDEN=0, CLKVF always reads 0. 0: External oscillator is not valid. 1: External oscillator is valid.
4	TMRSET	**RTC Timer Set.** Set this bit to 1 to initiate an RTC timer set operation, which copies the value in SETCAP to the RTC timer. The timer must be running (RUN = 1) in order to set the timer value. This bit is cleared by hardware when the transfer operation is done. 0: RTC timer set operation is complete. 1: Start the RTC timer set.
3	TMRCAP	**RTC Timer Capture.** Set this bit to 1 to initiate an RTC timer capture operation, which copies the current RTC timer value to SETCAP. This bit is cleared by hardware when the transfer operation is done. 0: RTC timer capture operation is complete. 1: Start the RTC timer capture.
2	ALM2I	**Alarm 2 Interrupt Flag.** 0: Alarm 2 event has not occurred. 1: Alarm 2 event occurred.
1	ALM1I	**Alarm 1 Interrupt Flag.** 0: Alarm 1 event has not occurred. 1: Alarm 1 event occurred.
0	ALM0I	**Alarm 0 Interrupt Flag.** 0: Alarm 0 event has not occurred. 1: Alarm 0 event occurred.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Register 31.3. RTC0_ALARM0: RTC Alarm 0

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	ALARM0[31:16]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	ALARM0[15:0]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Register ALL Access Address RTC0_ALARM0 = 0x4002_9020																

Table 31.5. RTC0_ALARM0 Register Bit Descriptions

Bit	Name	Function
31:0	ALARM0	**RTC Alarm 0.** The RTC Alarm 0 event will occur when ALARM0 matches the RTC timer value.

Register 31.4. RTC0_ALARM1: RTC Alarm 1

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	ALARM1[31:16]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	ALARM1[15:0]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address
RTC0_ALARM1 = 0x4002_9030

Table 31.6. RTC0_ALARM1 Register Bit Descriptions

Bit	Name	Function
31:0	ALARM1	**RTC Alarm 1.** The RTC Alarm 1 event will occur when ALARM1 matches the RTC timer value.

Register 31.5. RTC0_ALARM2: RTC Alarm 2

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	ALARM2[31:16]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	ALARM2[15:0]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address
RTC0_ALARM2 = 0x4002_9040

Table 31.7. RTC0_ALARM2 Register Bit Descriptions

Bit	Name	Function
31:0	ALARM2	**RTC Alarm 2.** The RTC Alarm 2 event will occur when ALARM2 matches the RTC timer value.

Register 31.6. RTC0_SETCAP: RTC Timer Set/Capture Value

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	SETCAP[31:16]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	SETCAP[15:0]															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

RTC0_SETCAP = 0x4002_9050

Table 31.8. RTC0_SETCAP Register Bit Descriptions

Bit	Name	Function
31:0	SETCAP	**RTC Timer Set/Capture Value.** The value in SETCAP will be written to the RTC timer when TMRSET is set to 1. The operation will be complete when TMRSET is cleared to 0 by the hardware. The value of the RTC timer will be copied to SETCAP when TMRCAP is set to 1. The operation will be complete when TMRCAP is cleared to 0 by the hardware.

Register 31.7. RTC0_LFOCONTROL: LFOSC Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	LFOSCEN	Reserved														
Type	RW	R														
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address
RTC0_LFOCONTROL = 0x4002_9060

Table 31.9. RTC0_LFOCONTROL Register Bit Descriptions

Bit	Name	Function
31	LFOSCEN	**Low Frequency Oscillator Enable.** 0: Disable the Low Frequency Oscillator (LFOSCn). 1: Enable the Low Frequency Oscillator (LFOSCn).
30:0	Reserved	Must write reset value.

31.9. RTC0 Register Memory Map

Table 31.10. RTC0 Memory Map

Register Name	RTC0_CONFIG	RTC0_CONTROL	RTC0_ALARM0	RTC0_ALARM1	RTC0_ALARM2
ALL Address	0x4002_9000	0x4002_9010	0x4002_9020	0x4002_9030	0x4002_9040
Access Methods	ALL | SET | CLR	ALL | SET | CLR	ALL	ALL	ALL
Bit 31	RTCEN	Reserved	ALARM0	ALARM1	ALARM2
Bit 30	CLKSEL				
Bit 29	RTCOEN				
Bit 28	Reserved				
Bit 27					
Bit 26	ALM2EN				
Bit 25	ALM1EN				
Bit 24	ALM0EN				
Bit 23	Reserved				
Bit 22					
Bit 21					
Bit 20					
Bit 19					
Bit 18	AGCEN				
Bit 17	CRYSEN				
Bit 16	BDEN				
Bit 15	Reserved				
Bit 14					
Bit 13					
Bit 12					
Bit 11					
Bit 10					
Bit 9					
Bit 8		LRDYF			
Bit 7	RTCLC	HSMDEN			
Bit 6		OSCFI			
Bit 5		CLKVF			
Bit 4		TMRSET			
Bit 3	ASEN	TMRCAP			
Bit 2	MCLKEN	ALM2I			
Bit 1	RUN	ALM1I			
Bit 0	ALM0AREN	ALM0I			

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

Table 31.10. RTC0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RTC0_SETCAP	0x4002_9050	ALL	SETCAP																															
RTC0_LFOCONTROL	0x4002_9060	ALL	LFOSCEN	Reserved																														

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

32. SAR Analog-to-Digital Converter (SARADC0 and SARADC1)

This section describes the SAR Analog to Digital Converter (SARADC) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the SARADC block, which is used by both SARADC0 and SARADC1 on all device families covered in this document.

32.1. SARADC Features

The SARADC module includes the following features:

- Single-ended 10-bit or 12-bit operation.
- Operation in low power modes at lower conversion speeds.
- Can be synchronized to the EPCA0 synchronization output, to take samples at precise times in the PWM waveform.
- Selectable asynchronous hardware conversion trigger with hardware channel select.
- Output data window comparator allows automatic range checking.
- Support for Burst Mode, which produces one set of accumulated data per conversion-start trigger with programmable power-on settling and tracking time.
- Conversion complete, multiple conversion complete, and FIFO overflow and underflow flags and interrupts supported.
- Flexible output data formatting.
- Sequencer allows up to 8 sources to be automatically scanned using one of four channel characteristic profiles without software intervention.
- Eight-word conversion data FIFO for DMA operations.
- SARADC0 and SARADC1 can work together synchronously or by interleaving samples.

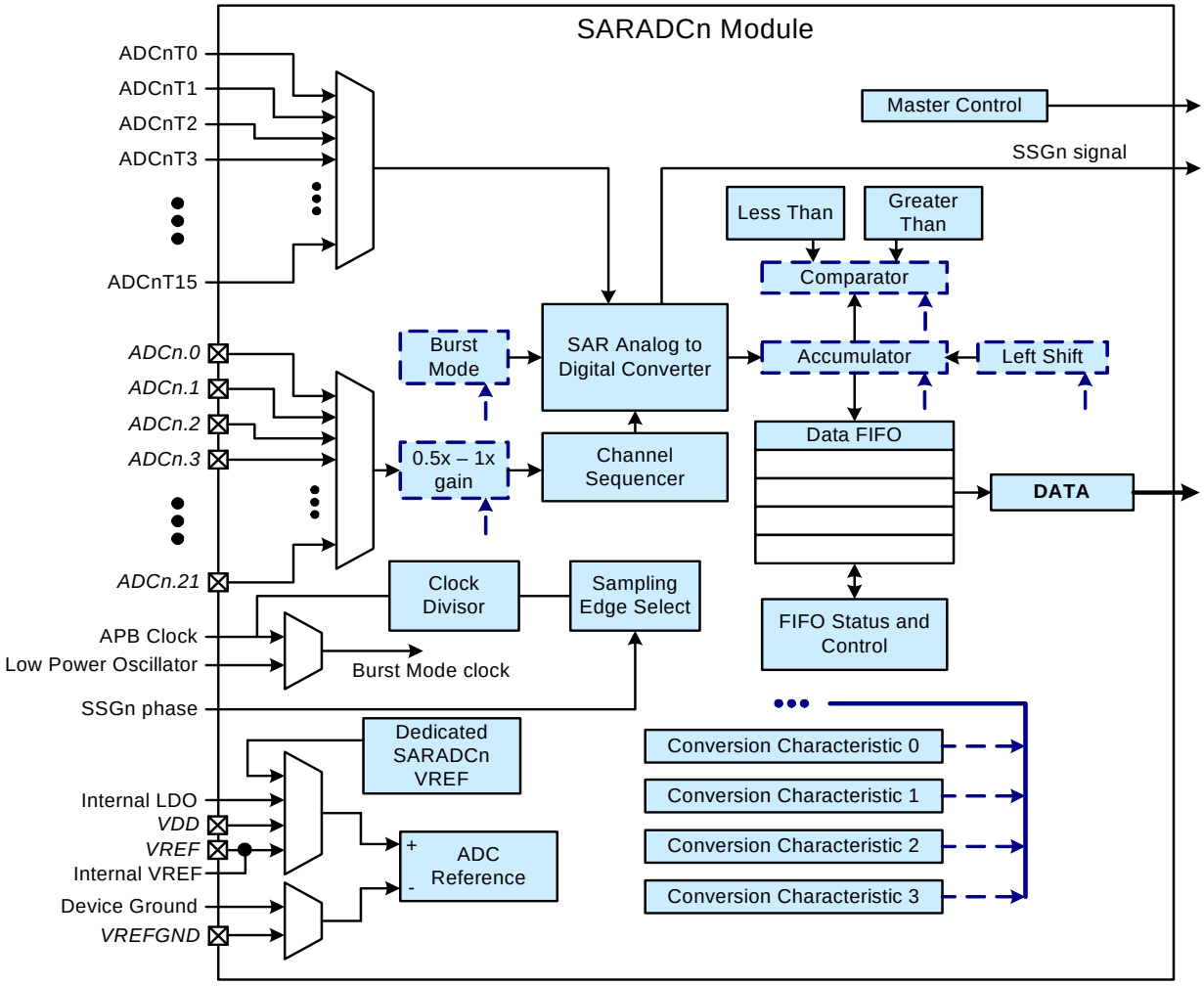


Figure 32.1. SARADC Block Diagram

32.2. Tracking and Conversion Time

A single ADC conversion consists of two phases; the tracking phase, and the conversion phase. During the tracking phase, the ADC's sampling capacitor connects to the selected multiplexer channel and charges to the voltage present at that node. During the conversion phase, the sampling capacitor disconnects from the multiplexer channel and connects to the SAR converter circuitry.

32.2.1. Input Settling Time

It is important for the application to allow enough settling time at the ADC input during the tracking phase. This will depend largely on the external source impedance and the desired accuracy level. The input to the SARADC is shown in Figure 32.2. The sample switch is closed during the tracking phase and open during the conversion phase. Values for C_{IN}, C_{SAR}, and R_{MUX} are different depending on the type of input channel and the gain range. These values can be found in the device data sheet electrical specifications tables. The system designer should assume that the capacitor C_{SAR} is discharged between every conversion.

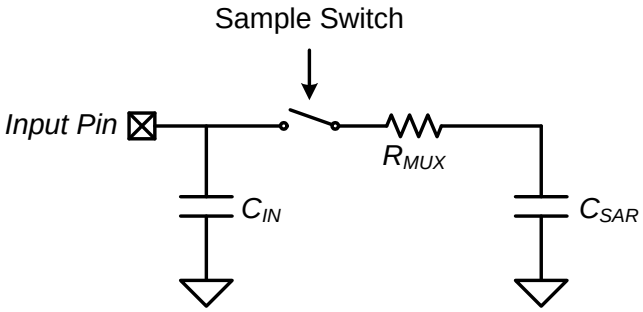


Figure 32.2. SARADC Input Model

32.2.2. SAR Clock Generation

The SAR clock speed dictates the timing of the conversion phase, which lasts for 13 SAR clock cycles. The SAR clock speed is programmable as a divided version of the APB clock using the CLKDIV field in the CONFIG register. The SAR clock should be configured to be as fast as possible according to the electrical specifications in the device data sheet. Faster SAR clock speeds allow the converter more time in the tracking phase and reduce the amount of time the converter is actively converting, thereby reducing system power.

32.2.3. Tracking Mode (Non-Burst Operation)

When the ADC operates in non-burst mode, two tracking options are available and are selected by the TRKMD bit in the CONTROL register: normal tracking and delayed tracking. In normal tracking mode, the ADC tracks any time a conversion is not taking place, and the start-of-conversion event triggers the beginning of the conversion phase. The ADC returns to tracking immediately after a conversion finishes. The tracking time in this mode is therefore determined by the sample rate minus the conversion time.

In delayed tracking mode, the start-of-conversion event will trigger the ADC to track for three SAR clock cycles, followed by the conversion phase. Upon completion of a conversion, the ADC will go into an idle state, waiting for the next start-of-conversion trigger. Timing for the two tracking modes is shown in Figure 32.3.

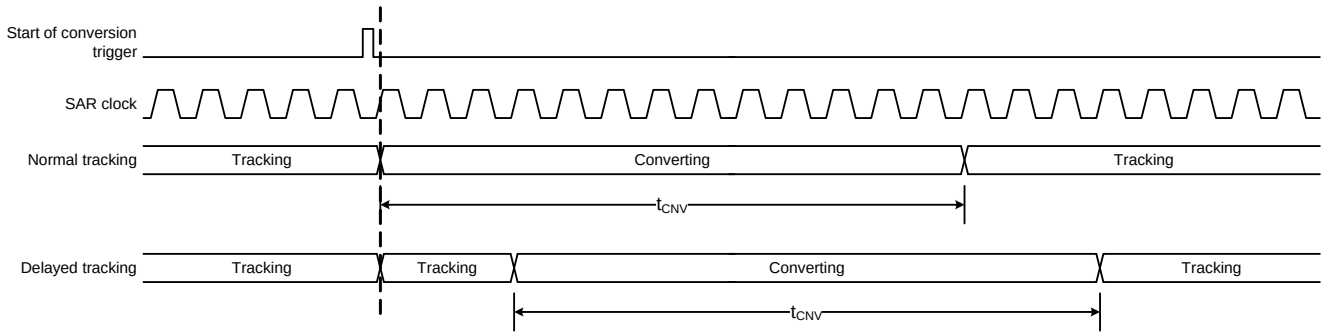


Figure 32.3. Non-Burst Tracking and Conversion Timing

32.2.4. Start-of-Conversion Source

Conversions can be initiated by several different internal and external trigger sources, which vary between device families. The SCSEL field in the CONTROL register selects the start-of-conversion source to be used by the ADC. In "on-demand" trigger mode, conversions are initiated when firmware sets the ADBUSY bit in the CONTROL register to 1. In all other conversion modes, the selected start-of-conversion trigger (timer overflow, external pin transition, SSG, etc.) will begin a conversion. The frequency of the selected start-of-conversion trigger determines the sampling rate of the ADC and should not exceed the maximum listed in the electrical specification for the device. SARADC0 and SARADC1 start of conversion sources vary by package, and are shown in Table 32.1 and Table 32.2.

Table 32.1. SARADC0 Start of Conversion Sources

Trigger	External Convert Start Description	SiM3U1x7/C1x7 Pin Name	SiM3U1x6/C1x6 Pin Name	SiM3U1x4/C1x4 Pin Name
ADC0T0	Internal Convert Start	“On Demand” by writing 1 to ADBUSY		
ADC0T1	Internal Convert Start	Timer 0 Low overflow		
ADC0T2	Internal Convert Start	Timer 0 High overflow		
ADC0T3	Internal Convert Start	Timer 1 Low overflow		
ADC0T4	Internal Convert Start	Timer 1 High overflow		
ADC0T5	Internal Convert Start	EPCA0 synchronization pulse		
ADC0T6	Internal Convert Start	I2C0 Timer overflow		
ADC0T7	Internal Convert Start	I2C1 Timer overflow		
ADC0T8	Internal Convert Start	SSG phase defined by ADSP bits (cannot be used in Burst Mode).		
ADC0T15	External Convert Start	PB1.13	PB1.6	PB0.12

Table 32.2. SARADC1 Start of Conversion Sources

Trigger	External Convert Start Description	SiM3U1x7/C1x7 Pin Name	SiM3U1x6/C1x6 Pin Name	SiM3U1x4/C1x4 Pin Name
ADC1T0	Internal Convert Start	“On Demand” by writing 1 to ADBUSY		
ADC1T1	Internal Convert Start	Timer 0 Low overflow		
ADC1T2	Internal Convert Start	Timer 0 High overflow		
ADC1T3	Internal Convert Start	Timer 1 Low overflow		
ADC1T4	Internal Convert Start	Timer 1 High overflow		
ADC1T5	Internal Convert Start	EPCA0 synchronization pulse		
ADC1T6	Internal Convert Start	I2C0 Timer overflow		
ADC1T7	Internal Convert Start	I2C1 Timer overflow		
ADC1T8	Internal Convert Start	SSG phase defined by ADSP bits (cannot be used in Burst Mode).		
ADC1T15	External Convert Start	PB1.14	PB1.7	PB0.13

32.2.5. 12-bit Mode

The ADC normally operates as a 10-bit converter, and achieves its highest sampling rate in the 10-bit mode. A 12-bit mode is available, which increases the resolution of the converter to 12 bits at the expense of conversion speed.

This 12-bit mode is a special condition of burst mode operation. When operating the converter in 12-bit mode, the BURSTEN bit in the CONTROL register must be set to 1. One input sample and four conversion cycles are required per 12-bit conversion word, and the set of four conversions will be initiated by the start-of-conversion trigger. The converter uses a patented technique to increase the resolution and linearity of the converter by two bits using only four conversion cycles, whereas a traditional straight average operation would require 16 samples to increase noise resolution by 2 bits, and would have no effect on linearity. Additionally, when used to sample DC input signals, the converter can be configured to resample the input four times per 12-bit conversion, which can provide additional filtering of Gaussian noise present at the input. The AD12BSSEL field on the CONTROL register is used to configure whether four separate input samples or a single input sample is used for the 12-bit result. Figure 32.4 shows the difference in timing between these two options.

Note: When using single-sampling (AD12BSSEL = 1), the TRKMD bit should be cleared to 0 to ensure proper signal tracking.

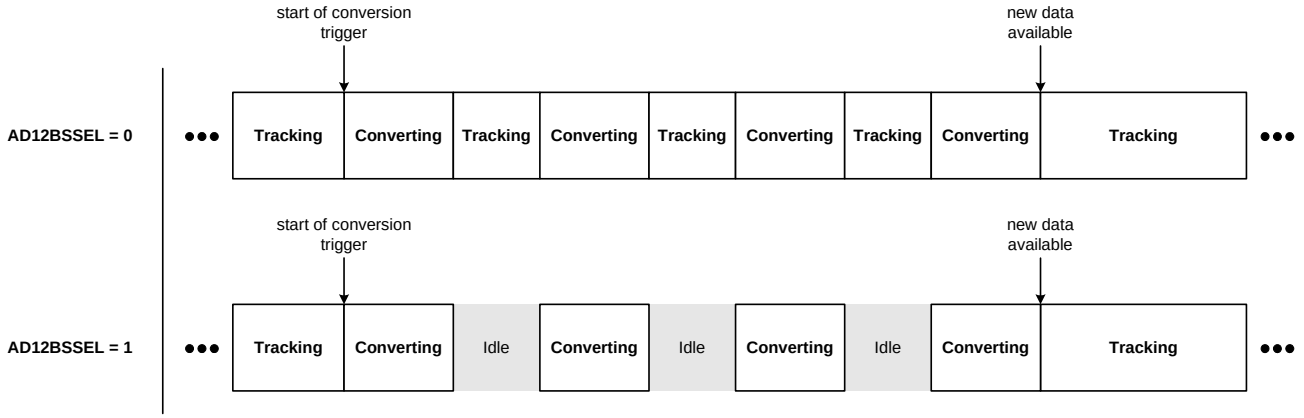


Figure 32.4. 12-Bit Sampling Options

32.3. Burst Mode

The ADC implements a "burst mode" feature which enables lower power operation of the system. When a conversion trigger event occurs, the ADC will power on (if needed), track for a selected period of time, perform one or more conversions, and then return to the idle or powered-down state. The repeat counter (CHRxRPT) dictates the number of conversions performed for the channel being converted. Burst mode is enabled by setting the BURSTEN bit in the CONTROL register to 1.

In burst mode, the ADCEN bit controls the power consumption of the ADC between conversions. When ADCEN is cleared to 0, the ADC is powered down after each burst. If ADCEN is set to 1, the ADC will remain powered on between bursts.

In burst mode, the ADC can use a high-speed, low-power oscillator for conversion timing, enabling the system designer to run the core from a slow clock source or put the core in a low-power state to conserve power. Burst mode can also be configured to use the APB clock as a source. The clock used for burst mode is selected using the BCLKSEL bit in the CONFIG register.

32.3.1. Data Accumulation

When burst mode is enabled, ADC samples can automatically be accumulated by the converter as they are taken. The accumulation mode is controlled by the ACCMD bit in the CONTROL register. This bit should be cleared to 0 to enable accumulation. Firmware must write the initial value (typically zero) to the ACC register prior to a conversion being taken or a scan sequence being initiated. The number of conversions accumulated is determined by the repeat counter field of the selected conversion characteristic register. For example, if CHxRPT is set to sample four times, four conversions will be accumulated.

When using the ADC's scan function, the accumulator will automatically be cleared to zero before the sequencer moves to the next selected channel. In other modes, this does not occur, and firmware must clear the ACC register, as necessary. Note that the ACC register should not be written while conversions are in progress, and its contents cannot be read.

32.3.2. Burst Mode Tracking

Tracking time in burst mode is different than non-burst mode operation of the ADC. For the first conversion, the tracking time is determined by the power-on time selected by the PWRTIME field in the CONTROL register. For all subsequent conversions during an ADC burst, the tracking time is dictated by the setting of the BMTK and the TRKMD bits. This timing is described in the BMTK bit description in the CONTROL register. Figure 32.5 illustrates the burst mode tracking timing when ADCEN is cleared to 0 (power down between conversions), the burst clock is the low power oscillator, and the APB clock is operating at a low frequency.

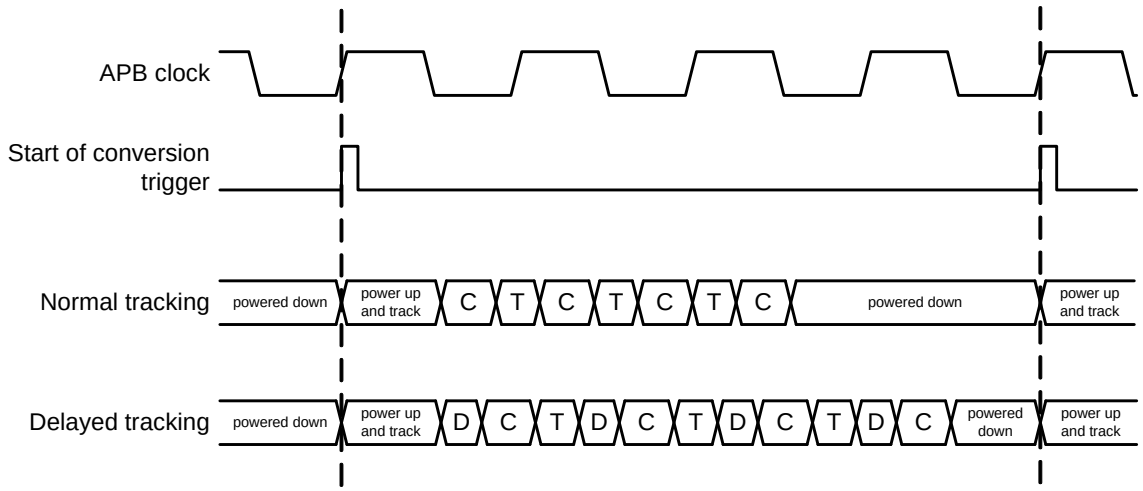


Figure 32.5. Burst Mode Tracking and Conversion Timing (Four Samples)

32.4. Channel Sequencer

The channel sequencer allows the user to define up to eight different combinations of ADC configurations and multiplexer channels, then scan through them with an ADC scan operation, relieving software of the overhead necessary to change multiplexer channels and other parameters.

32.4.1. Multiplexer Input Settings

Each ADC module (SARADC0, SARADC1) can select between multiple external inputs (up to 16 are available, depending on the package type) and internal inputs. These inputs become the positive inputs to the single-ended SARADC0 or SARADC1 module. The possible input selections for SARADC0 and SARADC1 are shown in Table 32.3 and Table 32.4. Note that for some selections, other device circuitry must be enabled.

32.4.2. Timeslot Settings

The Channel Sequencer Time Slot Setup registers SQ7654 and SQ3210 configure the eight channel sequencer time slots each ADC. Each time slot defines the multiplexer channel to be converted, as well as which conversion characteristic to use for the conversion.

If the TSnMUX field in a time slot is set to the terminate scan value (0x1F), this indicates that no conversion is to be performed on this channel. When the sequencer encounters an 0x1F value in the time slot mux selection or converts the final time slot, it will either halt (single scan mode) or wrap back to time slot 0 (continuous scan mode).

The scan done interrupt flag (SDI in the STATUS register) will be set to 1 when a single scan operation is complete or when firmware clears SCANEN. Figure 32.6 shows a typical setup for a single scan using three sequencer time slots.

32.4.3. Conversion Characteristic Settings

The Conversion Characteristic Setup registers CHAR10 and CHAR32 define some of the characteristics of how the SARADC conversions will be performed. Each register defines two conversion characteristics (for example, CHAR10 defines conversion characteristic 1 and 0). The conversion characteristics select the gain, accumulation levels, post-conversion shifting, number of data bits, and whether to use the window comparator hardware. See the CHAR10 and CHAR32 register descriptions for more details on the selectable options.

32.4.4. Channel Scan Mode

The ADC implements channel sequencer logic which is capable of "scanning" through one or more ADC configurations automatically. When SCANEN is set to 1, the channel sequencer is active. The sequencer begins with time slot 0, and continues scanning through all eight time slots in sequence until it reaches the last channel or the final time slot or the terminate scan value (0x1F).

32.4.4.1. Single Scan Mode

If the SCANMD field in the CONFIG register is set to 0, the ADC performs a single scan through the channels. When using single-scan mode, each scan must be initiated by a 0-to-1 transition of the SCANEN bit in the CONFIG register. The scan will begin on the next conversion trigger event and end when the final channel in the sequencer has been converted. Note that a conversion trigger event is required for each channel in the scan, and the length of time between each conversion trigger event needs to be greater than or equal to the time required for the longest conversion. SCANEN will return to 0 upon completion of a scan in the single scan mode.

32.4.4.2. Continuous Scan Mode

If the SCANMD field in the CONFIG register is set to 1, the ADC will continue to wrap through all channels of the sequencer indefinitely. The continuous scan operation must be initiated by a 0-to-1 transition of the SCANEN bit in the CONFIG register. When the channel sequencer reaches the end of the sequence or the terminate scan value (0x1F), the ADC will begin again with the first channel. The scans will continue until firmware clears the SCANEN bit to 0.

32.4.4.3. Important notes on the use of Scan Mode

Note the following important restrictions on the use of Scan Mode:

1. Scan should not be used if interleaved or simultaneous modes are enabled.
2. Burst mode should be enabled (BURSTEN = 1) if using scan.
3. A start-of-conversion trigger is required for each time slot in the sequence. Each trigger needs to be

spaced farther apart in time than the amount of time required for the longest conversion in the sequence. For example, if one time slot uses a 64-sample accumulation, the length of time between each conversion needs to be longer than the time required for one 64-sample accumulation.

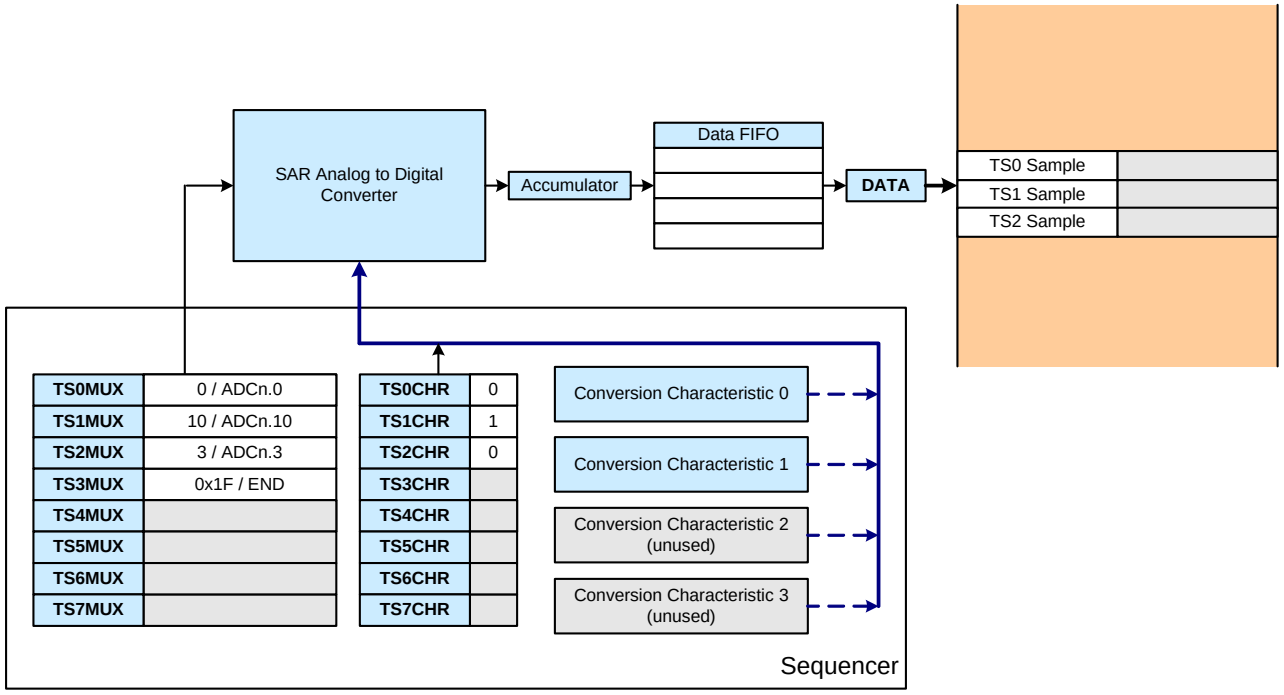


Figure 32.6. Channel Scan Setup and Timing

32.4.5. Single Channel Mode

When SCANEN is cleared to 0, the channel sequencer is not active. In this mode, time slot 0 is used for all conversions performed by the ADC. Any of the conversion characteristic setup registers can be used to define the conversion parameters in single configuration mode. The single conversion complete interrupt flag (SCCI in the STATUS register) will be set to 1 after each conversion is complete.

Setting the TSnMUX to 0x1F will terminate the scan at that time slot.

Table 32.3. SARADC0 Input Channels

SARADC0 Input	SARADC0 Input Description	SiM3U1x7/C1x7 Pin Name	SiM3U1x6/C1x6 Pin Name	SiM3U1x4/C1x4 Pin Name
ADC0.0	Normal Input	PB0.0	Reserved	PB0.6
ADC0.1	Normal Input	PB0.1	Reserved	PB0.7
ADC0.2	Normal Input	PB0.2	PB0.0	Reserved
ADC0.3	Normal Input	PB0.3	PB0.1	Reserved
ADC0.4	Normal Input	PB0.4	PB0.2	Reserved
ADC0.5	Normal Input	PB0.5	PB0.3	Reserved
ADC0.6	Normal Input	PB0.7	PB0.4	Reserved

Table 32.3. SARADC0 Input Channels (Continued)

SARADC0 Input	SARADC0 Input Description	SiM3U1x7/C1x7 Pin Name	SiM3U1x6/C1x6 Pin Name	SiM3U1x4/C1x4 Pin Name
ADC0.7	Normal Input	PB0.8	PB0.5	Reserved
ADC0.8	Normal Input	PB0.9	PB0.6	PB0.0
ADC0.9	Normal Input	PB0.11	PB0.8	PB0.2
ADC0.10	Normal Input	PB0.12	PB0.9	PB0.3
ADC0.11	Normal Input	PB1.1	Reserved	Reserved
ADC0.12	High Quality Input	PB1.3	PB0.14	Reserved
ADC0.13	High Quality Input	PB1.4	PB0.15	Reserved
ADC0.14	High Quality Input	PB1.5	PB1.0	PB0.8
ADC0.15	High Quality Input	PB1.6	PB1.1	PB0.9
ADC0.16	Internal Channel	IVC0.0 Output (IVC0C0)		
ADC0.17	Internal Channel	VSS		
ADC0.18	Internal Channel	1.8 V Output of LDO		
ADC0.19	Internal Channel	VDD		
ADC0.20	Internal Channel	Temperature Sensor Output		
ADC0.21	Internal Channel	Voltage at VIOHD / 4 (1)		

Notes:

1. The VIOHD/4 option requires the VIOHD divider to be enabled within the PBHD block (PBHD4_PBDRV.PBVTRKEN).

Table 32.4. SARADC1 Input Channels

SARADC1 Input	SARADC1 Input Description	SiM3U1x7/C1x7 Pin Name	SiM3U1x6/C1x6 Pin Name	SiM3U1x4/C1x4 Pin Name
ADC1.0	Normal Input	PB2.2	PB1.11	PB1.1
ADC1.1	Normal Input	PB2.1	PB1.10	PB1.0
ADC1.2	Normal Input	PB2.0	PB1.9	PB0.15
ADC1.3	Normal Input	PB1.15	PB1.8	PB0.14
ADC1.4	Normal Input	PB1.14	PB1.7	PB0.13
ADC1.5	Normal Input	PB1.13	PB1.6	PB0.12
ADC1.6	Normal Input	PB1.12	PB0.10	PB0.4
ADC1.7	Normal Input	PB1.11	PB1.5	PB0.11

Table 32.4. SARADC1 Input Channels

SARADC1 Input	SARADC1 Input Description	SiM3U1x7/C1x7 Pin Name	SiM3U1x6/C1x6 Pin Name	SiM3U1x4/C1x4 Pin Name
ADC1.8	Normal Input	PB1.10	PB1.4	PB0.10
ADC1.9	Normal Input	PB1.9	Reserved	Reserved
ADC1.10	Normal Input	PB1.8	PB1.3	Reserved
ADC1.11	Normal Input	PB1.7	PB1.2	Reserved
ADC1.12	High Quality Input	PB1.3	PB0.14	Reserved
ADC1.13	High Quality Input	PB1.4	PB0.15	Reserved
ADC1.14	High Quality Input	PB1.5	PB1.0	PB0.8
ADC1.15	High Quality Input	PB1.6	PB1.1	PB0.9
ADC1.16	Internal Channel	IVC0.1 Output (IVC0C1)		
ADC1.17	Internal Channel	Voltage at VREGIN / 4 (1)		
ADC1.18	Internal Channel	EXTVREG0 Current Sense (2)		
ADC1.19	Internal Channel	VIO		
ADC1.20	Internal Channel	Temperature Sensor Output		
ADC1.21	Internal Channel	Voltage at VIOHD / 4 (3)		

Notes:

1. The VREGIN/4 option requires the VREGIN sense circuitry to be enabled within the VREG0 block (VREG0_CONTROL.SENSEEN).
2. The EXTVREG0 Current Sense option requires the current sense circuit in the EXTVREG0 block to be enabled (EXTVREG0_CSCONTROL.ADCISNSEN).
3. The VIOHD/4 option requires the VIOHD divider to be enabled within the PBHD block (PBHD4_PBDRV.PBVTRKEN).

32.5. Sample Sync Generator

The Sample Sync Generator (SSG) module can be used to synchronize sampling of multiple ADC modules with other external and internal events, as well as IDAC conversions. The SSG module synchronizes everything, including its own output signals, with the SAR clock of an ADC. Note that when using the SSG module, SARADC0 will always be the master ADC and SARADC1 will always be the slave ADC.

32.5.1. Sampling Phase Selection

When using the SSG, the ADC sampling rate will be 1/16th the SAR clock frequency, and the ADC must be configured for 10-bit operation. The ADC can sample on any of sixteen different phases within the conversion time. These phases are spaced according to the speed of the ADC's SAR clock. By default, the ADC will sample at phase 0 of the SSG module. To sample on a different phase, firmware should set the SPEN bit in the CONFIG to 1 and select the desired phase using the SPSEL field. Figure 32.7 shows the relationship between the SAR clock, and the SSG phases.

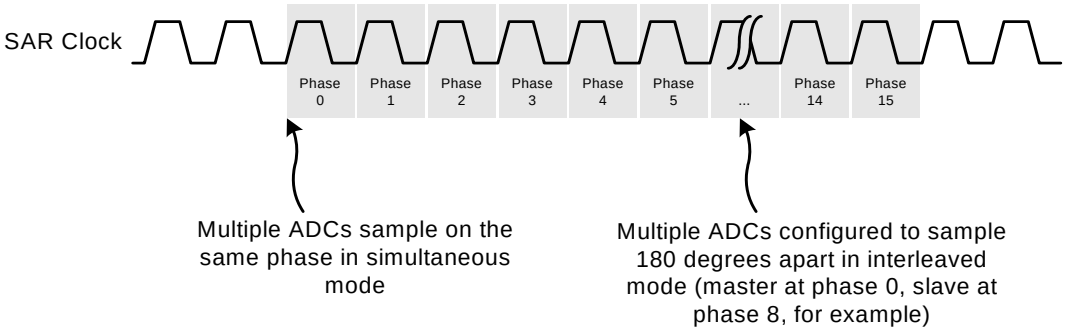


Figure 32.7. ADC Sample Timing And SSG Phase Relationship

32.5.2. Dual ADC Operation

On device families with more than one ADC module, the SSG can be used to synchronize the operation of multiple ADCs. This can be useful for sampling two different channels simultaneously (simultaneous mode) or for interleaving two ADCs on the same channel to obtain a faster throughput (interleaved mode). To synchronize two ADCs, the SSG module must be enabled and configured. Note that scan should not be used if interleaved or simultaneous modes are enabled.

When using the SSG module, SARADC0 will always be the master and SARADC1 will always be the slave. The SPSEL field in the CONFIG register of both the master and slave is used to select the desired sampling phase, as detailed in “32.5.1. Sampling Phase Selection” . Both ADCs should have the SSG module selected as their start-of-conversion triggers. On both master and slave ADCs, the SSGEN bit in the CONFIG register should be set to enable generation of conversion triggers from the SSG module. Note that when enabling the ADCs, the SSGEN bit for the master must be set first, followed by the slave ADC’s SSGEN bit. When disabling the ADCs, the reverse is true: the slave ADC’s SSGEN must be cleared first, followed by the master’s SSGEN bit.

32.5.2.1. Simultaneous Mode

To operate two ADCs simultaneously, their respective SPSEL fields should be set to the same value. This configures the ADCs to sample at the same time. If it is necessary that the ADC data from both ADCs be packed into a single 32-bit data word, the SIMCEN bit in the CONFIG register should also be set to 1 on the master ADC, and the desired sample order should be configured using the PACKMD field. Only the options for two samples per 32-bit word are valid if SIMCEN is set, and the master ADC's configuration for PACKMD determines the order (the slave ADC’s PACKMD setting doesn’t matter). If SIMCEN is cleared to 0 on the master ADC, samples from the slave will not be packed into the same data word as the master samples and will be available in the slave ADC FIFO.

32.5.2.2. Interleaved Mode

To operate two ADCs in interleaved mode, their respective SPSEL fields are most commonly set to values 8 phases apart. This configures the ADCs to sample at regular intervals between one another. If it is necessary that the ADC data from both ADCs be packed into a single 32-bit data word, the INTLVEN bit in the CONFIG register should also be set to 1 on the master ADC, and the desired sample order should be configured using the master ADC's PACKMD field (the slave's PACKMD setting doesn't matter). Any selection of PACKMD is valid when INTLVEN is set; the order in which samples are taken will determine the order they appear in the FIFO. If INTLVEN is cleared to 0 on the master ADC, samples from the slave will not be packed into the same data word as the master samples and will be available in the slave ADC FIFO.

32.5.2.3. Important notes on the use of the SSG

Note the following important restrictions on the use of the SSG:

1. Scan should be disabled if interleaved or simultaneous modes are enabled.
2. SARADC0 will always be the master and SARADC1 will always be the slave.
3. 12-bit conversions are not supported.
4. When enabling SSG mode, SSGEN must be set first on the Master ADC, then can be set on the Slave ADC. The reverse is true when disabling SSG mode.

32.5.2.4. Dual ADC + SSG Configuration Sequence

The following steps show the configuration of dual SARADCs with the SSG. If using only a single ADC with the SSG, the slave ADC configuration steps may be omitted.

1. Configure all clock source, clock divider and voltage reference settings for each ADC.
2. Clear the scan mode enable (SCANEN) bit to disable scan mode for each ADC.
3. Set the Sampling Phase Enable (SPEN) bit for each ADC.
4. Configure the desired sampling phase in the Sample Phase Select (SPSEL) for each ADC.
5. Set the start of conversion source (SCEL) in each ADC to use the SSG.
6. If desired, configure the master ADC's output packing mode (PACKMD) bit field to enable packing data from the slave ADCs into the master ADC data register.
7. For each ADC, configure the Conversion Characteristic 0 Repeat Counter (CHR0PFT) to 0 to accumulate a single sample.
8. For each ADC, set the Conversion Characteristic 0 Resolution Selection (CHR0RSEL) bit to 0 to select 10-bit mode.
9. If using Interleaved mode, set the Interleaved Conversion Packing Enable (INTLVEN) bit on the master ADC, and clear the INTLVEN bit on the slave ADC.
10. Alternately, if using Simultaneous mode, set the Simultaneous Conversion Packing Enable (SIMCEN) bit on the master ADC, and clear the SIMCEN bit on the slave ADC
11. Set the Synchronous Sample Generator Enable (SSGEN) bit for the master ADC. Note that the slave ADC's SSGEN bit must be set after the master's SSGEN bit is set.
12. Set the Synchronous Sample Generator Enable (SSGEN) bit for the slave ADC.
13. If desired, configure and enable DMA for the ADC. If the slave ADC's data is packed into the master's ADC's data register, DMA only needs to be enabled for the master ADC.
14. Configure and enable the SSG module as described in the SSGn chapter.

32.6. Voltage Reference Configuration

The ADC has the option to use several voltage reference sources: the on-chip VREF module, an external voltage reference, a dedicated internal reference for the SARADC block, an internal 1.8V LDO regulator, or the VDD voltage supply. The VREFSEL field in the CONTROL register selects the voltage reference for the ADC. The dedicated SARADC reference will be automatically enabled if it is selected. Optionally, when using the external VREF pin as the reference source, the reference ground is selectable between an internal ground node tied to VSS

and the external VREFGND pin. The REFGNDSEL bit in the CONTROL register determines which option is used. When REFGNDSEL is set to use VREFGND and an external input is being measured, the VREFGND pin signal is also used as the ADC's signal ground reference. Using the external VREFGND can provide for cleaner ADC conversions with an external reference source. The various VREF configuration options are shown in Figure 32.8.

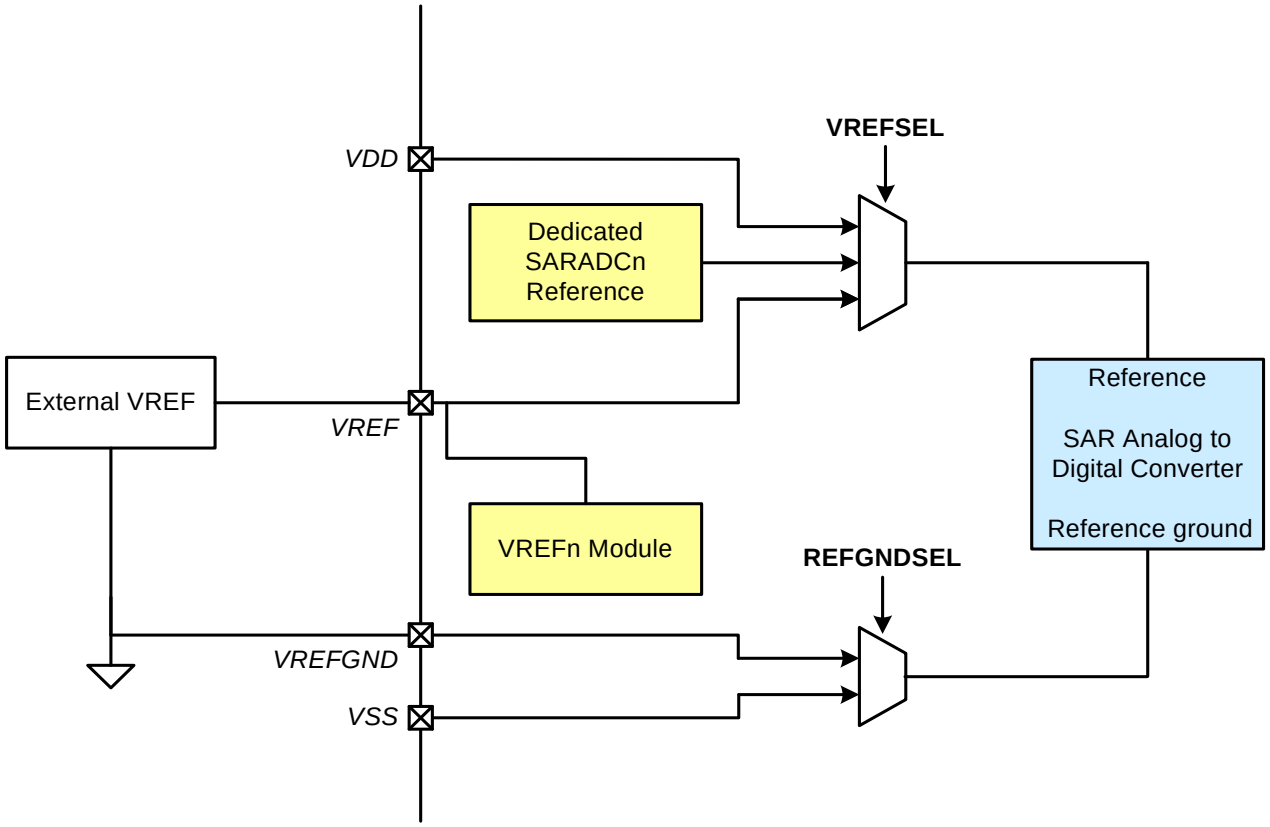


Figure 32.8. Voltage Reference Options

32.7. Power Configuration

When the ADC is disabled, it will remain in a powered-down, inactive state. The ADC is enabled by setting the ADCEN bit in the CONTROL register to 1, and there are several fields in the CONTROL register to reduce operational power when the ADC is enabled. The MREFLPEN bit can be used to reduce the power to the internal buffers when the SAR clock is operated at a slower speed. Additionally, the BIASSEL field has four selectable power levels that can scale power in regards to the SAR clock speed. The LPMDEN bit is used to reduce the current required during the tracking phase. If the application allows for longer tracking times, LPMDEN can be set to 1.

32.8. Data Output

The ADC allows for several different configurations and post-processing options on the output data. In the basic configuration, individual samples are written into a FIFO at the end of each conversion, and can be read through the DATA register or transferred to RAM using the DMA. The FIFOLVL field in the FIFOSTATUS register indicates the number of 32-bit words currently available in the FIFO. Each data word may contain one or two ADC samples. The order and number of the data words is selected by the PACKMD field in the CONFIG register. When the MSB of PACKMD is 0, only one sample is written per word, and the LSB of the PACKMD field determines whether the sample is written to the upper half or lower half of the 32-bit word. When the MSB of PACKMD is 1, two samples are packed into each output word. The order in which the samples are packed is determined by the LSB of PACKMD. The DPSTS bit in the FIFOSTATUS register indicates the target for the next sample within a data word (upper half or lower half). Figure 32.9 and Figure 32.10 show the data packing options.

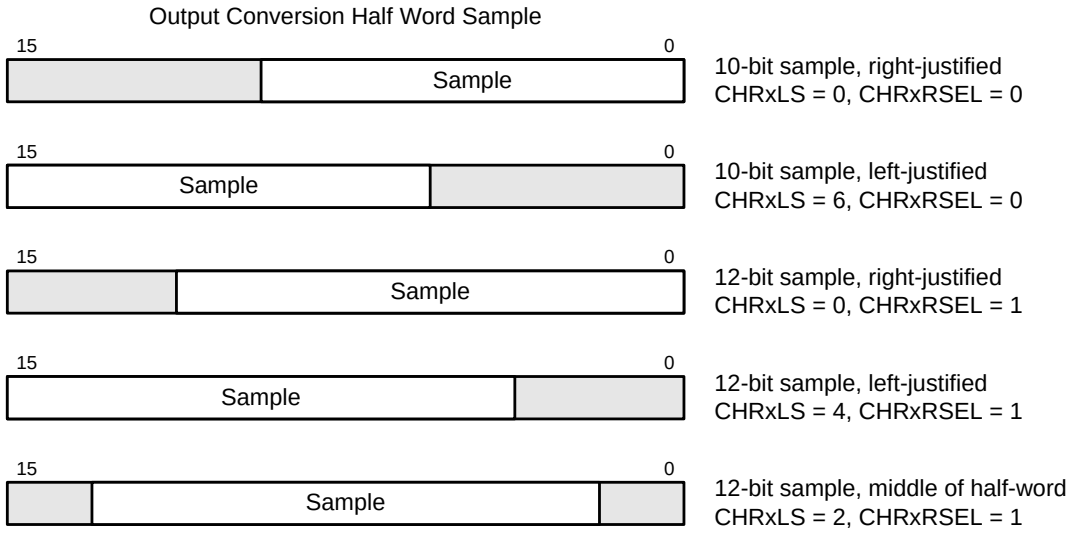


Figure 32.9. Sample Formatting

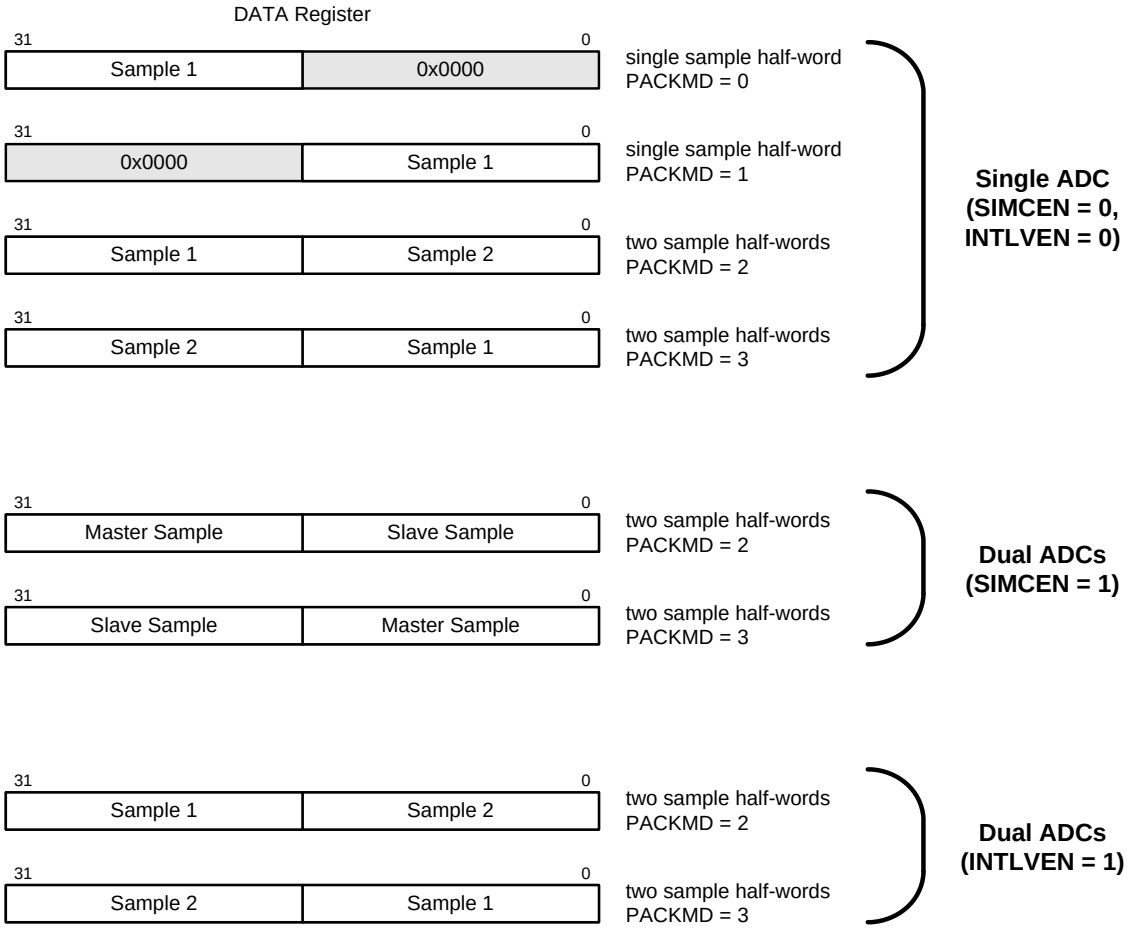


Figure 32.10. ADC Output Data Packing Options

32.8.1. Output Data Window Comparator

The ADC includes an output data window comparator. Using the window comparator, the ADC output data can be automatically compared against a specified upper and lower limit and trigger an interrupt, when desired. The window comparator limits are set by the WCLIMITS register. The WCGT field in WCLIMITS sets a "greater than" comparison limit, and the WCLT field sets a "less than" comparison limit. These two limit values are always compared against a right-justified output after any accumulation has been performed on the data. The window comparator is enabled for individual channel characteristics as detailed in “32.4. Channel Sequencer” .

The window comparator limits work together to determine when an interrupt will be triggered. Firmware can configure the ADC to generate an interrupt when the output is within the two limits or outside of the limits. To generate an interrupt within the two limits, WCLT should be programmed to a higher value than WCGT. To generate an interrupt outside of the two limits, WCGT should be programmed to a higher value than WCLT. Figure 32.11, Figure 32.12, Figure 32.13, and Figure 32.14 show examples of configuring the window comparator limits for different situations.

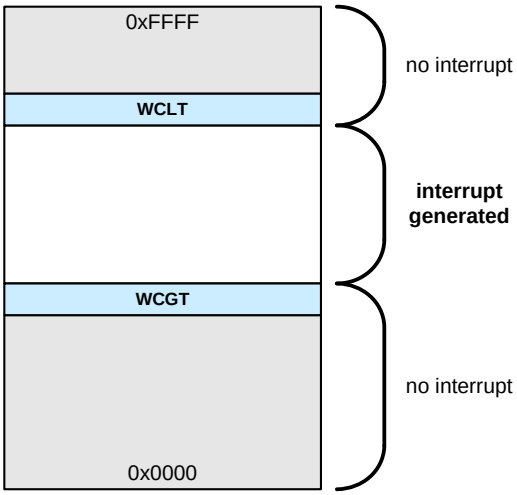


Figure 32.11. Example Window Comparator Limits for Inside Range (WCGT < WCLT)

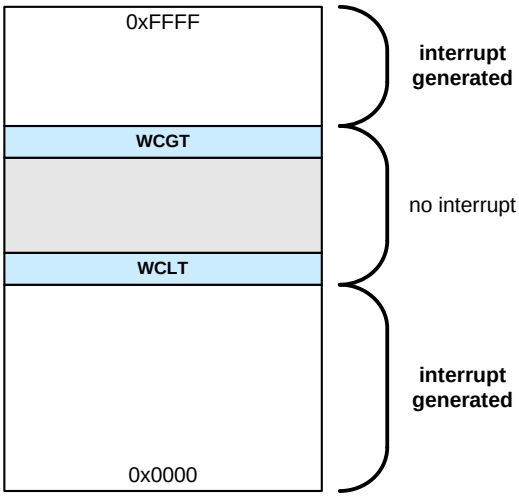


Figure 32.12. Example Window Comparator Limits for Outside Range (WCGT > WCLT)

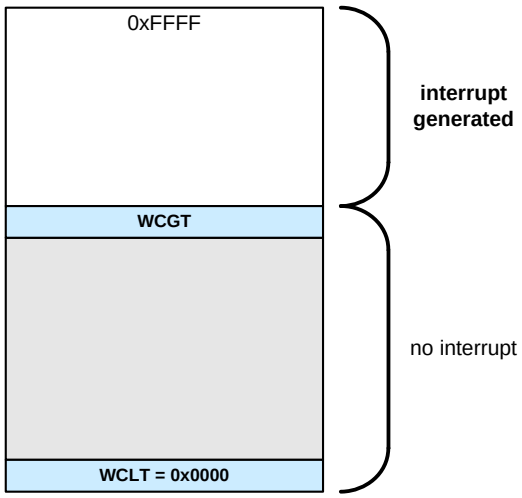


Figure 32.13. Example Window Comparator Limits for Above Value (WCGT = Value, WCLT = 0)

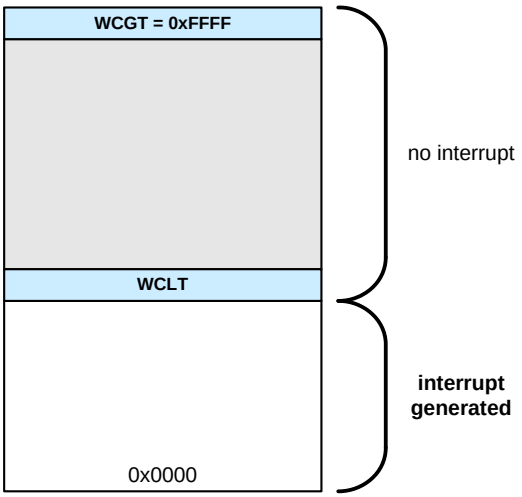


Figure 32.14. Example Window Comparator Limits for Below Value (WCLT = Value, WCGT = Full Scale)

32.9. Interrupts

The ADC interrupt can be triggered by five different, independently maskable interrupt sources. Two of these interrupts indicate error conditions, while the other three are primarily used for non-DMA operation. All interrupt status flags are located in the STATUS register and must be cleared by software. Descriptions of each interrupt condition are below:

- FURI: The FIFO underrun interrupt flag is set when a read from the DATA register is initiated while the FIFO is empty (FIFOLVL = 0). The read of the DATA register in this event will return the previous ADC result.
- FORI: The FIFO overrun interrupt flag is set when a new ADC data word (containing one or two samples) is to be written to the FIFO and the FIFO is full. If a FIFO overrun occurs, the data word that triggered the overrun will be lost.

- SDI: The scan done interrupt flag is set when scan mode is enabled and a channel scan sequence completes a single scan operation. In continuous scan mode, the SDI flag will be set when firmware exits scan.
- SCCI: The single conversion complete interrupt flag is set at the end of every conversion or accumulated conversion (when accumulation is enabled).
- WCI: The window comparator interrupt is set when a window comparison event happens and the current sequencer channel has enabled the interrupt.

32.10. DMA Configuration and Usage

DMA can be used to pipe the ADC data out of the FIFO into RAM, allowing more bandwidth for the core to perform other tasks. Note that the DMA will only start a transfer when there are 4 elements in the FIFO. For the SARADC module, the DMA should be configured as follows:

- Source size (SRCSIZE) and destination size (DSTSIZE) are 2 for a word transfer.
- The source address increment (SRCAIMD) is 3 for non-incrementing mode.
- The destination address increment (DSTAIMD) is 2 for word increments.
- NCOUNT (where NCOUNT+1 is the number of 4-byte words) and RPOWER (where 2^{RPOWER} is the number of data transfers) set as described below. Note that RPOWER > 2 is not valid setting.
  - RPOWER = 0 and NCOUNT = 0. As soon as the FIFO reaches 4 words, the first word will be transferred using the DMA. In this configuration, there will always be 3 lagging elements in the FIFO.
  - RPOWER = 1 and NCOUNT = 1. As soon as the FIFO reaches 4 words, the first two words will be transferred using the DMA. In this configuration, there will always be 2 lagging elements in the FIFO.
  - RPOWER = 2 and NCOUNT = 3. As soon as the FIFO reaches 4 words, all elements in the FIFO will be transferred using the DMA. In this configuration, there will be no lagging elements in the FIFO.

Once the DMA is configured, writing a 1 to DMAEN in the CONFIG register will enable the DMA transfers from the ADC. The FIFO will continue to be serviced by the DMA until the specified transfer operation is complete.

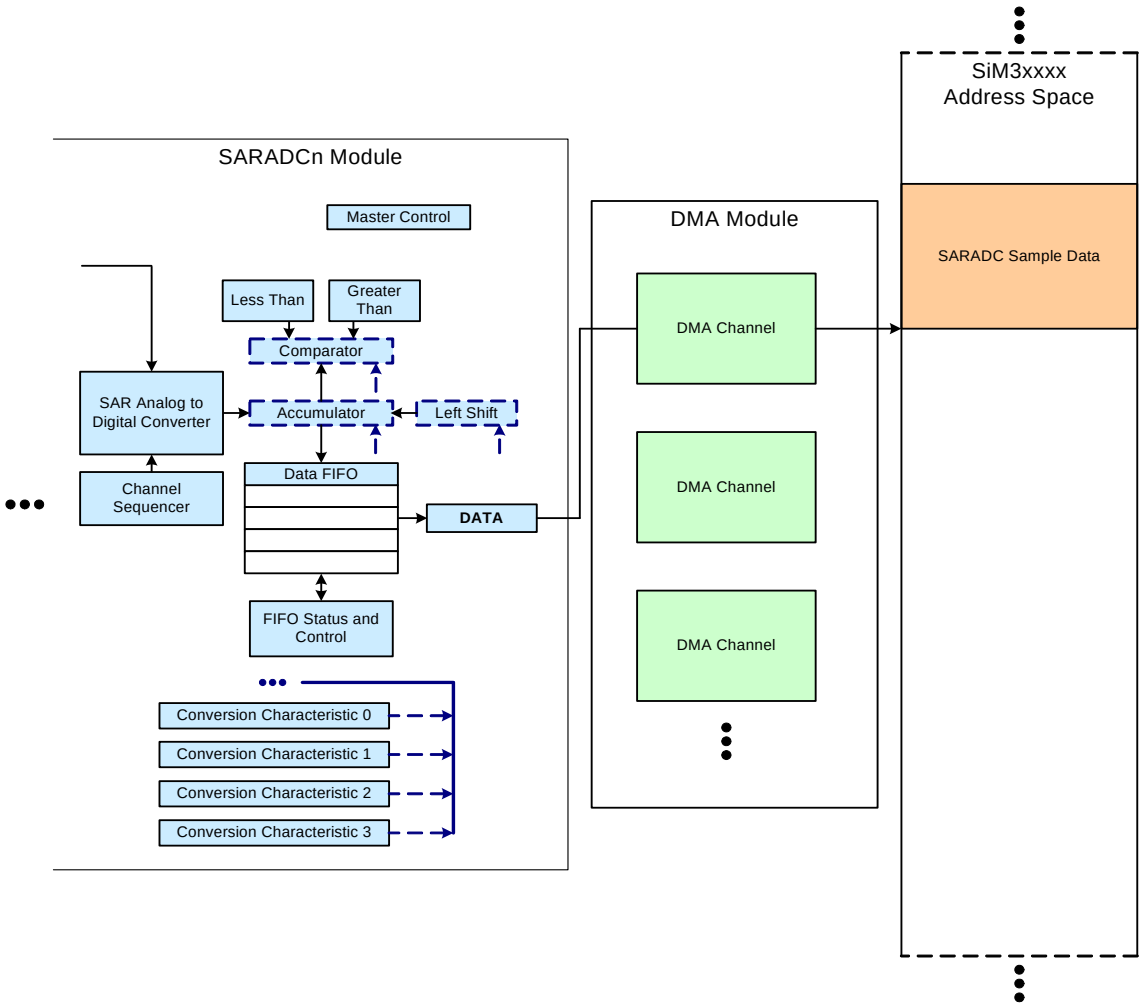


Figure 32.15. SAR ADC DMA Configuration

32.11. SARADC0 and SARADC1 Registers

This section contains the detailed register descriptions for SARADC0 and SARADC1 registers.

Register 32.1. SARADCn_CONFIG: Module Configuration

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved	FURIEN	FORIEN	SDIEN	SCCIEN	CLKDIV										
Type	R	RW	RW	RW	RW	RW										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	BCLKSEL	DMAEN	Reserved	SCANMD	Reserved	SCANEN	INTLVEN	SIMCEN	PACKMD		SSGEN	SPEN	SPSEL			
Type	RW	RW	R	RW	R	RW	RW	RW	RW		RW	RW	RW			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

SARADC0_CONFIG = 0x4001_A000
SARADC1_CONFIG = 0x4001_B000
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 32.5. SARADCn_CONFIG Register Bit Descriptions

Bit	Name	Function
31	Reserved	Must write reset value.
30	FURIEN	**FIFO Underrun Interrupt Enable.** 0: Disable the data FIFO underrun interrupt. 1: Enable the data FIFO underrun interrupt.
29	FORIEN	**FIFO Overrun Interrupt Enable.** 0: Disable the data FIFO overrun interrupt. 1: Enable the data FIFO overrun interrupt.
28	SDIEN	**Scan Done Interrupt Enable.** This bit enables the generation of an interrupt when the channel sequencer has cycled through all of the specified time slots. 0: Disable the ADC scan complete interrupt. 1: Enable the ADC scan complete interrupt.
27	SCCIEN	**Single Conversion Complete Interrupt Enable.** 0: Disable the ADC single data conversion complete interrupt. 1: Enable the ADC single data conversion complete interrupt.

Table 32.5. SARADCn_CONFIG Register Bit Descriptions

Bit	Name	Function
26:16	CLKDIV	**SAR Clock Divider.** This field sets the ADC clock divider value. It should be configured to be as close to the maximum SAR clock speed as the datasheet will allow. When CLKDIV < 3, the APB clock is used as the SAR clock. When CLKDIV ≥ 3, the SAR clock frequency is given by the following equation: $F_{CLKSAR} = \frac{2 \times F_{APB}}{CLKDIV + 1}$
15	BCLKSEL	**Burst Mode Clock Select.** 0: Burst mode uses the Low Power Oscillator. 1: Burst mode uses the APB clock.
14	DMAEN	**DMA Interface Enable .** 0: Disable the ADC module DMA interface. 1: Enable the ADC module DMA interface.
13	Reserved	Must write reset value.
12	SCANMD	**Scan Mode Select.** 0: The channel sequencer will cycle through all of the specified time slots once. 1: The channel sequencer will cycle through all of the specified time slots in a loop until SCANEN is cleared to 0.
11	Reserved	Must write reset value.
10	SCANEN	**Scan Mode Enable.** Setting this bit to 1 enables the ADC to scan through the specified time slots in the channel sequencer. The sequence begins on a 0-to-1 transition of this bit, so it must be 0 before a write to 1 to have any effect. 0: Disable ADC scan mode. 1: Enable ADC scan mode. The ADC will scan through the defined time slots in sequence on every start of conversion.
9	INTLVEN	**Interleaved Conversion Packing Enable.** This bit enables packing of the conversions from two ADCs in interleaved mode. It should be set to 1 on the master ADC if dual ADC sample packing is desired. This bit should always be set to 0 on the slave ADC. 0: Disable interleaved mode conversion packing. 1: Enable interleaved mode conversion packing.
8	SIMCEN	**Simultaneous Conversion Packing Enable.** This bit enables packing of the conversions from two ADCs in simultaneous mode. It should be set to 1 on the master ADC if dual ADC sample packing is desired. This bit should always be set to 0 on the slave ADC. 0: Disable simultaneous mode conversion packing. 1: Enable simultaneous mode conversion packing.

Table 32.5. SARADCn_CONFIG Register Bit Descriptions

Bit	Name	Function
7:6	PACKMD	**Output Packing Mode.** This field specifies how the ADC output data will be packed into the data registers. 00: Data is written to the upper half-word and the lower half-word is filled with 0's. An SCI interrupt is triggered when data is written, if enabled. 01: Data is written to the lower half-word, and the upper half-word is filled with 0's. An SCI interrupt is triggered when data is written, if enabled. 10: Two data words are packed into the register with the upper half-word representing the earlier data, and the lower half-word representing the later data. If SIMCEN is set to 1, the upper half-word represents data from the master ADC and the lower half-word represents data from the slave ADC. The ADC write to the lower half-word will trigger the SCI interrupt, if enabled. 11: Two data words are packed into the register with the lower half-word representing the earlier data, and the upper half-word representing the later data. If SIMCEN is set to 1, the lower half-word represents data from the master ADC and the upper half-word represents data from the slave ADC. The ADC write to the upper half-word will trigger the SCI interrupt, if enabled.
5	SSGEN	**Synchronous Sample Generator Enable.** This bit enables the ADC to create a conversion trigger based on the SSG module phase output signal. In dual ADC modes, SSGEN must be set to 1 for both ADCs. Note that when enabling SSG mode, the master's SSGEN must be set first, followed by the slave's SSGEN. When disabling SSG mode, the slave's SSGEN must be cleared first, followed by the master's SSGEN. 0: Disables conversion trigger generation from the SSG module phase output. 1: Enables conversion trigger generation from the SSG module phase output.
4	SPEN	**Sampling Phase Enable.** 0: Disable Phase Select. The ADC will always sample on the start-of-conversion trigger selected by the SCSEL field. 1: Enable Phase Select. The ADC will sample according to the phase selected by the SPSEL field.

Table 32.5. SARADCn_CONFIG Register Bit Descriptions

Bit	Name	Function
3:0	SPSEL	**Sampling Phase Select.** Allows the ADC to delay sampling from the start-of-conversion source to one of 16 different phases. This feature can only be used in conjunction with the sample sync generator (SSG). 0000: The ADC samples at SSG phase 0. 0001: The ADC samples at SSG phase 1. 0010: The ADC samples at SSG phase 2. 0011: The ADC samples at SSG phase 3. 0100: The ADC samples at SSG phase 4. 0101: The ADC samples at SSG phase 5. 0110: The ADC samples at SSG phase 6. 0111: The ADC samples at SSG phase 7. 1000: The ADC samples at SSG phase 8. 1001: The ADC samples at SSG phase 9. 1010: The ADC samples at SSG phase 10. 1011: The ADC samples at SSG phase 11. 1100: The ADC samples at SSG phase 12. 1101: The ADC samples at SSG phase 13. 1110: The ADC samples at SSG phase 14. 1111: The ADC samples at SSG phase 15.

Register 32.2. SARADCn_CONTROL: Measurement Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	VREFSEL		Reserved		MREFLPEN	LPMDEN	BIASSEL		ADBUSY	TRKMD	ACCMD	Reserved	VCMEN	AD12BSSEL	ADCEN	BURSTEN
Type	RW		RW		RW	RW	RW		RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	PWRTIME				SCSEL				BMTK						CLKESEL	REFGNDSEL
Type	RW				RW				RW						RW	RW
Reset	1	1	1	1	0	0	0	0	0	1	1	1	1	0	0	0

Register ALL Access Addresses

SARADC0_CONTROL = 0x4001_A010
SARADC1_CONTROL = 0x4001_B010
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 32.6. SARADCn_CONTROL Register Bit Descriptions

Bit	Name	Function
31:30	VREFSEL	**Voltage Reference Select.** 00: Select the internal, dedicated SARADC voltage reference as the ADC reference. 01: Select the VDD pin as the ADC reference. 10: Select the output of the internal LDO regulator (~1.8 V) as the ADC reference. 11: Select the VREF pin as the ADC reference. This option is used for either an external VREF or the on-chip VREF driving out to the VREF pin.
29:28	Reserved	Must write reset value.
27	MREFLPEN	**MUX and VREF Low Power Enable.** This bit is used to limit the power of the internal buffers used on the reference and the mux. It can be set to 1 to reduce power consumption when the SAR clock is slowed down. 0: Disable low power mode. 1: Enable low power mode (SAR clock ≤ 4 MHz).

Table 32.6. SARADCn_CONTROL Register Bit Descriptions

Bit	Name	Function
26	LPMDEN	**Low Power Mode Enable.** This bit can be used to reduce power to one of the ADC's internal nodes. It can be set to 1 to reduce power when tracking times in the application are longer (slower sample rates). 0: Disable low power mode. 1: Enable low power mode (requires extended tracking time).
25:24	BIASSEL	**Bias Power Select.** This field can be used to adjust the ADC's power consumption based on the conversion speed. Higher bias currents allow for faster conversion times. 00: Select bias current mode 0. Recommended to use modes 1, 2, or 3. 01: Select bias current mode 1 (SARCLK = 16 MHz). 10: Select bias current mode 2. 11: Select bias current mode 3 (SARCLK = 4 MHz).
23	ADBUSY	**ADC Busy.** This bit indicates that the ADC is currently converting (not tracking). Writing 1 to this bit in "On Demand" trigger mode initiates a conversion.
22	TRKMD	**ADC Tracking Mode.** 0: Normal Tracking Mode: When the ADC is enabled, a conversion begins immediately following the start-of-conversion signal. 1: Delayed Tracking Mode: When the ADC is enabled, a conversion begins 3 SAR clock cycles following the start-of-conversion signal. The ADC is allowed to track during this time.
21	ACCMD	**Accumulation Mode.** This bit is used to enable or disable accumulation when burst mode is enabled (BURSTEN = 1). 0: Conversions will be accumulated for the specified number of cycles in burst mode according to the channel configuration. 1: Conversions will not be accumulated in burst mode.
20	Reserved	Must write reset value.
19	VCMEN	**Common Mode Buffer Enable.** 0: Disable the common mode buffer. 1: Enable the common mode buffer.
18	AD12BSSEL	**12-Bit Mode Sample Select.** This bit defines how the ADC samples the analog input in 12-bit mode. 0: The ADC re-samples the input before each of the four conversions. 1: The ADC samples once before the first conversion and converts four times.
17	ADCEN	**ADC Enable.** 0: Disable the ADC (low-power shutdown). 1: Enable the ADC (active and ready for data conversions).
16	BURSTEN	**Burst Mode Enable.**

Table 32.6. SARADCn_CONTROL Register Bit Descriptions

Bit	Name	Function
15:12	PWRTIME	**Burst Mode Power Up Time.** This field sets the time delay required for ADC to power up from a low power state. $T_{PWRTIME} = \frac{8 \times PWRTIME}{F_{APB}}$
11:8	SCSEL	**Start-Of-Conversion Source Select.** This field specifies the event used to initiate conversions. 0000: An ADC conversion triggers from the ADCnT0 trigger source. 0001: An ADC conversion triggers from the ADCnT1 trigger source. 0010: An ADC conversion triggers from the ADCnT2 trigger source. 0011: An ADC conversion triggers from the ADCnT3 trigger source. 0100: An ADC conversion triggers from the ADCnT4 trigger source. 0101: An ADC conversion triggers from the ADCnT5 trigger source. 0110: An ADC conversion triggers from the ADCnT6 trigger source. 0111: An ADC conversion triggers from the ADCnT7 trigger source. 1000: An ADC conversion triggers from the ADCnT8 trigger source. 1001: An ADC conversion triggers from the ADCnT9 trigger source. 1010: An ADC conversion triggers from the ADCnT10 trigger source. 1011: An ADC conversion triggers from the ADCnT11 trigger source. 1100: An ADC conversion triggers from the ADCnT12 trigger source. 1101: An ADC conversion triggers from the ADCnT13 trigger source. 1110: An ADC conversion triggers from the ADCnT14 trigger source. 1111: An ADC conversion triggers from the ADCnT15 trigger source.
7:2	BMTK	**Burst Mode Tracking Time.** This field Sets the time delay between consecutive conversions performed in Burst Mode. $T_{BMTK} = \frac{64 - BMTK + (3 \times TRKMD)}{F_{APB}}$ Note: The Burst Mode track delay is not inserted prior to the first conversion. The required tracking time for the first conversion should be defined with the ADPWM field.
1	CLKESEL	**Sampling Clock Edge Select.** This bit selects which edge of the APB clock is used during sampling. Note that if the core is halted and a conversion completes while this bit is set to 1, the SCCI bit will not set. It is recommended to leave this bit at 0 when debugging SAR-related firmware. 0: Select the rising edge of the APB clock. 1: Select the falling edge of the APB clock.

Table 32.6. SARADCn_CONTROL Register Bit Descriptions

Bit	Name	Function
0	REFGNDSEL	**Reference Ground Select.** This bit selects the reference ground for ADC conversions. The internal ground is always used for temperature sensor measurements. 0: The internal device ground is used as the ground reference for ADC conversions. 1: The VREFGND pin is used as the ground reference for ADC conversions.

Register 32.3. SARADCn_SQ7654: Channel Sequencer Time Slots 4-7 Setup

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved	TS7MUX					TS7CHR		Reserved	TS6MUX					TS6CHR	
Type	R	RW					RW		R	RW					RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved	TS5MUX					TS5CHR		Reserved	TS4MUX					TS4CHR	
Type	R	RW					RW		R	RW					RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

SARADC0_SQ7654 = 0x4001_A020
SARADC1_SQ7654 = 0x4001_B020

Table 32.7. SARADCn_SQ7654 Register Bit Descriptions

Bit	Name	Function
31	Reserved	Must write reset value.
30:26	TS7MUX	**Time Slot 7 Input Channel.** A value of x in this field selects the ADCn.x channel as the ADCn input during this time slot. Set this field to 0x1F to terminate the scan at this slot.
25:24	TS7CHR	**Time Slot 7 Conversion Characteristic.** Selects which of the conversion characteristic settings is used for this time slot. 00: Select conversion characteristic 0 for time slot 7. 01: Select conversion characteristic 1 for time slot 7. 10: Select conversion characteristic 2 for time slot 7. 11: Select conversion characteristic 3 for time slot 7.
23	Reserved	Must write reset value.
22:18	TS6MUX	**Time Slot 6 Input Channel.** A value of x in this field selects the ADCn.x channel as the ADCn input during this time slot. Set this field to 0x1F to terminate the scan at this slot.

Table 32.7. SARADCn_SQ7654 Register Bit Descriptions

Bit	Name	Function
17:16	TS6CHR	**Time Slot 6 Conversion Characteristic.** Selects which of the conversion characteristic settings is used for this time slot. 00: Select conversion characteristic 0 for time slot 6. 01: Select conversion characteristic 1 for time slot 6. 10: Select conversion characteristic 2 for time slot 6. 11: Select conversion characteristic 3 for time slot 6.
15	Reserved	Must write reset value.
14:10	TS5MUX	**Time Slot 5 Input Channel.** A value of x in this field selects the ADCn.x channel as the ADCn input during this time slot. Set this field to 0x1F to terminate the scan at this slot.
9:8	TS5CHR	**Time Slot 5 Conversion Characteristic.** Selects which of the conversion characteristic settings is used for this time slot. 00: Select conversion characteristic 0 for time slot 5. 01: Select conversion characteristic 1 for time slot 5. 10: Select conversion characteristic 2 for time slot 5. 11: Select conversion characteristic 3 for time slot 5.
7	Reserved	Must write reset value.
6:2	TS4MUX	**Time Slot 4 Input Channel.** A value of x in this field selects the ADCn.x channel as the ADCn input during this time slot. Set this field to 0x1F to terminate the scan at this slot.
1:0	TS4CHR	**Time Slot 4 Conversion Characteristic.** Selects which of the conversion characteristic settings is used for this time slot. 00: Select conversion characteristic 0 for time slot 4. 01: Select conversion characteristic 1 for time slot 4. 10: Select conversion characteristic 2 for time slot 4. 11: Select conversion characteristic 3 for time slot 4.

Register 32.4. SARADCn_SQ3210: Channel Sequencer Time Slots 0-3 Setup

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved	TS3MUX					TS3CHR		Reserved	TS2MUX					TS2CHR	
Type	R	RW					RW		R	RW					RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved	TS1MUX					TS1CHR		Reserved	TS0MUX					TS0CHR	
Type	R	RW					RW		R	RW					RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

SARADC0_SQ3210 = 0x4001_A030
SARADC1_SQ3210 = 0x4001_B030

Table 32.8. SARADCn_SQ3210 Register Bit Descriptions

Bit	Name	Function
31	Reserved	Must write reset value.
30:26	TS3MUX	**Time Slot 3 Input Channel.** A value of x in this field selects the ADCn.x channel as the ADCn input during this time slot. Set this field to 0x1F to terminate the scan at this slot.
25:24	TS3CHR	**Time Slot 3 Conversion Characteristic.** Selects which of the conversion characteristic settings is used for this time slot. 00: Select conversion characteristic 0 for time slot 3. 01: Select conversion characteristic 1 for time slot 3. 10: Select conversion characteristic 2 for time slot 3. 11: Select conversion characteristic 3 for time slot 3.
23	Reserved	Must write reset value.
22:18	TS2MUX	**Time Slot 2 Input Channel.** A value of x in this field selects the ADCn.x channel as the ADCn input during this time slot. Set this field to 0x1F to terminate the scan at this slot.

Table 32.8. SARADCn_SQ3210 Register Bit Descriptions

Bit	Name	Function
17:16	TS2CHR	**Time Slot 2 Conversion Characteristic.** Selects which of the conversion characteristic settings is used for this time slot. 00: Select conversion characteristic 0 for time slot 2. 01: Select conversion characteristic 1 for time slot 2. 10: Select conversion characteristic 2 for time slot 2. 11: Select conversion characteristic 3 for time slot 2.
15	Reserved	Must write reset value.
14:10	TS1MUX	**Time Slot 1 Input Channel.** A value of x in this field selects the ADCn.x channel as the ADCn input during this time slot. Set this field to 0x1F to terminate the scan at this slot.
9:8	TS1CHR	**Time Slot 1 Conversion Characteristic.** Selects which of the conversion characteristic settings is used for this time slot. 00: Select conversion characteristic 0 for time slot 1. 01: Select conversion characteristic 1 for time slot 1. 10: Select conversion characteristic 2 for time slot 1. 11: Select conversion characteristic 3 for time slot 1.
7	Reserved	Must write reset value.
6:2	TS0MUX	**Time Slot 0 Input Channel.** A value of x in this field selects the ADCn.x channel as the ADCn input during this time slot. Set this field to 0x1F to terminate the scan at this slot. Time Slot 0 is also used to specify the parameters for single (non-scan mode) conversions.
1:0	TS0CHR	**Time Slot 0 Conversion Characteristic.** Selects which of the conversion characteristic settings is used for this time slot. 00: Select conversion characteristic 0 for time slot 0. 01: Select conversion characteristic 1 for time slot 0. 10: Select conversion characteristic 2 for time slot 0. 11: Select conversion characteristic 3 for time slot 0.

Register 32.5. SARADCn_CHAR32: Conversion Characteristic 2 and 3 Setup

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved							CHR3WCIEN	CHR3RSEL	CHR3LS			CHR3RPT			CHR3GN
Type	R							RW	RW	RW			RW			RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved							CHR2WCIEN	CHR2RSEL	CHR2LS			CHR2RPT			CHR2GN
Type	R							RW	RW	RW			RW			RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

SARADC0_CHAR32 = 0x4001_A040
SARADC1_CHAR32 = 0x4001_B040
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 32.9. SARADCn_CHAR32 Register Bit Descriptions

Bit	Name	Function
31:25	Reserved	Must write reset value.
24	CHR3WCIEN	**Conversion Characteristic 3 Window Comparator Interrupt Enable.** Enable window comparison interrupts for this channel. 0: Disable window comparison interrupts. 1: Enabled window comparison interrupts. The window comparator will be used to check the ADC result on channels that use this characteristic.
23	CHR3RSEL	**Conversion Characteristic 3 Resolution Selection.** Select between 10- and 12-bit mode. 0: Select 10-bit Mode. 1: Select 12-bit Mode (burst mode must be enabled).
22:20	CHR3LS	**Conversion Characteristic 3 Left-Shift Bits.** This field specifies the number of bits to shift the result left at conversion completion. A zero value produces a fully right-justified result.

Table 32.9. SARADCn_CHAR32 Register Bit Descriptions

Bit	Name	Function
19:17	CHR3RPT	**Conversion Characteristic 3 Repeat Counter.** This field determines the number of samples the converter will accumulate in burst mode when the accumulation option is enabled. 000: Accumulate one sample. 001: Accumulate four samples. 010: Accumulate eight samples. 011: Accumulate sixteen samples. 100: Accumulate thirty-two samples (10-bit mode only). 101: Accumulate sixty-four samples (10-bit mode only). 110-111: Reserved.
16	CHR3GN	**Conversion Characteristic 3 Gain.** 0: The on-chip PGA gain is 1. 1: The on-chip PGA gain is 0.5.
15:9	Reserved	Must write reset value.
8	CHR2WCIEN	**Conversion Characteristic 2 Window Comparator Interrupt Enable.** Enable window comparison interrupts for this channel. 0: Disable window comparison interrupts. 1: Enabled window comparison interrupts. The window comparator will be used to check the ADC result on channels that use this characteristic.
7	CHR2RSEL	**Conversion Characteristic 2 Resolution Selection.** Select between 10- and 12-bit mode. 0: Select 10-bit Mode. 1: Select 12-bit Mode (burst mode must be enabled).
6:4	CHR2LS	**Conversion Characteristic 2 Left-Shift Bits.** This field specifies the number of bits to shift the result left at conversion completion. A zero value produces a fully right-justified result.
3:1	CHR2RPT	**Conversion Characteristic 2 Repeat Counter.** This field determines the number of samples the converter will accumulate in burst mode when the accumulation option is enabled. 000: Accumulate one sample. 001: Accumulate four samples. 010: Accumulate eight samples. 011: Accumulate sixteen samples. 100: Accumulate thirty-two samples (10-bit mode only). 101: Accumulate sixty-four samples (10-bit mode only). 110-111: Reserved.
0	CHR2GN	**Conversion Characteristic 2 Gain.** 0: The on-chip PGA gain is 1. 1: The on-chip PGA gain is 0.5.

Register 32.6. SARADCn_CHAR10: Conversion Characteristic 0 and 1 Setup

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved							CHR1WCIEN	CHR1RSEL	CHR1LS			CHR1RPT			CHR1GN
Type	R							RW	RW	RW			RW			RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved							CHR0WCIEN	CHR0RSEL	CHR0LS			CHR0RPT			CHR0GN
Type	R							RW	RW	RW			RW			RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

SARADC0_CHAR10 = 0x4001_A050
SARADC1_CHAR10 = 0x4001_B050
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 32.10. SARADCn_CHAR10 Register Bit Descriptions

Bit	Name	Function
31:25	Reserved	Must write reset value.
24	CHR1WCIEN	**Conversion Characteristic 1 Window Comparator Interrupt Enable.** Enable window comparison interrupts for this channel. 0: Disable window comparison interrupts. 1: Enabled window comparison interrupts. The window comparator will be used to check the ADC result on channels that use this characteristic.
23	CHR1RSEL	**Conversion Characteristic 1 Resolution Selection.** Select between 10- and 12-bit mode. 0: Select 10-bit Mode. 1: Select 12-bit Mode (burst mode must be enabled).
22:20	CHR1LS	**Conversion Characteristic 1 Left-Shift Bits.** This field specifies the number of bits to shift the result left at conversion completion. A zero value produces a fully right-justified result.

Table 32.10. SARADCn_CHAR10 Register Bit Descriptions

Bit	Name	Function
19:17	CHR1RPT	**Conversion Characteristic 1 Repeat Counter.** This field determines the number of samples the converter will accumulate in burst mode when the accumulation option is enabled. 000: Accumulate one sample. 001: Accumulate four samples. 010: Accumulate eight samples. 011: Accumulate sixteen samples. 100: Accumulate thirty-two samples (10-bit mode only). 101: Accumulate sixty-four samples (10-bit mode only). 110-111: Reserved.
16	CHR1GN	**Conversion Characteristic 1 Gain.** 0: The on-chip PGA gain is 1. 1: The on-chip PGA gain is 0.5.
15:9	Reserved	Must write reset value.
8	CHR0WCIEN	**Conversion Characteristic 0 Window Comparator Interrupt Enable.** Enable window comparison interrupts for this channel. 0: Disable window comparison interrupts. 1: Enabled window comparison interrupts. The window comparator will be used to check the ADC result on channels that use this characteristic.
7	CHR0RSEL	**Conversion Characteristic 0 Resolution Selection.** Select between 10- and 12-bit mode. 0: Select 10-bit Mode. 1: Select 12-bit Mode (burst mode must be enabled).
6:4	CHR0LS	**Conversion Characteristic 0 Left-Shift Bits.** This field specifies the number of bits to shift the result left at conversion completion. A zero value produces a fully right-justified result.
3:1	CHR0RPT	**Conversion Characteristic 0 Repeat Counter.** This field determines the number of samples the converter will accumulate in burst mode when the accumulation option is enabled. 000: Accumulate one sample. 001: Accumulate four samples. 010: Accumulate eight samples. 011: Accumulate sixteen samples. 100: Accumulate thirty-two samples (10-bit mode only). 101: Accumulate sixty-four samples (10-bit mode only). 110-111: Reserved.
0	CHR0GN	**Conversion Characteristic 0 Gain.** 0: The on-chip PGA gain is 1. 1: The on-chip PGA gain is 0.5.

Register 32.7. SARADCn_DATA: Output Data Word

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	DATA[31:16]															
Type	R															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	DATA[15:0]															
Type	R															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Register ALL Access Addresses

SARADC0_DATA = 0x4001_A060

SARADC1_DATA = 0x4001_B060

Table 32.11. SARADCn_DATA Register Bit Descriptions

Bit	Name	Function
31:0	DATA	**Output Data Word.** The DATA register represents the oldest information available in the FIFO. When DATA is read, FIFOLVL decrements by one, and the FIFO pointer will point to the next value in the FIFO. Data is packed according to the PACKMD field.

Register 32.8. SARADCn_WCLIMITS: Window Comparator Limits

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	WCGT															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	WCLT															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

SARADC0_WCLIMITS = 0x4001_A070
SARADC1_WCLIMITS = 0x4001_B070

Table 32.12. SARADCn_WCLIMITS Register Bit Descriptions

Bit	Name	Function
31:16	WCGT	**Greater-Than Window Comparator Limit.** This field is the right-justified "greater than" parameter for the window comparator. ADC output data will be compared against this value when it is available. When WCLT is greater than WCGT, an ADC result between the two limits will cause a window compare interrupt, if enabled. When WCLT is less than WCGT, an ADC result above or below the two limits (but not in between) will cause a window compare interrupt, if enabled.
15:0	WCLT	**Less-Than Window Comparator Limit.** This field is the right-justified "less than" parameter for the window comparator. ADC output data will be compared against this value when it is available. When WCLT is greater than WCGT, an ADC result between the two limits will cause a window compare interrupt, if enabled. When WCLT is less than WCGT, an ADC result above or below the two limits (but not in between) will cause a window compare interrupt, if enabled.

Register 32.9. SARADCn_ACC: Accumulator Initial Value

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	ACC															
Type	W															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

SARADC0_ACC = 0x4001_A080
SARADC1_ACC = 0x4001_B080

Table 32.13. SARADCn_ACC Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15:0	ACC	**Accumulator Initial Value.** This write-only field is used to set the accumulator to an initial value. In most cases, this field should be written to zero before beginning a conversion or a scan sequence when accumulation is enabled (ACCMD = 0).

Register 32.10. SARADCn_STATUS: Module Status

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved											FURI	FORI	SDI	SCCI	WCI
Type	R											RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

SARADC0_STATUS = 0x4001_A090
SARADC1_STATUS = 0x4001_B090
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 32.14. SARADCn_STATUS Register Bit Descriptions

Bit	Name	Function
31:5	Reserved	Must write reset value.
4	FURI	**FIFO Underrun Interrupt Flag.** This bit is set to 1 by hardware when a FIFO underrun event has occurred, and can be used to trigger an interrupt if enabled. This bit must be cleared by software.
3	FORI	**FIFO Overrun Interrupt Flag.** This bit is set to 1 by hardware when a FIFO overrun event has occurred, and can be used to trigger an interrupt if enabled. This bit must be cleared by software.
2	SDI	**Scan Done Interrupt Flag.** This bit is set to 1 by hardware when a scan operation is complete, and can be used to trigger an interrupt if enabled. This bit must be cleared by software.
1	SCCI	**Single Conversion Complete Interrupt Flag.** This bit is set to 1 by hardware at the end of each conversion, and can be used to trigger an interrupt if enabled. This bit must be cleared by software.
0	WCI	**Window Compare Interrupt Flag.** This bit is set to 1 by hardware when a window comparator event has occurred, and can be used to trigger an interrupt if enabled. This bit must be cleared by software.

Notes:
1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Register 32.11. SARADCn_FIFOSTATUS: FIFO Status

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved										DRDYF	DPSTS	FIFOLVL			
Type	R										R	R	R			
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Register ALL Access Addresses

SARADC0_FIFOSTATUS = 0x4001_A0A0

SARADC1_FIFOSTATUS = 0x4001_B0A0

Table 32.15. SARADCn_FIFOSTATUS Register Bit Descriptions

Bit	Name	Function
31:6	Reserved	Must write reset value.
5	DRDYF	**Data Ready Flag.** This bit indicates that new data is ready to be written into the FIFO. It is set after the data conversion is complete and cleared by any new conversion start trigger. 0: New data is not produced yet. 1: New data is ready.
4	DPSTS	**Data Packing Status.** This is a read only status bit indicating to which half-word the hardware will write the next ADC output data. 0: The next ADC conversion will be written to the lower half-word. 1: The next ADC conversion will be written to the upper half-word.
3:0	FIFOLVL	**FIFO Level.** This is the number of ADC words in the FIFO. Each word may contain one or two samples depending on the packing mode (PACKMD).

32.12. SARADCn Register Memory Map

Table 32.16. SARADCn Memory Map

Register Name	SARADCn_SQ3210	SARADCn_SQ7654	SARADCn_CONTROL	SARADCn_CONFIG
ALL Offset	0x30	0x20	0x10	0x0
Access Methods	ALL	ALL	ALL | SET | CLR	ALL | SET | CLR
Bit 31	Reserved	Reserved	VREFSEL	Reserved
Bit 30	TS3MUX	TS7MUX		FURIEN
Bit 29			Reserved	FORIEN
Bit 28				SDIEN
Bit 27	TS3CHR	TS7CHR	MREFLPEN	SCCIEN
Bit 26			LPMDEN	CLKDIV
Bit 25			BIASSEL	
Bit 24			ADBUSY	
Bit 23	Reserved	Reserved	TRKMD	
Bit 22	TS2MUX	TS6MUX	ACCMD	
Bit 21			Reserved	
Bit 20			VCMEN	
Bit 19			AD12BSSEL	
Bit 18	TS2CHR	TS6CHR	ADCEN	
Bit 17			BURSTEN	
Bit 16	Reserved	Reserved	PWRTIME	BCLKSEL
Bit 15	TS1MUX	TS5MUX		DMAEN
Bit 14			SCSEL	Reserved
Bit 13				SCANMD
Bit 12	TS1CHR	TS5CHR	BMTK	Reserved
Bit 11				SCANEN
Bit 10				INTLVEN
Bit 9				SIMCEN
Bit 8	Reserved	Reserved	CLKESEL	PACKMD
Bit 7	TS0MUX	TS4MUX	REFGNDSEL	
Bit 6				SSGEN
Bit 5				SPEN
Bit 4				SPSEL
Bit 3				
Bit 2				
Bit 1	TS0CHR	TS4CHR		
Bit 0				

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: SARADC0 = 0x4001_A000, SARADC1 = 0x4001_B000

Table 32.16. SARADCn Memory Map

Register Name	SARADCn_WCLIMITS	SARADCn_DATA	SARADCn_CHAR10	SARADCn_CHAR32
ALL Offset	0x70	0x60	0x50	0x40
Access Methods	ALL	ALL	ALL | SET | CLR	ALL | SET | CLR
Bit 31	WCGT	DATA	Reserved	Reserved
Bit 30				
Bit 29				
Bit 28				
Bit 27				
Bit 26				
Bit 25				
Bit 24			CHR1WCIEN	CHR3WCIEN
Bit 23			CHR1RSEL	CHR3RSEL
Bit 22			CHR1LS	CHR3LS
Bit 21				
Bit 20				
Bit 19			CHR1RPT	CHR3RPT
Bit 18				
Bit 17				
Bit 16			CHR1GN	CHR3GN
Bit 15	WCLT		Reserved	Reserved
Bit 14				
Bit 13				
Bit 12				
Bit 11				
Bit 10				
Bit 9				
Bit 8			CHR0WCIEN	CHR2WCIEN
Bit 7			CHR0RSEL	CHR2RSEL
Bit 6			CHR0LS	CHR2LS
Bit 5				
Bit 4				
Bit 3			CHR0RPT	CHR2RPT
Bit 2				
Bit 1			CHR0GN	CHR2GN
Bit 0				

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: SARADC0 = 0x4001_A000, SARADC1 = 0x4001_B000

Table 32.16. SARADCn Memory Map

Register Name	ALL Offset	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SARADCn_ACC	0x80	ALL	Reserved																ACC															
SARADCn_STATUS	0x90	ALL | SET | CLR	Reserved																											FURI	FORI	SDI	SCCI	WCI
SARADCn_FIFOSTATUS	0xA0	ALL	Reserved																										DRDYF	DPSTS	FIFOLVL			

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: SARADC0 = 0x4001_A000, SARADC1 = 0x4001_B000

33. Serial Peripheral Interface (SPI0, SPI1 and SPI2)

This section describes the Serial Peripheral Interface (SPI) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the SPI block, which is used by SPI0, SPI1 and SPI2 on all device families covered in this document.

33.1. SPI Features

The SPI module includes the following features:

- Supports 3- or 4-wire master or slave modes.
- Supports up to 10 MHz clock in master mode and one-tenth of the APB clock in slave mode.
- Support for all clock phase polarity and slave select (NSS) polarity modes.
- 16-bit programmable clock rate.
- Programmable MSB-first or LSB-first shifting.
- 8-byte FIFO buffers for both transmit and receive data paths to support high speed transfers.
- Programmable FIFO threshold level to request data service for DMA transfers.
- Support for multiple masters on the same data lines.

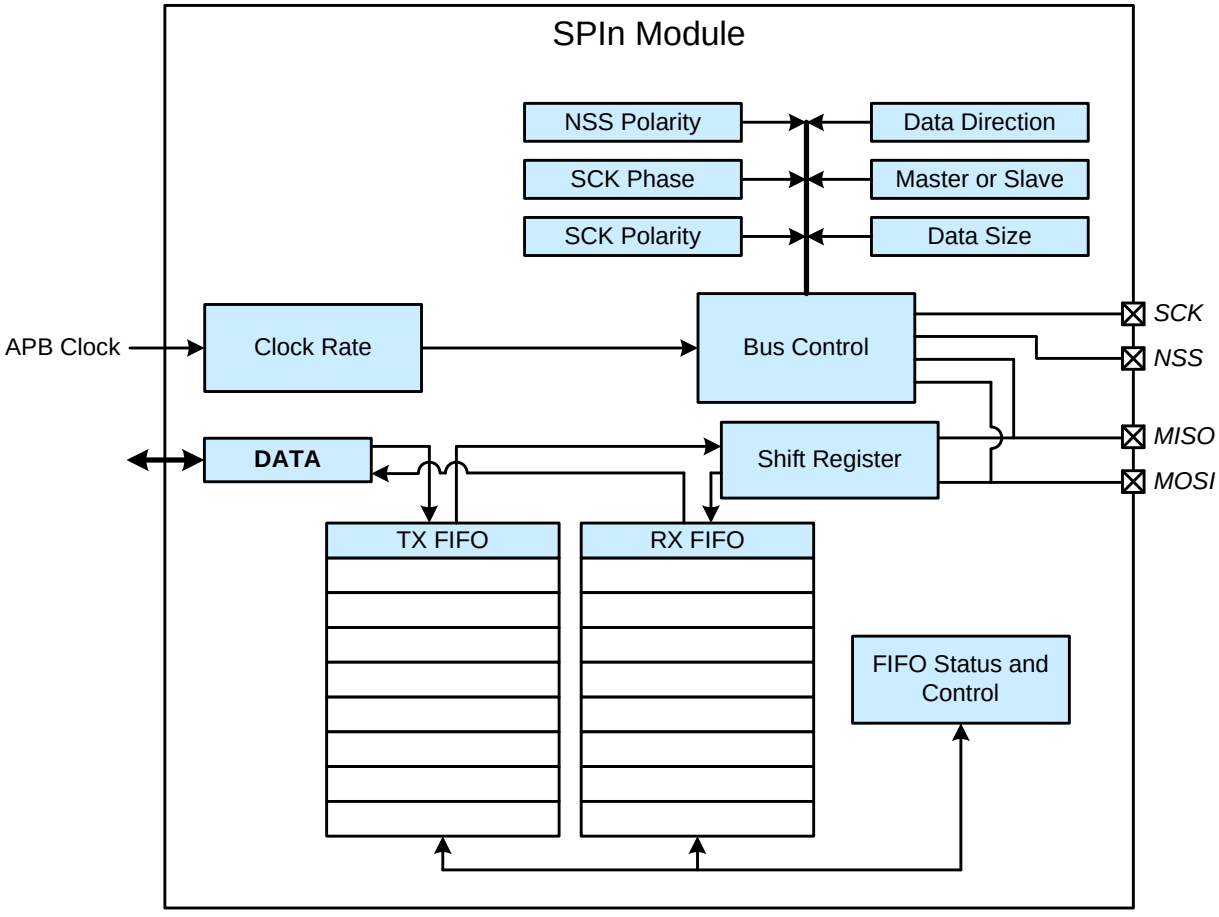


Figure 33.1. SPI Block Diagram

33.2. Signal Descriptions

The four signals used by the SPI module are MOSI, MISO, SCK, NSS. Figure 33.2 shows a typical SPI transfer.

33.2.1. Master Out, Slave In (MOSI)

The master-out, slave-in (MOSI) signal is an output from a master device and an input to the slave devices on the bus. It is used to serially transfer data from the master to the slaves. This MOSI pin is an output when the module operates as a master and an input when operating as a slave.

33.2.2. Master In, Slave Out (MISO)

The master-in, slave-out (MISO) signal is an output from a slave device and an input to the master device. It is used to serially transfer data from the slave to the master. The MISO pin is an input when the SPI module operates as a master and an output when operating as a slave. The hardware places the MISO pin in a high-impedance state when the module is disabled or when the module operates in 4-wire mode as a slave that is not selected.

33.2.3. Serial Clock (SCK)

The serial clock (SCK) signal is an output from the master device and an input to the slave devices. It is used to synchronize the transfer of data between the master and slave on the MOSI and MISO pins. The SCK pin is an output driving the clock when operating as a master and an input receiving the clock when operating as a slave.

33.2.4. Slave Select (NSS)

The slave select (NSS) signal can be an output from a master device (in 4-wire single master mode), an input to a master device (in 4-wire multiple master), an input to a slave device (in 4-wire slave mode), or unused/unconnected (in 3-wire master or 3-wire slave mode). The slave select mode (NSSMD) field in the CONFIG register configures the NSS pin for the desired mode.

If NSS is used as either an output or input, the pin should have an external pull-up resistor to VIO.

The NSS signal may be optionally connected to a physical pin. The device port configuration module has more information.

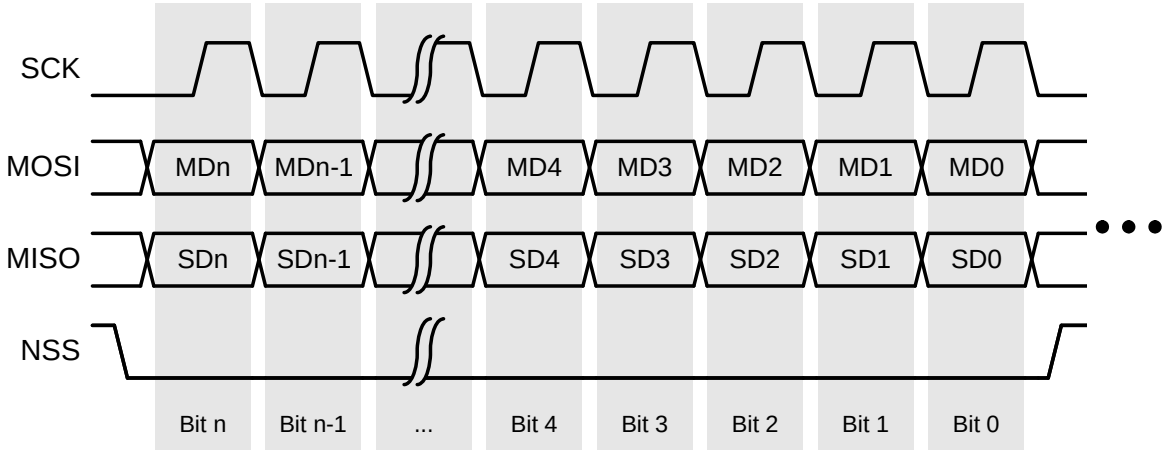


Figure 33.2. 4-Wire SPI Transfer

33.3. Clocking

The APB clock is the clock source for the SPI module. The SCK output clock in master mode is a divided version of this clock. In both master and slave modes, the clock signal present on the SCK pin determines when data is shifted out of or into the data shift register.

33.3.1. Master Mode Clocking

In master mode, an internal clock rate generator is used to divide down the APB clock to produce the desired SCK rate, and the hardware drives the SCK pin as an output from the device. The 16-bit clock divider (CLKDIV) field in the CLKRATE register sets the SCK frequency as a fraction of the APB clock. The CLKDIV bit description describes the equation for the SCK frequency as a function of the APB clock.

33.3.2. Slave Mode Clocking

The CLKDIV field is not used in slave mode, and the SCK pin becomes an input to the device. A different device should be configured as the SPI bus master and is expected to drive the SCK input at the desired frequency. The maximum input SCK rate in slave mode is equal to the APB clock frequency divided by 10.

33.4. Signal Format

The SPI module has flexible data formatting options. The data length, clock phase, clock polarity, shift direction, and NSS polarity are all selectable via fields in the CONFIG register.

33.4.1. Data Size and Shift Direction

The data size (DSIZE) field configures the transfer data length to be between 1 and 16 bits. DSIZE should be set to one less than the desired data length; for an 8-bit data length, firmware should set DSIZE to 7.

The data direction select (DDIRSEL) field configures the data shift direction for both transmitted and received data. The hardware can shift data MSB first (DDIRSEL = 0) or LSB first (DDIRSEL = 1).

33.4.2. Slave Select Polarity

The slave select polarity (NSSPOL) bit determines the polarity of the NSS pin for both master mode where NSS is an output and for slave mode where NSS is an input. The pin can be active low (NSSPOL = 0) or active high (NSSPOL = 1).

33.4.3. Clock Phase and Polarity Configuration

The CLKPHA and CLKPOL bits configure the SCK pin phase and polarity, respectively. Clearing CLKPHA to 0 places the SCK edge at the center of the data bit, and setting CLKPHA to 1 places the SCK edge at the data bit transition edge. The clock can also be idle low (CLKPOL = 0) or idle high (CLKPOL = 1).

The CLKPHA and CLKPOL bits must be set in both master and slave modes. In master mode, these bits determine the characteristics of the clock driven on of the SCK output. In slave mode, these bits set the characteristics of the clock that the module expects to receive on the SCK input. The clock phase and polarity settings determine when the data transitions take place with respect to the SCK phase.

Figure 33.3 illustrates all clock polarity and phase combinations when DDIRSEL is cleared to 0. Figure 33.4 illustrates all clock polarity and phase combinations when DDIRSEL is set to 1.

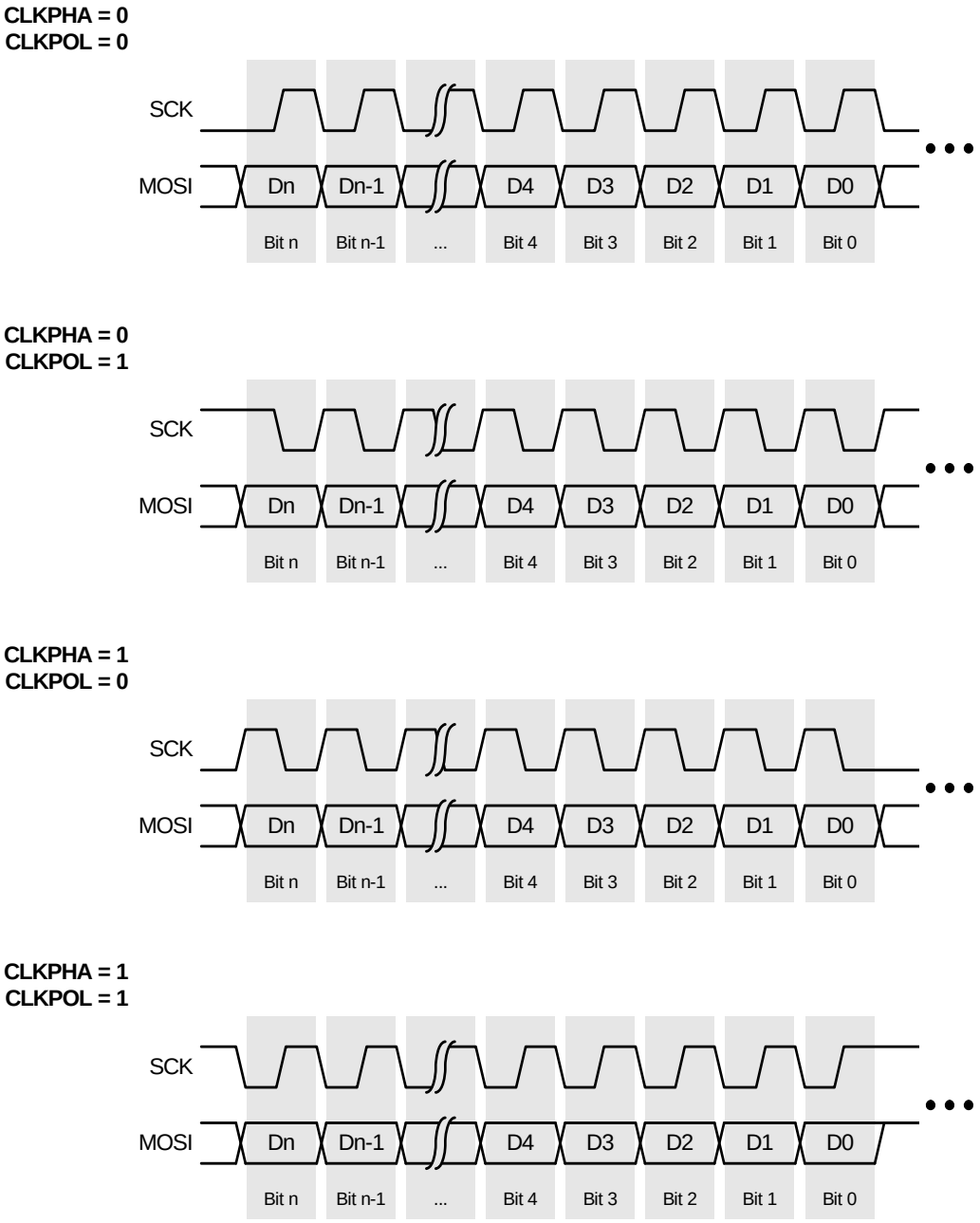


Figure 33.3. SPI Clock Polarity and Phase Combinations (DDIRSEL = 0, Master Mode)

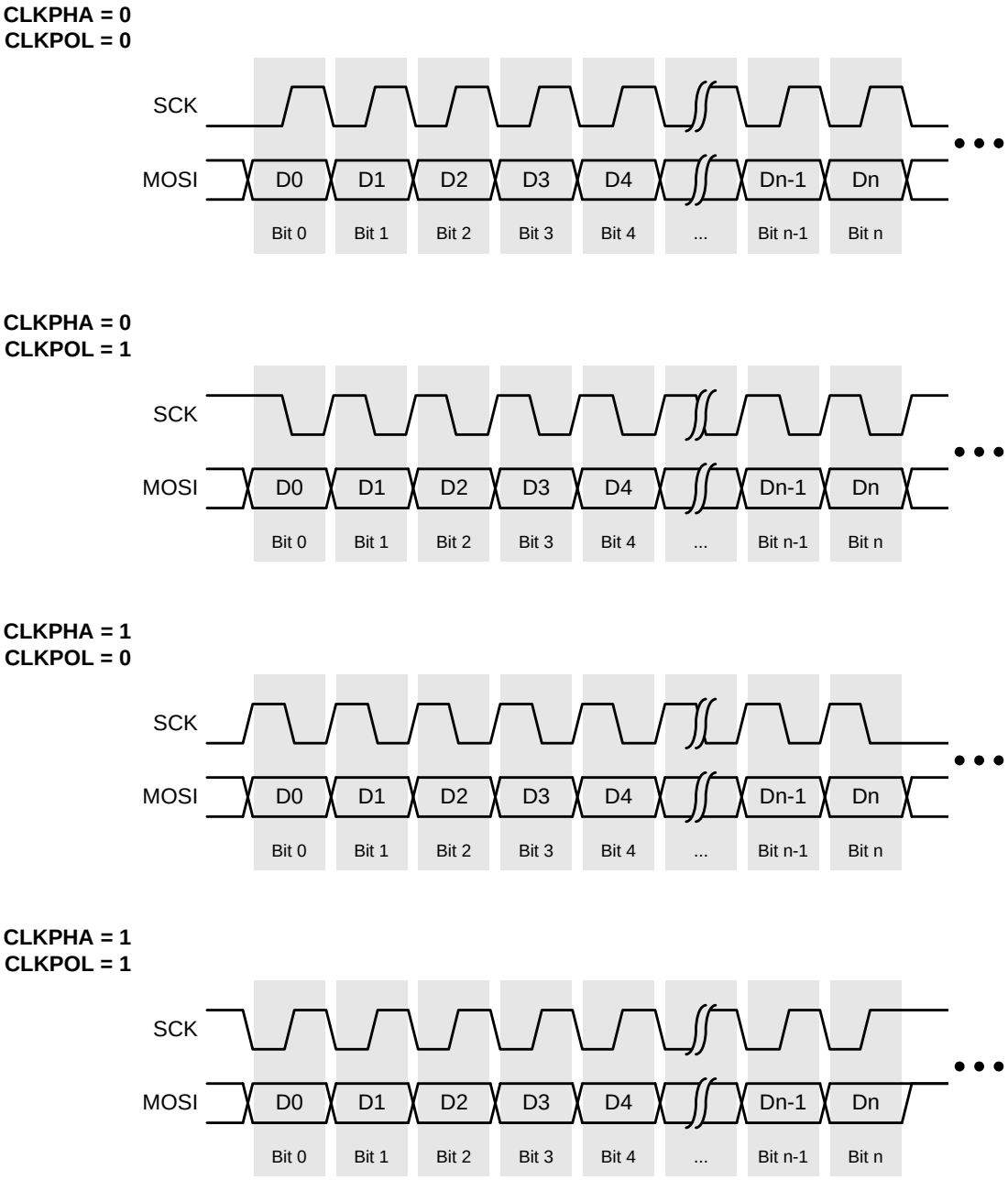


Figure 33.4. SPI Clock Polarity and Phase Combinations (DDIRSEL = 1, Master Mode)

33.5. Master Mode Configurations and Data Transfer

Firmware can set the MSTEN bit to 1 to configure the module as a master. The module supports three different options of master mode operation.

33.5.1. 3-Wire Single Master Mode

In 3-wire single master mode (NSSMD = 0), the slave select (NSS) pin is not required and may not be connected to port pins by the device port configuration module. In this mode, the device should be connected to a single slave device that either doesn't support an NSS input or has its NSS input tied low. To support multiple slaves in 3-wire single master mode, each slave device must have an NSS input, the NSS inputs must be connected to a unique port pin on the master device, and each pin must be controlled from firmware (bit-banged).

Figure 33.5 illustrates the master-slave connection diagram for 3-wire single master mode.

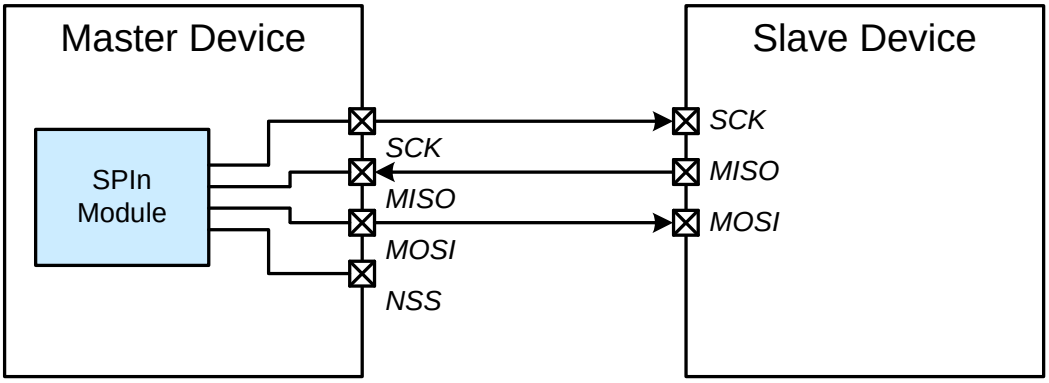


Figure 33.5. 3-Wire Single Master Mode Connection Diagram

33.5.2. 4-Wire Single Master Mode

In 4-wire single master mode (NSSMD = 2 or 3), the slave select (NSS) pin is configured as an output and should be connected to the NSS input of the first slave device. An additional slave device added on the SPI bus should have its NSS input driven by a firmware-controlled port pin output from the master device. The least-significant bit of NSSMD determines the state of the master's NSS pin in this mode.

Figure 33.6 shows the master-slave connection diagram for 4-wire single master mode.

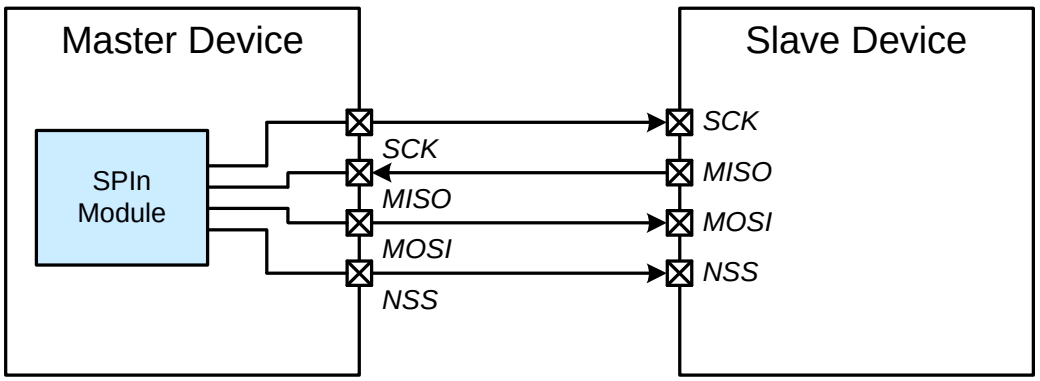


Figure 33.6. 4-Wire Single Master Mode Connection Diagram

33.5.3. 4-Wire Multiple Master Mode

Set the NSSMD (Slave Select Mode) field to 0x01 to configure the NSS pin for 4-wire slave / multi master mode.

In 4-wire multiple master mode (NSSMD = 1), the slave select (NSS) pin is configured as an input and is used to disable the SPI module while another SPI master accesses the bus. When the NSS input is driven low by the bus master, two events occur:

1. Hardware clears the master mode enable (MSTEN) and SPI enable (SPIEN) bits to disable the SPI module. The module must be manually re-enabled by firmware.
2. Hardware sets the mode fault interrupt (MDFI) flag. This will generate an interrupt if the mode fault interrupt is enabled (MDFIEN = 1).

Slave devices on the SPI bus should have their NSS inputs driven by a firmware-controlled port pin output from the master devices.

Figure 33.7 illustrates the connection diagram for 4-wire multiple master mode.

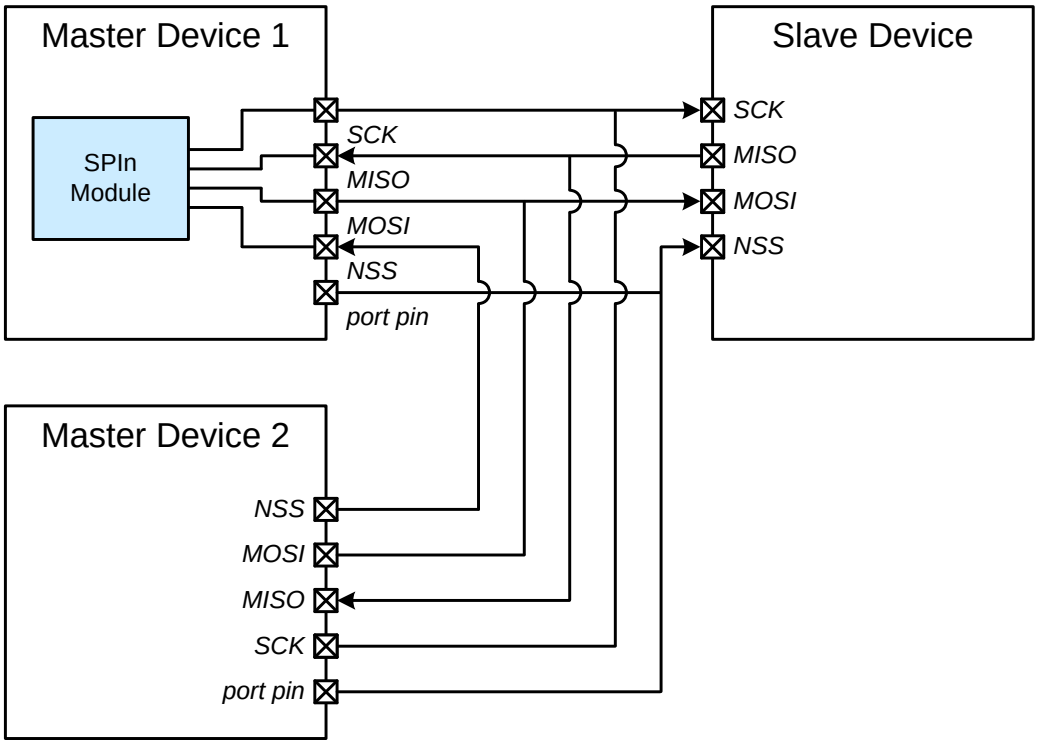


Figure 33.7. 4-Wire Multiple Master Mode Connection Diagram

33.5.4. Master Mode Data Transfer

A master device initiates all data transfers on a SPI bus. When the SPI module is first enabled by setting the SPIEN bit to 1, the transmit and receive FIFOs are empty. The hardware sets the transmit FIFO write request interrupt (TFRQI) flag when the number of empty slots in the transmit FIFO is at or above the transmit FIFO threshold (TFTH), which causes an interrupt, if enabled (TFRQIEN = 1). Firmware can then write to the DATA register in right-justified bytes, half-words, or full words to transfer data to the transmit FIFO.

When the shift register is empty, the hardware retrieves data from the transmit FIFO and begins a transmission. The module immediately shifts the data out serially on the MOSI line while driving the serial clock on SCK. When both the transmit FIFO and the shift register are empty, the hardware sets the underrun interrupt (URI) flag, resulting in an interrupt if the underrun interrupt is enabled (URIEN = 1).

The SPI bus is full-duplex, so the addressed SPI slave device may also be simultaneously transferring the contents of its shift register back to the SPI master using the MISO line as the master transfers data to a slave using MOSI. The shift register empty interrupt (SREI) flag serves as both a transmit-complete flag and receive-data-ready flag. When a received byte is fully transferred into the shift register, the hardware automatically moves it into the receive FIFO where it may be accessed by reading the DATA register.

33.6. Slave Mode Configurations and Data Transfer

Clearing the master mode enable (MSTEN) bit configures the SPI module for slave mode. Firmware should first fully configure the module (data length, clock polarity and phase, and slave mode) before setting the SPIEN bit to initiate SPI operations.

33.6.1. 3-Wire Slave Mode

In 3-wire slave mode (NSSMD = 0), the slave select (NSS) pin is not required and may not be connected to physical pins by the device's port configuration module. Because there is no means to uniquely address multiple slave devices in this mode, the device's SPI module should be the only slave device present on the bus in 3-wire slave mode. In addition, the master's pins should be in an idle state before enabling the module in 3-wire slave mode, since any unexpected transitions on SCK can cause erroneous bits to be shifted into the shift register without the NSS signal to gate the clock on SCK.

Figure 33.8 illustrates the connection diagram for 3-wire slave mode.

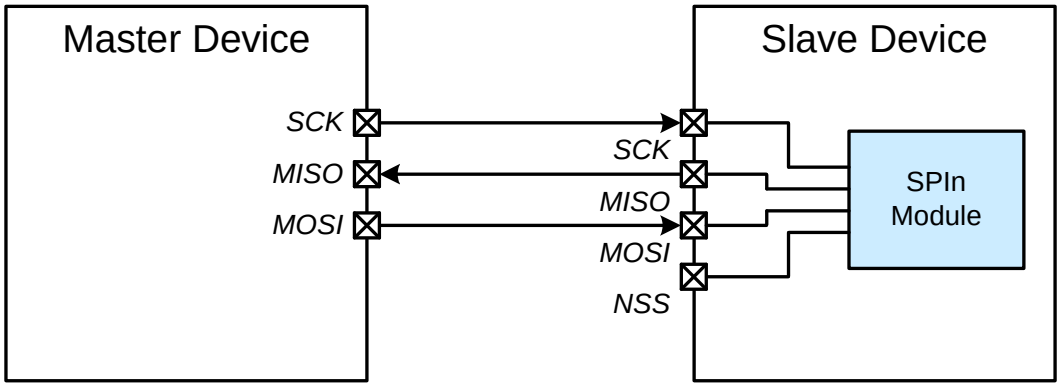


Figure 33.8. 3-Wire Single Slave Mode Connection Diagram

33.6.1.1. 4-Wire Slave Mode

In 4-wire slave mode (NSSMD = 2), the slave select (NSS) pin is configured as an input and should be connected to the NSS output of the master device. Any additional slaves should have their NSS pins be connected to one of the master's general purpose port pins controlled by firmware.

Asserting the NSS signal enables the SPI module, and deasserting NSS disables the module. The polarity of the NSS input can be set using the NSSPOL bit. The NSS signal must be asserted for at least two APB clocks before the first active edge of SCK for each byte transfer.

Figure 33.9 shows the connection diagram for 4-wire slave mode.

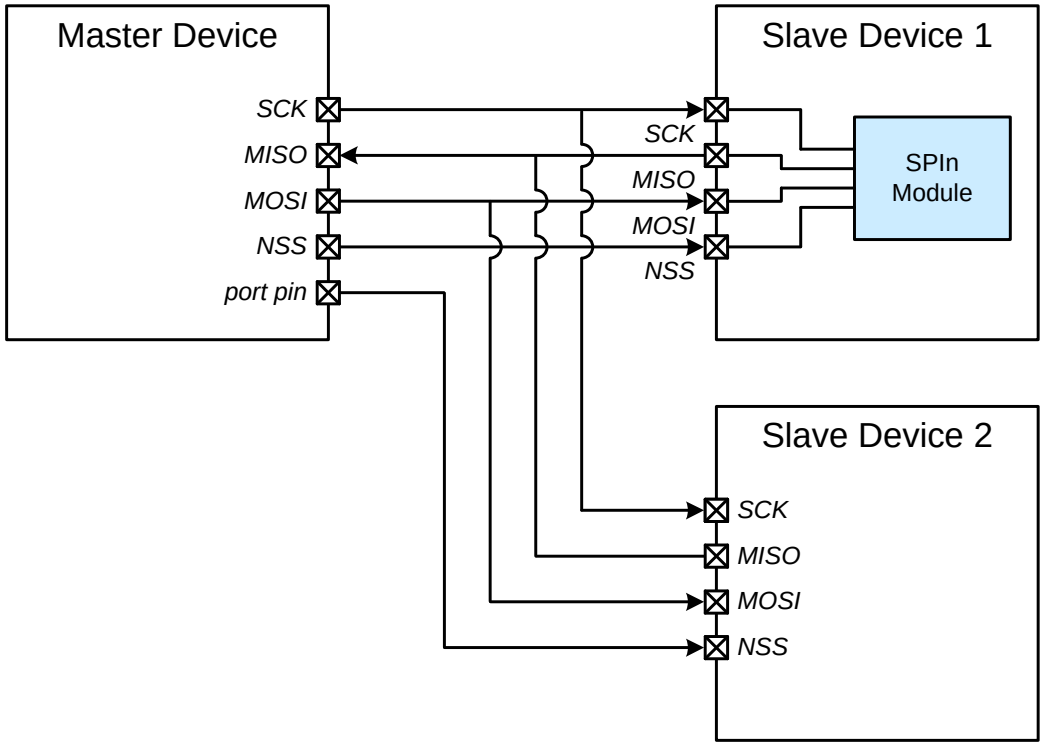


Figure 33.9. 4-Wire Slave Mode Connection Diagram

33.6.2. Slave Mode Data Transfer

In a SPI slave device, the master-generated clock input on the slave's SCK signal drives the data bytes received on MOSI and transmitted out through MISO. The hardware copies the data into the receive FIFO after receiving the number of bits specified by the DSIZE field. When the number of filled slots in the receive FIFO reaches or exceeds the programmed receive FIFO threshold (RFTH), hardware sets the receive FIFO read request interrupt (RFRQI) flag and generates an interrupt if RFRQIEN is set to 1. Firmware can read from the DATA register in right-justified bytes, half-words, or full words to pop data from the receive FIFO.

A slave device cannot initiate transfers and should pre-load the data into the transmit FIFO by writing to the DATA register before the master begins a transfer. If the shift register is empty, the hardware moves the first data in the transmit FIFO to the shift register in preparation for the data transfer.

33.7. Interrupts

The SPI module has several interrupt sources that can generate a SPI interrupt. All sources can be enabled or disabled by a corresponding interrupt enable bit except for the illegal FIFO access interrupts (TFILI and RFILI), which are always enabled.

33.7.1. Transmit Interrupts

The transmit FIFO write request (TFRQI) flag indicates that the FIFO has more room for data. Hardware sets this flag when the number of empty slots in the transmit FIFO is greater than or equal to the number of slots specified in the TFTH field. If DMA operations are enabled, a DMA request will be generated when hardware sets the flag. This flag can also generate an interrupt if the TFRQIEN bit is set to 1 until the number of empty slots in the transmit FIFO level drops below the TFTH setting.

33.7.2. Receive Interrupts

This receive FIFO read request interrupt (RFRQI) flag indicates that the receive FIFO has data available to be read by firmware. Hardware sets this bit when the number of filled slots in the receive FIFO is greater than or equal to the number of slots specified in the RFTH field. A DMA request will be generated if DMA is enabled (DMAEN = 1). If the receive FIFO read request interrupt is enabled (RFRQIEN = 1), an interrupt will also be generated until the number of filled slots in the receive FIFO drops below the RFTH setting.

33.7.3. Other Interrupts

The slave selected interrupt (SLVSELI) flag indicates when the slave select signal (NSS) is active. This flag represents a deglitched version of the NSS pin, rather than the instantaneous pin value. Firmware can read the instantaneous pin value by reading the NSSSTS bit. If the slave selected interrupt enable (SLVSELIEN) bit is also set, an interrupt will be generated. The interrupt will continuously trigger as long as the master asserts the NSS pin.

The shift register empty interrupt (SREI) and underrun interrupt (URI) flags indicate when the hardware has shifted all data out of the shift register and there's no data waiting in the transmit FIFO. The hardware will also generate an interrupt when SREI is set if SREIEN is set to 1 or when URI is set if URIEN is set to 1. These bits must be cleared by firmware.

33.7.4. Error Interrupts

The SPI module also has several error interrupts. The transmit FIFO overrun interrupt (TFORI) flag indicates that a transmit FIFO overrun occurred, and any new data written to the DATA register will not be placed in the transmit FIFO. The hardware will generate an interrupt if the transmit FIFO overrun interrupt enable (TFORIEN) bit is set. This flag must be cleared by firmware.

When the receive FIFO is full and new data arrives in the shift register, the hardware sets the receive FIFO overrun interrupt (RFORI) flag and ignores the data. This flag can also generate an interrupt if RFORIEN is set to 1. This flag must be cleared by firmware.

The mode fault interrupt (MDFI) flag indicates when a master mode collision occurs on the bus. The hardware sets this flag when NSS is low in multi-master mode (MSTEN = 1 and NSSMD = 1). An interrupt will be generated when MDFI sets, if enabled (MDFIEN= 1). This flag must be cleared by firmware.

The illegal transmit and receive FIFO access interrupts (TFILI and RFILI) are always enabled. These errors can occur when accesses to the DATA register are not right-justified, and an interrupt will be generated if either of these bits is set to 1. These flags must be cleared by firmware.

33.8. Debug Mode

Firmware can set the DBGMD bit to force the SPI module to halt on a debug breakpoint. Clearing the DBGMD bit forces the module to continue operating while the core halts in debug mode.

33.9. Module Reset

The SPI module can be reset by setting the RESET bit in the CONFIG register to 1. This bit resets the SPIEN and MSTEN bits in the CONFIG register, all bits in the CONTROL register, and flushes the receive and transmit FIFOs. The affected bits and fields are inaccessible during the reset process. Firmware should poll the RESET bit until hardware clears it, indicating the reset operation is complete.

33.10. DMA Configuration and Usage

When DMA is enabled (DMAEN = 1), the transmitter will generate a DMA request when the number of empty slots in the transmit FIFO is greater than or equal to the number of slots specified in the TFTH field. The DMA must service the request for data before the last data in the shift register finishes transmission, causing the device to set the underrun interrupt flag (URI). As soon as firmware loads data into the transmit FIFO, the shift register will pop the first data from the transmit FIFO and resume transmissions. Firmware can determine the number of filled slots in the transmit FIFO by reading the transmit FIFO counter (TFCNT) field. In the event of an error, the transmit FIFO flush (TFIFOFL) bit flushes all the data in the transmit FIFO.

During reception with DMA enabled, the receiver will generate a DMA request when the number of filled slots in the receive FIFO is greater than or equal to the number of slots specified in the RFTH field. If the receive FIFO is full, the DMA must service the DMA request before an overrun occurs. After an overrun condition, the hardware will not write the last data to the receive FIFO and will set the receive FIFO overrun interrupt (RFORI) flag, resulting in an interrupt if the RFORIEN bit is set to 1. At the end of a data transfer, a DMA request will not be generated if the number of filled slots in the receive FIFO is below the number of slots specified in the RFTH field. After clearing the SPIEN bit to disable the module, firmware can pop any remaining data from the receive FIFO using the DATA register until the receive FIFO counter (RFCNT) field reaches zero. The receive FIFO can also be emptied using the receive FIFO flush (RFIFOFL) bit.

Figure 33.10 shows the SPI DMA configuration.

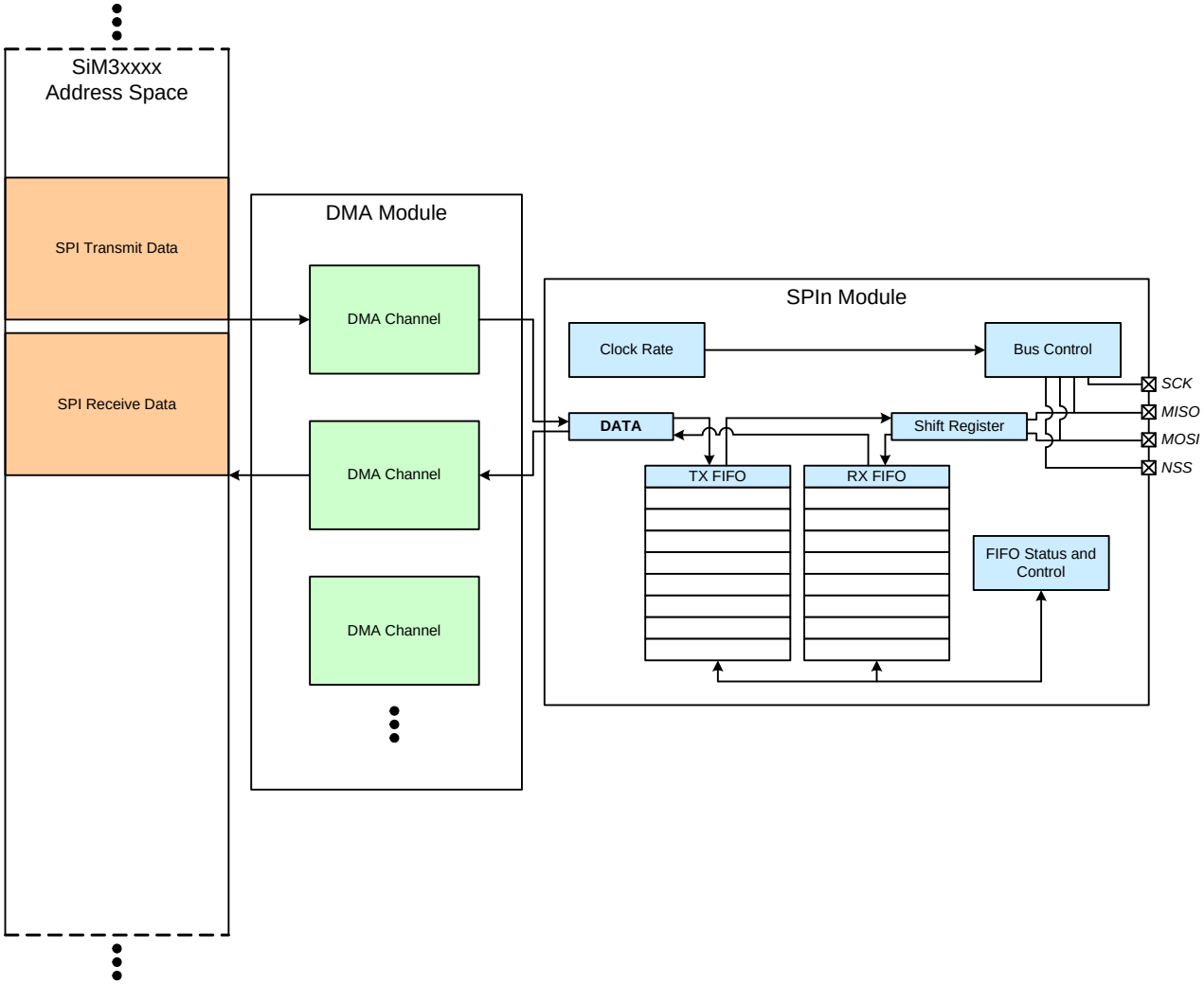


Figure 33.10. SPI DMA Configuration

33.11. SPI0, SPI1 and SPI2 Registers

This section contains the detailed register descriptions for SPI0, SPI1 and SPI2 registers.

Register 33.1. SPIn_DATA: Input/Output Data

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	DATA[31:16]															
Type	RW															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	DATA[15:0]															
Type	RW															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Register ALL Access Addresses

SPI0_DATA = 0x4000_4000
SPI1_DATA = 0x4000_5000
SPI2_DATA = 0x4000_6000

Table 33.1. SPIn_DATA Register Bit Descriptions

Bit	Name	Function
31:0	DATA	**Input/Output Data.** The DATA register is used to write to the transmit buffer and read from the receive buffer. Data can be in byte, half-word, or word format and must be right-justified. For every byte of data written, the TFCNT counter will increase. Reading will access data in the receive buffer. For every byte read, the RFCNT field will decrease.

Notes:

1. Reads of this register modify the state of hardware. Debug logic should take care when reading this register.

Register 33.2. SPIn_CONTROL: Module Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved							DBGMD	TFCNT				RFCNT			
Type	R							RW	R				R			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	BUSYF	NSSSTS	Reserved				TFILI	RFILI	SREI	URI	MDFI	SLVSELI	TFORI	TFRQI	RFORI	RFRQI
Type	R	R	R				RW	RW	R	RW	RW	R	RW	R	RW	R
Reset	0	1	0	0	0	0	0	0	1	0	0	0	0	1	0	0

Register ALL Access Addresses
SPI0_CONTROL = 0x4000_4010
SPI1_CONTROL = 0x4000_5010
SPI2_CONTROL = 0x4000_6010
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 33.2. SPIn_CONTROL Register Bit Descriptions

Bit	Name	Function
31:25	Reserved	Must write reset value.
24	DBGMD	**SPI Debug Mode.** 0: The SPI module will continue to operate while the core is halted in debug mode. 1: A debug breakpoint will cause the SPI module to halt.
23:20	TFCNT	**Transmit FIFO Counter.** Indicates the number of bytes in the transmit FIFO.
19:16	RFCNT	**Receive FIFO Counter.** Indicates the number of bytes in the receive FIFO.
15	BUSYF	**SPI Busy.** 0: The SPI is not busy and a transfer is not in progress. 1: The SPI is currently busy and a transfer is in progress.
14	NSSSTS	**NSS Instantaneous Pin Status.** This represents the instantaneous logic level at the NSS pin. 0: NSS is currently a logic low. 1: NSS is currently a logic high.

Notes:
1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 33.2. SPIn_CONTROL Register Bit Descriptions

Bit	Name	Function
13:10	Reserved	Must write reset value.
9	TFILI	**Illegal Transmit FIFO Access Interrupt Flag.** This bit indicates that an illegal write or read of the transmit FIFO has occurred. An interrupt will be generated if this bit is set to 1. This bit must be cleared by firmware.
8	RFILI	**Illegal Receive FIFO Access Interrupt Flag.** This bit indicates that an illegal write or read of the receive FIFO has occurred. An interrupt will be generated if this bit is set to 1. This bit must be cleared by firmware.
7	SREI	**Shift Register Empty Interrupt Flag.** Indicates that all the data has been transferred out of the shift register and there is no data waiting in the TX FIFO. If SREIEN is enabled, an interrupt will be generated. 0: There is data still present in the transmit FIFO. 1: All data has been transferred out of the shift register and there is no data waiting in the transmit FIFO.
6	URI	**Underrun Interrupt Flag.** This bit is set to 1 to indicate the end of a data transfer when the transmit FIFO and the shift register are empty. If URIEN is enabled, an interrupt will be generated. This bit must be cleared by firmware.
5	MDFI	**Mode Fault Interrupt Flag.** This bit is set to logic 1 by hardware when a master mode collision is detected (NSS is low, MSTEN = 1, and NSSMD [1:0] = 01b). An interrupt will be generated if MDFIEN is enabled. This bit must be cleared by firmware.
4	SLVSELI	**Slave Selected Interrupt Flag.** Indicates when the slave select signal (NSS) is active. This bit does not represent the instantaneous pin value, but is a deglitched version of the pin. An interrupt will be generated if SLVSELIEN is set to 1. 0: The slave select signal (NSS) is not active. 1: The slave select signal (NSS) is active.
3	TFORI	**Transmit FIFO Overrun Interrupt Flag.** Indicates that a transmit FIFO overrun has occurred and the new data will not be placed in the FIFO. If TFORIEN is enabled, an interrupt will be generated. This bit must be cleared by firmware.
2	TFRQI	**Transmit FIFO Write Request Interrupt Flag.** This flag indicates that the TX FIFO is at or below the number of bytes defined by the TFTH field. If DMA is enabled, a DMA request will be generated. If TFRQIEN is set to 1, an interrupt will be generated until the FIFO level fills above TFTH. 0: The TX FIFO has fewer bytes than the level defined by TFTH. 1: The TX FIFO has equal or more bytes than the level defined by TFTH.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 33.2. SPIn_CONTROL Register Bit Descriptions

Bit	Name	Function
1	RFORI	**Receive FIFO Overrun Interrupt Flag.** This flag Indicates that a receive FIFO overrun has occurred and the new data will not be placed in the FIFO. If RFORIEN is enabled, an interrupt will be generated. This bit must be cleared by firmware.
0	RFRQI	**Receive FIFO Read Request Interrupt Flag.** This flag indicates that the RX FIFO is at or above the number of bytes defined by the RFTH field. If DMA is enabled, a DMA request will be generated. An interrupt will be generated if RFRQIEN is set to 1 until the FIFO level drops below RFTH. 0: The RX FIFO has fewer bytes than the level defined by RFTH. 1: The RX FIFO has equal or more bytes than the level defined by RFTH.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Register 33.3. SPIn_CONFIG: Module Configuration

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	RESET	TFIFOFL	RFIFOFL	Reserved				DMAEN	DSIZE				TFTH		RFTH	
Type	RW	RW	RW	R				RW	RW				RW		RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	NSSMD		DDIRSEL	NSSPOL	CLKPHA	CLKPOL	MSTEN	SPIEN	SREIEN	URIEN	MDFIEN	SLVSELIEN	TFORIEN	TFRQIEN	RFORIEN	RFRQIEN
Type	RW		RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

SPI0_CONFIG = 0x4000_4020
SPI1_CONFIG = 0x4000_5020
SPI2_CONFIG = 0x4000_6020
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 33.3. SPIn_CONFIG Register Bit Descriptions

Bit	Name	Function
31	RESET	**Module Soft Reset.** Setting this bit to 1 resets the SPIEN and MSTEN bits in the CONFIG register, all bits in the CONTROL register, and flushes the RX and TX FIFOs. This bit is cleared by hardware when the operation completes. 0: SPI module is not in soft reset. 1: SPI module is in soft reset and some of the module bits cannot be accessed until this bit is cleared to 0 by hardware.
30	TFIFOFL	**Transmit FIFO Flush.** Setting this bit to 1 flushes the transmit FIFO of all data. This bit is cleared by hardware when the operation completes.
29	RFIFOFL	**Receive FIFO Flush.** Setting this bit to 1 flushes the receive FIFO of all data. This bit is cleared by hardware when the operation completes.
28:25	Reserved	Must write reset value.

Table 33.3. SPIn_CONFIG Register Bit Descriptions

Bit	Name	Function
24	DMAEN	**DMA Enable.** 0: Disable DMA requests. 1: Enable DMA requests when the transmit buffer is empty or the receive buffer is full.
23:20	DSIZE	**Data Size.** This field specifies the length of a data transfer. The SPI supports data transfers of any length between 1 and 16 bits. The data transfer size is equal to DSIZE + 1.
19:18	TFTH	**Transmit FIFO Threshold.** This field sets the trip point for transmit FIFO requests. 00: A DMA / TFRQ request asserts when ≥ 1 FIFO slot is empty. 01: A DMA / TFRQ request asserts when ≥ 2 FIFO slots are empty. 10: A DMA / TFRQ request asserts when ≥ 4 FIFO slots are empty. 11: A DMA / TFRQ request asserts when all FIFO slots are empty.
17:16	RFTH	**Receive FIFO Threshold.** This field sets the trip point for receive FIFO requests. 00: A DMA / RFRQ request asserts when ≥ 1 FIFO slot is filled. 01: A DMA / RFRQ request asserts when ≥ 2 FIFO slots are filled. 10: A DMA / RFRQ request asserts when ≥ 4 FIFO slots are filled. 11: A DMA / RFRQ request asserts when all FIFO slots are filled.
15:14	NSSMD	**Slave Select Mode.** This bit selects the behavior of the NSS pin. 00: 3-wire Slave or 3-wire Master. 01: 4-wire slave (NSS input). This setting can also be used for multi-master configurations. 10: 4-wire master with NSS low (NSS output). 11: 4-wire master with NSS high (NSS output).
13	DDIRSEL	**Data Direction Select.** 0: Data will be shifted MSB first. 1: Data will be shifted LSB first.
12	NSSPOL	**Slave Select Polarity Select.** 0: NSS is active low. 1: NSS is active high.
11	CLKPHA	**SPI Clock Phase.** 0: The first edge of SCK is the sample edge (center of data bit). 1: The first edge of SCK is the shift edge (edge of data bit).
10	CLKPOL	**SPI Clock Polarity.** 0: The SCK line is low in the idle state. 1: The SCK line is high in the idle state.
9	MSTEN	**Master Mode Enable.** 0: Operate in slave mode. 1: Operate in master mode.

Table 33.3. SPIn_CONFIG Register Bit Descriptions

Bit	Name	Function
8	SPIEN	**SPI Enable.** 0: Disable the SPI. 1: Enable the SPI.
7	SREIEN	**Shift Register Empty Interrupt Enable.** 0: Disable the shift register empty interrupt. 1: Enable the shift register empty interrupt.
6	URIEN	**Underrun Interrupt Enable.** 0: Disable the underrun interrupt. 1: Enable the underrun interrupt.
5	MDFIEN	**Mode Fault Interrupt Enable.** 0: Disable the mode fault interrupt. 1: Enable the mode fault interrupt.
4	SLVSELIEN	**Slave Selected Interrupt Enable.** 0: Disable the slave select interrupt. 1: Enable the slave select interrupt.
3	TFORIEN	**Transmit FIFO Overrun Interrupt Enable.** 0: Disable the transmit FIFO overrun interrupt. 1: Enable the transmit FIFO overrun interrupt.
2	TFRQIEN	**Transmit FIFO Write Request Interrupt Enable.** 0: Disable the transmit FIFO data request interrupt. 1: Enable the transmit FIFO data request interrupt.
1	RFORIEN	**Receive FIFO Overrun Interrupt Enable.** 0: Disable the receive FIFO overrun interrupt. 1: Enable the receive FIFO overrun interrupt.
0	RFRQIEN	**Receive FIFO Read Request Interrupt Enable.** 0: Disable the receive FIFO request interrupt. 1: Enable the receive FIFO request interrupt.

Register 33.4. SPIn_CLKRATE: Module Clock Rate Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	CLKDIV															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

SPI0_CLKRATE = 0x4000_4030
SPI1_CLKRATE = 0x4000_5030
SPI2_CLKRATE = 0x4000_6030

Table 33.4. SPIn_CLKRATE Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15:0	CLKDIV	**Clock Divider.** This field sets the frequency of the SPI clock output in master mode, according to the equation: $F_{SCK} = \frac{F_{APB}}{2 \times (CLKDIV + 1)}$

Register 33.5. SPIn_FSTATUS: FIFO Status

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	TFWPTR				TFRPTR				RFWPTR				RFRPTR			
Type	R				R				R				R			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

SPI0_FSTATUS = 0x4000_4040
SPI1_FSTATUS = 0x4000_5040
SPI2_FSTATUS = 0x4000_6040

Table 33.5. SPIn_FSTATUS Register Bit Descriptions

Bit	Name	Function
31:16	Reserved	Must write reset value.
15:12	TFWPTR	**Transmit FIFO Write Pointer.** This field indicates the next FIFO firmware will access on a transmit FIFO write.
11:8	TFRPTR	**Transmit FIFO Read Pointer.** This field indicates the next FIFO slot the SPI hardware will read in the transmit FIFO.
7:4	RFWPTR	**Receive FIFO Write Pointer.** This field indicates the next slot the SPI hardware will write in the receive FIFO.
3:0	RFRPTR	**Receive FIFO Read Pointer.** This field indicates the next FIFO slot firmware will access on a receive FIFO read.

33.12. SPIn Register Memory Map

Table 33.6. SPIn Memory Map

Register Name	SPIn_FSTATUS	SPIn_CLKRATE	SPIn_CONFIG	SPIn_CONTROL	SPIn_DATA																										
ALL Offset	0x40	0x30	0x20	0x10	0x0																										
Access Methods	ALL	ALL	ALL | SET | CLR	ALL | SET | CLR	ALL	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
	Reserved	Reserved	RESET	Reserved	DATA																										
			TFIFOFL																												
			RFIFOFL																												
			Reserved																												
			DMAEN	DBGMD																											
			DSIZE	TFCNT																											
			TFTH	RFCNT																											
			RFTH																												
	TFWPTR	CLKDIV	NSSMD	BUSYF																											
			DDIRSEL	NSSTS																											
			NSSPOL	Reserved																											
			CLKPHA																												
			CLKPOL																												
	TFRPTR		MSTEN	TFILI																											
			SPIEN	RFILI																											
	RFWPTR		SREIEN	SREI																											
			URIEN	URI																											
			MDFIEN	MDFI																											
			SLVSELIEN	SLVSELI																											
	RFRPTR		TFORIEN	TFORI																											
			TFRQIEN	TFRQI																											
			RFORIEN	RFORI																											
			RFRQIEN	RFRQI																											

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: SPI0 = 0x4000_4000, SPI1 = 0x4000_5000, SPI2 = 0x4000_6000

34. Sample Sync Generator (SSG0)

This section describes the Sample Sync Generator (SSG) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the SSG block, which is used by all device families covered in this document.

34.1. SSG Features

The Sample Sync Generator module has the following features:

- Connects multiple modules together to perform synchronized actions.
- Outputs a clock synchronized to the SAR clock of a master SARADC to any number of SARADC modules.
- Includes a phase counter, pulse generator, and up to four programmable outputs.

The SSG module includes a phase counter and a pulse generator. The phase counter is a 4-bit free-running counter clocked from a selected SARADC module clock. Counting up from zero, the phase counter marks sixteen equally-spaced events that can be used as conversion start triggers for any number of SARADC modules. The programmable pulse generator creates a 50% duty cycle pulse with a period of 16 phase counter ticks. Up to four outputs available to internal and external devices can be driven by the pulse generator with programmable polarity and a defined output setting when the pulse generator is stopped.

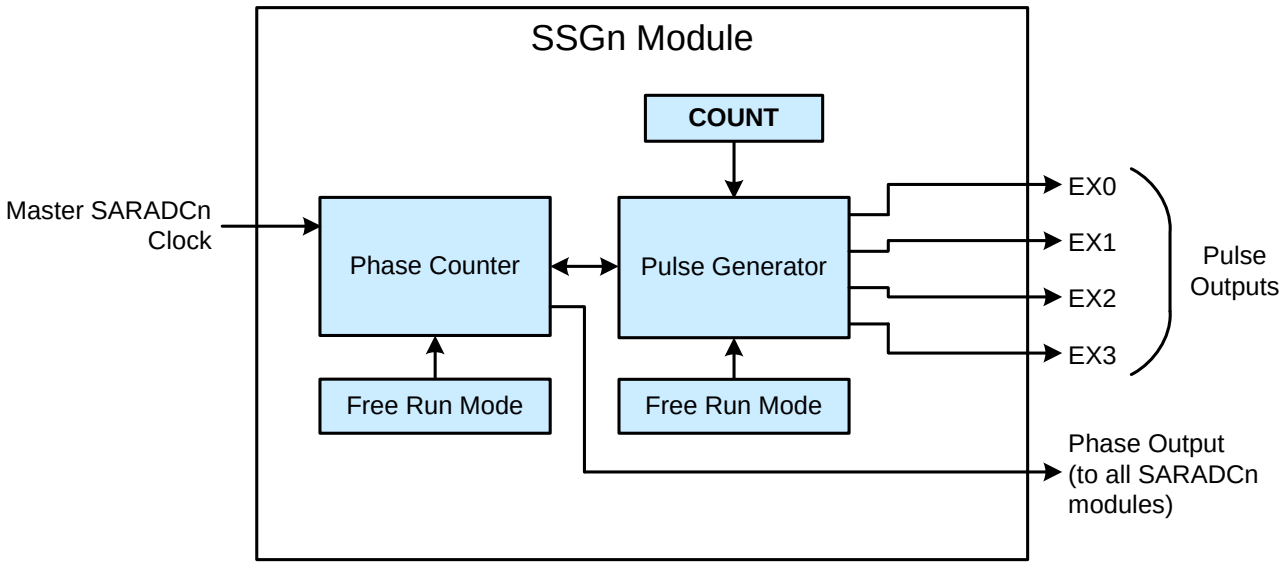


Figure 34.1. SSG Block Diagram

34.2. Phase Generator

The phase generator section of the SSG module is clocked by the SAR clock of a designated master SARADC block. This feeds into a 4-bit counter which generates 16 different phases from the master SAR clock. One cycle of the phase generator lasts 16 SAR clocks, and is equivalent to a single ADC conversion. Every SARADC in the device has the option of setting its conversion start trigger to any one of these 16 phases, allowing multiple ADCs to synchronize conversion timing with one another. The phase generator can be configured to run only when the pulse generator is operating, or it can be used in free running mode, where it will run continuously while the master ADC is enabled. The PUGFREN bit in the CONFIG register should be cleared to 0 for pulse generator mode, or set to 1 for free running mode.

34.3. Pulse Generator

The pulse generation logic in the SSG module consists of a programmable 12-bit counter and logic to produce four output signals. Each output signal can be independently configured to remain under software control or generate pulses, and with selectable polarity. The output signals can be used to trigger other hardware blocks on the device (such as an IDAC) or routed to external pins using the crossbar to synchronize ADC samples with external circuitry, such as linear image sensors.

34.3.1. Pulse Speed

The pulse generator logic will output one pulse per phase generator cycle (16 SAR clocks) by default. The duty cycle of the pulse is 50%, and toggles coincident with the SAR clock edges at the beginning of phase 0 and the beginning of phase 8. For applications where two ADCs are used in interleaved mode to increase throughput, the pulse generator logic can be configured to produce two pulses per phase generator cycle, or one pulse every 8 SAR clocks. The SSEL bit in the CONFIG register should be cleared to 0 for a single pulse per cycle, and set to 1 for two pulses per cycle. Figure 34.2 shows the timing of the pulse generator with both SSEL settings.

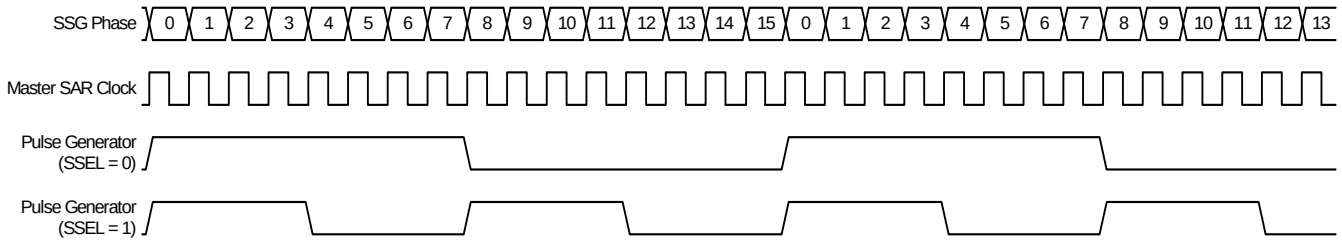


Figure 34.2. Pulse Generator Timing vs. SSEL Bit State

34.3.2. Output Signal Control

The four output signals (EX0, EX1, EX2 and EX3) can be individually enabled to output the generated pulse train, or an inverted version of the pulse train. When the pulse generator is idle, or when the output signals are not enabled to generate pulses, the state of the signals can be controlled directly by software.

Each output has a corresponding enable bit (EXnEN) in the CONTROL register, which enables generation of a pulse train at that output. Likewise, each output has a bit which allows the user to control the polarity of the output signal (EXnINVEN). In an idle state, or when the corresponding EXnEN bit is cleared to 0, the EXnINVEN bit directly controls the logic level at the pin. When EXnEN is set and pulses are actively being generated, the EXnINVEN bit has the effect of inverting the output pulse signal at the corresponding pin. Figure 34.3 shows the effect that the EXnEN and EXnINVEN bits have on the output waveforms.

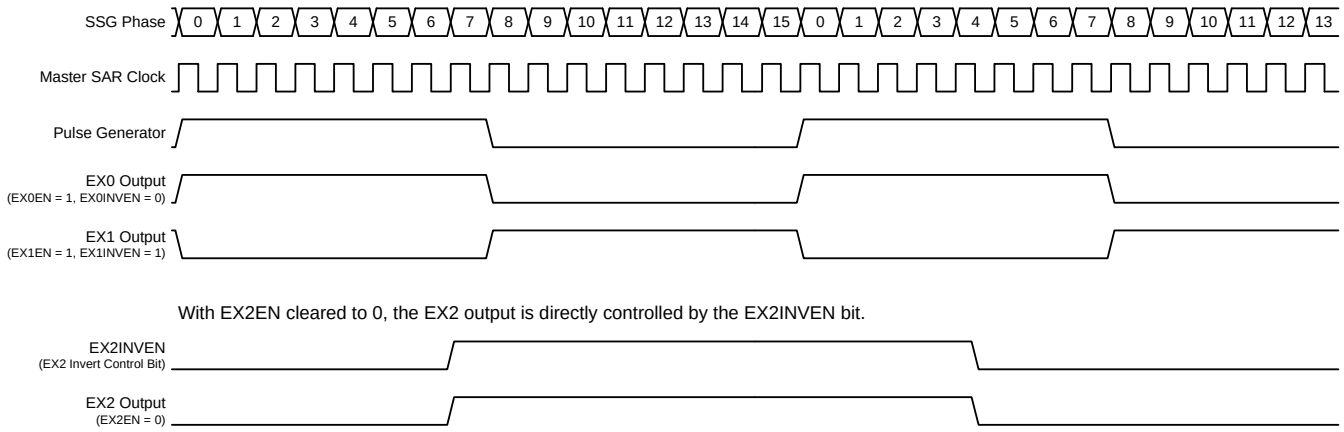


Figure 34.3. Timing Effects of the EXnEN and EXnNIVEN Bits (SSEL = 0)

34.3.3. Counter Operation

The pulse generator logic includes a 12-bit counter which can be used to operate the pulse generator for a specific number of phase generator cycles. The number of cycles is set using the COUNT field in the CONFIG register, where COUNT is equal to the number of phase generator cycles minus one (COUNT = 49 corresponds to 50 phase generator cycles). It is important to note that when the SSEL bit is 0, the number of pulses generated by the pulse generator will be equal to the selected number of phase generator cycles. If SSEL is 1, the number of pulses generated will be double the number of phase generator cycles. To configure the SSG pulse generation logic with the counter option:

1. Disable all EXnEN bits.
2. Clear the PUGFREN bit to 0 (to stop after a specified number of pulses).
3. Configure the EXnINVEN bits to the desired polarity.
4. Configure the COUNT field to the desired number of phase generator cycles –1.
5. Enable the pulse generator on one or more outputs using the EXnEN bits.

Pulse generation will begin within one phase generator cycle. The STATUS bit in the CONTROL register will read back 1 during the pulse generation, and return to 0 when complete. Pulse generation can be disabled at any time by clearing all of the EXnEN bits in the CONTROL register, which resets and stops the internal pulse counter. The COUNT field should only be updated when the pulse generator is idle. Figure 34.4 shows how pulses are generated in count mode, with the COUNT field set to 3 (4 pulses).

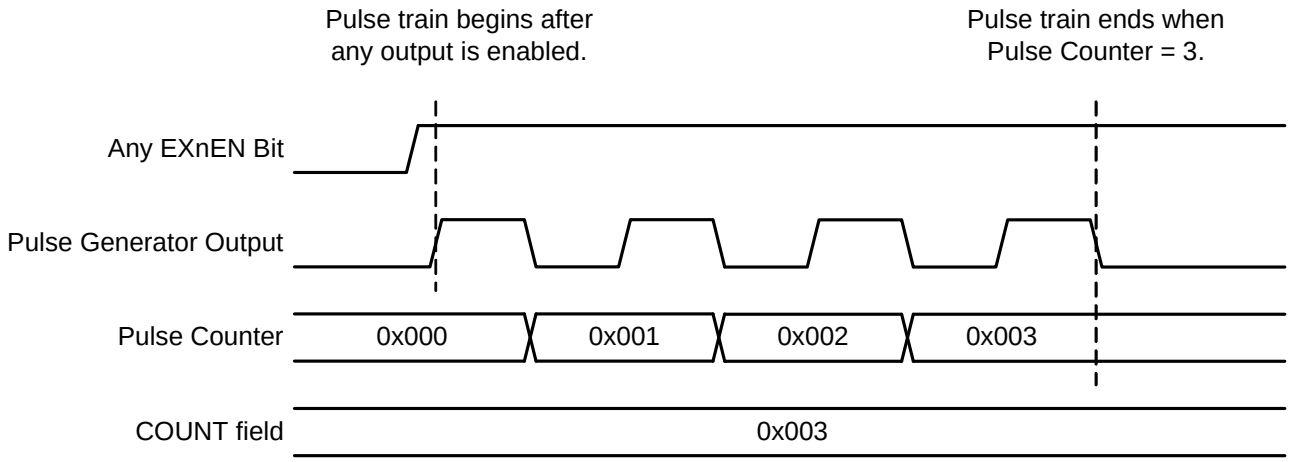


Figure 34.4. Pulse Counter Operation in Count Mode (PUGFREN = 0)

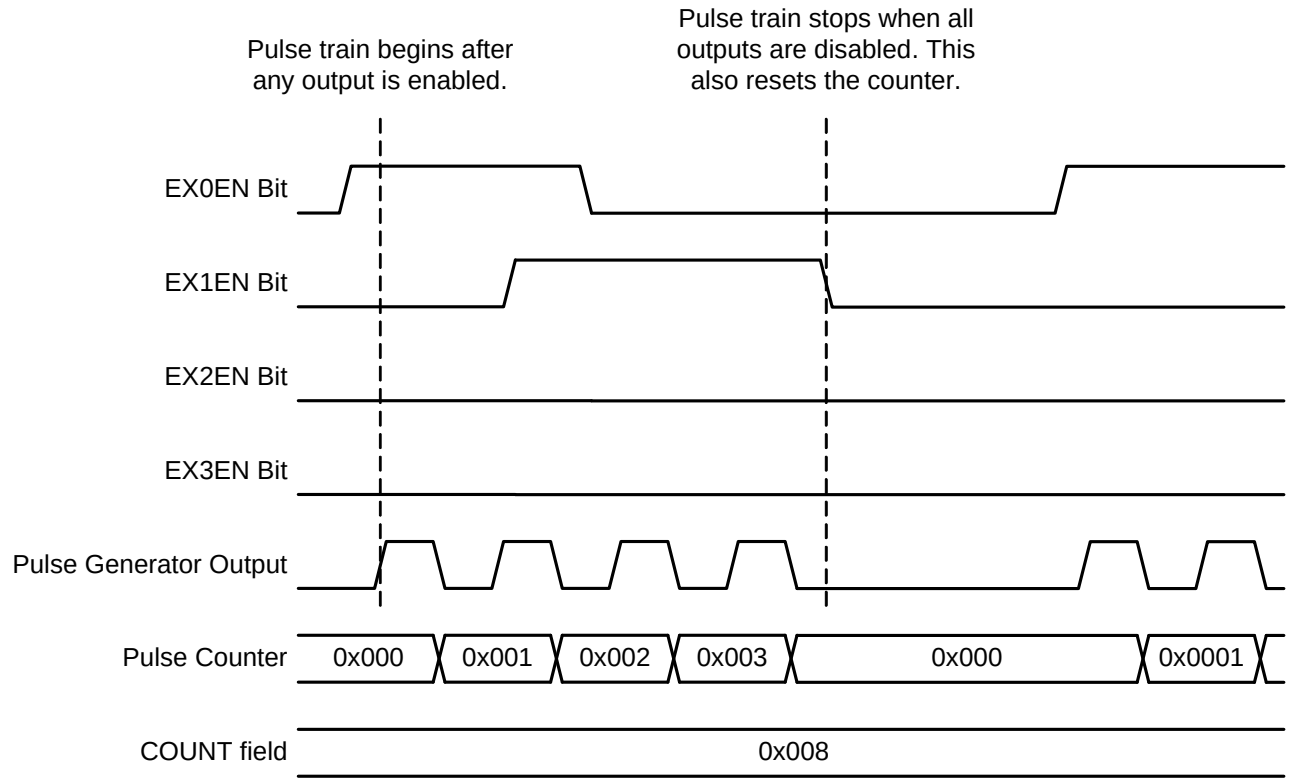


Figure 34.5. Pulse Counter Operation with Multiple Channels

34.3.4. Free Running Operation

In certain applications, it is necessary to run the pulse generator logic continuously. In such applications, the COUNT register is not needed, and the setup for the module is simpler. Setting the PUGFREN bit in the CONFIG register to 1 will enable the pulse generator in free running mode. If PUGFREN is set to 1 and any of the EXnEN bits are set to 1, pulse generation will occur. Figure 34.6 shows how pulses are generated in free running mode.

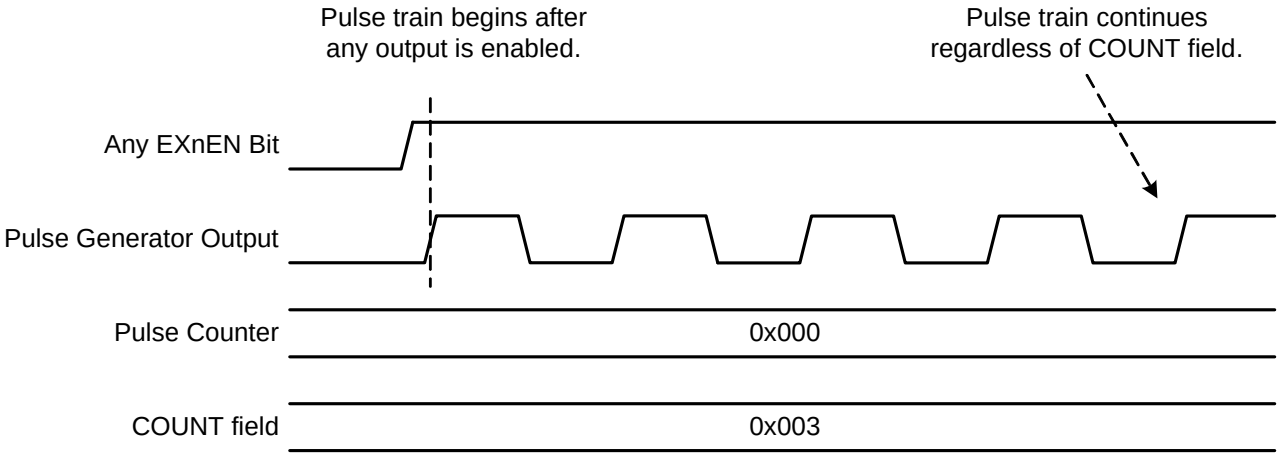


Figure 34.6. Pulse Counter Operation in Free Running Mode (PUGFREN = 1)

34.4. SSG0 Registers

This section contains the detailed register descriptions for SSG0 registers.

Register 34.1. SSG0_CONFIG: Module Configuration

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved	PUGFREN	PHGFREN	SSEL	COUNT											
Type	R	RW	RW	RW	RW											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address
SSG0_CONFIG = 0x4001_E000

Table 34.1. SSG0_CONFIG Register Bit Descriptions

Bit	Name	Function
31:15	Reserved	Must write reset value.
14	PUGFREN	**Pulse Generator Free-Run Enable.** 0: The COUNT field determines the number of pulses generated by the Pulse Generator. 1: The Pulse Generator always generates pulses regardless of COUNT unless all outputs are disabled (EX0EN, EX1EN, EX2EN, and EX3EN are all 0).
13	PHGFREN	**Phase Generator Free-Run Enable.** 0: The Phase Generator runs only when pulse generation occurs. 1: The Phase Generator runs when an ADC is enabled, regardless of the Pulse Generator settings.
12	SSEL	**Speed Select.** 0: The SSG module runs at normal speed, where each pulse and phase cycle consists of 16 ADC clocks. 1: The SSG module runs at double speed, where each pulse and phase cycle consists of 8 ADC clocks.

Table 34.1. SSG0_CONFIG Register Bit Descriptions

Bit	Name	Function
11:0	COUNT	**Pulse Generator Counter.** This field specifies the number of pulses to be generated by the Pulse Generator. If the SSG module is also serving as the convert-start source for an ADC, this field also represents the number of convert-start signals to be generated. If SSEL is cleared to 0, a 0 in this field represents one pulse and 0xFFF represents 4096 pulses. The number of generated pulses is given by: Number of Pulses = COUNT + 1. If SSEL is set to 1, a 0 in this field represents two pulses and 0xFFF represents 8192 pulses. The number of generated pulses is given by: Number of Pulses = 2 * (COUNT + 1). A read of this field always returns the last value written or 0 after a reset. This field should be written only if all of the SSG outputs are disabled (EX0EN, EX1EN, EX2EN, and EX3EN are all 0). After writing to the COUNT field and enabling one or more outputs, software should check the STATUS flag to determine when the current Pulse Generator operation is complete.

Register 34.2. SSG0_CONTROL: Module Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved							STATUS	EX3EN	EX2EN	EX1EN	EX0EN	EX3INVEN	EX2INVEN	EX1INVEN	EX0INVEN
Type	R							R	RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

SSG0_CONTROL = 0x4001_E010

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 34.2. SSG0_CONTROL Register Bit Descriptions

Bit	Name	Function
31:9	Reserved	Must write reset value.
8	STATUS	**SSG Module Status.** The SSG module will automatically become idle if all outputs are disabled (EX0EN, EX1EN, EX2EN, and EX3EN are all 0), or the COUNT field is zero and the Pulse Generator is not in free-run mode (PUGFREN is 0). 0: The SSG module is idle and the Pulse Generator is not operating. 1: The SSG module is active and the Pulse Generator is counting.
7	EX3EN	**Output 3 Enable.** Enabling a Pulse Generator output allows it to serve as an input to an internal module. To also enable the output on a pin, the output must be enabled on a Crossbar. 0: Disable the EX3 Pulse Generator output. 1: Enable the EX3 Pulse Generator output.
6	EX2EN	**Output 2 Enable.** Enabling a Pulse Generator output allows it to serve as an input to an internal module. To also enable the output on a pin, the output must be enabled on a Crossbar. 0: Disable the EX2 Pulse Generator output. 1: Enable the EX2 Pulse Generator output.
5	EX1EN	**Output 1 Enable.** Enabling a Pulse Generator output allows it to serve as an input to an internal module. To also enable the output on a pin, the output must be enabled on a Crossbar. 0: Disable the EX1 Pulse Generator output. 1: Enable the EX1 Pulse Generator output.

Table 34.2. SSG0_CONTROL Register Bit Descriptions

Bit	Name	Function
4	EX0EN	**Output 0 Enable.** Enabling a Pulse Generator output allows it to serve as an input to an internal module. To also enable the output on a pin, the output must be enabled on a Crossbar. 0: Disable the EX0 Pulse Generator output. 1: Enable the EX0 Pulse Generator output.
3	EX3INVEN	**Output 3 Invert Enable.** 0: Do not invert the Pulse Generator output on EX3. 1: Invert the Pulse Generator output on EX3.
2	EX2INVEN	**Output 2 Invert Enable.** 0: Do not invert the Pulse Generator output on EX2. 1: Invert the Pulse Generator output on EX2.
1	EX1INVEN	**Output 1 Invert Enable.** 0: Do not invert the Pulse Generator output on EX1. 1: Invert the Pulse Generator output on EX1.
0	EX0INVEN	**Output 0 Invert Enable.** 0: Do not invert the Pulse Generator output on EX0. 1: Invert the Pulse Generator output on EX0.

34.5. SSG0 Register Memory Map

Table 34.3. SSG0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
SSG0_CONFIG	0x4001_E000	ALL	Reserved																	PUGFREN	PHGFREN	SSEL	COUNT											
SSG0_CONTROL	0x4001_E010	ALL | SET | CLR	Reserved																							STATUS	EX3EN	EX2EN	EX1EN	EX0EN	EX3INVEN	EX2INVEN	EX1INVEN	EX0INVEN

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

35. Timers (TIMER0 and TIMER1)

This section describes the TIMER module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx

This section describes version “A” of the TIMER block, which is used by TIMER0 and TIMER1 on all device families covered in this document.

35.1. Timer Features

Each timer module (TIMER) is independent, and includes the following features:

- Operation as a single 32-bit or two independent 16-bit timers.
- Clocking options include the APB clock, the APB clock scaled using an 8-bit clock divider, the external oscillator, or falling edges on an external input pin (synchronized to the APB clock).
- Auto-reload functionality in both 32-bit and 16-bit modes.
- Up/Down count capability, controlled by an external input pin.
- Rising and falling edge capture modes on an external pin (TIMER0 and TIMER1).
- Low or high pulse capture modes.
- Frequency and duty cycle capture modes.
- One shot mode, triggered from EPCA Sync signal.
- Square wave output mode, which is capable of toggling an external pin at a given rate with 50% duty cycle.
- 32-bit or 16-bit pulse-width modulation modes.

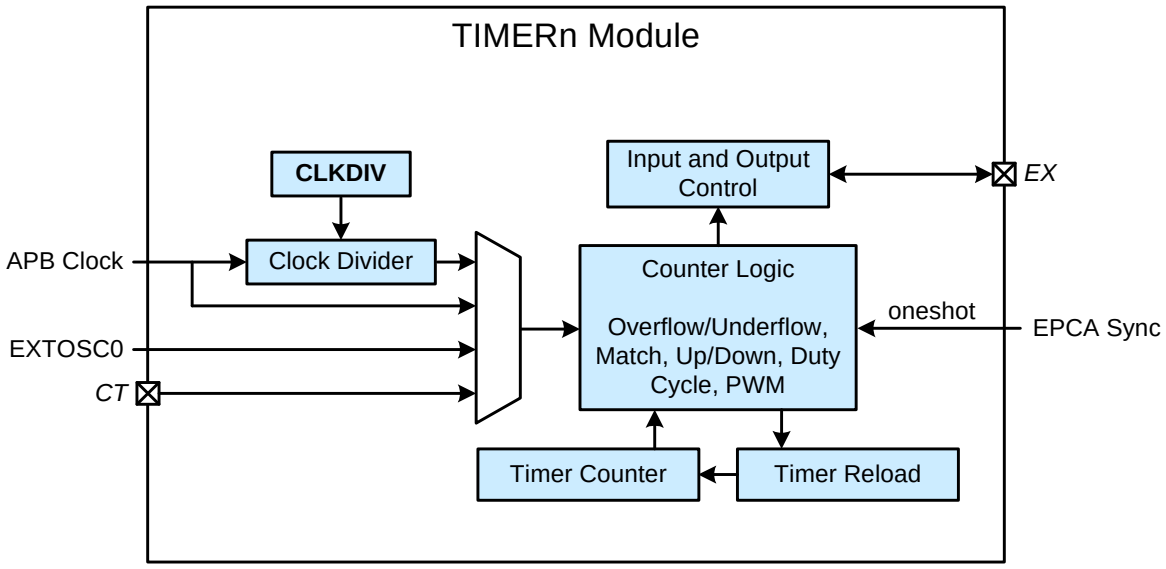


Figure 35.1. Timer Block Diagram

35.2. Clocking

The timer module can be clocked from several internal and external sources, selected by the HCLK and LCLK bits. The SPLITEN bit allows the two halves of the 32-bit timer to be split and clocked as individual 16-bit timers. When operating as a single 32-bit timer (SPLITEN = 0), HCLK controls the clock to the entire timer. When configured for split mode (SPLITEN = 1), HCLK controls the clock to the high-side 16-bit timer, and LCLK controls the clock to the low-side 16-bit timer.

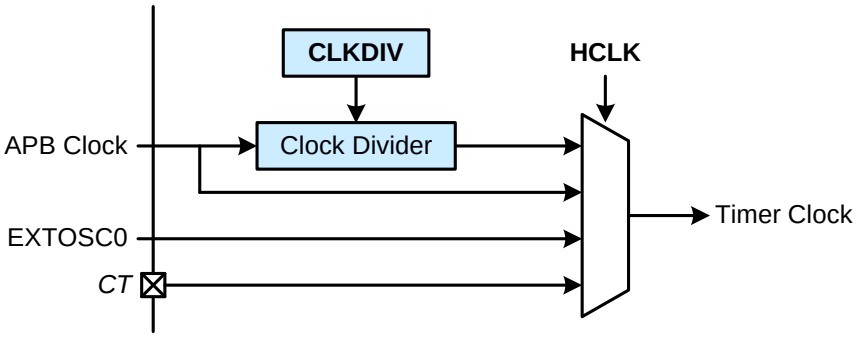


Figure 35.2. Clock Source Selection (SPLITEN = 0)

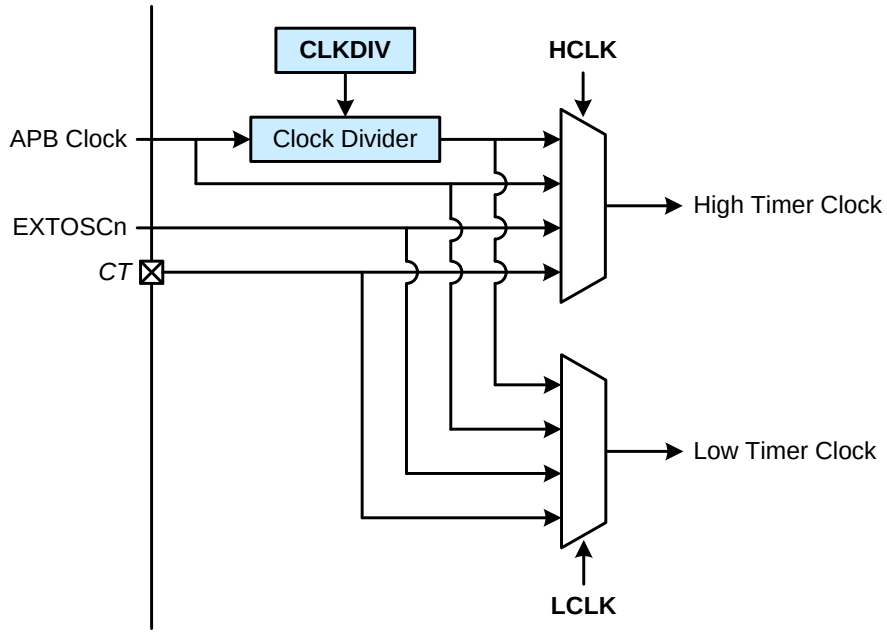


Figure 35.3. Clock Source Selection (SPLITEN = 1)

35.2.1. APB Clock

When the APB clock signal is selected as the timer clock source, the timer will clock from the APB clock source defined by the Clock Control module.

35.2.2. External Clock (EXTOSCn)

When the external clock is selected as the timer clock source, the timer will clock from the external clock source (EXTOSCn), regardless of the clock selection of the core. In this mode the external clock source is synchronized to the selected APB clock. In order to guarantee that the external clock transitions are recognized by the device, the external clock signal must be high or low for at least one APB clock. This limits the maximum frequency of an external clock in this mode to one-half the APB clock.

35.2.3. Clock Divider

The timer module includes an 8-bit configurable clock divider, allowing the timer to be clocked by a divided version of the APB clock. The clock divider consists of an 8-bit up counter and reload register and runs only when selected as the timer clock source. The counter value is incremented on every tick of the APB clock. It can be directly read and written through the CLKDIVCT field, and the reload value is stored in the CLKDIVRL field. On an overflow from 0xFF, the CLKDIVCT counter is automatically reloaded with the value in CLKDIVRL. The resulting timer clock rate is determined by Equation 35.1.

$$F_{TIMER} = \frac{F_{APB}}{(256 - CLKDIVRL)}$$

Equation 35.1. Clock Divider Output Clock Rate

35.2.4. External Pin (CT)

When the CT pin is selected as the timer clock source, the timer is incremented on falling edges of the pin. The CT pin is synchronized to the selected APB clock in this mode. In order to guarantee that the CT pin transitions are recognized by the device, CT must be at its new logic state for at least two APB clock cycles. Thus, the maximum frequency at which CT can toggle is one fourth the APB clock speed.

35.3. Configuring Timer Interrupts

There are two interrupt sources each for the high and low 16-bit words in the timer module. Each interrupt source can be individually enabled to generate a timer interrupt. The available interrupt flags are HOVFI, LOVFI, HEXI, and LEXI. The overflow flags (HOVFI and LOVFI) are set to 1 any time a positive overflow occurs (an increment when the timer is all 1's). The overflow interrupt enable bits (HOVFIEN and LOVFIEN) enable the corresponding overflow flags to be recognized by the interrupt controller. The extra flags (HEXI and LEXI) are set to 1 under conditions defined by the selected timer mode. The corresponding enable bits (HEXIEN and LEXIEN) enable the extra flags to be recognized by the interrupt controller. Refer to the timer mode descriptions in Section 35.5 for specifics on the overflow and extra flag triggers for each mode.

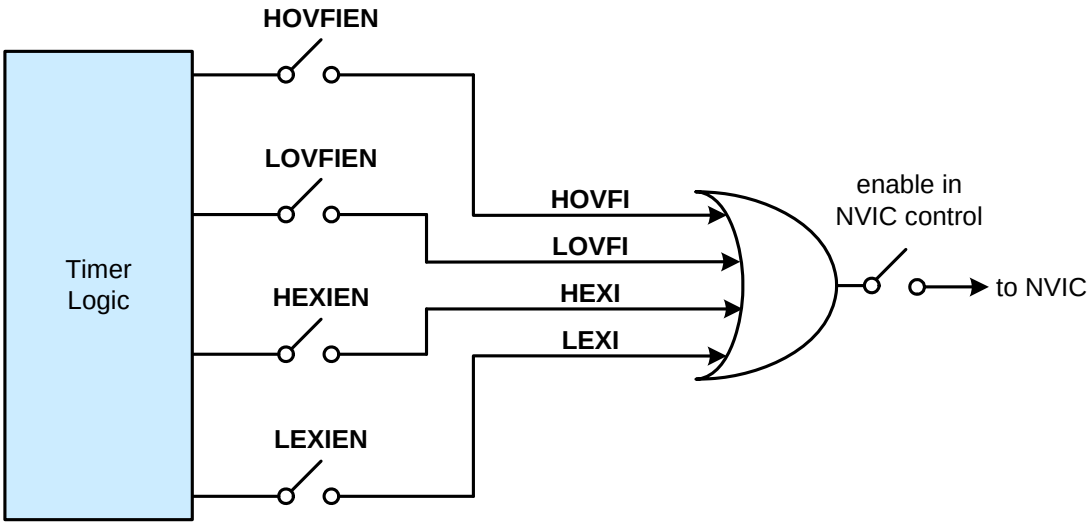


Figure 35.4. Timer Interrupt Configuration

35.4. Timer Synchronization

Each 16-bit timer includes a synchronization mechanism that allows all timers on the device to start and stop simultaneously. This synchronization is controlled using the master enable bits HMSTREN (high timer) and LMSTREN (low timer) and the master run control bit (MSTRUN) in the master TIMER module (TIMER0). When operating in 32-bit mode, HMSTREN controls the full timer. The MSTRUN bit affects all timer modules (0 through m) and is defined in the CONFIG register. If the master enable bit (HMSTREN, LMSTREN) is set for a timer, both MSTRUN in the master timer (TIMER0) and HRUN must be set to 1 for the timer to run.

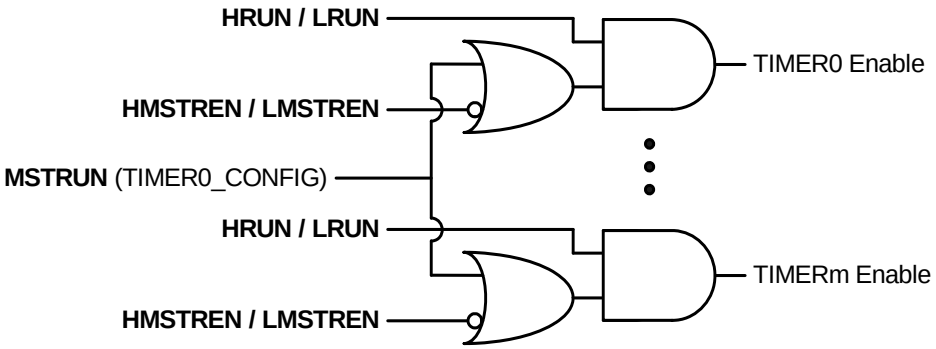


Figure 35.5. Timer Synchronization Block Diagram

35.5. Timer Modes

The timer mode selection is accomplished using the mode select bits in conjunction with the SPLITEN bit. All timer modes are available when operating as a 32-bit timer (SPLITEN = 0). Modes that use the EX pin as an input can be used with either the high or low timer in 16-bit split mode (SPLITEN = 1). In these modes, the two 16-bit timers share the EX input pin. Modes using the pin as an output are only available to the high timer in 16-bit split mode. Table 35.1 shows the different timer modes with their availability and external pin functions.

Table 35.1. Timer Mode Selection and Availability

Timer Mode	Mode Control Field	32-bit Mode (SPLITEN = 0)	16-bit Mode (SPLITEN = 1)	EX Pin
Auto-Reload	0000b	Available	Available for High and Low	Not Used
Up/Down	0001b	Available	Available for High and Low	Direction Input
Falling Edge Capture	0010b	Available	Available for High and Low	Capture Input
Rising Edge Capture	0011b	Available	Available for High and Low	Capture Input
Low Pulse Capture	0100b	Available	Available for High and Low	Capture Input
High Pulse Capture	0101b	Available	Available for High and Low	Capture Input
Duty Cycle Capture	0110b	Available	Available for High and Low	Capture Input
One Shot	0111b	Available	Available for High and Low	Not Used
Square Wave	1000b	Available	Available for High only	Square Wave Output
Pulse Width Modulation	1001b	Available	Available for High only	PWM Output

When operating in a 32-bit mode, the entire 32-bit COUNT register is used to store the current timer value. The 32-bit CAPTURE register function is defined by the mode in which the timer is operating. For split 16-bit mode, the corresponding high and low 16-bits of these registers are used independently by the high and low timer.

35.5.1. EX Input Synchronization

When used as an input in a timer mode, the EX signal is synchronized with the APB clock. The timer response delay to EX signal changes is 3-4 APB clocks, and the EX signal must remain high or low for two APB rising edges to be recognized by the timer. The timer event (capture, start or stop, or count direction change) associated with the EX signal will occur on the 4th rising APB edge after the EX signal change, as shown in Figure 35.6.

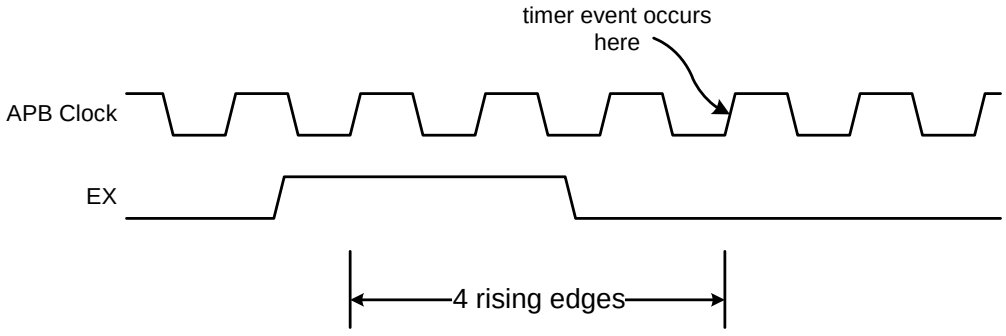


Figure 35.6. EX Signal Synchronization Timing

35.5.2. Auto-Reload Mode

In auto-reload mode, the timer counts up to all 1's. When an overflow occurs, the count register reloads from the value stored in the auto-reload register, and a timer overflow interrupt is generated.

If operating in 16-bit mode, Equation 35.2 describes the high timer's overflow rate. The low timer's overflow rate is based on the LCCR value instead of HCCR.

$$\text{Overflow Rate} = \frac{F_{TIMER}}{65536 - HCCR}$$

Equation 35.2. 16-bit Auto-Reload Overflow Rate

Equation 35.3 describes the timer's overflow rate if operating in 32-bit mode.

$$\text{Overflow Rate} = \frac{F_{TIMER}}{4294967296 - CAPTURE}$$

Equation 35.3. 32-bit Auto-Reload Overflow Rate

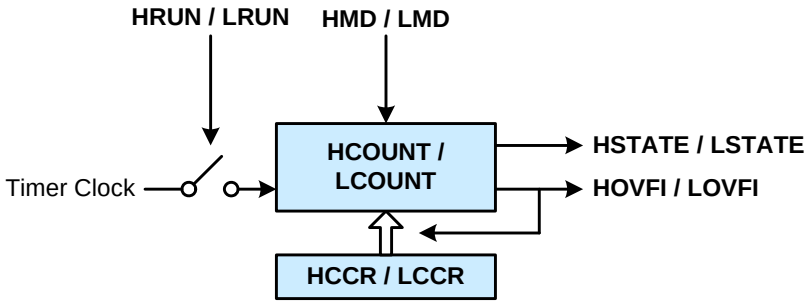


Figure 35.7. Auto-Reload Mode Block Diagram

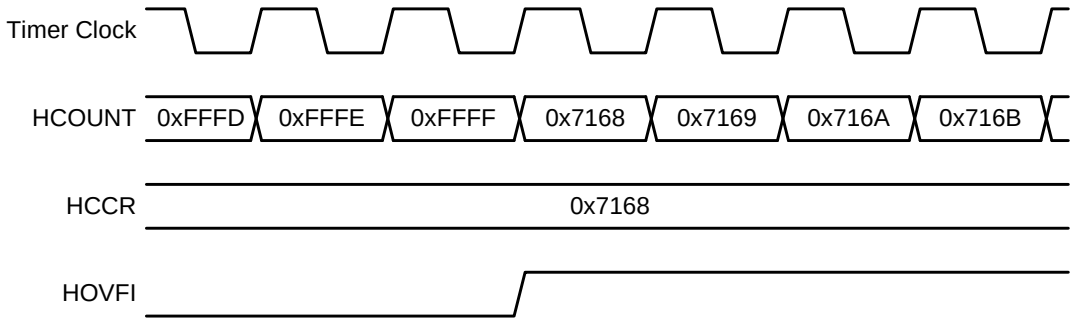


Figure 35.8. 16-bit Auto-Reload Mode Timing Diagram

35.5.3. Up/Down Mode

In up/down mode, the timer counts up or down, depending on the state of the external EX pin (high = count up, low = count down). If an overflow occurs, the count register is reloaded from the value stored in the auto-reload register, and a timer overflow is generated. If the timer counts down past the auto-reload value, the count register is reloaded with all 1's, and a timer underflow is generated (HEXI or LEXI set to 1).

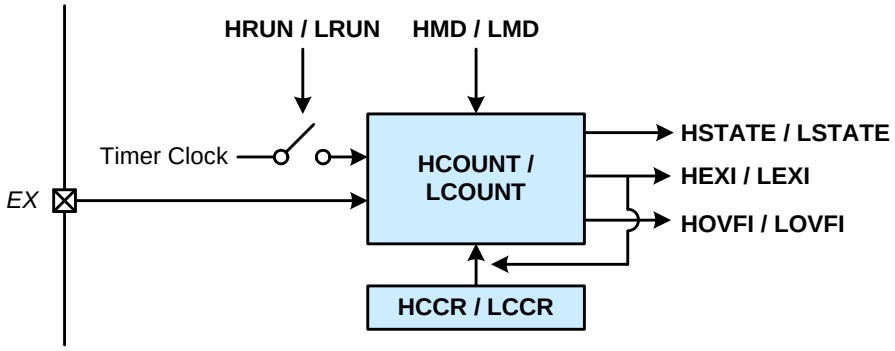
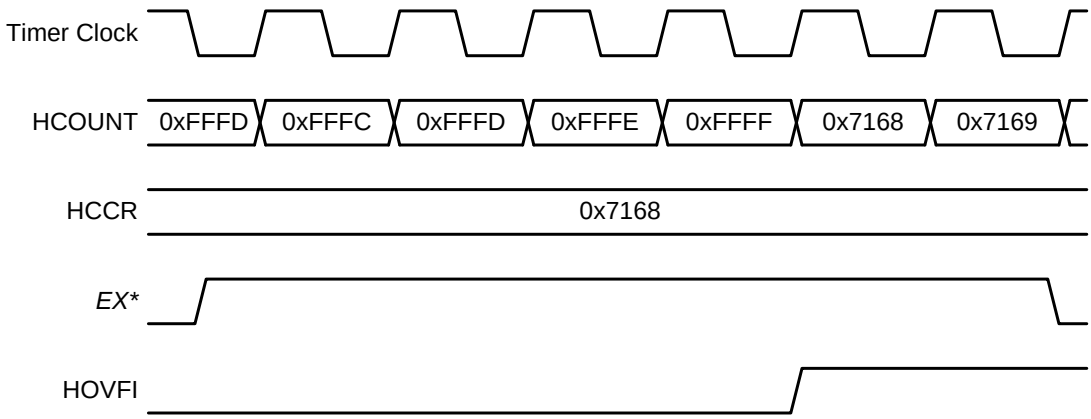


Figure 35.9. Up/Down Mode Block Diagram



*Note: The timer response delay to EX input signal changes is 3-4 APB clock cycles.

Figure 35.10. 16-bit Up/Down Mode Timing Diagram

35.5.4. Edge Capture Mode (Rising or Falling)

When operated in either falling or rising edge capture mode, the timer module counts up to all 1's. The overflow flag is set when an overflow occurs, and the timer reloads to all 0's. If the selected edge (rising or falling) occurs on the EX pin, the timer value is captured to the capture/compare/reload register and the HEXI or LEXI flag is set.

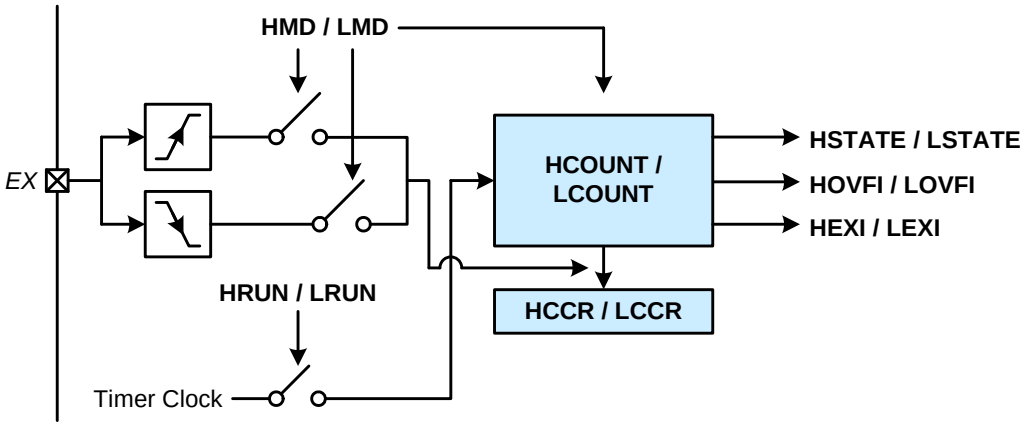
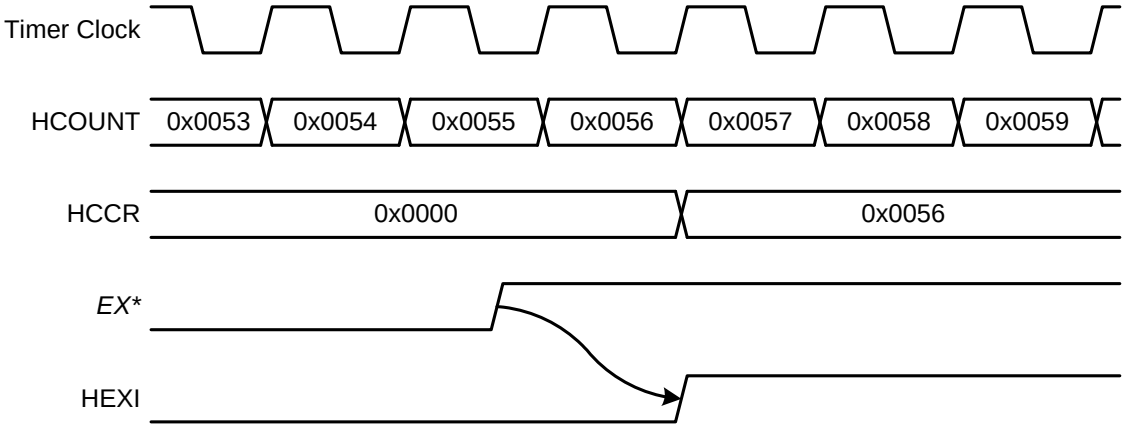


Figure 35.11. Edge Capture Mode Block Diagram



*Note: The timer response delay to EX input signal changes is 3-4 APB clock cycles.

Figure 35.12. 16-bit Edge Capture Mode Timing Diagram (Rising Edge Shown)

35.5.5. Pulse Capture Mode (High or Low)

When operating in low or high pulse capture modes, the timer looks for a sequence of two edges that define a pulse on the external EX pin. In low pulse capture mode, the first edge is the first falling edge the pin sees, while the second edge is the rising edge that follows. In high pulse capture mode, the first edge is a rising edge and the second edge is a falling edge.

The timer is configured to first wait for the first edge, and capture the current timer value to the capture/compare/reload register. The HSTATE or LSTATE bit is set high when this first edge occurs. After the capture occurs, the timer continues to run until the second edge occurs. The timer will halt on the second edge, effectively capturing the second edge position in the main timer counter. On the second edge, the HEXI or LEXI flag is set. The difference between the two captured edges can be used to determine the pulse width.

Note: The timer automatically clears its HRUN or LRUN bit when the second edge is detected.

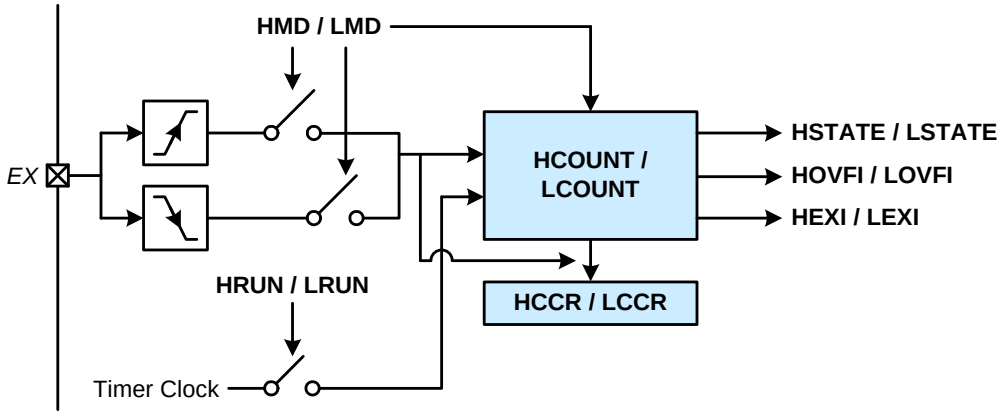
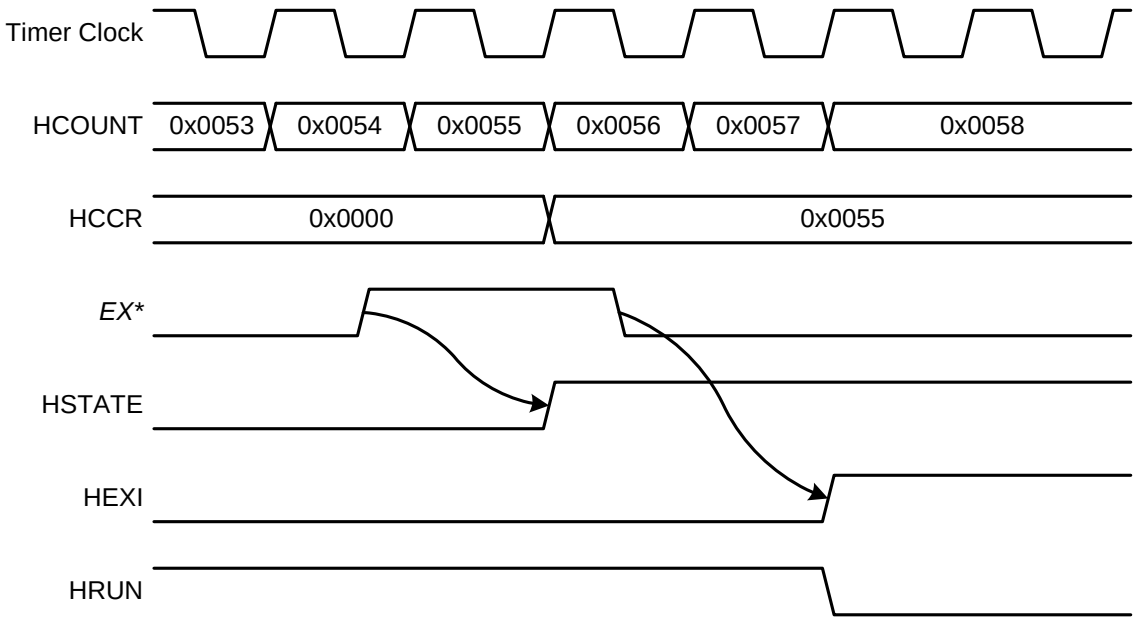


Figure 35.13. Pulse Capture Mode Block Diagram



*Note: The timer response delay to EX input signal changes is 3-4 APB clock cycles.

Figure 35.14. Pulse Capture Mode Timing Diagram (High Pulse Shown)

35.5.6. Duty Cycle Capture Mode

When operating in duty cycle capture mode, the timer initially sits idle (not clocking with HRUN and LRUN cleared to 0) and waits for a sequence of three edges on the EX pin. The timer begins running (by setting its HRUN or LRUN bit) on the first rising edge. The second edge, which is a falling edge, triggers a capture of the current timer value into the capture/compare/reload register. When the capture occurs, the HSTATE or LSTATE bit is set high. The timer halts by clearing its HRUN or LRUN bit on the third edge (another rising edge), effectively capturing the third edge position in the main timer counter. On the third edge, the HEXI or LEXI flag is also set.

The difference between the initial timer value and the final timer value is equivalent to the duration between two rising edges, and can be used to determine the frequency or period of the sampled signal. The difference between the capture value and the initial timer value is the high time of the signal, while the difference between the final timer value and the capture value is the low time of the signal. These can be used to determine the duty cycle of the sampled signal.

Note: The timer automatically sets and clears its HRUN or LRUN bits when edges occur on the EX pin.

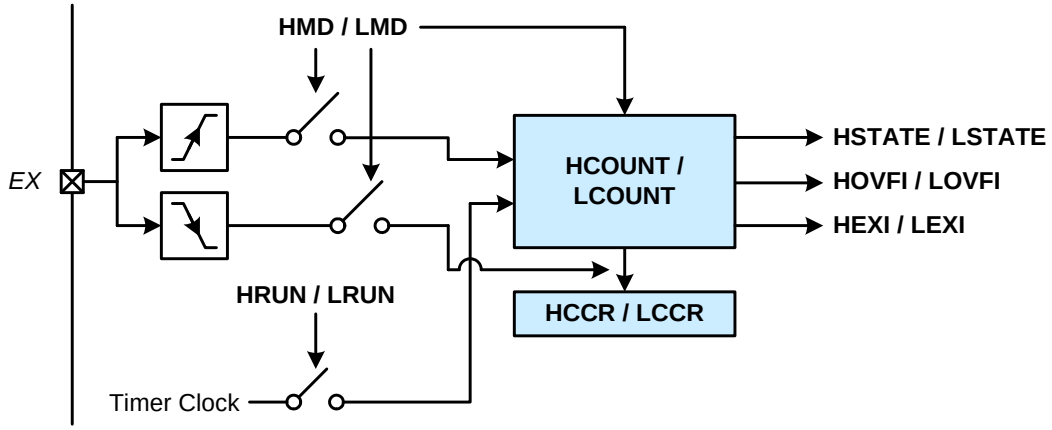
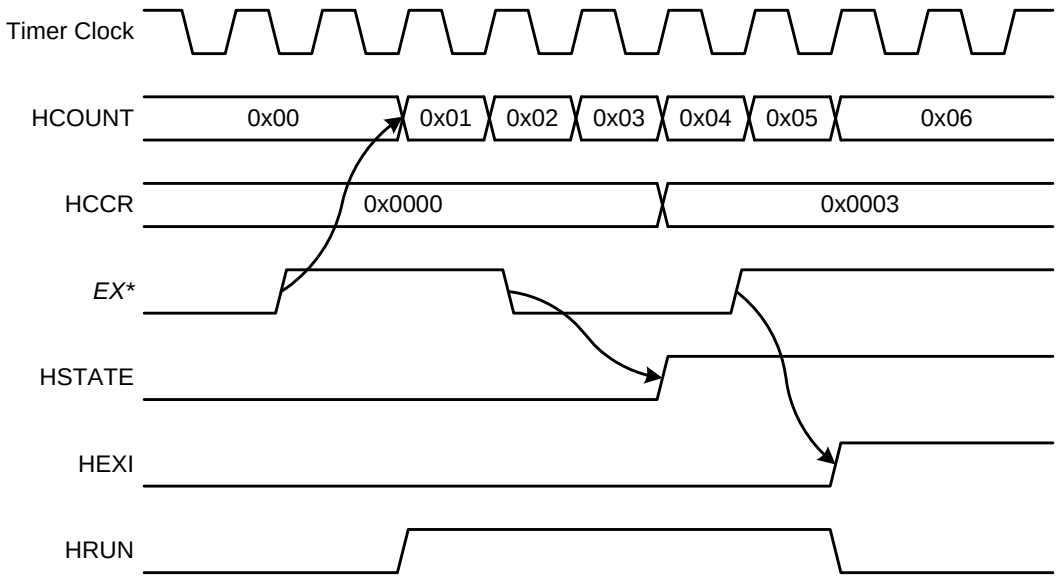


Figure 35.15. Duty Cycle Capture Mode Block Diagram



*Note: The timer response delay to EX input signal changes is 3-4 APB clock cycles.

Figure 35.16. Duty Cycle Capture Mode Timing Diagram

35.5.7. One Shot Mode

In one shot mode, the timer is initially idle (not clocking with HRUN and LRUN cleared to 0) and waiting for a pulse on its internal oneshot input. When a pulse is detected on the oneshot input, the timer begins running (by setting its HRUN or LRUN bit), and operates as though it were in auto-reload mode. The timer counts up to all 1's, and overflows. When the overflow occurs, the count register reloads from the value stored in the auto-reload register, and will immediately halt (by clearing HRUN or LRUN to 0). If enabled, a timer overflow interrupt is generated. The timer will remain idle until the next oneshot pulse occurs.

In the SiM3U1xx device family, all of the timer oneshot signals are tied to the EPCA Sync output. This allows the timers to implement a delayed trigger for different peripherals off the EPCA Sync event.

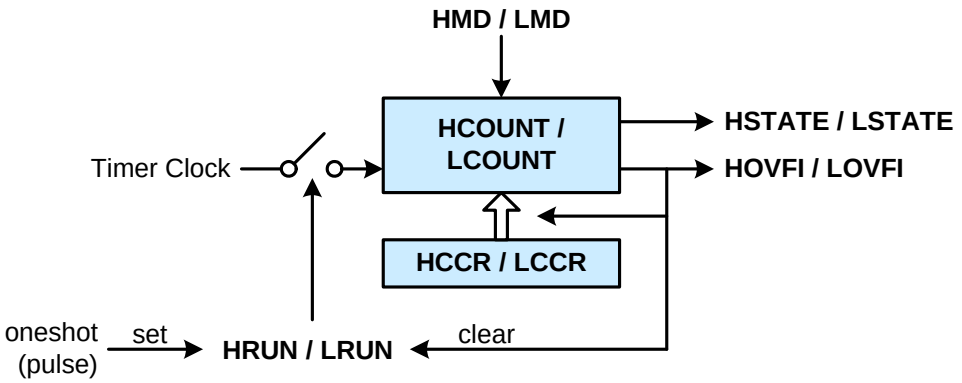


Figure 35.17. One Shot Mode Block Diagram

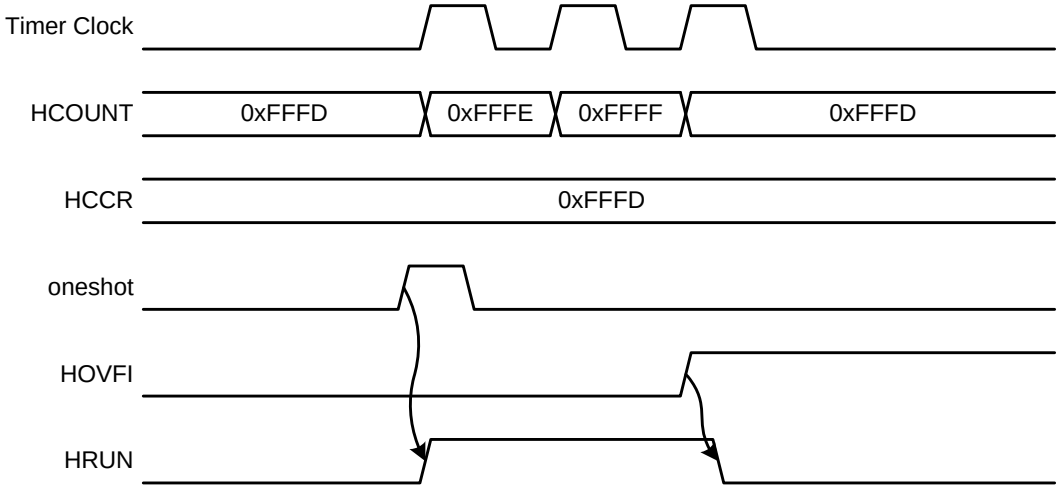


Figure 35.18. One Shot Mode Timing Diagram

35.5.8. Square Wave Output Mode

In square wave output mode, the timer operates the same as in auto-reload mode, except that the EX pin is toggled when an overflow event occurs. The state of the pin can be read or written using the HSTATE or LSTATE bit.

If operating in 16-bit mode, Equation 35.4 describes the high timer's output frequency in square wave output mode (the low timer cannot be used in 16-bit square wave output mode).

$$F_{OUT} = \frac{1}{2} \times \frac{F_{TIMER}}{65536 - HCCR}$$

Equation 35.4. 16-bit Square Wave Output Frequency

Equation 35.5 describes the timer's output frequency in square wave output mode if operating as a 32-bit timer.

$$F_{OUT} = \frac{1}{2} \times \frac{F_{TIMER}}{4294967296 - CAPTURE}$$

Equation 35.5. 32-bit Square Wave Output Frequency

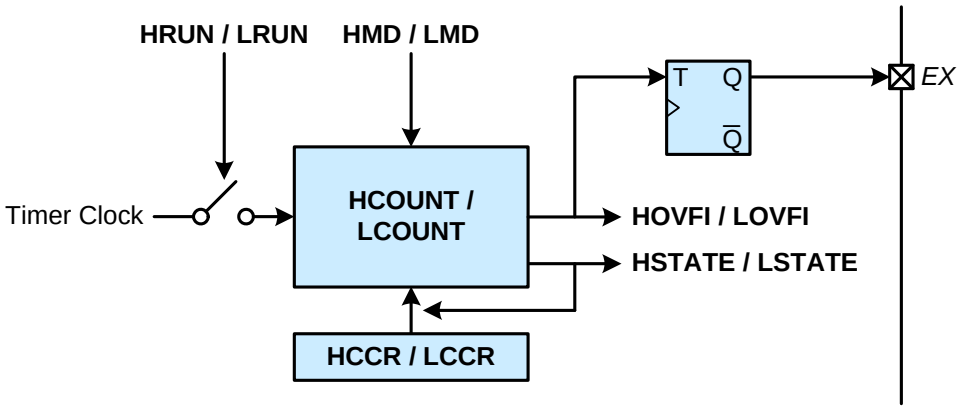


Figure 35.19. Square Wave Output Mode Block Diagram

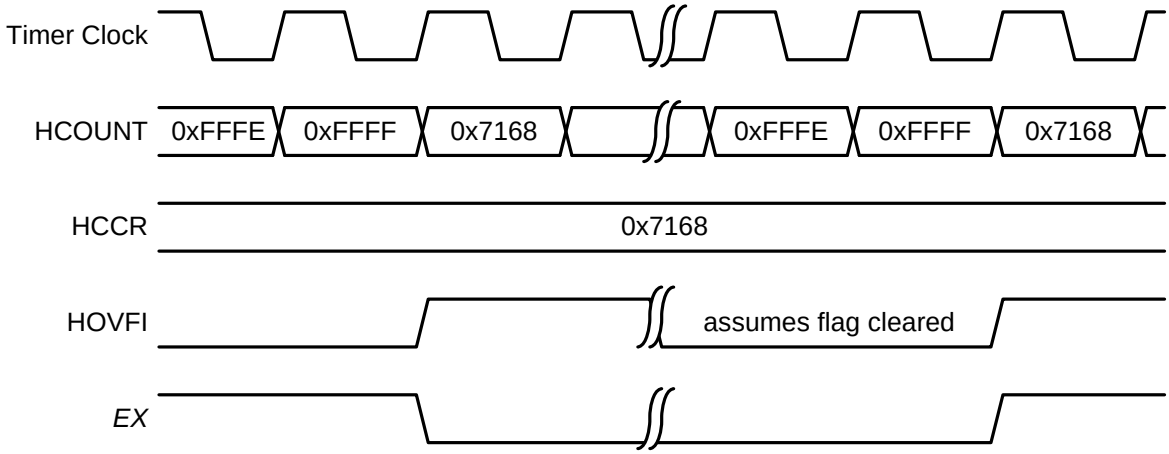


Figure 35.20. Square Wave Output Mode Timing Diagram

35.5.9. Pulse Width Modulation (PWM) Mode

In PWM mode, a PWM waveform is generated at the EX pin. The timer counts up to all 1's. Upon an overflow, the timer is loaded with all 0's, the pin is cleared to a low state, and the overflow flag is set. When the value of the count register is equal to the capture/compare/reload register, the EX pin is set to a high state and the HEXI or LEXI flag is set. It is possible to control the duty cycle of the waveform by writing to the HCCR or LCCR fields, and the EX pin can be read or written using the HSTATE or LSTATE bit.

If operating in 16-bit mode, Equation 35.6 describes the high timer's PWM duty cycle (the low timer cannot be used in 16-bit PWM mode).

$$\text{Duty Cycle} = \frac{65536 - \text{HCCR}}{65536}$$

Equation 35.6. 16-bit PWM Duty Cycle

Equation 35.7 describes the timer's output frequency in square wave output mode if operating as a 32-bit timer.

$$\text{Duty Cycle} = \frac{4294967296 - \text{CAPTURE}}{4294967296}$$

Equation 35.7. 32-bit PWM Duty Cycle

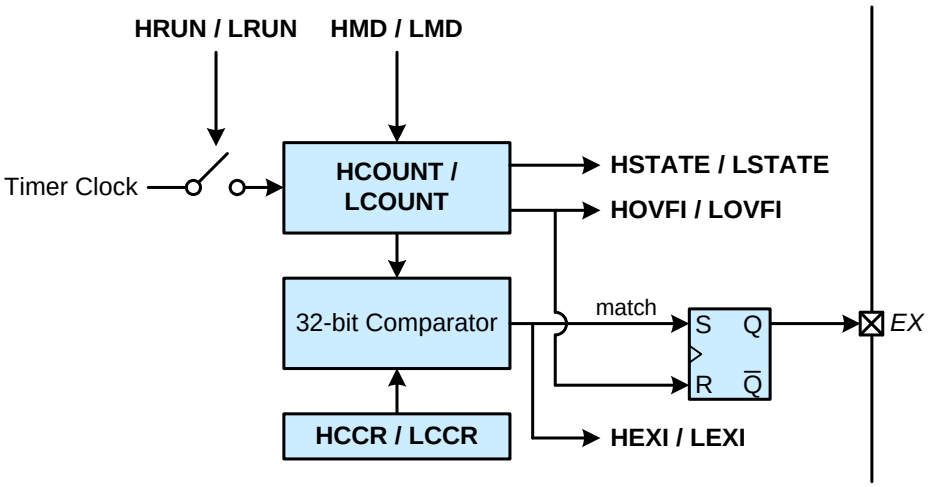


Figure 35.21. PWM Mode Block Diagram

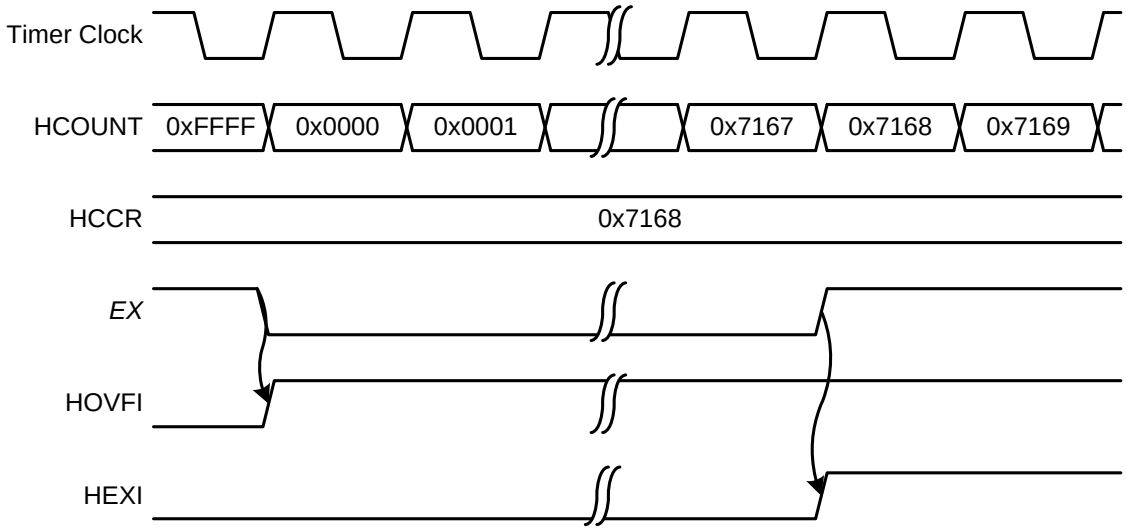


Figure 35.22. PWM Mode Timing Diagram

35.6. TIMER0 and TIMER1 Registers

This section contains the detailed register descriptions for TIMER0 and TIMER1 registers.

Register 35.1. TIMERn_CONFIG: High and Low Timer Configuration

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	HOVFI	HEXI	HRUN	HSTATE	HMD				HOVFIEN	HEXIEN	DBGMD	HMSTREN	MSTRUN	Reserved	HCLK	
Type	RW	RW	RW	RW	RW				RW	RW	RW	RW	RW	R	RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	LOVFI	LEXI	LRUN	LSTATE	Reserved	LMD			LOVFIEN	LEXIEN	SPLITEN	LMSTREN	Reserved		LCLK	
Type	RW	RW	RW	RW	R	RW			RW	RW	RW	RW	R		RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

TIMER0_CONFIG = 0x4001_4000
TIMER1_CONFIG = 0x4001_5000
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 35.2. TIMERn_CONFIG Register Bit Descriptions

Bit	Name	Function
31	HOVFI	**High Timer Overflow Interrupt Flag.** If split mode is enabled (SPLITEN = 1), this bit indicates the high 16-bit timer has wrapped or reloaded after reaching all 1's. If split mode is disabled (SPLITEN = 0), this value indicates the 32-bit timer has wrapped or reloaded after reaching all 1's. The timer module can set this bit in all modes. This bit must be cleared by software.
30	HEXI	**High Timer Extra Interrupt Flag.** This bit indicates the high 16-bit timer (or 32-bit timer if SPLITEN = 0) has been captured, reloaded with all 1's when counting down, or the timer matched the capture register in PWM mode. This interrupt flag can be set by the timer module in all modes except Auto-Reload and Toggle. This bit must be cleared by firmware.

Notes:
1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 35.2. TIMERn_CONFIG Register Bit Descriptions

Bit	Name	Function
29	HRUN	**High Run Control.** This bit is the run control for the high timer. When split mode is disabled (SPLITEN = 0), this bit also controls the low timer. 0: Stop the high timer or entire 32-bit timer. 1: The high timer runs if HMSTREN = 0 or MSTRUN (in Timer 0) = 1. The full 32-bit timer runs if split mode is disabled and (HMSTREN = 0 or MSTRUN = 1).
28	HSTATE	**High Multi Purpose State Indicator.** In PWM mode, the HSTATE bit is cleared each time the timer overflows from all 1's to all 0's. The HSTATE bit is then set when the timer increments while matching the capture register value. In Low/High/DC capture modes, the HSTATE bit indicates that the timer has captured due to the leading edge of TnEX. In all other modes, HSTATE is complemented each time the timer wraps or reloads when equal to all 1's (i.e. each time the HOVFI flag is also set). When in PWM or square wave output mode, the HSTATE bit defines the output state of the TnEX pin.
27:24	HMD	**High Timer Mode.** This field controls the mode of the high timer when split mode is enabled and the entire 32-bit timer when split mode is disabled. 0000: The high 16-bit timer or entire 32-bit timer is in Auto-Reload Mode. 0001: The high 16-bit timer or entire 32-bit timer is in Up/Down Count Mode. 0010: The high 16-bit timer or entire 32-bit timer is in Falling Edge Capture Mode. 0011: The high 16-bit timer or entire 32-bit timer is in Rising Edge Capture Mode. 0100: The high 16-bit timer or entire 32-bit timer is in Low Time Capture Mode. 0101: The high 16-bit timer or entire 32-bit timer is in High Time Capture Mode. 0110: The high 16-bit timer or entire 32-bit timer is in Duty Cycle Capture Mode. 0111: The high 16-bit timer or entire 32-bit timer is in Oneshot Mode. 1000: The high 16-bit timer or entire 32-bit timer is in Square Wave Output Mode. 1001: The high 16-bit timer or entire 32-bit timer is in PWM Mode. 1010-1111: Reserved.
23	HOVFIEN	**High Timer Overflow Interrupt Enable.** 0: The state of HOVFI does not affect the high timer interrupt. 1: A high timer interrupt request is generated if HOVFI is set to 1.
22	HEXIEN	**High Timer Extra Interrupt Enable.** 0: The state of the HEXI flag does not affect the high timer interrupt. 1: A high timer interrupt request is generated if HEXI is set to 1.
21	DBGMD	**Timer Debug Mode.** 0: The timer will continue to operate while the core is halted in debug mode. 1: A debug breakpoint will cause the Timer to halt.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 35.2. TIMERn_CONFIG Register Bit Descriptions

Bit	Name	Function
20	HMSTREN	**High Master Enable.** This bit determines whether the run master control (MSTRUN in the master TIMER0 module) must be set in addition to the timer's run bit (HRUN) before the high timer will run. This controls the entire 32-bit timer in 32-bit mode. 0: MSTRUN does not need to be set for the timer to run. 1: MSTRUN must be set for the timer to run.
19	MSTRUN	**Master Run Control.** MSTRUN is connected from the master timer to all slave timers. This bit is only used in the master timer (TIMER0). 0: Disable the master run control for all timers. 1: Enable the master run control for all timers.
18	Reserved	Must write reset value.
17:16	HCLK	**High Clock Source.** Selects the clock source of the high 16-bit timer if split mode is enabled (SPLITEN = 1) and the entire 32-bit timer if split mode is disabled (SPLITEN = 0). 00: Select the APB clock as the timer source. 01: Select the external oscillator clock as the timer source. The external oscillator must run slower than one-half the APB clock. 10: Select the dedicated 8-bit prescaler as the timer source. 11: Select falling edges of the CT signal as the timer clock source.
15	LOVFI	**Low Timer Overflow Interrupt.** The hardware sets this bit when the low 16-bit timer has wrapped or reloaded after reaching all 1's. This bit is set by the module regardless of the state of SPLITEN and can be set in all modes. This bit must be cleared by firmware.
14	LEXI	**Low Timer Extra Interrupt Flag.** This bit is set by hardware when the low 16-bit timer has been captured, reloaded with all 1's when counting down, or the timer matched the capture register in PWM mode. This interrupt flag can be set by hardware in all modes except Auto-Reload and Square Wave. This flag is not set by hardware when split mode is disabled (SPLITEN = 0). This bit must be cleared by firmware.
13	LRUN	**Run Control Low.** LRUN is the run control for the Low Timer in split mode. 0: Stop the low timer if split mode is enabled (SPLITEN = 1). 1: The low timer runs if split mode is enabled (SPLITEN = 1) and (LMSTREN = 0 or MSTRUN = 1 in Timer 0).
12	LSTATE	**Low Multi Purpose State Indicator.** In Low/High/DC capture modes, the LSTATE bit indicates that the low timer has captured due to the leading edge of TnEX. In all other modes, LSTATE is complemented each time the timer wraps or reloads when equal to all 1's (i.e. each time the LOVFI flag is also set).

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 35.2. TIMERn_CONFIG Register Bit Descriptions

Bit	Name	Function
11	Reserved	Must write reset value.
10:8	LMD	**Low Timer Mode.** This field controls the mode of the low timer when split mode is enabled (SPLITEN = 1). 000: The low timer is in Auto-Reload Mode. 001: The low timer is in Up/Down Count Mode. 010: The low timer is in Falling Edge Capture Mode. 011: The low timer is in Rising Edge Capture Mode. 100: The low timer is in Low Time Capture Mode. 101: The low timer is in High Time Capture Mode. 110: The low timer is in Duty Cycle Capture Mode. 111: The low timer is in Oneshot Mode.
7	LOVFIEN	**Low Timer Overflow Interrupt Enable.** 0: The state of LOVFI does not affect the low timer interrupt. 1: A low timer interrupt request is generated if LOVFI = 1.
6	LEXIEN	**Low Timer Extra Interrupt Enable.** 0: The state of the LEXI flag does not affect the low timer interrupt. 1: A low timer interrupt request is generated if LEXI is set to 1.
5	SPLITEN	**Split Mode Enable.** This bit selects between a single 32-bit timer or two 16-bit timers. 0: The timer operates as a single 32-bit timer controlled by the high timer fields. 1: The timer operates as two independent 16-bit timers.
4	LMSTREN	**Low Run Master Enable.** This bit determines whether the run master control (MSTRUN in the master TIMER0 module) must be set in addition to the timer's run bit (LRUN) before the low timer will run. 0: MSTRUN does not need to be set for the low timer to run. 1: MSTRUN must be set for the low timer to run.
3:2	Reserved	Must write reset value.
1:0	LCLK	**Low Clock Source.** Select the clock source for the low 16-bit timer if SPLITEN is set to 1. If SPLITEN is cleared to 0, the low timer source is controlled by HCLK. 00: Select the APB clock as the timer source. 01: Select the external oscillator clock as the timer source. The external oscillator must run slower than one-half the APB clock. 10: Select the dedicated 8-bit prescaler as the timer source. 11: Select falling edges of the CT signal as the timer clock source.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Register 35.2. TIMERn_CLKDIV: Module Clock Divider Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved								CLKDIVCT							
Type	R								RW							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved								CLKDIVRL							
Type	R								RW							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

TIMER0_CLKDIV = 0x4001_4010
TIMER1_CLKDIV = 0x4001_5010

Table 35.3. TIMERn_CLKDIV Register Bit Descriptions

Bit	Name	Function
31:24	Reserved	Must write reset value.
23:16	CLKDIVCT	**Clock Divider Counter.** This field provides direct access to the 8-bit prescaler counter. The counter repeatedly counts from the value in CLKDIVRL to all 1's.
15:8	Reserved	Must write reset value.
7:0	CLKDIVRL	**Clock Divider Reload Value.** This field holds the reload value for the 8-bit clock divider. The clock divider output frequency is given by: $F_{TIMER} = \frac{F_{APB}}{(256 - CLKDIVRL)}$

Register 35.3. TIMERn_COUNT: Timer Value

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	HCOUNT															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	LCOUNT															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

TIMER0_COUNT = 0x4001_4020
TIMER1_COUNT = 0x4001_5020

Table 35.4. TIMERn_COUNT Register Bit Descriptions

Bit	Name	Function
31:16	HCOUNT	**High Timer Count.** This field holds the high timer state.
15:0	LCOUNT	**Low Timer Count.** This field holds the low timer state.

Register 35.4. TIMERn_CAPTURE: Timer Capture/Reload Value

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	HCCR															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	LCCR															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

TIMER0_CAPTURE = 0x4001_4030
TIMER1_CAPTURE = 0x4001_5030

Table 35.5. TIMERn_CAPTURE Register Bit Descriptions

Bit	Name	Function
31:16	HCCR	**High Timer Capture/Reload.** This field holds the high timer capture or reload value.
15:0	LCCR	**Low Timer Capture/Reload.** This field holds the low timer capture or reload value.

35.7. TIMERn Register Memory Map

Table 35.6. TIMERn Memory Map

Register Name	ALL Offset	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
TIMERn_CONFIG	0x0	ALL | SET | CLR	HOVFI	HEXI	HRUN	HSTATE	HMD				HOVFIEN	HEXIEN	DBGMD	HMSTREN	MSTRUN	Reserved	HCLK		LOVFI	LEXI	LRUN	LSTATE	Reserved	LMD			LOVFIEN	LEXIEN	SPLITEN	LMSTREN	Reserved		LCLK	
TIMERn_CLKDIV	0x10	ALL	Reserved								CLKDIVCT								Reserved								CLKDIVRL							
TIMERn_COUNT	0x20	ALL	HCOUNT																LCOUNT															
TIMERn_CAPTURE	0x30	ALL	HCCR																LCCR															

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: TIMER0 = 0x4001_4000, TIMER1 = 0x4001_5000

36. Universal Synchronous/Asynchronous Receiver/Transmitter (USART0 and USART1)

This section describes the USART module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the USART block, which is used by both USART0 and USART1 on all device families covered in this document.

36.1. USART Features

The USART module includes the following features:

- Independent transmitter and receiver configurations with separate 16-bit baud rate generators.
- Synchronous or asynchronous transmissions and receptions.
- Clock master or slave operation with programmable polarity and edge controls.
- Up to 5 Mbaud (synchronous or asynchronous, TX or RX, and master or slave) or 1 Mbaud Smartcard (TX or RX).
- Individual enables for generated clocks during start, stop, and idle states.
- Internal transmit and receive FIFOs with flush capability and support for byte, half-word, and word reads and writes.
- Data bit lengths from 5 to 9 bits.
- Programmable inter-packet transmit delays.
- Auto-baud detection with support for the LIN SYNC byte.
- Automatic parity generation (with enable).
- Automatic start and stop generation (with separate enables).
- Transmit and receive hardware flow-control.
- Independent inversion correction for TX, RX, RTS, and CTS signals.
- IrDA modulation and demodulation with programmable pulse widths.
- Smartcard ACK/NACK support.
- Parity error, frame error, overrun, and underrun detection.
- Multi-master and half-duplex support.
- Multiple loop-back modes supported.
- Multi-processor communications support.

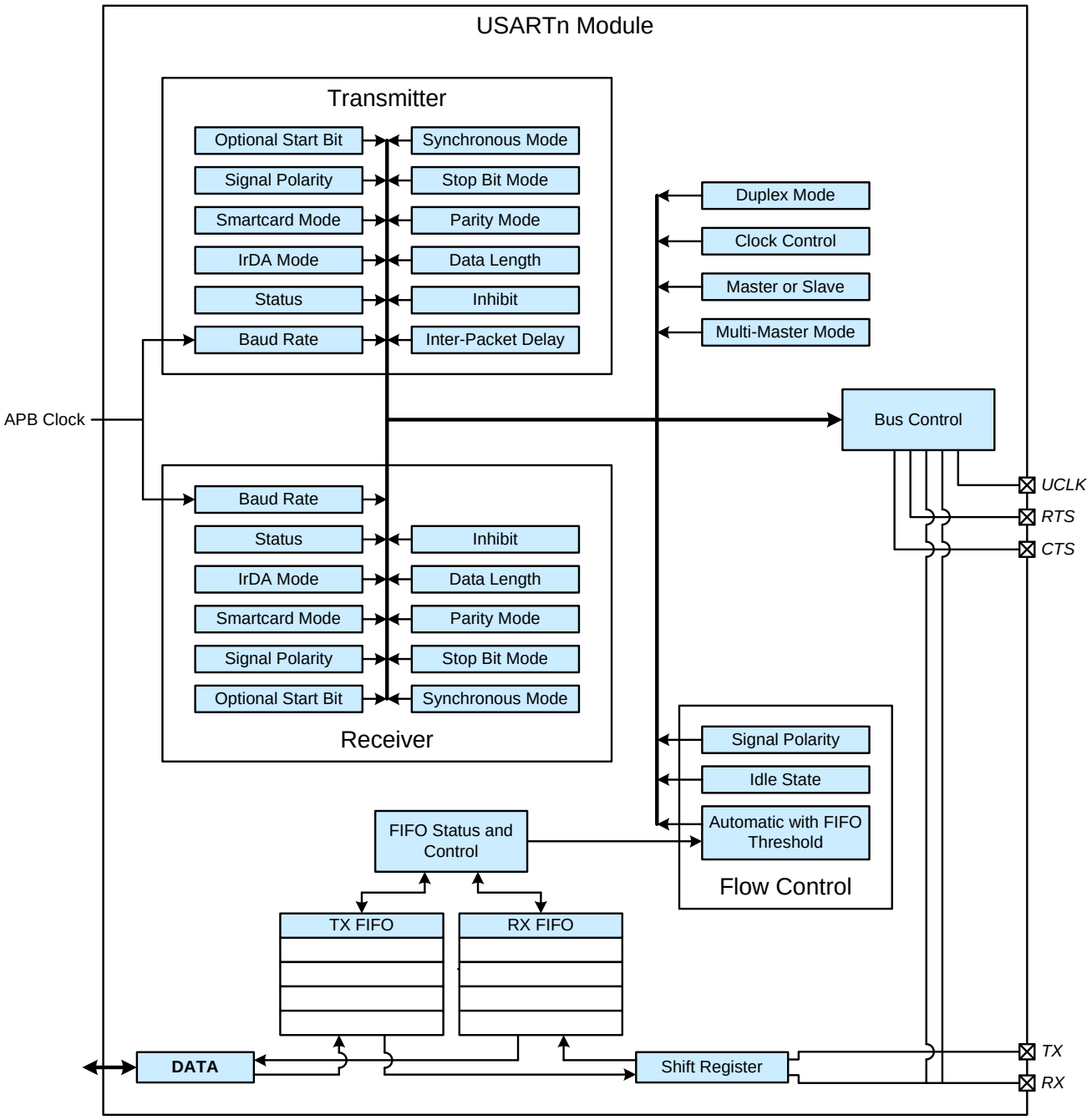


Figure 36.1. USART Block Diagram

36.2. Basic Data Format

The USART module data consists of four parts: start bit, data, parity bit, and stop bit.

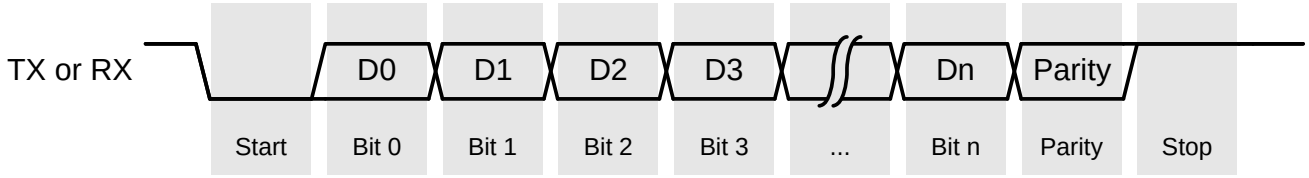


Figure 36.2. Basic Asynchronous USART Communication

Start bits are optional and can be enabled using the transmitter start enable (TSTRTEN) and receiver start enable (RSTRTEN) bits.

The data length is variable and can be set to 5 to 9 bits using the transmitter data length (TDATLN) or receiver data length (RDATLN) bit fields.

The transmitter will transmit a parity bit when the transmitter parity enable (TPAREN) bit is set. The transmitter parity mode (TPARMD) field configures the parity for the transmit data as either odd, even, set (parity always set to 1), or clear (parity always cleared to 0).

Similarly, the receiver will expect to receive a parity bit when receiver parity is enabled (RPAREN = 1). The RPARMD field controls the receiver parity mode and sets it to odd, even, set (parity always expected to be set to 1), or clear (parity always expected to be cleared to 0).

The stop bits are also fully configurable using the transmitter stop enable (TSTPEN) and receiver stop enable (RSTPEN) bits. The transmitter stop mode (TSTPMD) and receiver stop mode (RSTPMD) fields configure the stop length to 0.5, 1, 1.5, or 2 bits. Transmitting partial stop bits (0.5 or 1.5) causes subsequent transmitted bits to slip by one-half of a bit-time.

36.3. Baud Rate

The USART can operate as two independent, one-way channels since the transmitter and the receiver each have their own internal baud rate generators. For example, the transmitter can operate in synchronous mode at one baud rate while the receiver operates in asynchronous mode at a different baud rate.

If only one-half of the USART module operates in synchronous mode, the associated baud rate generator controls the clock frequency. If both the transmitter and receiver operate in synchronous mode, then the transmitter's baud rate generator controls the UCLK frequency.

In asynchronous or synchronous master modes, the transmitter baud rate control (TBAUD) bit field sets the transmitter baud rate. In asynchronous master mode, the receiver baud rate control (RBAUD) bit field sets the receiver baud rate. The receiver baud rate control (RBAUD) bit field sets the receiver baud rate in synchronous master mode only if the transmitter is not also in synchronous master mode. The equations in the TBAUD and RBAUD field descriptions determine the transmitter or receiver baud rate. The associated baud rate generator is unused in synchronous slave mode.

36.3.1. Receiver Auto-Baud Detection

The hardware receiver auto-baud detection automatically derives the appropriate RBAUD value from the received data. When receive auto-baud detection is enabled (RABDEN = 1), the hardware requires the first incoming data to be a value that guarantees an RX state transition between each bit: 0x55 when in non-IrDA mode and 0x00 when in IrDA mode. The receiver auto-baud detection supports either 8- or 9-bit data lengths with any combination of parity or stop bits.

Figure 36.3 shows the USART receive auto-baud detection input timing.

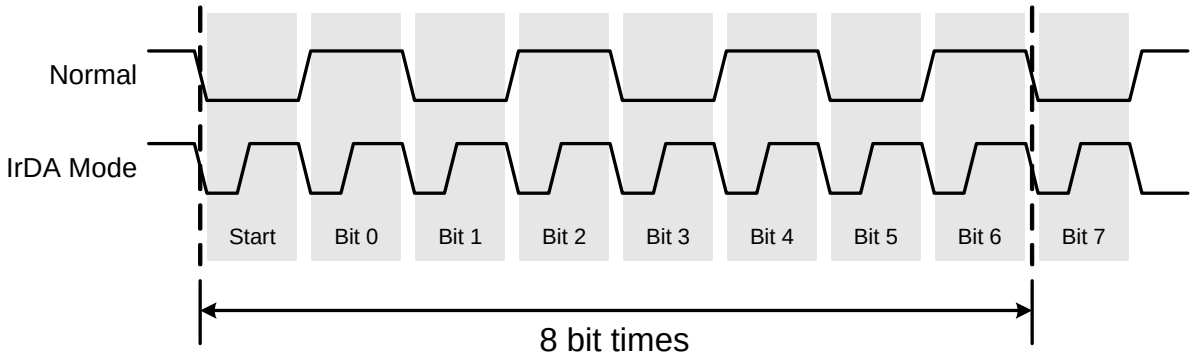


Figure 36.3. Receiver Auto-Baud Detection Input Timing

The receiver auto-baud detection calculates the appropriate value for RBAUD by counting the number of clock cycles between the leading edge of the start bit and the leading edge of the 8th data bit and dividing the clock cycle count by either 16 (when RIRDAEN = 0) or 128 (when RIRDAEN = 1). The hardware stores the calculated value to RBAUD and clears the receive auto-baud detection enable bit (RABDEN = 0). The new baud rate is used starting at the 8th bit of the incoming packet. The receiver will treat the received data (0x55 or 0x00) like any other received data and check it for parity errors and store it in the FIFO. If the first incoming byte causes the RBAUD value to overflow, indicating an auto-baud error, hardware will set the receive frame error interrupt (RFRMERI) flag regardless of the sampled state of the stop bits.

36.4. Interrupts

The USART module contains several interrupt sources that cause a vector to the USART interrupt. All of the interrupt flags may be masked by clearing a corresponding interrupt enable bit.

36.4.1. Transmit Interrupt Sources

The transmit data request interrupt (TDREQI) can be enabled by setting the TDREQIEN bit to 1. This flag indicates that the transmitter is requesting more FIFO data. The TDREQI flag automatically clears when the number of full entries in the transmit FIFO is more than the TFTH setting.

The transmit complete interrupt (TCPTI) flag indicates the following:

1. If TCPTTH = 0, this flag indicates a single transmit completed since TCPTI was last cleared.
2. If TCPTTH = 1, this flag indicates a transmit of the last available data in the FIFO completed since TCPTI was last cleared.

In either scenario, the hardware will generate an interrupt if the TCPTIEN bit is set to 1 when TCPTI is set.

The transmitter has two error condition flags. The underrun error interrupt (TUREI) flag indicates when the transmitter is required to send data but no data is available in the transmit FIFO. The Smartcard parity error interrupt (TSCERI) flag sets when the hardware detects the TX pin state as 0 at the end of the first whole stop bit period when operating in Smartcard mode (TSCEN = 1). Both of these interrupts are enabled with the transmit error interrupt enable (TERIEN) bit. Transmissions are automatically inhibited when any enabled error flag causes the interrupt to assert.

Finally, the transmitter has a transmit FIFO error interrupt (TFERI) flag which is set whenever an illegal FIFO write (e.g, a write to a full FIFO) is detected. Note that TFERI does not have an interrupt enable/disable control bit, so if the TFERI flag is set an interrupt will always be generated (assuming the USART module interrupt is enabled).

36.4.2. Receive Interrupt Sources

The receive data request interrupt (RDREQI) flag indicates that at least the number of receive FIFO slots set by the receive FIFO threshold (RFTH) are full, and firmware can read data from the receive FIFO. This interrupt is enabled by setting the receive data request interrupt enable (RDREQIEN) to 1. This flag automatically clears when the number of filled slots in the receive FIFO drops below the RFTH setting.

The receiver has three error interrupt sources. The receive overrun error interrupt (ROREI) flag indicates when the receive FIFO and shift register is full, and the shift register has been erroneously overwritten by additional receive data. The receive parity error interrupt (RPARERI) indicates that an invalid parity bit has been received. Finally, the receive frame error interrupt (RFRMERI) indicates when an expected whole stop bit is low, when a frame match mode fails, or when an auto-baud error occurs. The receive error interrupt enable (RERIEN) bit enables these flags as interrupt sources.

Finally, the receiver has a receive FIFO error interrupt (RFERI) flag which is set whenever an illegal FIFO read (e.g, a read from an empty FIFO) is detected. Note that RFERI does not have an interrupt enable/disable control bit, so if the RFERI flag is set an interrupt will always be generated (assuming the USART module interrupt is enabled).

36.5. Flow Control

36.5.1. Transmitter Hardware Flow Control

The CTS enable (CTSEN) bit enables hardware flow control in the transmitter. When CTS is enabled, the transmitter will only begin transmissions if the CTS input is low after the optional inversion using the CTSINVEN bit.

Firmware can read the current state of the CTS pin using the CTS bit. The CTS value read will be inverted from the CTS pin if the signal is inverted (CTSINVEN = 1).

36.5.2. Receiver Hardware Flow Control

Firmware can set the RTS enable (RTSEN) bit to enable hardware flow control in the receiver. When flow control is enabled, the receiver asserts RTS under any of the following conditions:

1. The receiver is disabled (REN = 0).
2. The receiver is inhibited (RINH = 1).
3. The receiver is stalled during a debug halt (DBGMD = 1).
4. An error flag sets causing the receive error interrupt to assert (RERIEN = 1 and RFRMERI, ROREI, or RPARERI is also set).
5. The receiver can store zero (if RTSTH = 0) or up to one (if RTSTH = 1) more data.

The RTS signal holds its last value when disabled by clearing RTSEN to 0. The RTS bit allows firmware to read the state of the RTS signal. Firmware can also write the RTS bit when flow control is disabled (RTSEN = 0) to set the state of the RTS pin. Note that the value written into the RTS bit will be inverted at the RTS pin if the RTS invert enable (RTSINVEN) bit is set to 1.

36.5.3. Inter-Packet Delay Generator

The transmitter supports a configurable delay between transmissions that may be used to limit the transmission rate in applications that do not support hardware flow control.

The hardware calculates the inter-packet delay using a counter operating at the current baud rate. As a result, the IPDELAY field represents multiples of the transmitter bit times. For example, setting the inter-packet delay (IPDELAY) field to 5 results in a delay equal to 5 bit times. Setting the IPDELAY field to 0 disables the inter-packet delay.

If in synchronous slave mode, UCLK must be running to use the inter-packet delay feature.

36.6. Debug Mode

Firmware can set the DBGMD bit to force the USART module to halt on a debug breakpoint. The module will complete any active transmissions and receptions before stopping. Clearing the DBGMD bit forces the USART module to continue operating while the core halts in debug mode. Note that when IPDELAY is set to zero, the USART may not halt until the FIFO is empty. If IPDELAY is non-zero, the USART will complete the current transfer but will not empty the FIFO.

36.7. Sending Data

To begin a transmit operation, firmware should set all the necessary transmitter configuration bits, enable the transmitter (TEN = 1), and write the outgoing data to the DATA register. Hardware will automatically transfer the data from the DATA register to the transmit FIFO. A FIFO entry is immediately loaded from the FIFO into the shift register any time data is available in the transmit FIFO and the shift register is empty.

The data transmission begins when all of the following conditions are met:

1. The shift register is loaded with data.
2. The transmitter is enabled (TEN = 1).
3. The transmitter is not inhibited (TINH = 0).
4. The CTS pin is deasserted if flow control is enabled (CTSEN = 1).
5. The module is not halted because of a debug breakpoint if DBGMD is set to 1.

36.7.1. Transmitter FIFO Management

The USART transmit FIFO is implemented as a circular buffer with four entries, allowing data to be pushed as a continuous stream. The DATA register forms a single port into the transmit and receive FIFOs. Writes to the DATA register push data into the transmit FIFO, and reads from the DATA register pop data from the receive FIFO.

The transmitter supports byte, half-word, or word writes to the FIFO. Writes to the FIFO should always be right-justified, so byte-wide writes should always write DATA[7:0], half-word writes should always write DATA[15:0], and word writes should always write DATA[31:0]. The transmitter ignores all other writes to the DATA register, including left-justified byte writes or writes containing more data than will fit in the empty slots in the FIFO. Any illegal FIFO write sets the transmit FIFO error interrupt flag (TFERI) and causes an interrupt, if enabled.

The TDATLN bit controls how the written data is mapped into the FIFO. The transmitter supports data lengths ranging from 5 to 9 bits; the optional start, parity, and stop bits are not stored in the FIFO. When transmitting less than 9 bits of data, the hardware transmits all 5 to 8 bits written to the FIFO by firmware.

When transmitting 9-bit data, the transmitter supports two modes of operation:

1. In normal 9-bit mode (TDATLN = 4), firmware writes 9-bit data to the FIFO with each half-word containing a right-justified 9-bit entry.
2. In fixed 9-bit mode (TDATLN = 5), bus writes to the FIFO are assumed to contain only the least-significant 8 bits of each data entry, and hardware inserts the 9th bit value (set by TBIT) into the FIFO on each FIFO write.

When the data length is 8 bits (or 9 bits when using TDATLN = 5), one, two, or four FIFO entries can be written in a single bus operation by using byte, half-word, or word operations, respectively. For example, writing a whole word (4 bytes) to DATA[31:0] will write four entries in the FIFO with each byte as an entry.

When the data length is 9 bits (using TDATLN = 4), two FIFO entries can be written in a single bus operation by using a word operation, or one entry can be written using a half-word operation. With a half-word write, the hardware will push DATA[8:0] to the FIFO. With a word write, the hardware will push DATA[8:0] to the FIFO, followed by the value in DATA[24:16]. The other bits in the DATA register in this mode are unused and have no effect.

The transfer FIFO count (TCNT) field maintains a count of the number of occupied entries in the transmit FIFO. Firmware may read TCNT to determine the allowable width of writes (byte, half-word, or word) given the available number of empty FIFO entries.

The transmit FIFO threshold (TFTH) field configures the threshold at which the hardware asserts the transmit data request interrupt (TDREQI) or DMA data request and may be configured to 1, 2, or 4 empty FIFO slots.

Firmware can set the transmit FIFO flush (TFIFOFL) bit to flush the contents of the transmit FIFO. A FIFO flush will also clear the contents of the shift register if the transmitter is idle and is not actively transmitting data.

36.7.2. Transmitter Status

The USART module has two status bits which may be monitored to determine the status of the transmitter.

36.7.2.1. Transmitter Busy Flag (TBUSYF)

The transmitter busy flag (TBUSYF) will be asserted when a single byte transmission is in progress. Note that if transmitted multiple bytes with IPDELAY>0, TBUSYF will return to 0 between each transmitted byte. For this reason, TBUSYF is not a reliable indicator of a completed transmission, and the method described below using the TCPTI bit is recommended.

36.7.2.2. Transmitter Complete Interrupt Flag (TCPTI)

The transmit complete interrupt (TCPTI) flag indicates the following:

1. If TCPTTH = 0, this flag indicates a single transmit completed since TCPTI was last cleared.
2. If TCPTTH = 1, this flag indicates a transmit of the last available data in the FIFO completed since TCPTI was last cleared.

To determine when the transmitter has completed transmission of all data in the FIFO, firmware may set TCPTTH=1, write the data into the FIFO, and then wait for the TCPTI bit to be asserted. Note that TCPTI must be manually cleared by firmware.

36.8. Receiving Data

Firmware should first configure all necessary receiver configuration bits before enabling the receiver (REN = 1). Incoming data will be de-glitched and then shifted into the 10-bit receive shift register. When the data reception is complete, the data will be aligned and padded according to the data length specified in the RDATLN bit field, and then pushed into the circular receive FIFO buffer if an empty entry is available. When both the receive FIFO and the shift register are full, any subsequent received data replaces the shift register contents and causes the hardware to set the receive overrun error interrupt (ROREI) flag.

Firmware may read from the FIFO any time full entries are available regardless of the state of the receiver.

The receiver supports one-shot receptions, where the receiver accepts a single byte. Firmware can enable this by setting the ROSEN bit. The hardware automatically clears this bit after the next reception is completed and accepted. If MATMD is set to 1 for MCE mode and a match does not occur, the hardware will not accept the data and will not clear the ROSEN bit.

36.8.1. Receiver FIFO Management

The receive FIFO is implemented as a circular buffer with four entries. The receive and transmit FIFOs are both accessed using the same DATA register; a read from the DATA register pops data from the receive FIFO, and a write to the DATA register pushes data into the transmit FIFO. When reading from the FIFO, the least-significant entry is popped from the FIFO first. The receiver supports byte, half-word, or word DMA and bus reads from the FIFO.

The RDATLN field determines how the data is mapped into the FIFO. The receiver supports data lengths ranging from 5 to 9 bits; the optional start, parity, and stop bits are not stored in the FIFO. When receiving less than 9 bits, a firmware read from the FIFO returns all 5 to 8 bits of the receive data.

When receiving 9-bit data, the receiver supports two modes of operation:

1. In normal 9-bit mode (RDATLN = 4), firmware reads 9-bit data from the FIFO, with each half-word containing a right-justified 9-bit entry.
2. In fixed 9-bit mode (RDATLN = 5), only the least-significant 8 bits of received data are stored in the FIFO. The 9th bit is compared against RBIT to generate an interrupt or set error flags according to the MATMD field.

When the data length is 8 bits (or 9 bits when RDATLN = 5), firmware can read one, two, or four FIFO entries in a single bus operation by using byte, half-word, or word operations, respectively.

When the data length is 9 bits (using normal 9-bit mode with RDATLN = 4), each half-word is treated as a right-justified entry. When reading 9-bit data using a half-word bus operation, the hardware places the first entry popped from the FIFO into DATA[8:0]. When reading a whole word, hardware places the first entry popped from the FIFO into DATA[8:0] and the second entry into DATA[24:16]. The rest of the DATA bits are unused in this mode and are set to zero.

The receive FIFO count (RCNT) field maintains a count of the number of occupied entries in the receive FIFO. Firmware may read RCNT to determine the allowable read widths given the available number of full FIFO entries. The receive FIFO threshold (RFTH) field configures the threshold at which hardware asserts a read request interrupt (RDREQI) or DMA data request and may be configured to 1, 2, or 4 occupied slots.

Firmware can set the receive FIFO flush (RFIFOFL) bit to flush the contents of the receive FIFO. A FIFO flush will also clear the contents of the shift register if the receiver is idle and is not actively receiving data.

36.8.2. Receiver Status

The USART module includes a status bit which may be monitored to determine the status of the receiver.

36.8.2.1. Receiver Busy Flag (RBUSYF)

The receiver busy flag (RBUSYF) will be asserted when a single byte receive is in progress. Note that when receiving multiple bytes, any delay between the bytes may cause RBUSYF to return to 0 between each received byte.

36.9. Synchronous Communications

Synchronous communication is similar to basic asynchronous protocol with the addition of a synchronous UCLK clock signal. The UCLK signal is an output in master mode and an input when operating as a slave.

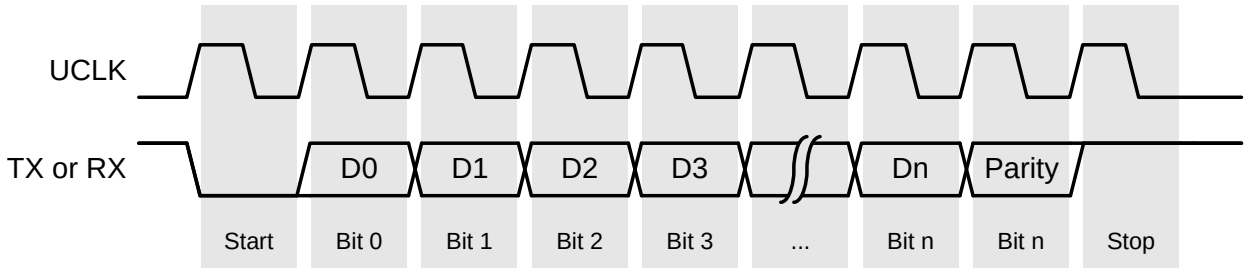


Figure 36.4. Synchronous USART Communication

Firmware can configure the USART transmitter or receiver for synchronous mode by setting the transmitter synchronous mode (TSYNCCEN) or receiver synchronous mode (RSYNCEN) enable bits.

The transmitter and receiver share a common clock generator and will share a common UCLK pin and clock configuration if both are configured for synchronous operation. Either the transmitter or receiver may request that the hardware generate a clock when configured as a synchronous clock master. If only one-half of the module is operating in synchronous mode, that half's associated baud rate generator controls the clock frequency. If both the transmitter and receiver are operating in synchronous mode, then the transmitter's baud rate generator controls the UCLK frequency.

When both the transmitter and receiver are disabled or in asynchronous mode, the UCLK pin state may be read or written via the UCLK bit in the FLOWCN register.

Note that in synchronous mode, the transmitter and receiver baud rates must not exceed APBCLK / 16.

36.9.1. Synchronous Clock Master Mode

The operational mode (OPMD) configures the USART module as either the synchronous master or slave. Setting this bit to 1 enables the module as the synchronous master. In synchronous clock master mode, the TBAUD register setting determines the clock frequency.

By default, the master will hold the UCLK pin idle between transmissions. Firmware can set the idle clock control bit (ISTCLK) to 1 to generate a clock between transmissions.

To configure the transmitter to generate a clock during the start or stop bits, set the start state clock control (STRTSTCLK) or the stop state clock control (STPSTCLK) bits to 1.

36.9.2. Synchronous Clock Slave Mode

In synchronous clock slave mode, the UCLK pin is configured as an input, and the receiver or transmitter expect to receive a clock signal from the synchronous master. In synchronous slave mode, Firmware should set the idle clock control bit (ISTCLK) to 1 to prevent the slave from transmitting without waiting for the clock edge

.

If the module is configured as a synchronous slave (OPMD = 0), is not configured to transmit a start bit (TSTRTEN = 0), and receives a UCLK edge causing it shift out a data bit with no data available, the transmitter will set the underrun error flag (TUREI) and generate an interrupt, if enabled (TERIEN = 1).

36.9.3. Synchronous Clock Configuration

The synchronous clock generator supports configurable UCLK polarity and phase using two control bits: sampling clock edge select (CLKESEL) and clock idle state (CLKIDLE). In synchronous master mode (OPMD = 1), these bits determine the characteristics of the generated clock; in synchronous slave mode (OPMD = 0), these bits determine the expected characteristics of the received clock.

The CLKESEL bit determines whether the generated or received clock falls (CLKESEL = 0) or rises (CLKESEL = 1) in the middle of each transmitted bit. The CLKIDLE bit determines whether the idle state of the generated or received clock is low (CLKIDLE = 0) or high (CLKIDLE = 1).

Figure 36.5 illustrates these clock configurations.

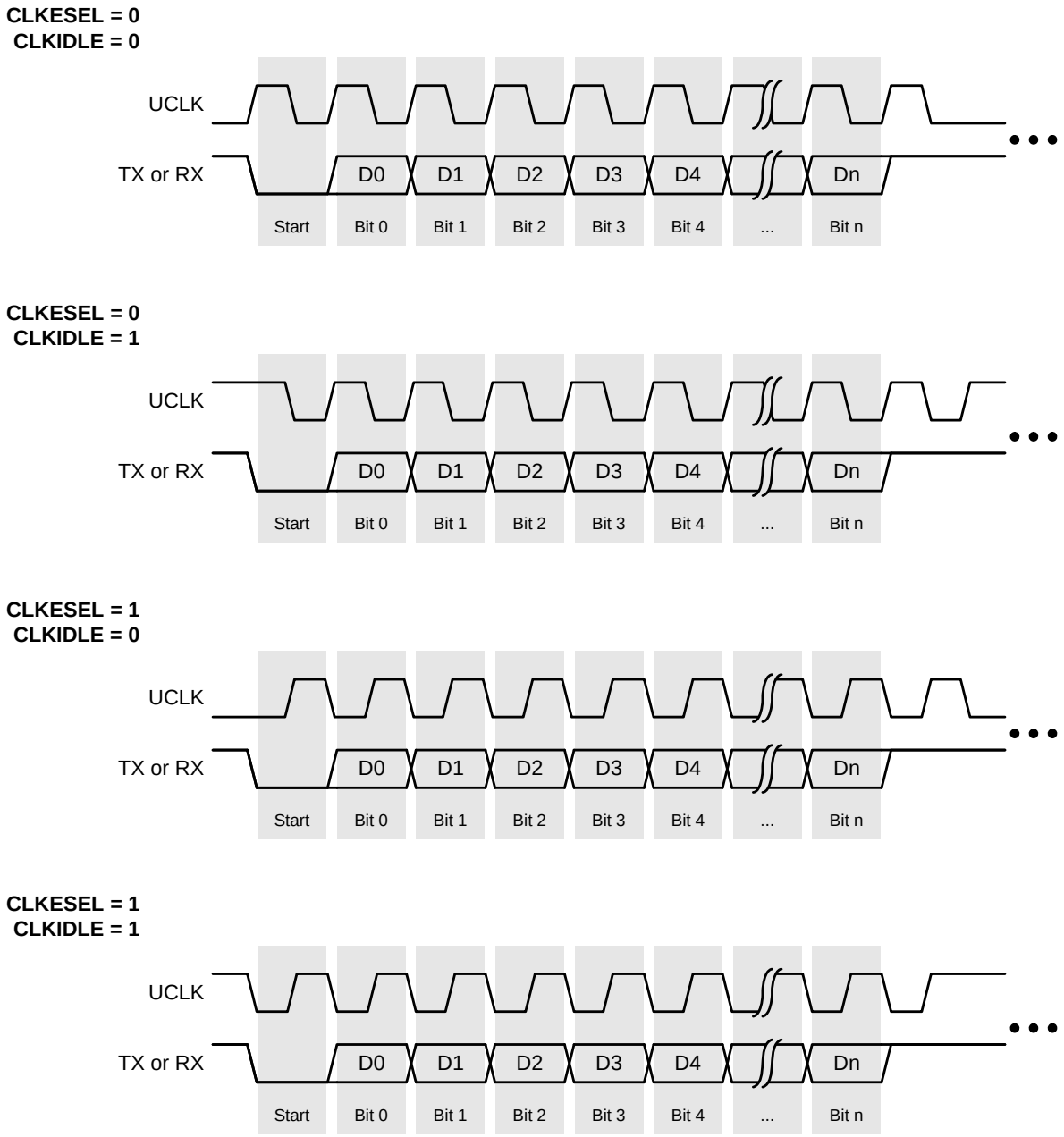


Figure 36.5. USART Clock Configurations

36.10. Additional Communication Support

36.10.1. Multi-Master Mode

The transmitter supports applications in which multiple transmitters are driving the same TX and UCLK signals. This multi-master mode is suited for applications using a Smartcard or LIN, which perform one-wire, half-duplex communications.

Multi-master mode requires the use of idle signal tristates (ITSEN = 1). When this bit is set to 1 and the transmitter is idle, the transmitter automatically tristates the TX pin when not transmitting. The transmitter will also automatically tristate the UCLK pin if the transmitter is the clock master. The TX (and UCLK pin, if the transmitter is the synchronous master) pin must have an external pull-up resistor to ensure the pin remains in an idle state when tristated.

Because a synchronous master will always attempt to drive a clock when it sees data on its RX input, if using multi-master mode in a synchronous application, the transmitting USART must set itself to slave mode between transmissions to prevent contention on the clock line.

36.10.2. Multi-Processor Communications

In a multi-processor application (multiple slaves), a master first transmits an address byte to select the target. In an address byte, the 9th bit is always set to a logic 1; in a data byte, the 9th bit is always set to logic 0. The USART module supports multi-processor communication through the use of match operations. Although the match operations are most commonly used with the 9th data bit as an address indicator, the match operations are available regardless of the actual configured data length.

Match operations are enabled or disabled in the receiver by configuring the match mode (MATMD) bit field.

When MATMD is set to 1, the hardware operates in MCE mode and only accepts and stores the incoming receive data if the last data bit matches the value of RBIT. Otherwise, the hardware ignores incoming data, no interrupts are generated, and no FIFO activity occurs. This mode can be used to only accept data when the last data bit marks the frame as containing an address value. Firmware would then compare this address against an assigned value and reconfigure the receiver as needed if the values match. The MATMD setting is automatically switched to Frame mode after a match occurs in the MCE setting.

In Frame mode (MATMD = 2), all incoming data is accepted and stored, but a receive frame error (RFRMERI) asserts if the last data bit matches the value of RBIT. This operation can be used to assert an error when an unexpected address frame arrives.

In Store mode (MATMD = 3), the hardware accepts all incoming data and stores the last data bit in the RBIT field. The last data bit may also be stored in the FIFO depending on the value of RDATLN. For example, if RDATLN is set to 9 bits with the 9th bit stored in the FIFO (RDATLN = 4), and MATMD is set to store, the 9th data bit will be stored in both RBIT and the FIFO.

36.10.3. IrDA

The USART module transmitter and receiver support simple, asynchronous communications with IrDA devices.

36.10.3.1. IrDA Modulation

Firmware can place the transmitter in IrDA mode by setting the TIRDAEN bit. In IrDA mode, the transmitter performs a RZ (return-to-zero) modulation on the TX signal.

By default, transmitting a 0 causes a pulse to be generated low for a fraction of the bit time and transmitting a 1 leaves TX high without generating a low pulse. The transmit IrDA pulse width (TIRDAPW) bit field configures the pulse width as either 1/16, 1/8, 3/16, or 1/4 of the bit time.

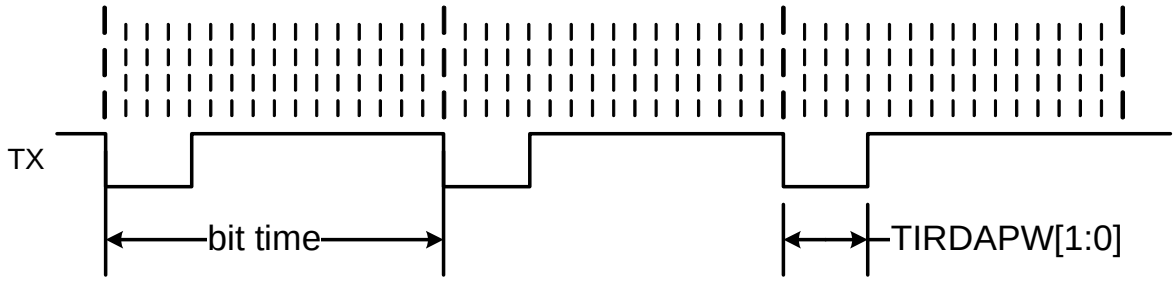


Figure 36.6. USART Transmit IrDA Pulse Width Timing

As shown in Figure 36.6, the default TX idles high and generates a low pulse when a 0 is transmitted. The polarity of TX may be inverted by setting the transmitter invert enable (TINVEN) bit, causing TX to idle low and generate a high pulse when sending a 0.

36.10.3.2. IrDA Demodulation

Setting the receiver IrDA enable (RIRDAEN) to 1 places the receiver in IrDA mode. In this mode, the receiver performs a simple RZ-to-NRZ (return-to-zero to non-return-to-zero) demodulation on the RX signal.

36.10.4. Smartcard Mode

36.10.4.1. Smartcard Transmitter

The transmitter Smartcard mode is enabled by setting the transmitter Smartcard parity response enable (TSCEN = 1) and consists of the ability to capture the ACK or NACK response from a Smartcard during the stop bit periods. Generally, Smartcard communications will also use multi-master and half-duplex modes with the TX and RX pins tied together externally.

Figure 36.7 illustrates the signaling on the TX pin in Smartcard mode.

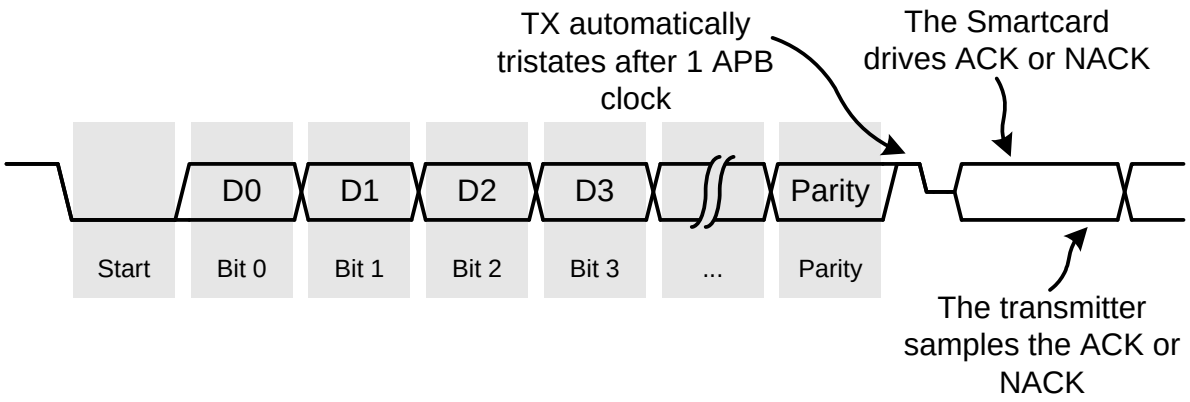


Figure 36.7. Smartcard Transmit Signalling

The transmitter should be configured to transmit 2 stop bits (or 1.5 stop bits if inter-packet delays are enabled) in Smartcard mode. The transmitter will tristate the TX pin one APB clock cycle after the start of the first stop bit. The Smartcard is expected to drive the TX pin low starting at the mid-point of the first stop bit if a receive error occurred. The transmitter captures the state of the TX pin at the end of the first stop bit period and sets the Smartcard parity error interrupt (TSCERI) flag accordingly. An interrupt is generated if the error flag is set and the transmit error interrupt is enabled (TERIEN = 1). The transmitter will re-enable its TX pin driver at the end of the stop bits except when operating in multi-master mode.

36.10.4.2. Smartcard Receiver

The receiver Smartcard mode is enabled by setting receiver Smartcard parity response (RSCEN) to 1 and allows the receiver to return an ACK or NACK response to a Smartcard during the stop bit periods. Figure 36.8 shows signaling on the RX pin in Smartcard mode.

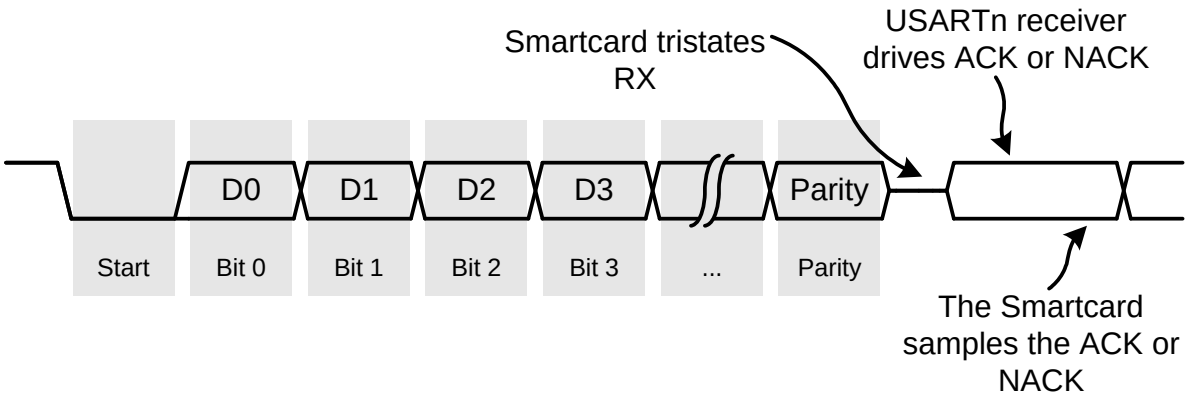


Figure 36.8. Smartcard Receive Signalling

When using Smartcard mode, firmware should configure the receiver to expect 2 stop bits. The receiver will drive RX low if a parity error occurs or high if no parity error occurs for one bit-time starting at the mid-point of the first stop bit. The Smartcard is expected to sample the state of RX at the end of the first stop bit.

36.10.5. LIN Support

The USART transmitter and receiver can be used with LIN (Local Interconnect Network) systems.

When using LIN, firmware should configure the transmitter to use 8 data bits, 1 start bit, 1 stop bit, and no parity bits.

The receiver auto-baud detection also supports the LIN SYNC byte (0x55).

36.10.6. Half Duplex Mode

The USART module supports applications in which half-duplex communications occur across a shared TX/RX signal. In half-duplex mode, the hardware automatically inhibits the receiver during transmit operations and the transmitter during receive operations.

The duplex mode bit (DUPLEXMD) enables half-duplex mode for the module. Firmware should also enable idle tristates (ITSEN = 1) to force the transmitter to tristate the TX signal when not transmitting. The RX and TX pins must be shorted externally in half-duplex mode.

36.10.7. Loop Back Support

The USART module supports several internal loop back options for testing purposes. The loop back modes are configured in the loop back mode (LBMD) field. Table 36.1 describes the different loop back options.

Table 36.1. Internal Loop Back Modes

LBMD value	Loop Back Mode
0	Disabled
1	Receive Loop Back
2	Transmit Loop Back
3	Full Loop Back

When loop back mode is disabled, the module operates normally.

In receive loop back mode, the receiver input path is disconnected from the RX signal and internally connected to the transmitter. Any data transmitted in this mode will be sent out on the TX signal and also received by the device. The physical RX pin tristates in this mode. TX, CTS, RTS, and UCLK are connected to the corresponding external pins, if the signals are enabled to connect to pins in the device's port configuration.

With transmit loop back, the transmitter output path is disconnected from the TX signal, and the RX input signal is internally looped back to the TX pin. Data received on the RX signal will be received by the device and also sent directly back out on the TX pin. RX, CTS, RTS, and UCLK are connected to the corresponding external pins, if enabled.

In full loop back mode, the transmitter output is internally routed back to the receiver input. Neither the transmitter nor receiver are connected to external device pins. If enabled, the RX device pin is similarly looped back to the TX pin. Any data transmitted on TX will be sent directly back in on RX. The CTS input and RTS output pins are likewise looped back at both the transmitter and receiver and at the device pins, if enabled.

Figure 36.9 illustrates the internal connections for each loop back mode.

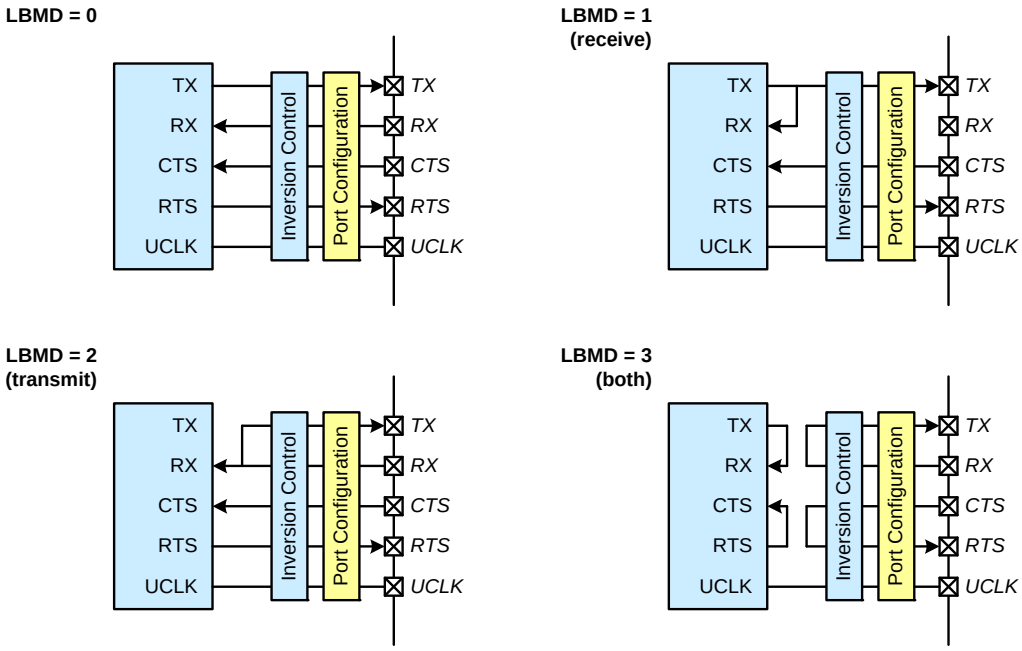


Figure 36.9. Internal Loop Back Connections

36.11. DMA Configuration and Usage

The USART module supports an independent DMA interface for the transmitter and receiver. The DMA operations should target the DATA register in non-incrementing mode. The number of bytes transferred per DMA read or write should coordinate with the transmit and receive FIFO thresholds (TFTH and RFTH) to ensure data is transferred as quickly as possible. For example, if transmitting 9-bit data, firmware can configure the TFTH field to cause a DMA data request when two FIFO entries are empty and the DMA to move words containing two 9-bit data values.

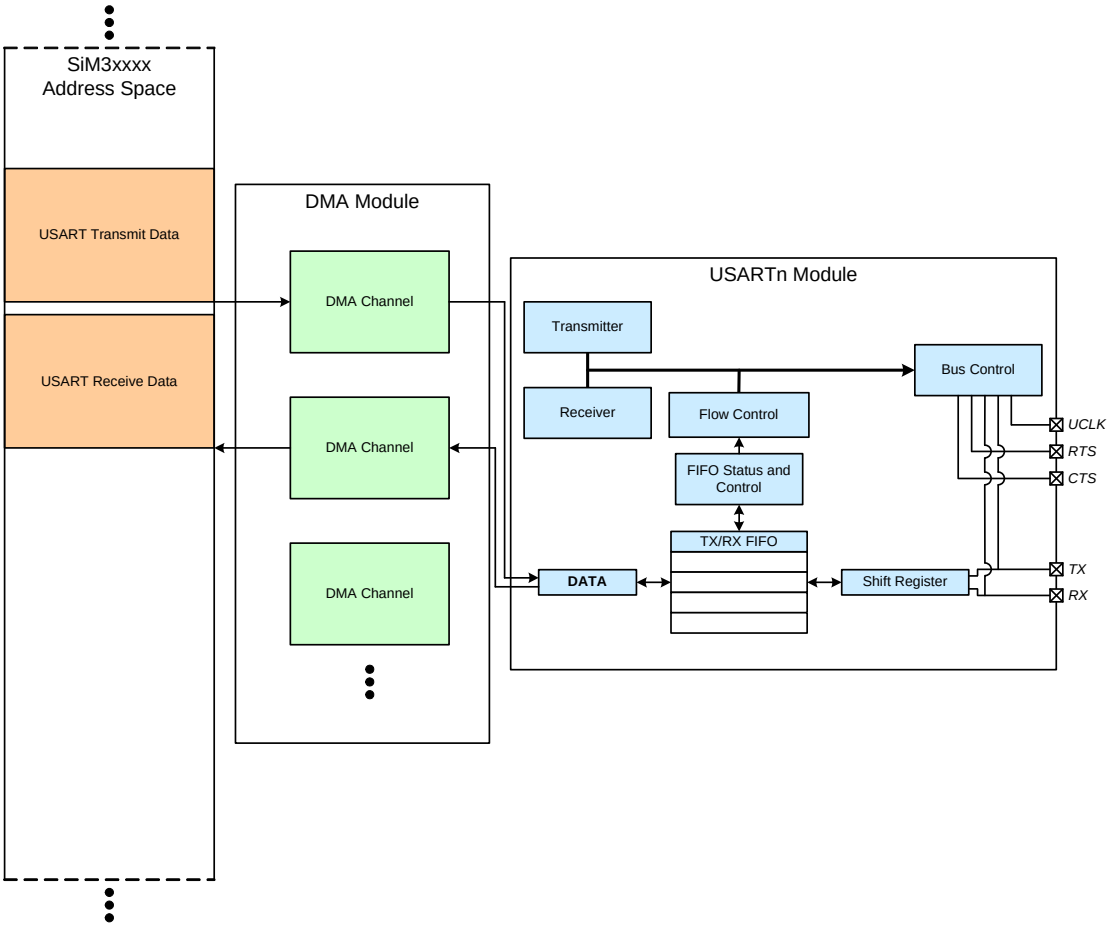


Figure 36.10. USART DMA Configuration

When the transmitter DMA enable (TDMAEN) bit is set and the number of empty entries in the transmit FIFO is equal to or greater than the setting in the transmit FIFO threshold (TFTH) field, the following two events occur:

1. The transmitter automatically requests a single DMA operation.
2. The transmitter sets the read-only transmit data request interrupt flag (TDREQI), which asserts an interrupt, if enabled (TDREQIEN = 1).

Firmware can enable DMA requests for the receiver by setting the RDMAEN bit to 1. When in DMA mode and the number of full entries in the receive FIFO is equal to or greater than the setting in the receive FIFO threshold (RFTH), the following two events occur:

1. The receiver automatically requests a single DMA operation. The receiver does not need to be enabled for this DMA request to occur.
2. The receiver sets the read-only receive data request interrupt flag (RDREQI), causing a receive data request interrupt, if enabled (RDREQIEN = 1).

The transmit and receive FIFO error flags are still set by hardware in DMA mode in the event of a FIFO error.

36.12. USART0 and USART1 Registers

This section contains the detailed register descriptions for USART0 and USART1 registers.

Register 36.1. USARTn_CONFIG: Module Configuration

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	TSYNCEN	TINVEN	TIRDAEN	TSCEN	Reserved	TDATLN			Reserved	TPARMD		TSTPMD		TSTPEN	TPAREN	TSTRTEN
Type	RW	RW	RW	RW	R	RW			R	RW		RW		RW	RW	RW
Reset	0	0	0	0	0	0	1	1	0	0	0	0	1	1	0	1
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	RSYNCEN	RINVEN	RIRDAEN	RSCEN	Reserved	RDATLN			Reserved	RPARMD		RSTPMD		RSTPEN	RPAREN	RSTRTEN
Type	RW	RW	RW	RW	R	RW			R	RW		RW		RW	RW	RW
Reset	0	0	0	0	0	0	1	1	0	0	0	0	1	1	0	1

Register ALL Access Addresses

USART0_CONFIG = 0x4000_0000
USART1_CONFIG = 0x4000_1000
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 36.2. USARTn_CONFIG Register Bit Descriptions

Bit	Name	Function
31	TSYNCEN	**Transmitter Synchronous Mode Enable.** 0: The transmitter operates in asynchronous mode. 1: The transmitter operates in synchronous mode.
30	TINVEN	**Transmitter Invert Enable.** 0: Do not invert the TX pin signals (the TX idle state is high). 1: Invert the TX pin signals (the TX idle state is low).
29	TIRDAEN	**Transmitter IrDA Enable.** 0: Disable IrDA transmit mode. 1: Enable IrDA transmit mode.
28	TSCEN	**Transmitter Smartcard Parity Response Enable.** 0: The transmitter does not check for a Smartcard parity error response. 1: The transmitter checks for a Smartcard parity error response.
27	Reserved	Must write reset value.

Table 36.2. USARTn_CONFIG Register Bit Descriptions

Bit	Name	Function
26:24	TDATLN	**Transmitter Data Length.** Select the number of data bits sent during transmission. 000: 5 bits. 001: 6 bits. 010: 7 bits. 011: 8 bits. 100: 9 bits. The 9th bit is taken from the FIFO data (normal mode). 101: 9 bits. The 9th bit is set by the value of TBIT (fixed mode). 110-111: Reserved.
23	Reserved	Must write reset value.
22:21	TPARMD	**Transmitter Parity Mode.** This field selects the type of parity sent during transmissions. 00: Odd Parity. 01: Even Parity. 10: Set (Parity = 1). 11: Clear (Parity = 0).
20:19	TSTPMD	**Transmitter Stop Mode.** This bit selects the transmitted stop bit length. 00: 0.5 stop bit. 01: 1 stop bit. 10: 1.5 stop bits. 11: 2 stop bits.
18	TSTPEN	**Transmitter Stop Enable.** 0: Do not send stop bits during transmissions. 1: Send stop bits during transmissions.
17	TPAREN	**Transmitter Parity Enable.** 0: Do not send a parity bit during transmissions. 1: Send a parity bit during transmissions.
16	TSTRTEN	**Transmitter Start Enable.** 0: Do not generate a start bit during transmissions. 1: Generate a start bit during transmissions.
15	RSYNCEN	**Receiver Synchronous Mode Enable.** 0: The receiver operates in asynchronous mode. 1: The receiver operates in synchronous mode.
14	RINVEN	**Receiver Invert Enable.** 0: Do not invert the RX pin signals (the RX idle state is high). 1: Invert the RX pin signals (the RX idle state is low).
13	RIRDAEN	**Receiver IrDA Enable.** 0: The receiver does not operate in IrDA mode. 1: The receiver operates in IrDA mode.

Table 36.2. USARTn_CONFIG Register Bit Descriptions

Bit	Name	Function
12	RSCEN	**Receiver Smartcard Parity Response Enable.** 0: The receiver does not send a Smartcard parity error response. 1: The receiver sends a Smartcard parity response.
11	Reserved	Must write reset value.
10:8	RDATLN	**Receiver Data Length.** Select the expected length of received data. 000: 5 bits. 001: 6 bits. 010: 7 bits. 011: 8 bits. 100: 9 bits. The 9th bit is stored in the FIFO (normal mode). 101: 9 bits. The 9th bit is not stored in the FIFO (fixed mode). This mode is used when the 9th bit is only used for match operations (see MATMD). 110-111: Reserved.
7	Reserved	Must write reset value.
6:5	RPARMD	**Receiver Parity Mode.** This field selects the type of parity expected during reception. 00: Odd Parity. 01: Even Parity. 10: Set (Parity = 1). 11: Clear (Parity = 0).
4:3	RSTPMD	**Receiver Stop Mode.** Select the number of stop bits expected during reception. 00: 0.5 stop bit. 01: 1 stop bit. 10: 1.5 stop bits. 11: 2 stop bits.
2	RSTPEN	**Receiver Stop Enable.** 0: Do not expect stop bits during receptions. 1: Expect stop bits during receptions.
1	RPAREN	**Receiver Parity Enable.** 0: Do not expect a parity bit during receptions. 1: Expect a parity bit during receptions.
0	RSTRTEN	**Receiver Start Enable.** 0: Do not expect a start bit during receptions. 1: Expect a start bit during receptions.

Register 36.2. USARTn_MODE: Module Mode Select

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	OPMD	ITSEN	CLKESEL	CLKIDLE	DUPLEXMD	Reserved			ISTCLK	STRTSTCLK	STPSTCLK	Reserved	LBMD		Reserved	DBGMD
Type	RW	RW	RW	RW	RW	R			RW	RW	RW	R	RW		R	RW
Reset	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

USART0_MODE = 0x4000_0010

USART1_MODE = 0x4000_1010

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 36.3. USARTn_MODE Register Bit Descriptions

Bit	Name	Function
31	OPMD	**Operational Mode.** Select between master and slave operation in synchronous mode. This bit has no effect in asynchronous mode. 0: The USART operates as a slave. 1: The USART operates as a master.
30	ITSEN	**Idle TX/UCLK Tristate Enable.** 0: The TX and UCLK (if in synchronous master mode) pins are always an output in this mode, even when idle. 1: The TX pin is tristated when idle. If ISTCLK is cleared to 0 and the transmitter is configured in synchronous master mode, the UCLK pin will also be tristated when idle.
29	CLKESEL	**Clock Edge Select.** This bit selects the edge control for the clock in synchronous mode. 0: The clock falls in the middle of each bit. 1: The clock rises in the middle of each bit.
28	CLKIDLE	**Clock Idle State.** Select the idle state for the clock in synchronous mode. 0: The synchronous clock is low when idle. 1: The synchronous clock is high when idle.

Table 36.3. USARTn_MODE Register Bit Descriptions

Bit	Name	Function
27	DUPLEXMD	**Duplex Mode.** If this bit is set to 1, the ITSEN bit must also be set to 1 to tristate the TX signal during receive operations. 0: Full-duplex mode. The transmitter and receiver can operate simultaneously. 1: Half-duplex mode. The transmitter automatically inhibits when the receiver is active and the receiver automatically inhibits when the transmitter is active.
26:24	Reserved	Must write reset value.
23	ISTCLK	**Idle Clock Control.** In synchronous master mode this bit configures whether or not the master generates clocks between transmissions. In synchronous slave mode this bit determines whether or not the slave waits for a clock edge before transmissions. If both ISTCLK and ITSEN are 1, the UCLK signal will not tristate when idle. 0: When the USART is a clock master and CLKESEL is not equal to CLKIDLE, the clock is held idle between transmissions. When the USART is a clock master and CLKESEL equals CLKIDEL, the clock will still be generated between transmissions. When the USART is a clock slave, the USART will begin transmissions without waiting for the next clock edge. 1: When the USART is a clock master, the clock is generated between transmissions or receptions. When the USART is a clock slave, the USART will wait until the next clock edge before transmitting.
22	STRTSTCLK	**Start State Clock Control.** Set whether or not the master generates clocks during the start bit in synchronous mode. 0: When the USART is a clock master, the clock is held idle during a start bit. 1: When the USART is a clock master, the clock is generated during a start bit.
21	STPSTCLK	**Stop State Clock Control.** Set whether or not the master generates clocks during the stop bit in synchronous mode. 0: When the USART is a clock master, the clock is not generated during stop bits. 1: When the USART is a clock master, the clock is generated during stop bits.
20	Reserved	Must write reset value.
19:18	LBMD	**Loop Back Mode.** Select from different internal loop-back options for diagnostic and debug purposes. 00: Loop back is disabled and the TX and RX signals are connected to the corresponding external pins. 01: Receive loop back. The receiver input path is disconnected from the RX pin and internally connected to the transmitter. Data transmitted will be sent out on TX and also received by the device. 10: Transmit loop back. The transmitter output path is disconnected from the TX pin and the RX input pin is internally looped back out to the TX pin. Data received at RX will be received by the device and also sent directly back out on TX. 11: Full loop back. Internally, the transmitter output is routed back to the receiver input. Neither the transmitter nor receiver are connected to external device pins. The device pin RX is looped back to TX in a similar fashion. Data transmitted on TX will be sent directly back in on RX.

Table 36.3. USARTn_MODE Register Bit Descriptions

Bit	Name	Function
17	Reserved	Must write reset value.
16	DBGMD	**USART Debug Mode.** 0: The USART module will continue to operate while the core is halted in debug mode. 1: A debug breakpoint will cause the USART module to halt. Any active transmissions and receptions will complete first.
15:0	Reserved	Must write reset value.

Register 36.3. USARTn_FLOWCN: Flow Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved		TIRDAPW		Reserved				CTSEN	Reserved	CTSINVEN	Reserved		UCLK	TX	CTS
Type	R		RW		R				RW	R	RW	R		RW	RW	R
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	X	1	X
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved								RTSEN	RTSTH	RTSINVEN	Reserved			RX	RTS
Type	R								RW	RW	RW	R			R	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	X	1

Register ALL Access Addresses

USART0_FLOWCN = 0x4000_0020

USART1_FLOWCN = 0x4000_1020

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 36.4. USARTn_FLOWCN Register Bit Descriptions

Bit	Name	Function
31:30	Reserved	Must write reset value.
29:28	TIRDAPW	**Transmit IrDA Pulse Width.** This field sets the IrDA pulse width as a fraction of the bit period. 00: The IrDA pulse width is 1/16th of a bit period. 01: The IrDA pulse width is 1/8th of a bit period. 10: The IrDA pulse width is 3/16th of a bit period. 11: The IrDA pulse width is 1/4th of a bit period.
27:24	Reserved	Must write reset value.
23	CTSEN	**CTS Enable.** 0: The CTS pin state does not affect transmissions. 1: Transmissions will begin only if the CTS pin (after optional inversion) is low.
22	Reserved	Must write reset value.
21	CTSINVEN	**CTS Invert Enable.** 0: The USART does not invert CTS. 1: The USART inverts CTS.
20:19	Reserved	Must write reset value.

Table 36.4. USARTn_FLOWCN Register Bit Descriptions

Bit	Name	Function
18	UCLK	**UCLK State.** This bit may be written when both the transmitter and receiver are disabled (TEN=REN=0) and in asynchronous mode (TSYNCEN=RSYCNEN=0). Note that the written value will not affect the UCLK output unless the USART is configured as a Master (OPMD=1) and idle tri-state is disabled (ITSEN=0). 0: The UCLK pin is low. 1: The UCLK pin is high.
17	TX	**TX State.** Firmware can write this bit to change the TX state only when the transmitter is disabled (TEN = 0 and the idle TX tristate enable bit is cleared (ITSEN=0). 0: The TX pin (before optional inversion) is low. 1: The TX pin (before optional inversion) is high.
16	CTS	**CTS State.** 0: Indicates the CTS pin state (after optional inversion) is low. 1: Indicates the CTS pin state (after optional inversion) is high.
15:8	Reserved	Must write reset value.
7	RTSEN	**RTS Enable.** 0: The RTS state is not changed by hardware. The RTS bit can be written only when hardware RTS is disabled (RTSEN = 0). 1: Hardware sets RTS when the receive FIFO is at or above the threshold set by RTSTH and clears RTS otherwise.
6	RTSTH	**RTS Threshold Control.** 0: RTS is de-asserted when the receive FIFO and shift register are full and no more incoming data can be stored. 1: RTS is de-asserted when the receive FIFO and shift register are nearly full and only one more data can be received.
5	RTSINVEN	**RTS Invert Enable.** 0: The USART does not invert the RTS signal before driving the pin. 1: The USART inverts the RTS signal driving the pin.
4:2	Reserved	Must write reset value.
1	RX	**RX Pin Status.** 0: RX pin (after optional inversion) is low. 1: RX pin (after optional inversion) is high.
0	RTS	**RTS State.** This bit is writeable only when RTSEN is cleared to 0. 0: RTS pin (before optional inversion) is driven low. 1: RTS pin (before optional inversion) is driven high.

Register 36.4. USARTn_CONTROL: Module Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	TEN	TINH	Reserved	TBIT	TBUSYF	Reserved			TCPTIEN	TDREQIEN	TERIEN	TCPTTH	TCPTI	TDREQI	TUREI	TSCERI
Type	RW	RW	R	RW	R	R			RW	RW	RW	RW	RW	R	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	REN	RINH	ROSEN	RBIT	RBUSYF	RABDEN	MATMD		Reserved	RDREQIEN	RERIEN	Reserved	RDREQI	ROREI	RPARERI	RFRMERI
Type	RW	RW	RW	RW	R	RW	RW		R	RW	RW	R	R	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

USART0_CONTROL = 0x4000_0030
USART1_CONTROL = 0x4000_1030
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 36.5. USARTn_CONTROL Register Bit Descriptions

Bit	Name	Function
31	TEN	**Transmitter Enable.** 0: Disable the transmitter. When cleared, the transmitter immediately aborts any active transmission. Clearing this bit does not automatically flush the transmit FIFO. 1: Enable the transmitter. The transmitter will initiate a transmission when data becomes available in the transmit FIFO.
30	TINH	**Transmit Inhibit.** 0: The transmitter operates normally. 1: Transmissions are inhibited. The transmitter will stall after any current transmission is complete.
29	Reserved	Must write reset value.

Notes:
1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 36.5. USARTn_CONTROL Register Bit Descriptions

Bit	Name	Function
28	TBIT	**Last Transmit Bit.** This bit is used to set the 9th bit during each FIFO write when TDATLN is set to 5. The TBIT value is stored in the FIFO during each FIFO write; the TBIT value at the time of transmission is not used. If TDATLN is not set to 5, the written bit from the core interface is used. This allows the DMA or the core to pack 9-bit outgoing data into bytes when the outgoing 9th bit is constant.
27	TBUSYF	**Transmitter Busy Flag.** 0: The USART transmitter is idle. 1: The USART transmitter is active and transmitting.
26:24	Reserved	Must write reset value.
23	TCPTIEN	**Transmit Complete Interrupt Enable.** 0: Disable the transmit complete interrupt. 1: Enable the transmit complete interrupt. A transmit interrupt is generated when TCPTI is set to 1.
22	TDREQIEN	**Transmit Data Request Interrupt Enable.** 0: Disable the transmit data request interrupt. 1: Enable the transmit data request interrupt. A transmit interrupt is asserted when TDREQI is set to 1.
21	TERIEN	**Transmit Error Interrupt Enable.** 0: Disable the transmit error interrupt. 1: Enable the transmit error interrupt. A transmit interrupt is generated when TUREI or TSCERI is set to 1.
20	TCPTTH	**Transmit Complete Threshold.** 0: A transmit is completed (TCPTI = 1) at the end of each transmission. 1: A transmit is completed (TCPTI = 1) only at the end of a transmission when no more data is available to transmit.
19	TCPTI	**Transmit Complete Interrupt Flag.** This bit is set by hardware if a byte is transmitted (TCCPTH = 0) or if the last available byte was transmitted (TCPTTH = 1). This bit must be cleared by firmware.
18	TDREQI	**Transmit Data Request Interrupt Flag.** 0: The transmitter is not requesting more FIFO data. 1: The transmitter is requesting more FIFO data.
17	TUREI	**Transmit Underrun Error Interrupt Flag.** Hardware sets this bit to 1 when a transmit FIFO underrun occurs. This bit must be cleared by firmware.
16	TSCERI	**Smartcard Parity Error Interrupt Flag.** This bit is set by hardware when a Smartcard parity error occurs. This bit must be cleared by firmware.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 36.5. USARTn_CONTROL Register Bit Descriptions

Bit	Name	Function
15	REN	**Receiver Enable.** This bit enables the USART receiver for multiple transactions. 0: Disable the receiver. The receiver can receive one data transaction only if ROSEN is set. 1: Enable the receiver.
14	RINH	**Receiver Inhibit.** 0: The receiver operates normally. 1: RTS is immediately asserted when RINH is set. The receiver will complete any ongoing reception, but ignore all traffic after that.
13	ROSEN	**Receiver One-Shot Enable.** 0: Disable one-shot receive mode. 1: Enable one-shot receive mode.
12	RBIT	**Last Receive Bit.** This bit is used according to the match mode (MATMD) setting.
11	RBUSYF	**Receiver Busy Flag.** 0: The USART receiver is idle. 1: The USART receiver is receiving data.
10	RABDEN	**Receiver Auto-Baud Enable.** 0: Disable receiver auto-baud. 1: Enable receiver auto-baud.
9:8	MATMD	**Match Mode.** The hardware automatically switches from MCE mode to Frame mode when a MCE match occurs. 00: Disable the match function. 01: (MCE) Data whose last data bit equals RBIT is accepted and stored. 10: (Frame) A framing error is asserted if the last received bit matches RBIT. 11: (Store) Store the last incoming data bit in RBIT. This mode can be used incon-junction with the RDATLN setting.
7	Reserved	Must write reset value.
6	RDREQIEN	**Receive Data Request Interrupt Enable.** 0: Disable the read data request interrupt. 1: Enable the read data request interrupt. A receive interrupt is generated when RDREQI is set to 1.
5	RERIEN	**Receive Error Interrupt Enable.** 0: Disable the receive error interrupt. 1: Enable the receive error interrupt. A receive error interrupt is asserted when ROREI, RFRMERI, or RPARERI is set to 1.
4	Reserved	Must write reset value.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 36.5. USARTn_CONTROL Register Bit Descriptions

Bit	Name	Function
3	RDREQI	**Receive Data Request Interrupt Flag.** 0: Fewer than RFTH FIFO slots are filled with data. 1: At least RFTH FIFO slots are filled with data.
2	ROREI	**Receive Overrun Error Interrupt Flag.** This bit is set to 1 by hardware when a receive overrun error occurs. This bit must be cleared by firmware.
1	RPARERI	**Receive Parity Error Interrupt Flag.** This bit is set to 1 by hardware when the receiver encounters a parity error. This bit must be cleared by firmware.
0	RFRMERI	**Receive Frame Error Interrupt Flag.** Hardware sets this bit to 1 when a receive frame error occurs. There are two sources for this error: when an expected whole stop bit is low, or when a frame match-mode fails. This bit must be cleared by firmware.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Register 36.5. USARTn_IPDELAY: Inter-Packet Delay

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved								IPDELAY							
Type	R								RW							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

USART0_IPDELAY = 0x4000_0040

USART1_IPDELAY = 0x4000_1040

Table 36.6. USARTn_IPDELAY Register Bit Descriptions

Bit	Name	Function
31:24	Reserved	Must write reset value.
23:16	IPDELAY	**Inter-Packet Delay.** This field configures the transmitter to delay between transmissions by the specified number of bit times. A value of 0 means no delay is added, and a value of 5 means 5 bit times are added. Bit times are configured in the TBAUD register. If in synchronous slave mode, UCLK must be running to use the inter-packet delay feature.
15:0	Reserved	Must write reset value.

Register 36.6. USARTn_BAUDRATE: Transmit and Receive Baud Rate

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	TBAUD															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	RBAUD															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses
USART0_BAUDRATE = 0x4000_0050
USART1_BAUDRATE = 0x4000_1050

Table 36.7. USARTn_BAUDRATE Register Bit Descriptions

Bit	Name	Function
31:16	TBAUD	**Transmitter Baud Rate Control.** This field sets the transmitter baud rate according to the equation: $\text{Baud Rate} = \frac{F_{APB}}{N \times (TBAUD + 1)}$ N = 2 if TIRDAEN = 0, and N = 16 if TIRDAEN = 1. Note that in synchronous mode, the transmitter baud rate must not exceed APBCLK / 16.
15:0	RBAUD	**Receiver Baud Rate Control.** This field sets the receiver baud rate according to the equation: $\text{Baud Rate} = \frac{F_{APB}}{N \times (RBAUD + 1)}$ N = 2 if RIRDAEN = 0, and N = 16 if RIRDAEN = 1. Note that in synchronous mode, the receiver baud rate must not exceed APBCLK / 16.

Register 36.7. USARTn_FIFOCN: FIFO Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved					TSRFULLF	TFERI	TFIFOFL	TDMAEN	Reserved	TFTH		Reserved	TCNT		
Type	R					R	RW	RW	RW	R	RW		R	R		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved					RSRFULLF	RFERI	RFIFOFL	RDMAEN	Reserved	RFTH		Reserved	RCNT		
Type	R					R	RW	RW	RW	R	RW		R	R		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

USART0_FIFOCN = 0x4000_0060

USART1_FIFOCN = 0x4000_1060

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 36.8. USARTn_FIFOCN Register Bit Descriptions

Bit	Name	Function
31:27	Reserved	Must write reset value.
26	TSRFULLF	**Transmit Shift Register Full Flag.** This bit indicates when the transmitter shift register contains data. The bit is set when hardware loads the data from the FIFO to the transmit shift register and is cleared when the transmitter completely transmits the data.
25	TFERI	**Transmit FIFO Error Interrupt Flag.** This bit is set when an illegal FIFO write is detected, such as a non right-justified write or a write that contains more data than will fit in the empty FIFO entries. When this bit is set, an interrupt will be asserted. 0: A transmit FIFO error has not occurred since TFERI was last cleared. 1: A transmit FIFO error occurred.
24	TFIFOFL	**Transmit FIFO Flush.** Setting this bit to 1 flushes the transmit FIFO. If data is pending in the transmit shift register but a transmit has not begun, the shift register is also flushed. This bit always reads as 0.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 36.8. USARTn_FIFOCN Register Bit Descriptions

Bit	Name	Function
23	TDMAEN	**Transmitter DMA Enable.** Enable the transmit FIFO DMA request. This request is generated according to the TFTH setting. 0: Disable transmit FIFO DMA requests. 1: Enable transmit FIFO DMA requests.
22	Reserved	Must write reset value.
21:20	TFTH	**Transmit FIFO Threshold.** This field sets the FIFO threshold at which a DMA transfer request or a TDREQI interrupt is asserted. 00: A DMA request or transmit data request interrupt (TDREQI) is asserted when ≥ 1 FIFO slot is empty. 01: A DMA request or transmit data request interrupt (TDREQI) is asserted when ≥ 2 FIFO slots are empty. 10: A DMA request or transmit data request interrupt (TDREQ) is asserted when ≥ 4 FIFO slots are empty. 11: Reserved.
19	Reserved	Must write reset value.
18:16	TCNT	**Transmit FIFO Count.** This field indicates the number of entries in the transmit FIFO.
15:11	Reserved	Must write reset value.
10	RSRFULLF	**Receive Shift Register Full Flag.** This bit indicates when the receiver shift register contains data and remains high until the data is moved to the FIFO or is flushed. The flag is set as soon as incoming data is completely received and cleared when the data is transferred to the FIFO.
9	RFERI	**Receive FIFO Error Interrupt Flag.** This bit is set when hardware detects an illegal FIFO read, such as a non right-justified read or a read that demands more data than is available in the FIFO. When this bit is set, an interrupt will be asserted. 0: A receive FIFO error has not occurred since RFERI was last cleared. 1: A receive FIFO error occurred.
8	RFIFOFL	**Receive FIFO Flush.** Setting this bit to 1 flushes the receive FIFO and any data in the receive shift register. This bit always reads as 0.
7	RDMAEN	**Receiver DMA Enable.** Enable the receive FIFO DMA request. This request is generated according to the RFTH setting. 0: Disable receive FIFO DMA requests. 1: Enable receive FIFO DMA requests.
6	Reserved	Must write reset value.

Notes:
1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 36.8. USARTn_FIFOCN Register Bit Descriptions

Bit	Name	Function
5:4	RFTH	**Receive FIFO Threshold.** This is the threshold at which a DMA transfer or a RDREQI interrupt is asserted. 00: A DMA request or read data request interrupt (RDREQI) is asserted when ≥ 1 FIFO slot is full. 01: A DMA request or read data request interrupt (RDREQI) is asserted when ≥ 2 FIFO slots are full. 10: A DMA request or read data request interrupt (RDREQ) is asserted when ≥ 4 FIFO slots are full. 11: Reserved.
3	Reserved	Must write reset value.
2:0	RCNT	**Receive FIFO Count.** This field indicates the number of entries in the receive FIFO.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Register 36.8. USARTn_DATA: FIFO Input/Output Data

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	DATA[31:16]															
Type	RW															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	DATA[15:0]															
Type	RW															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Register ALL Access Addresses

USART0_DATA = 0x4000_0070
USART1_DATA = 0x4000_1070

Table 36.9. USARTn_DATA Register Bit Descriptions

Bit	Name	Function
31:0	DATA	**FIFO Data.** The 32-bit DATA register is a single port into the transmit and receive FIFOs. Reads and writes should always be right-justified. Byte-wide reads and writes should always access DATA[7:0], half-word reads and writes should always access DATA[15:0], and word reads and writes should always access DATA[31:0]. Writes to the DATA register push data into the transmit FIFO. Reads from the DATA register pop data from the receive FIFO. Multiple FIFO entries can be pushed or popped in a single write or read. For example, if the transmit bit-length is less than 9 bits, writing a whole word (4 bytes) to the DATA field will write four entries in the FIFO where each byte is a single entry. However, if the FIFO doesn't have room for 4 entries, the write is completely ignored and the TFERI error flag is set. Similarly, a half-word write will push 2 entries to the FIFO if the data length is < 9. If the data length is 9, each half-word is a single entry. When writing or reading multiple bytes from the FIFO, the least significant byte is written to or read from the FIFO first.

Notes:
1. Reads of this register modify the state of hardware. Debug logic should take care when reading this register.

36.13. USARTn Register Memory Map

Table 36.10. USARTn Memory Map

Register Name	USARTn_CONTROL	USARTn_FLOWCN	USARTn_MODE	USARTn_CONFIG
ALL Offset	0x30	0x20	0x10	0x0
Access Methods	ALL | SET | CLR	ALL | SET | CLR	ALL | SET | CLR	ALL | SET | CLR
Bit 31	TEN	Reserved	OPMD	TSYNCEN
Bit 30	TINH		ITSEN	TINVEN
Bit 29	Reserved	TIRDAPW	CLKESEL	TIRDAEN
Bit 28	TBIT		CLKIDLE	TSCEN
Bit 27	TBUSYF	Reserved	DUPLEXMD	Reserved
Bit 26	Reserved		Reserved	TDATLN
Bit 25				
Bit 24				
Bit 23	TCPTIEN	CTSEN	ISTCLK	Reserved
Bit 22	TDREQIEN	Reserved	STRTSTCLK	TPARMD
Bit 21	TERIEN	CTSINVEN	STPSTCLK	
Bit 20	TCPTTH	Reserved	Reserved	TSTPMD
Bit 19	TCPTI		LBMD	
Bit 18	TDREQI	UCLK		TSTPEN
Bit 17	TUREI	TX	Reserved	TPAREN
Bit 16	TSCERI	CTS	DBGMD	TSTRTEN
Bit 15	REN	Reserved	Reserved	RSYNCEN
Bit 14	RINH			RINVEN
Bit 13	ROSEN			RIRDAEN
Bit 12	RBIT			RSCEN
Bit 11	RBUSYF			Reserved
Bit 10	RABDEN			RDATLN
Bit 9	MATMD			
Bit 8				
Bit 7	Reserved	RTSEN		Reserved
Bit 6	RDREQIEN	RTSTH		RPARMD
Bit 5	RERIEN	RTSINVEN		
Bit 4	Reserved	Reserved		RSTPMD
Bit 3	RDREQI			
Bit 2	ROREI			RSTPEN
Bit 1	RPARERI	RX		RPAREN
Bit 0	RFRMERI	RTS		RSTRTEN

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: USART0 = 0x4000_0000, USART1 = 0x4000_1000

Table 36.10. USARTn Memory Map

Register Name	USARTn_DATA	USARTn_FIFOCN	USARTn_BAUDRATE	USARTn_IPDELAY	Register Name
ALL Offset	0x70	0x60	0x50	0x40	ALL Offset
Access Methods	ALL	ALL | SET | CLR	ALL	ALL	Access Methods
Bit 31	DATA	Reserved	TBAUD	Reserved	Bit 31
Bit 30		TSRFULLF			Bit 30
Bit 29		TFERI			Bit 29
Bit 28		TFIFOFL			Bit 28
Bit 27		TDMAEN			Bit 27
Bit 26		Reserved			Bit 26
Bit 25		TFTH			Bit 25
Bit 24		Reserved			Bit 24
Bit 23		TCNT		IPDELAY	Bit 23
Bit 22					Bit 22
Bit 21					Bit 21
Bit 20					Bit 20
Bit 19					Bit 19
Bit 18					Bit 18
Bit 17					Bit 17
Bit 16					Bit 16
Bit 15		Reserved	RBAUD	Reserved	Bit 15
Bit 14		RSRFULLF			Bit 14
Bit 13		RFERI			Bit 13
Bit 12		RFIFOFL			Bit 12
Bit 11		RDMAEN			Bit 11
Bit 10		Reserved			Bit 10
Bit 9		RFTH			Bit 9
Bit 8		Reserved			Bit 8
Bit 7		RCNT			Bit 7
Bit 6					Bit 6
Bit 5					Bit 5
Bit 4					Bit 4
Bit 3					Bit 3
Bit 2					Bit 2
Bit 1					Bit 1
Bit 0					Bit 0

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: USART0 = 0x4000_0000, USART1 = 0x4000_1000

37. Universal Asynchronous Receiver/Transmitter (UART0 and UART1)

This section describes the UART module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the UART block, which is used by both UART0 and UART1 on all device families covered in this document.

37.1. UART Features

The UART module is the same as the USARTn module without support for the synchronous modes. The UART module includes the following features:

- Independent transmitter and receiver configurations with separate 16-bit baud-rate generators.
- Asynchronous transmissions and receptions.
- Up to 5 Mbaud (TX or RX) or 1 Mbaud Smartcard (TX or RX).
- Internal transmit and receive FIFOs with flush capability and support for byte, half-word, and word reads and writes.
- Data bit lengths from 5 to 9 bits.
- Programmable inter-packet transmit delays.
- Auto-baud detection with support for the LIN SYNC byte.
- Automatic parity generation (with enable).
- Automatic start and stop generation.
- Transmit and receive hardware flow-control.
- Independent inversion correction for TX, RX, RTS, and CTS signals.
- IrDA modulation and demodulation with programmable pulse widths.
- Smartcard ACK/NACK support.
- Parity error, frame error, overrun, and underrun detection.
- Multi-master and half-duplex support.
- Multiple loop-back modes supported.

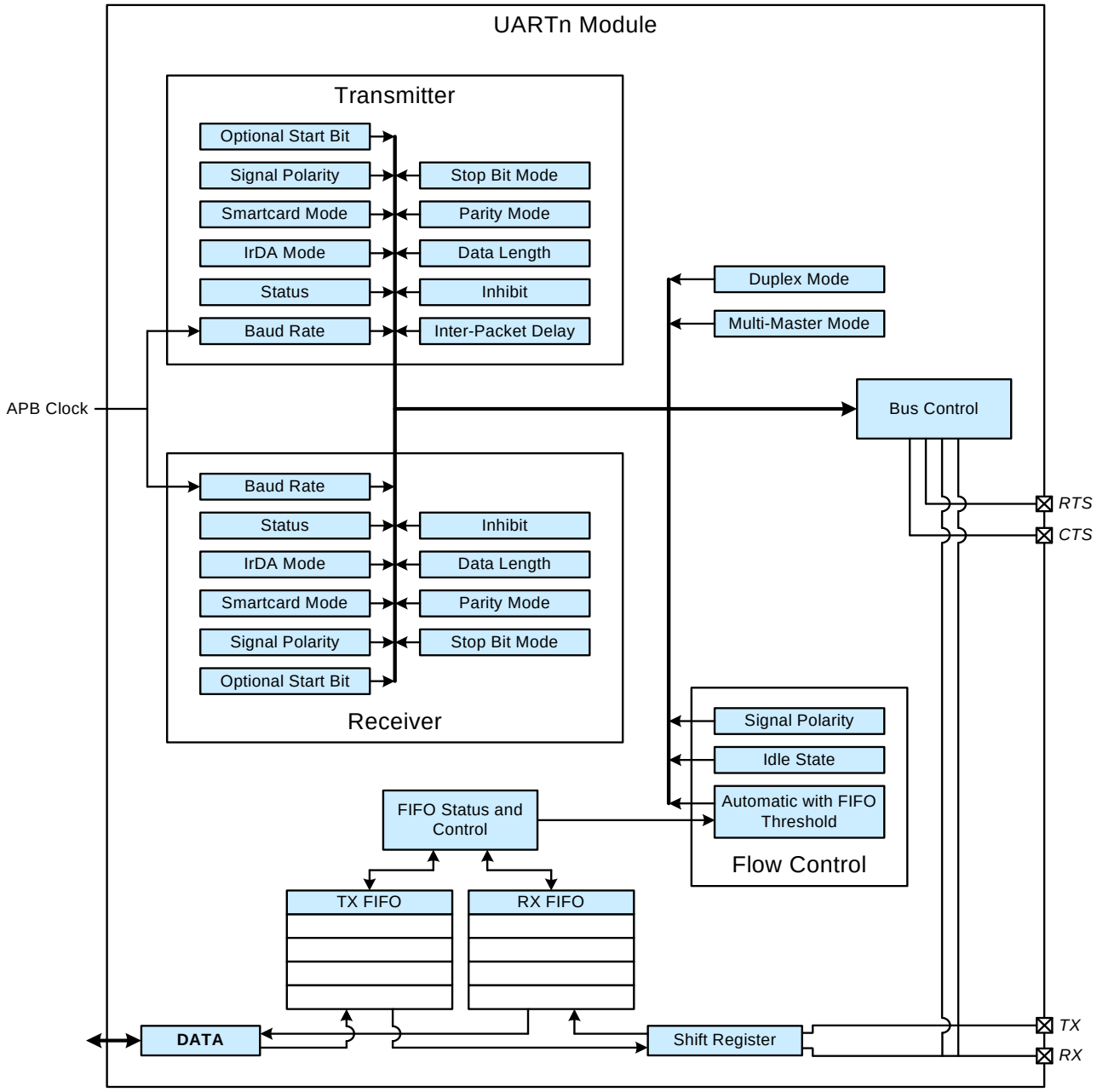


Figure 37.1. UART Block Diagram

37.2. Basic Data Format

The UART module data consists of four parts: start bit, data, parity bit, and stop bit.

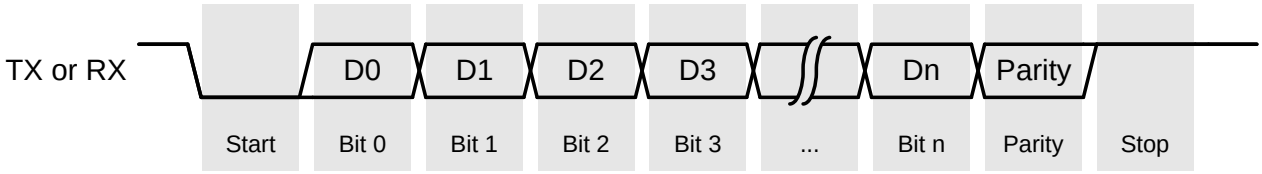


Figure 37.2. Basic Asynchronous UART Communication

Start bits are enabled using the transmitter start enable (TSTRTEN) and receiver start enable (RSTRTEN) bits.

The data length is variable and can be set to 5 to 9 bits using the transmitter data length (TDATLN) or receiver data length (RDATLN) bit fields.

The transmitter will transmit a parity bit when the transmitter parity enable (TPAREN) bit is set. The transmitter parity mode (TPARMD) field configures the parity for the transmit data as either odd, even, set (parity always set to 1), or clear (parity always cleared to 0).

Similarly, the receiver will expect to receive a parity bit when receiver parity is enabled (RPAREN = 1). The RPARMD field controls the receiver parity mode and sets it to odd, even, set (parity always expected to be set to 1), or clear (parity always expected to be cleared to 0).

The stop bits are also fully configurable using the transmitter stop enable (TSTPEN) and receiver stop enable (RSTPEN) bits. The transmitter stop mode (TSTPMD) and receiver stop mode (RSTPMD) fields configure the stop length to 0.5, 1, 1.5, or 2 bits. Transmitting partial stop bits (0.5 or 1.5) causes subsequent transmitted bits to slip by one-half of a bit-time.

37.3. Baud Rate

The UART can operate as two independent, one-way channels since the transmitter and the receiver each have their own internal baud rate generators. For example, the transmitter can operate at one baud rate while the receiver operates at a different baud rate.

In master mode, the transmitter baud rate control (TBAUD) bit field sets the transmitter baud rate, and the receiver baud rate control (RBAUD) bit field sets the receiver baud rate. The equations in the TBAUD and RBAUD field descriptions determine the transmitter or receiver baud rate.

37.3.1. Receiver Auto-Baud Detection

The hardware receiver auto-baud detection automatically derives the appropriate RBAUD value from the received data. When receive auto-baud detection is enabled (RABDEN = 1), the hardware requires the first incoming data to be a value that guarantees an RX state transition between each bit: 0x55 when in non-IrDA mode and 0x00 when in IrDA mode. The receiver auto-baud detection supports either 8- or 9-bit data lengths with any combination of parity or stop bits.

Figure 37.3 shows the UART receive auto-baud detection input timing.

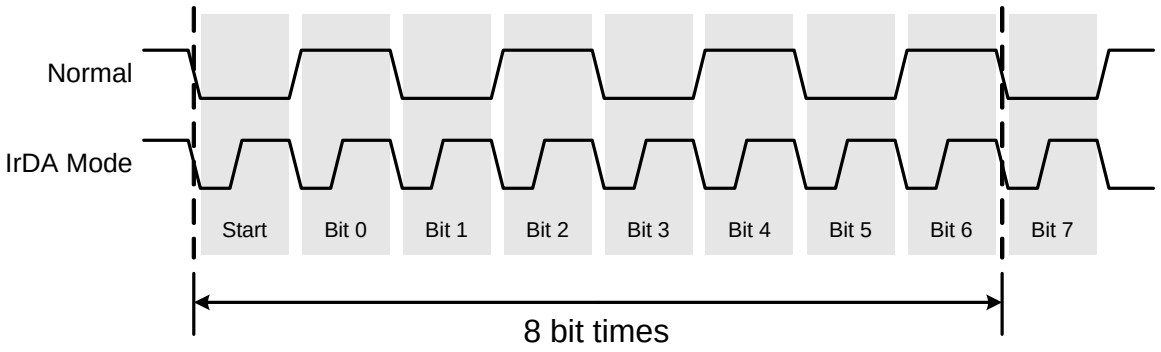


Figure 37.3. Receiver Auto-Baud Detection Input Timing

The receiver auto-baud detection calculates the appropriate value for RBAUD by counting the number of clock cycles between the leading edge of the start bit and the leading edge of the 8th data bit and dividing the clock cycle count by either 16 (when RIRDAEN = 0) or 128 (when RIRDAEN = 1). The hardware stores the calculated value to RBAUD and clears the receive auto-baud detection enable bit (RABDEN = 0). The new baud rate is used starting at the 8th bit of the incoming packet. The receiver will treat the received data (0x55 or 0x00) like any other received data and check it for parity errors and store it in the FIFO. If the first incoming byte causes the RBAUD value to overflow, indicating an auto-baud error, hardware will set the receive frame error interrupt (RFRMERI) flag regardless of the sampled state of the stop bits.

37.4. Interrupts

The UART module contains several interrupt sources that cause a vector to the UART interrupt. All of the interrupt flags may be masked by clearing a corresponding interrupt enable bit.

37.4.1. Transmit Interrupt Sources

The transmit data request interrupt (TDREQI) can be enabled by setting the TDREQIEN bit to 1. This flag indicates that the transmitter is requesting more FIFO data. The TDREQI flag automatically clears when the number of full entries in the transmit FIFO is more than the TFTH setting.

The transmit complete interrupt (TCPTI) flag indicates the following:

1. If TCPTTH = 0, this flag indicates a single transmit completed since TCPTI was last cleared.
2. If TCPTTH = 1, this flag indicates a transmit of the last available data in the FIFO completed since TCPTI was last cleared.

In either scenario, the hardware will generate an interrupt if the TCPTIEN bit is set to 1 when TCPTI is set.

The Smartcard parity error interrupt (TSCERI) flag sets when the hardware detects the TX pin state as 0 at the end of the first whole stop bit period when operating in Smartcard mode (TSCEN = 1). Both of these interrupts are enabled with the transmit error interrupt enable (TERIEN) bit. Transmissions are automatically inhibited when any enabled error flag causes the interrupt to assert.

Finally, the transmitter has a transmit FIFO error interrupt (TFERI) flag which is set whenever an illegal FIFO write (e.g, a write to a full FIFO) is detected. Note that TFERI does not have an interrupt enable/disable control bit, so if the TFERI flag is set an interrupt will always be generated (assuming the UART module interrupt is enabled).

37.4.2. Receive Interrupt Sources

The receive data request interrupt (RDREQI) flag indicates that at least the number of receive FIFO slots set by the receive FIFO threshold (RFTH) are full, and firmware can read data from the receive FIFO. This interrupt is enabled by setting the receive data request interrupt enable (RDREQIEN) to 1. This flag automatically clears when the number of filled slots in the receive FIFO drops below the RFTH setting.

The receiver has three error interrupt sources. The receive overrun error interrupt (ROREI) flag indicates when the receive FIFO and shift register is full, and the shift register has been erroneously overwritten by additional receive data. The receive parity error interrupt (RPARERI) indicates that an invalid parity bit has been received. Finally, the receive frame error interrupt (RFRMERI) indicates when an expected whole stop bit is low, when a frame match mode fails, or when an auto-baud error occurs. The receive error interrupt enable (RERIEN) bit enables these flags as interrupt sources.

Finally, the receiver has a receive FIFO error interrupt (RFERI) flag which is set whenever an illegal FIFO read (e.g, a read from an empty FIFO) is detected. Note that RFERI does not have an interrupt enable/disable control bit, so if the RFERI flag is set an interrupt will always be generated (assuming the UART module interrupt is enabled).

37.5. Flow Control

37.5.1. Transmitter Hardware Flow Control

The CTS enable (CTSEN) bit enables hardware flow control in the transmitter. When CTS is enabled, the transmitter will only begin transmissions if the CTS input is low after the optional inversion using the CTSINVEN bit.

Firmware can read the current state of the CTS pin using the CTS bit. The CTS value read will be inverted from the CTS pin if the signal is inverted (CTSINVEN = 1).

37.5.2. Receiver Hardware Flow Control

Firmware can set the RTS enable (RTSEN) bit to enable hardware flow control in the receiver. When flow control is enabled, the receiver asserts RTS under any of the following conditions:

1. The receiver is disabled (REN = 0).
2. The receiver is inhibited (RINH = 1).
3. The receiver is stalled during a debug halt (DBGMD = 1).
4. An error flag sets causing the receive error interrupt to assert (RERIEN = 1 and RFRMERI, ROREI, or RPARERI is also set).
5. The receiver can store zero (if RTSTH = 0) or up to one (if RTSTH = 1) more data.

The RTS signal holds its last value when disabled by clearing RTSEN to 0. The RTS bit allows firmware to read the state of the RTS signal. Firmware can also write the RTS bit when flow control is disabled (RTSEN = 0) to set the state of the RTS pin. Note that the value written into the RTS bit will be inverted at the RTS pin if the RTS invert enable (RTSINVEN) bit is set to 1.

37.5.3. Inter-Packet Delay Generator

The transmitter supports a configurable delay between transmissions that may be used to limit the transmission rate in applications that do not support hardware flow control.

The hardware calculates the inter-packet delay using a counter operating at the current baud rate. As a result, the IPDELAY field represents multiples of the transmitter bit times. For example, setting the inter-packet delay (IPDELAY) field to 5 results in a delay equal to 5 bit times. Setting the IPDELAY field to 0 disables the inter-packet delay.

37.6. Debug Mode

Firmware can set the DBGMD bit to force the UART module to halt on a debug breakpoint. The module will complete any active transmissions and receptions before stopping. Clearing the DBGMD bit forces the UART module to continue operating while the core halts in debug mode. Note that when IPDELAY is set to zero, the UART may not halt until the FIFO is empty. If IPDELAY is non-zero, the UART will complete the current transfer but will not empty the FIFO.

37.7. Sending Data

To begin a transmit operation, firmware should set all the necessary transmitter configuration bits, enable the transmitter (TEN = 1), and write the outgoing data to the DATA register. Hardware will automatically transfer the data from the DATA register to the transmit FIFO. A FIFO entry is immediately loaded from the FIFO into the shift register any time data is available in the transmit FIFO and the shift register is empty.

The data transmission begins when all of the following conditions are met:

1. The shift register is loaded with data.
2. The transmitter is enabled (TEN = 1).
3. The transmitter is not inhibited (TINH = 0).
4. The CTS pin is deasserted if flow control is enabled (CTSEN = 1).
5. The module is not halted because of a debug breakpoint if DBGMD is set to 1.

37.7.1. Transmitter FIFO Management

The UART transmit FIFO is implemented as a circular buffer with four entries, allowing data to be pushed as a continuous stream. The DATA register forms a single port into the transmit and receive FIFOs. Writes to the DATA register push data into the transmit FIFO, and reads from the DATA register pop data from the receive FIFO.

The transmitter supports byte, half-word, or word writes to the FIFO. Writes to the FIFO should always be right-justified, so byte-wide writes should always write DATA[7:0], half-word writes should always write DATA[15:0], and word writes should always write DATA[31:0]. The transmitter ignores all other writes to the DATA register, including left-justified byte writes or writes containing more data than will fit in the empty slots in the FIFO. Any illegal FIFO write sets the transmit FIFO error interrupt flag (TFERI) and causes an interrupt, if enabled.

The TDATLN bit controls how the written data is mapped into the FIFO. The transmitter supports data lengths ranging from 5 to 9 bits; the optional start, parity, and stop bits are not stored in the FIFO. When transmitting less than 9 bits of data, the hardware transmits all 5 to 8 bits written to the FIFO by firmware.

When transmitting 9-bit data, the transmitter supports two modes of operation:

1. In normal 9-bit mode (TDATLN = 4), firmware writes 9-bit data to the FIFO with each half-word containing a right-justified 9-bit entry.
2. In fixed 9-bit mode (TDATLN = 5), bus writes to the FIFO are assumed to contain only the least-significant 8 bits of each data entry, and hardware inserts the 9th bit value (set by TBIT) into the FIFO on each FIFO write.

When the data length is 8 bits (or 9 bits when using TDATLN = 5), one, two, or four FIFO entries can be written in a single bus operation by using byte, half-word, or word operations, respectively. For example, writing a whole word (4 bytes) to DATA[31:0] will write four entries in the FIFO with each byte as an entry.

When the data length is 9 bits (using TDATLN = 4), two FIFO entries can be written in a single bus operation by using a word operation, or one entry can be written using a half-word operation. With a half-word write, the hardware will push DATA[8:0] to the FIFO. With a word write, the hardware will push DATA[8:0] to the FIFO, followed by the value in DATA[24:16]. The other bits in the DATA register in this mode are unused and have no effect.

The transfer FIFO count (TCNT) field maintains a count of the number of occupied entries in the transmit FIFO. Firmware may read TCNT to determine the allowable width of writes (byte, half-word, or word) given the available number of empty FIFO entries.

The transmit FIFO threshold (TFTH) field configures the threshold at which the hardware asserts the transmit data request interrupt (TDREQI) and may be configured to 1, 2, or 4 empty FIFO slots.

Firmware can set the transmit FIFO flush (TFIFOFL) bit to flush the contents of the transmit FIFO. A FIFO flush will also clear the contents of the shift register if the transmitter is idle and is not actively transmitting data.

37.7.2. Transmitter Status

The UART module has two status bits which may be monitored to determine the status of the transmitter.

37.7.2.1. Transmitter Busy Flag (TBUSYF)

The transmitter busy flag (TBUSYF) will be asserted when a single byte transmission is in progress. Note that if transmitted multiple bytes with IPDELAY>0, TBUSYF will return to 0 between each transmitted byte. For this reason, TBUSYF is not a reliable indicator of a completed transmission, and the method described below using the TCPTI bit is recommended.

37.7.2.2. Transmitter Complete Interrupt Flag (TCPTI)

The transmit complete interrupt (TCPTI) flag indicates the following:

1. If TCPTTH = 0, this flag indicates a single transmit completed since TCPTI was last cleared.
2. If TCPTTH = 1, this flag indicates a transmit of the last available data in the FIFO completed since TCPTI was last cleared.

To determine when the transmitter has completed transmission of all data in the FIFO, firmware may set TCPTTH=1, write the data into the FIFO, and then wait for the TCPTI bit to be asserted. Note that TCPTI must be manually cleared by firmware.

37.8. Receiving Data

Firmware should first configure all necessary receiver configuration bits before enabling the receiver (REN = 1). Incoming data will be de-glitched and then shifted into the 10-bit receive shift register. When the data reception is complete, the data will be aligned and padded according to the data length specified in the RDATLN bit field, and then pushed into the circular receive FIFO buffer if an empty entry is available. When both the receive FIFO and the shift register are full, any subsequent received data replaces the shift register contents and causes the hardware to set the receive overrun error interrupt (ROREI) flag.

Firmware may read from the FIFO any time full entries are available regardless of the state of the receiver.

The receiver supports one-shot receptions, where the receiver accepts a single byte. Firmware can enable this by setting the ROSEN bit. The hardware automatically clears this bit after the next reception is completed and accepted. If MATMD is set to 1 for MCE mode and a match does not occur, the hardware will not accept the data and will not clear the ROSEN bit.

37.8.1. Receiver FIFO Management

The receive FIFO is implemented as a circular buffer with four entries. The receive and transmit FIFOs are both accessed using the same DATA register; a read from the DATA register pops data from the receive FIFO, and a write to the DATA register pushes data into the transmit FIFO. When reading from the FIFO, the least-significant entry is popped from the FIFO first. The receiver supports byte, half-word, or word bus reads from the FIFO.

The RDATLN field determines how the data is mapped into the FIFO. The receiver supports data lengths ranging from 5 to 9 bits; the optional start, parity, and stop bits are not stored in the FIFO. When receiving less than 9 bits, a firmware read from the FIFO returns all 5 to 8 bits of the receive data.

When receiving 9-bit data, the receiver supports two modes of operation:

1. In normal 9-bit mode (RDATLN = 4), firmware reads 9-bit data from the FIFO, with each half-word containing a right-justified 9-bit entry.
2. In fixed 9-bit mode (RDATLN = 5), only the least-significant 8 bits of received data are stored in the FIFO. The 9th bit is compared against RBIT to generate an interrupt or set error flags according to the MATMD field.

When the data length is 8 bits (or 9 bits when RDATLN = 5), firmware can read one, two, or four FIFO entries in a single bus operation by using byte, half-word, or word operations, respectively.

When the data length is 9 bits (using normal 9-bit mode with RDATLN = 4), each half-word is treated as a right-justified entry. When reading 9-bit data using a half-word bus operation, the hardware places the first entry popped from the FIFO into DATA[8:0]. When reading a whole word, hardware places the first entry popped from the FIFO into DATA[8:0] and the second entry into DATA[24:16]. The rest of the DATA bits are unused in this mode and are set to zero.

The receive FIFO count (RCNT) field maintains a count of the number of occupied entries in the receive FIFO. Firmware may read RCNT to determine the allowable read widths given the available number of full FIFO entries. The receive FIFO threshold (RFTH) field configures the threshold at which hardware asserts a read request interrupt (RDREQI) and may be configured to 1, 2, or 4 occupied slots.

Firmware can set the receive FIFO flush (RFIFOFL) bit to flush the contents of the receive FIFO. A FIFO flush will also clear the contents of the shift register if the receiver is idle and is not actively receiving data.

37.8.2. Receiver Status

The UART module includes a status bit which may be monitored to determine the status of the receiver.

37.8.2.1. Receiver Busy Flag (RBUSYF)

The receiver busy flag (RBUSYF) will be asserted when a single byte receive is in progress. Note that when receiving multiple bytes, any delay between the bytes may cause RBUSYF to return to 0 between each received byte.

37.9. Additional Communication Support

37.9.1. Multi-Master Mode

The transmitter supports applications in which multiple transmitters are driving the same TX signal. This multi-master mode is suited for applications using a Smartcard or LIN, which perform one-wire, half-duplex communications.

Multi-master mode requires the use of idle signal tristates (ITSEN = 1). When this bit is set to 1 and the transmitter is idle, the transmitter automatically tristates the TX pin when not transmitting. The TX pin must have an external pull-up resistor to ensure the pin remains in an idle state when tristated.

37.9.2. Multi-Processor Communications

In a multi-processor application (multiple slaves), a master first transmits an address byte to select the target. In an address byte, the 9th bit is always set to a logic 1; in a data byte, the 9th bit is always set to logic 0. The UART module supports multi-processor communication through the use of match operations. Although the match operations are most commonly used with the 9th data bit as an address indicator, the match operations are available regardless of the actual configured data length.

Match operations are enabled or disabled in the receiver by configuring the match mode (MATMD) bit field.

When MATMD is set to 1, the hardware operates in MCE mode and only accepts and stores the incoming receive data if the last data bit matches the value of RBIT. Otherwise, the hardware ignores incoming data, no interrupts are generated, and no FIFO activity occurs. This mode can be used to only accept data when the last data bit marks the frame as containing an address value. Firmware would then compare this address against an assigned value and reconfigure the receiver as needed if the values match. The MATMD setting is automatically switched to Frame mode after a match occurs in the MCE setting.

In Frame mode (MATMD = 2), all incoming data is accepted and stored, but a receive frame error (RFRMERI) asserts if the last data bit matches the value of RBIT. This operation can be used to assert an error when an unexpected address frame arrives.

In Store mode (MATMD = 3), the hardware accepts all incoming data and stores the last data bit in the RBIT field. The last data bit may also be stored in the FIFO depending on the value of RDATLN. For example, if RDATLN is set to 9 bits with the 9th bit stored in the FIFO (RDATLN = 4), and MATMD is set to store, the 9th data bit will be stored in both RBIT and the FIFO.

37.9.3. IrDA

The UART module transmitter and receiver support simple, asynchronous communications with IrDA devices.

37.9.3.1. IrDA Modulation

Firmware can place the transmitter in IrDA mode by setting the TIRDAEN bit. In IrDA mode, the transmitter performs a RZ (return-to-zero) modulation on the TX signal.

By default, transmitting a 0 causes a pulse to be generated low for a fraction of the bit time and transmitting a 1 leaves TX high without generating a low pulse. The transmit IrDA pulse width (TIRDAPW) bit field configures the pulse width as either 1/16, 1/8, 3/16, or 1/4 of the bit time.

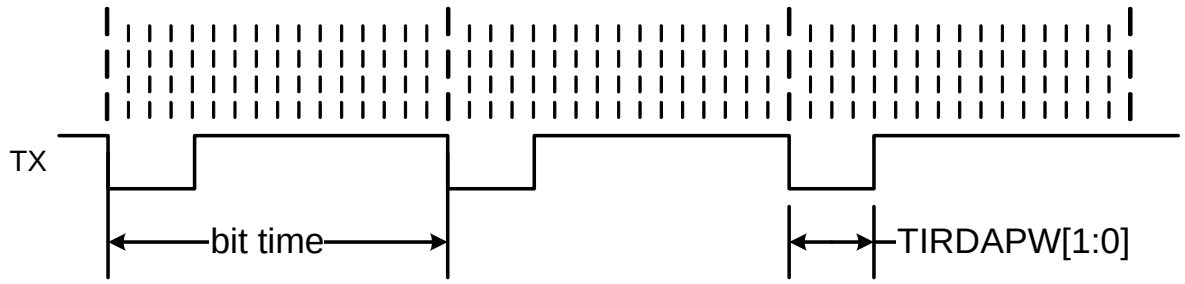


Figure 37.4. UART Transmit IrDA Pulse Width Timing

As shown in Figure 37.4, the default TX idles high and generates a low pulse when a 0 is transmitted. The polarity of TX may be inverted by setting the transmitter invert enable (TINVEN) bit, causing TX to idle low and generate a high pulse when sending a 0.

37.9.3.2. IrDA Demodulation

Setting the receiver IrDA enable (RIRDAEN) to 1 places the receiver in IrDA mode. In this mode, the receiver performs a simple RZ-to-NRZ (return-to-zero to non-return-to-zero) demodulation on the RX signal.

37.9.4. Smartcard Mode

37.9.4.1. Smartcard Transmitter

The transmitter Smartcard mode is enabled by setting the transmitter Smartcard parity response enable (TSCEN = 1) and consists of the ability to capture the ACK or NACK response from a Smartcard during the stop bit periods. Generally, Smartcard communications will also use multi-master and half-duplex modes with the TX and RX pins tied together externally.

Figure 37.5 illustrates the signaling on the TX pin in Smartcard mode.

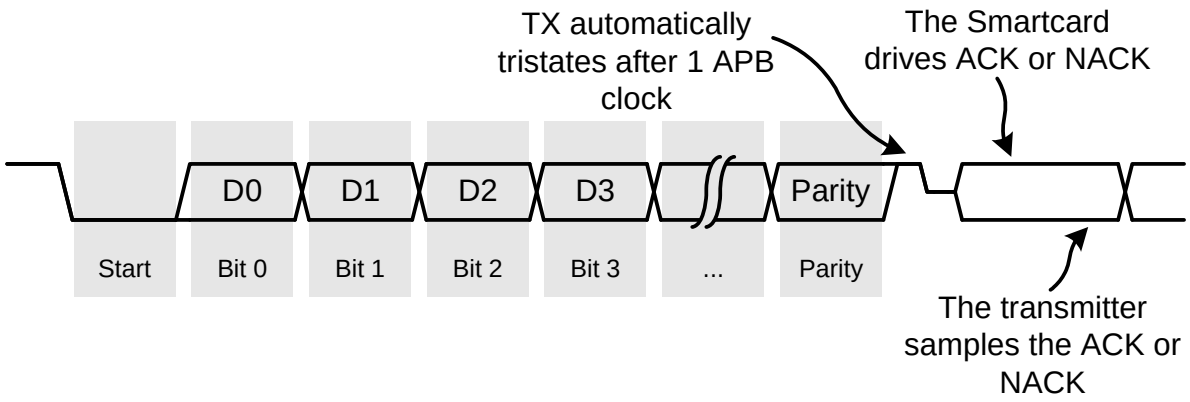


Figure 37.5. Smartcard Transmit Signalling

The transmitter should be configured to transmit 2 stop bits (or 1.5 stop bits if inter-packet delays are enabled) in Smartcard mode. The transmitter will tristate the TX pin one APB clock cycle after the start of the first stop bit. The Smartcard is expected to drive the TX pin low starting at the mid-point of the first stop bit if a receive error occurred. The transmitter captures the state of the TX pin at the end of the first stop bit period and sets the Smartcard parity error interrupt (TSCERI) flag accordingly. An interrupt is generated if the error flag is set and the transmit error interrupt is enabled (TERIEN = 1). The transmitter will re-enable its TX pin driver at the end of the stop bits except when operating in multi-master mode.

37.9.4.2. Smartcard Receiver

The receiver Smartcard mode is enabled by setting receiver Smartcard parity response (RSCEN) to 1 and allows the receiver to return an ACK or NACK response to a Smartcard during the stop bit periods. Figure 37.6 shows signaling on the RX pin in Smartcard mode.

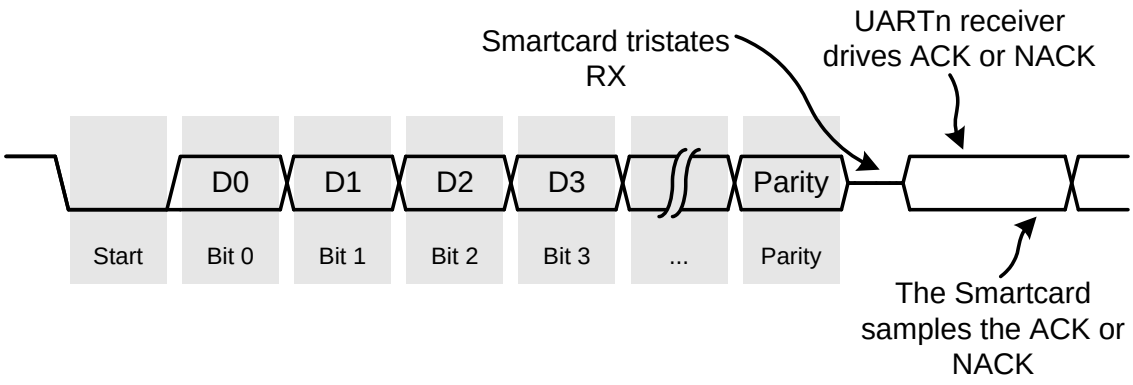


Figure 37.6. Smartcard Receive Signalling

When using Smartcard mode, firmware should configure the receiver to expect 2 stop bits. The receiver will drive RX low if a parity error occurs or high if no parity error occurs for one bit-time starting at the mid-point of the first stop bit. The Smartcard is expected to sample the state of RX at the end of the first stop bit.

37.9.5. LIN Support

The UART transmitter and receiver can be used with LIN (Local Interconnect Network) systems.

When using LIN, firmware should configure the transmitter to use 8 data bits, 1 start bit, 1 stop bit, and no parity bits.

The receiver auto-baud detection also supports the LIN SYNC byte (0x55).

37.9.6. Half Duplex Mode

The UART module supports applications in which half-duplex communications occur across a shared TX/RX signal. In half-duplex mode, the hardware automatically inhibits the receiver during transmit operations and the transmitter during receive operations.

The duplex mode bit (DUPLEXMD) enables half-duplex mode for the module. Firmware should also enable idle tristates (ITSEN = 1) to force the transmitter to tristate the TX signal when not transmitting. The RX and TX pins must be shorted externally in half-duplex mode.

37.9.7. Loop Back Support

The UART module supports several internal loop back options for testing purposes. The loop back modes are configured in the loop back mode (LBMD) field. Table 37.1 describes the different loop back options.

Table 37.1. Internal Loop Back Modes

LBMD value	Loop Back Mode
0	Disabled
1	Receive Loop Back
2	Transmit Loop Back
3	Full Loop Back

When loop back mode is disabled, the module operates normally.

In receive loop back mode, the receiver input path is disconnected from the RX signal and internally connected to the transmitter. Any data transmitted in this mode will be sent out on the TX signal and also received by the device. The physical RX pin tristates in this mode. TX, CTS, and RTS are connected to the corresponding external pins, if the signals are enabled to connect to pins in the device's port configuration.

With transmit loop back, the transmitter output path is disconnected from the TX signal, and the RX input signal is internally looped back to the TX pin. Data received on the RX signal will be received by the device and also sent directly back out on the TX pin. RX, CTS, and RTS are connected to the corresponding external pins, if enabled.

In full loop back mode, the transmitter output is internally routed back to the receiver input. Neither the transmitter nor receiver are connected to external device pins. If enabled, the RX device pin is similarly looped back to the TX pin. Any data transmitted on TX will be sent directly back in on RX. The CTS input and RTS output pins are likewise looped back at both the transmitter and receiver and at the device pins, if enabled.

Figure 37.7 illustrates the internal connections for each loop back mode.

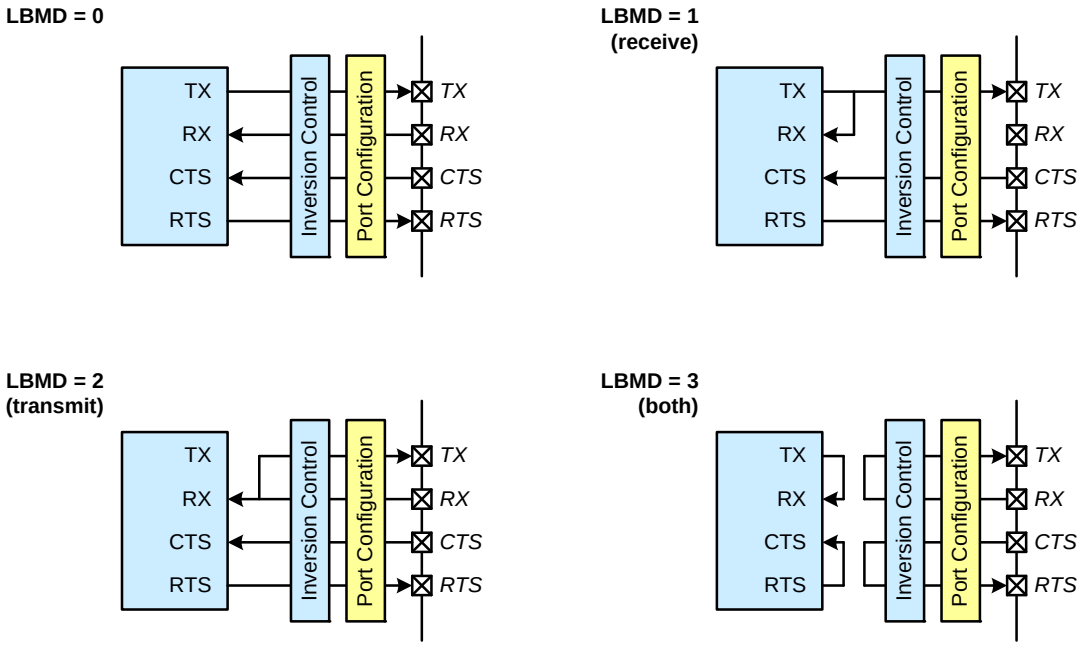


Figure 37.7. Internal Loop Back Connections

37.10. UART0 and UART1 Registers

This section contains the detailed register descriptions for UART0 and UART1 registers.

Register 37.1. UARTn_CONFIG: Module Configuration

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved	TINVEN	TIRDAEN	TSCEN	Reserved	TDATLN			Reserved	TPARMD		TSTPMD		TSTPEN	TPAREN	TSTRTEN
Type	RW	RW	RW	RW	R	RW			R	RW		RW		RW	RW	RW
Reset	0	0	0	0	0	0	1	1	0	0	0	0	1	1	0	1
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved	RINVEN	RIRDAEN	RSCEN	Reserved	RDATLN			Reserved	RPARMD		RSTPMD		RSTPEN	RPAREN	RSTRTEN
Type	RW	RW	RW	RW	R	RW			R	RW		RW		RW	RW	RW
Reset	0	0	0	0	0	0	1	1	0	0	0	0	1	1	0	1

Register ALL Access Addresses

UART0_CONFIG = 0x4000_2000
UART1_CONFIG = 0x4000_3000
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 37.2. UARTn_CONFIG Register Bit Descriptions

Bit	Name	Function
31	Reserved	Must write reset value.
30	TINVEN	**Transmitter Invert Enable.** 0: Do not invert the TX pin signals (the TX idle state is high). 1: Invert the TX pin signals (the TX idle state is low).
29	TIRDAEN	**Transmitter IrDA Enable.** 0: Disable IrDA transmit mode. 1: Enable IrDA transmit mode.
28	TSCEN	**Transmitter Smartcard Parity Response Enable.** 0: The transmitter does not check for a Smartcard parity error response. 1: The transmitter checks for a Smartcard parity error response.
27	Reserved	Must write reset value.

Table 37.2. UARTn_CONFIG Register Bit Descriptions

Bit	Name	Function
26:24	TDATLN	**Transmitter Data Length.** Select the number of data bits sent during transmission. 000: 5 bits. 001: 6 bits. 010: 7 bits. 011: 8 bits. 100: 9 bits. The 9th bit is taken from the FIFO data (normal mode). 101: 9 bits. The 9th bit is set by the value of TBIT (fixed mode). 110-111: Reserved.
23	Reserved	Must write reset value.
22:21	TPARMD	**Transmitter Parity Mode.** This field selects the type of parity sent during transmissions. 00: Odd Parity. 01: Even Parity. 10: Set (Parity = 1). 11: Clear (Parity = 0).
20:19	TSTPMD	**Transmitter Stop Mode.** This bit selects the transmitted stop bit length. 00: 0.5 stop bit. 01: 1 stop bit. 10: 1.5 stop bits. 11: 2 stop bits.
18	TSTPEN	**Transmitter Stop Enable.** 0: Do not send stop bits during transmissions. 1: Send stop bits during transmissions.
17	TPAREN	**Transmitter Parity Enable.** 0: Do not send a parity bit during transmissions. 1: Send a parity bit during transmissions.
16	TSTRTEN	**Transmitter Start Enable.** 0: Do not generate a start bit during transmissions. 1: Generate a start bit during transmissions.
15	Reserved	Must write reset value.
14	RINVEN	**Receiver Invert Enable.** 0: Do not invert the RX pin signals (the RX idle state is high). 1: Invert the RX pin signals (the RX idle state is low).
13	RIRDAEN	**Receiver IrDA Enable.** 0: The receiver does not operate in IrDA mode. 1: The receiver operates in IrDA mode.
12	RSCEN	**Receiver Smartcard Parity Response Enable.** 0: The receiver does not send a Smartcard parity error response. 1: The receiver sends a Smartcard Parity response.

Table 37.2. UARTn_CONFIG Register Bit Descriptions

Bit	Name	Function
11	Reserved	Must write reset value.
10:8	RDATLN	**Receiver Data Length.** Select the expected length of received data. 000: 5 bits. 001: 6 bits. 010: 7 bits. 011: 8 bits. 100: 9 bits. The 9th bit is stored in the FIFO (normal mode). 101: 9 bits. The 9th bit is not stored in the FIFO (fixed mode). This mode is used when the 9th bit is only used for match operations (see MATMD). 110-111: Reserved.
7	Reserved	Must write reset value.
6:5	RPARMD	**Receiver Parity Mode.** This field selects the type of parity expected during reception. 00: Odd Parity. 01: Even Parity. 10: Set (Parity = 1). 11: Clear (Parity = 0).
4:3	RSTPMD	**Receiver Stop Mode.** Select the number of stop bits expected during reception. 00: 0.5 stop bit. 01: 1 stop bit. 10: 1.5 stop bits. 11: 2 stop bits.
2	RSTPEN	**Receiver Stop Enable.** 0: Do not expect stop bits during receptions. 1: Expect stop bits during receptions.
1	RPAREN	**Receiver Parity Enable.** 0: Do not expect a parity bit during receptions. 1: Expect a parity bit during receptions.
0	RSTRTEN	**Receiver Start Enable.** 0: Do not expect a start bit during receptions. 1: Expect a start bit during receptions.

Register 37.2. UARTn_MODE: Module Mode Select

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved	ITSEN	Reserved		DUPLEXMD	Reserved							LBMD		Reserved	DBGMD
Type	RW	RW	RW		RW	R			RW			R	RW		R	RW
Reset	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses
UART0_MODE = 0x4000_2010
UART1_MODE = 0x4000_3010
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 37.3. UARTn_MODE Register Bit Descriptions

Bit	Name	Function
31	Reserved	Must write reset value.
30	ITSEN	**Idle TX Tristate Enable.** 0: The TX and UCLK (if in synchronous master mode) pins are always an output in this mode, even when idle. 1: The TX pin is tristated when idle. If ISTCLK is cleared to 0 and the transmitter is configured in synchronous master mode, the UCLK pin will also be tristated when idle.
29:28	Reserved	Must write reset value.
27	DUPLEXMD	**Duplex Mode.** If this bit is set to 1, the ITSEN bit must also be set to 1 to tristate the TX signal during receive operations. 0: Full-duplex mode. The transmitter and receiver can operate simultaneously. 1: Half-duplex mode. The transmitter automatically inhibits when the receiver is active and the receiver automatically inhibits when the transmitter is active.
26:20	Reserved	Must write reset value.

Table 37.3. UARTn_MODE Register Bit Descriptions

Bit	Name	Function
19:18	LBMD	**Loop Back Mode.** Select from different internal loop-back options for diagnostic and debug purposes. 00: Loop back is disabled and the TX and RX signals are connected to the corresponding external pins. 01: Receive loop back. The receiver input path is disconnected from the RX pin and internally connected to the transmitter. Data transmitted will be sent out on TX and also received by the device. 10: Transmit loop back. The transmitter output path is disconnected from the TX pin and the RX input pin is internally looped back out to the TX pin. Data received at RX will be received by the device and also sent directly back out on TX. 11: Full loop back. Internally, the transmitter output is routed back to the receiver input. Neither the transmitter nor receiver are connected to external device pins. The device pin RX is looped back to TX in a similar fashion. Data transmitted on TX will be sent directly back in on RX.
17	Reserved	Must write reset value.
16	DBGMD	**UART Debug Mode.** 0: The UART module will continue to operate while the core is halted in debug mode. 1: A debug breakpoint will cause the UART module to halt. Any active transmissions and receptions will complete first.
15:0	Reserved	Must write reset value.

Register 37.3. UARTn_FLOWCN: Flow Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved		TIRDAPW		Reserved				CTSEN	Reserved	CTSINVEN	Reserved			TX	CTS
Type	R		RW		R				RW	R	RW	R		RW	RW	R
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	X	1	X
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved								RTSEN	RTSTH	RTSINVEN	Reserved			RX	RTS
Type	R								RW	RW	RW	R			R	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	X	1

Register ALL Access Addresses
UART0_FLOWCN = 0x4000_2020
UART1_FLOWCN = 0x4000_3020
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 37.4. UARTn_FLOWCN Register Bit Descriptions

Bit	Name	Function
31:30	Reserved	Must write reset value.
29:28	TIRDAPW	**Transmit IrDA Pulse Width.** This field sets the IrDA pulse width as a fraction of the bit period. 00: The IrDA pulse width is 1/16th of a bit period. 01: The IrDA pulse width is 1/8th of a bit period. 10: The IrDA pulse width is 3/16th of a bit period. 11: The IrDA pulse width is 1/4th of a bit period.
27:24	Reserved	Must write reset value.
23	CTSEN	**CTS Enable.** 0: The CTS pin state does not affect transmissions. 1: Transmissions will begin only if the CTS pin (after optional inversion) is low.
22	Reserved	Must write reset value.
21	CTSINVEN	**CTS Invert Enable.** 0: The UART does not invert CTS. 1: The UART inverts CTS.
20:18	Reserved	Must write reset value.

Table 37.4. UARTn_FLOWCN Register Bit Descriptions

Bit	Name	Function
17	TX	**TX State.** Firmware can write this bit to change the TX state only when the transmitter is disabled (TEN = 0). 0: The TX pin (before optional inversion) is low. 1: The TX pin (before optional inversion) is high.
16	CTS	**CTS State.** 0: Indicates the CTS pin state (after optional inversion) is low. 1: Indicates the CTS pin state (after optional inversion) is high.
15:8	Reserved	Must write reset value.
7	RTSEN	**RTS Enable.** 0: The RTS state is not changed by hardware. The RTS bit can be written only when hardware RTS is disabled (RTSEN = 0). 1: Hardware sets RTS when the receive FIFO is at or above the threshold set by RTSTH and clears RTS otherwise.
6	RTSTH	**RTS Threshold Control.** 0: RTS is de-asserted when the receive FIFO and shift register are full and no more incoming data can be stored. 1: RTS is de-asserted when the receive FIFO and shift register are nearly full and only one more data can be received.
5	RTSINVEN	**RTS Invert Enable.** 0: The UART does not invert the RTS signal before driving the pin. 1: The UART inverts the RTS signal driving the pin.
4:2	Reserved	Must write reset value.
1	RX	**RX Pin Status.** 0: RX pin (after optional inversion) is low. 1: RX pin (after optional inversion) is high.
0	RTS	**RTS State.** This bit is writeable only when RTSEN is cleared to 0. 0: RTS pin (before optional inversion) is driven low. 1: RTS pin (before optional inversion) is driven high.

Register 37.4. UARTn_CONTROL: Module Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	TEN	TINH	Reserved	TBIT	TBUSYF	Reserved			TCPTIEN	TDREQIEN	TERIEN	TCPTTH	TCPTI	TDREQI	Reserved	TSCERI
Type	RW	RW	R	RW	R	R			RW	RW	RW	RW	RW	R	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	REN	RINH	ROSEN	RBIT	RBUSYF	RABDEN	MATMD		Reserved	RDREQIEN	RERIEN	Reserved	RDREQI	ROREI	RPARERI	RFRMERI
Type	RW	RW	RW	RW	R	RW	RW		R	RW	RW	R	R	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

UART0_CONTROL = 0x4000_2030
UART1_CONTROL = 0x4000_3030
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 37.5. UARTn_CONTROL Register Bit Descriptions

Bit	Name	Function
31	TEN	**Transmitter Enable.** 0: Disable the transmitter. When cleared, the transmitter immediately aborts any active transmission. Clearing this bit does not automatically flush the transmit FIFO. 1: Enable the transmitter. The transmitter will initiate a transmission when data becomes available in the transmit FIFO.
30	TINH	**Transmit Inhibit.** 0: The transmitter operates normally. 1: Transmissions are inhibited. The transmitter will stall after any current transmission is complete.
29	Reserved	Must write reset value.
28	TBIT	**Last Transmit Bit.** This bit is used to set the 9th bit during each FIFO write when TDATLN is set to 5. The TBIT value is stored in the FIFO during each FIFO write; the TBIT value at the time of transmission is not used. If TDATLN is not set to 5, the written bit from the core interface is used.

Notes:
1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 37.5. UARTn_CONTROL Register Bit Descriptions

Bit	Name	Function
27	TBUSYF	**Transmitter Busy Flag.** 0: The UART transmitter is idle. 1: The UART transmitter is active and transmitting.
26:24	Reserved	Must write reset value.
23	TCPTIEN	**Transmit Complete Interrupt Enable.** 0: Disable the transmit complete interrupt. 1: Enable the transmit complete interrupt. A transmit interrupt is generated when TCPTI is set to 1.
22	TDREQIEN	**Transmit Data Request Interrupt Enable.** 0: Disable the transmit data request interrupt. 1: Enable the transmit data request interrupt. A transmit interrupt is asserted when TDREQI is set to 1.
21	TERIEN	**Transmit Error Interrupt Enable.** 0: Disable the transmit error interrupt. 1: Enable the transmit error interrupt. A transmit interrupt is generated when TSCERI is set to 1.
20	TCPTTH	**Transmit Complete Threshold.** 0: A transmit is completed (TCPTI = 1) at the end of each transmission. 1: A transmit is completed (TCPTI = 1) only at the end of a transmission when no more data is available to transmit.
19	TCPTI	**Transmit Complete Interrupt Flag.** This bit is set by hardware if a byte is transmitted (TCCPTH = 0) or if the last available byte was transmitted (TCPTTH = 1). This bit must be cleared by firmware.
18	TDREQI	**Transmit Data Request Interrupt Flag.** 0: The transmitter is not requesting more FIFO data. 1: The transmitter is requesting more FIFO data.
17	Reserved	Must write reset value.
16	TSCERI	**Smartcard Parity Error Interrupt Flag.** This bit is set by hardware when a Smartcard parity error occurs. This bit must be cleared by firmware.
15	REN	**Receiver Enable.** This bit enables the UART receiver for multiple transactions. 0: Disable the receiver. The receiver can receive one data transaction only if ROSEN is set. 1: Enable the receiver.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 37.5. UARTn_CONTROL Register Bit Descriptions

Bit	Name	Function
14	RINH	**Receiver Inhibit.** 0: The receiver operates normally. 1: RTS is immediately asserted when RINH is set. The receiver will complete any ongoing reception, but ignore all traffic after that.
13	ROSEN	**Receiver One-Shot Enable.** 0: Disable one-shot receive mode. 1: Enable one-shot receive mode.
12	RBIT	**Last Receive Bit.** This bit is used according to the match mode (MATMD) setting.
11	RBUSYF	**Receiver Busy Flag.** 0: The UART receiver is idle. 1: The UART receiver is receiving data.
10	RABDEN	**Receiver Auto-Baud Enable.** 0: Disable receiver auto-baud. 1: Enable receiver auto-baud.
9:8	MATMD	**Match Mode.** The hardware automatically switches from MCE mode to Frame mode when a MCE match occurs. 00: Disable the match function. 01: (MCE) Data whose last data bit equals RBIT is accepted and stored. 10: (Frame) A framing error is asserted if the last received bit matches RBIT. 11: (Store) Store the last incoming data bit in RBIT. This mode can be used incon-junction with the RDATLN setting.
7	Reserved	Must write reset value.
6	RDREQIEN	**Receive Data Request Interrupt Enable.** 0: Disable the read data request interrupt. 1: Enable the read data request interrupt. A receive interrupt is generated when RDREQI is set to 1.
5	RERIEN	**Receive Error Interrupt Enable.** 0: Disable the receive error interrupt. 1: Enable the receive error interrupt. A receive error interrupt is asserted when ROREI, RFRMERI, or RPARERI is set to 1.
4	Reserved	Must write reset value.
3	RDREQI	**Receive Data Request Interrupt Flag.** 0: Fewer than RFTH FIFO slots are filled with data. 1: At least RFTH FIFO slots are filled with data.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 37.5. UARTn_CONTROL Register Bit Descriptions

Bit	Name	Function
2	ROREI	**Receive Overrun Error Interrupt Flag.** This bit is set to 1 by hardware when a receive overrun error occurs. This bit must be cleared by firmware.
1	RPARERI	**Receive Parity Error Interrupt Flag.** This bit is set to 1 by hardware when the receiver encounters a parity error. This bit must be cleared by firmware.
0	RFRMERI	**Receive Frame Error Interrupt Flag.** Hardware sets this bit to 1 when a receive frame error occurs. There are two sources for this error: when an expected whole stop bit is low, or when a frame match-mode fails. This bit must be cleared by firmware.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Register 37.5. UARTn_IPDELAY: Inter-Packet Delay

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved								IPDELAY							
Type	R								RW							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses
UART0_IPDELAY = 0x4000_2040
UART1_IPDELAY = 0x4000_3040

Table 37.6. UARTn_IPDELAY Register Bit Descriptions

Bit	Name	Function
31:24	Reserved	Must write reset value.
23:16	IPDELAY	**Inter-Packet Delay.** This field configures the transmitter to delay between transmissions by the specified number of bit times. A value of 0 means no delay is added, and a value of 5 means 5 bit times are added. Bit times are configured in the TBAUD register.
15:0	Reserved	Must write reset value.

Register 37.6. UARTn_BAUDRATE: Transmit and Receive Baud Rate

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	TBAUD															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	RBAUD															
Type	RW															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses
UART0_BAUDRATE = 0x4000_2050
UART1_BAUDRATE = 0x4000_3050

Table 37.7. UARTn_BAUDRATE Register Bit Descriptions

Bit	Name	Function
31:16	TBAUD	**Transmitter Baud Rate Control.** This field sets the transmitter baud rate according to the equation: $\text{Baud Rate} = \frac{F_{APB}}{N \times (TBAUD + 1)}$ N = 2 if TIRDAEN = 0, and N = 16 if TIRDAEN = 1.
15:0	RBAUD	**Receiver Baud Rate Control.** This field sets the receiver baud rate according to the equation: $\text{Baud Rate} = \frac{F_{APB}}{N \times (RBAUD + 1)}$ N = 2 if RIRDAEN = 0, and N = 16 if RIRDAEN = 1.

Register 37.7. UARTn_FIFOCN: FIFO Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved					TSRFULLF	TFERI	TFIFOFL	Reserved		TFTH		Reserved	TCNT		
Type	R					R	RW	RW	RW	R	RW		R	R		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved					RSRFULLF	RFERI	RFIFOFL	Reserved		RFTH		Reserved	RCNT		
Type	R					R	RW	RW	RW	R	RW		R	R		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses
UART0_FIFOCN = 0x4000_2060
UART1_FIFOCN = 0x4000_3060
This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 37.8. UARTn_FIFOCN Register Bit Descriptions

Bit	Name	Function
31:27	Reserved	Must write reset value.
26	TSRFULLF	**Transmit Shift Register Full Flag.** This bit indicates when the transmitter shift register contains data. The bit is set when hardware loads the data from the FIFO to the transmit shift register and is cleared when the transmitter completely transmits the data.
25	TFERI	**Transmit FIFO Error Interrupt Flag.** This bit is set when an illegal FIFO write is detected, such as a non right-justified write or a write that contains more data than will fit in the empty FIFO entries. When this bit is set, an interrupt will be asserted. 0: A transmit FIFO error has not occurred since TFERI was last cleared. 1: A transmit FIFO error occurred.
24	TFIFOFL	**Transmit FIFO Flush.** Setting this bit to 1 flushes the transmit FIFO. If data is pending in the transmit shift register but a transmit has not begun, the shift register is also flushed. This bit always reads as 0.

Notes:
1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 37.8. UARTn_FIFOCN Register Bit Descriptions

Bit	Name	Function
23:22	Reserved	Must write reset value.
21:20	TFTH	**Transmit FIFO Threshold.** This field sets the FIFO threshold at which a TDREQI interrupt is asserted. 00: A transmit data request interrupt (TDREQI) is asserted when ≥ 1 FIFO slot is empty. 01: A transmit data request interrupt (TDREQI) is asserted when ≥ 2 FIFO slots are empty. 10: A transmit data request interrupt (TDREQI) is asserted when ≥ 4 FIFO slots are empty. 11: Reserved.
19	Reserved	Must write reset value.
18:16	TCNT	**Transmit FIFO Count.** This field indicates the number of entries in the transmit FIFO.
15:11	Reserved	Must write reset value.
10	RSRFULLF	**Receive Shift Register Full .** This bit indicates when the receiver shift register contains data and remains high until the data is moved to the FIFO or is flushed. The flag is set as soon as incoming data is completely received and cleared when the data is transferred to the FIFO.
9	RFERI	**Receive FIFO Error Interrupt Flag.** This bit is set when hardware detects an illegal FIFO read, such as a non right-justified read or a read that demands more data than is available in the FIFO. When this bit is set, an interrupt will be asserted. 0: A receive FIFO error has not occurred since RFERI was last cleared. 1: A receive FIFO error occurred.
8	RFIFOFL	**Receive FIFO Flush.** Setting this bit to 1 flushes the receive FIFO and any data in the receive shift register. This bit always reads as 0.
7:6	Reserved	Must write reset value.
5:4	RFTH	**Receive FIFO Threshold.** This is the threshold at which a RDREQI interrupt is asserted. 00: A read data request interrupt (RDREQI) is asserted when ≥ 1 FIFO slot is full. 01: A read data request interrupt (RDREQI) is asserted when ≥ 2 FIFO slots are full. 10: A read data request interrupt (RDREQI) is asserted when ≥ 4 FIFO slots are full. 11: Reserved.
3	Reserved	Must write reset value.
2:0	RCNT	**Receive FIFO Count.** This field indicates the number of entries in the receive FIFO.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Register 37.8. UARTn_DATA: FIFO Input/Output Data

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	DATA[31:16]															
Type	RW															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	DATA[15:0]															
Type	RW															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Register ALL Access Addresses

UART0_DATA = 0x4000_2070
UART1_DATA = 0x4000_3070

Table 37.9. UARTn_DATA Register Bit Descriptions

Bit	Name	Function
31:0	DATA	**FIFO Data.** The 32-bit DATA register is a single port into the transmit and receive FIFOs. Reads and writes should always be right-justified. Byte-wide reads and writes should always access DATA[7:0], half-word reads and writes should always access DATA[15:0], and word reads and writes should always access DATA[31:0]. Writes to the DATA register push data into the transmit FIFO. Reads from the DATA register pop data from the receive FIFO. Multiple FIFO entries can be pushed or popped in a single write or read. For example, if the transmit bit-length is less than 9 bits, writing a whole word (4 bytes) to the DATA field will write four entries in the FIFO where each byte is a single entry. However, if the FIFO doesn't have room for 4 entries, the write is completely ignored and the TFERI error flag is set. Similarly, a half-word write will push 2 entries to the FIFO if the data length is < 9. If the data length is 9, each half-word is a single entry. When writing or reading multiple bytes from the FIFO, the least significant byte is written to or read from the FIFO first.

Notes:
1. Reads of this register modify the state of hardware. Debug logic should take care when reading this register.

37.11. UARTn Register Memory Map

Table 37.10. UARTn Memory Map

Register Name	UARTn_CONTROL	UARTn_FLOWCN	UARTn_MODE	UARTn_CONFIG
ALL Offset	0x30	0x20	0x10	0x0
Access Methods	ALL | SET | CLR	ALL | SET | CLR	ALL | SET | CLR	ALL | SET | CLR
Bit 31	TEN	Reserved	Reserved	Reserved
Bit 30	TINH		ITSEN	TINVEN
Bit 29	Reserved	TIRDAPW	Reserved	TIRDAEN
Bit 28	TBIT			TSCEN
Bit 27	TBUSYF	Reserved	DUPLEXMD	Reserved
Bit 26	Reserved		Reserved	TDATLN
Bit 25				
Bit 24				
Bit 23	TCPTIEN	CTSEN		Reserved
Bit 22	TDREQIEN	Reserved		TPARMD
Bit 21	TERIEN	CTSINVEN		
Bit 20	TCPTTH	Reserved	LBMD	TSTPMD
Bit 19	TCPTI			
Bit 18	TDREQI			TSTPEN
Bit 17	Reserved	TX	Reserved	TPAREN
Bit 16	TSCERI	CTS	DBGMD	TSTRTEN
Bit 15	REN	Reserved	Reserved	Reserved
Bit 14	RINH			RINVEN
Bit 13	ROSEN			RIRDAEN
Bit 12	RBIT			RSCEN
Bit 11	RBUSYF			Reserved
Bit 10	RABDEN			RDATLN
Bit 9	MATMD			
Bit 8				
Bit 7	Reserved	RTSEN		Reserved
Bit 6	RDREQIEN	RTSTH		RPARMD
Bit 5	RERIEN	RTSINVEN		
Bit 4	Reserved	Reserved		RSTPMD
Bit 3	RDREQI			
Bit 2	ROREI			RSTPEN
Bit 1	RPARERI	RX		RPAREN
Bit 0	RFRMERI	RTS		RSTRTEN

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: UART0 = 0x4000_2000, UART1 = 0x4000_3000

Table 37.10. UARTn Memory Map

Register Name	ALL Offset	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
UARTn_IPDELAY	0x40	ALL	Reserved								IPDELAY								Reserved															
UARTn_BAUDRATE	0x50	ALL	TBAUD																RBAUD															
UARTn_FIFOCN	0x60	ALL | SET | CLR	Reserved					TSRFULLF	TFERI	TFIFOFL	Reserved		TFTH		Reserved	TCNT			Reserved				RSRFULLF	RFERI	RFIFOFL	Reserved		RFTH		Reserved		RCNT		
UARTn_DATA	0x70	ALL	DATA																															

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: UART0 = 0x4000_2000, UART1 = 0x4000_3000

38. Universal Serial Bus Controller (USB0)

This section describes the Universal Serial Bus (USB) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx

This section describes version “A” of the USB block, which is used by all device families covered in this document.

Note: This document assumes a comprehensive understanding of the USB Protocol. Terms and abbreviations used in this document are defined in the USB Specification. We encourage you to review the latest version of the USB Specification before proceeding.

38.1. USB Features

The USB module includes the following features:

- Full and Low Speed functionality.
- USB 2.0 compliant USB peripheral support (no host capability).
- Direct module access to 2k bytes of RAM for dedicated FIFO memory.
- Dedicated USB oscillator with clock recovery to meet USB clocking requirements with no external components.
- Additional clocking options include PLL or external oscillator outputs.

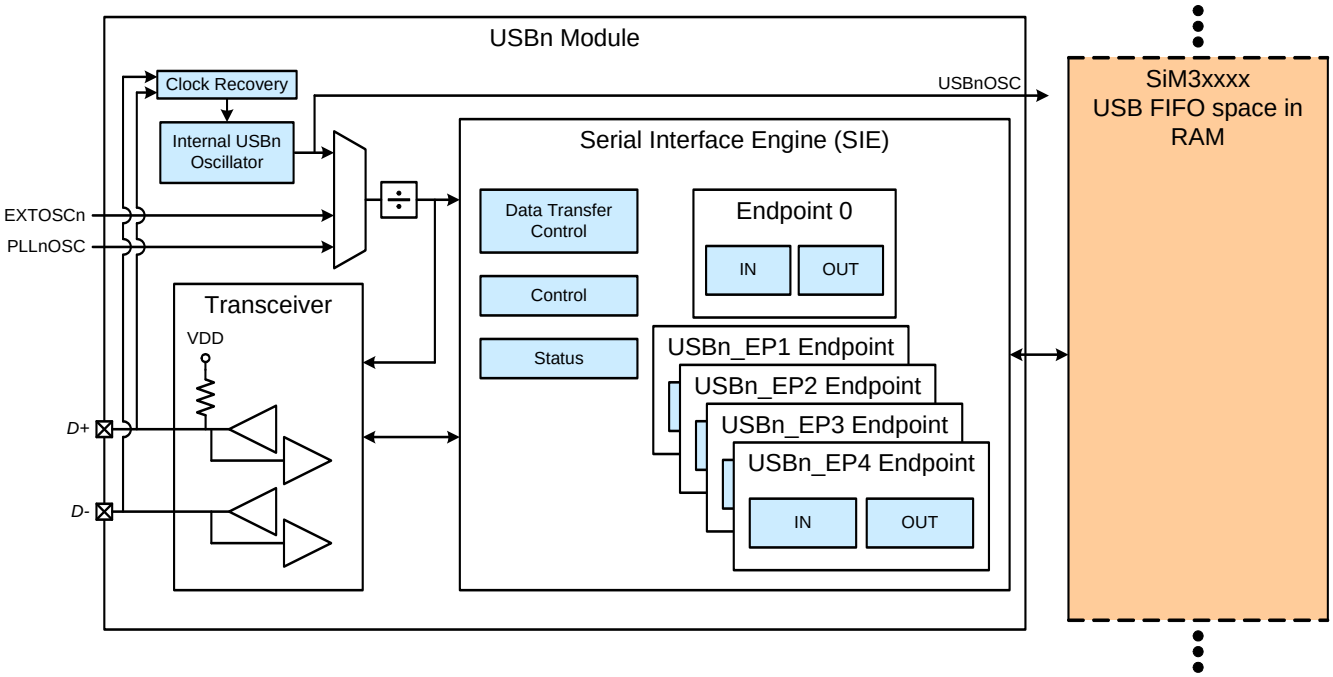


Figure 38.1. USB0 Block Diagram

38.2. Overview

The USB module has complete Full/Low Speed USB function for USB peripheral implementations (no host capability). The USB Function Controller (USB0) consists of a Serial Interface Engine (SIE), USB Transceiver (including matching resistors and configurable pull-up resistors), 2k FIFO block, and dedicated oscillator with clock recovery mechanism for crystal-less operation. No external components are required. The USB Function Controller and Transceiver is Universal Serial Bus Specification 2.0 compliant.

38.3. Clocking

USB0 is capable of communication as a Full or Low Speed USB function. Communication speed is selected via the SSEL bit in TCONTROL. When operating as a Low Speed function, the USB0 clock must be 6 MHz. When operating as a Full Speed function, the USB0 clock must be 48 MHz. The USB0 module can run from its dedicated oscillator, the PLL (Phase-Locked Loop) output clock, or an external oscillator.

38.3.1. Dedicated USB0 Clock

USB0 includes a programmable, dedicated internal oscillator that defaults as the USB0 clock after a system reset. The dedicated oscillator period can be programmed via the FADJUST register, which is factory calibrated to obtain a 48 MHz internal oscillator frequency. Note that the USB0 clock may be derived from the programmed dedicated oscillator divided by 1, 2, 4, or 8, as defined by the CLKDIV bits in register CLKSEL. The divide value defaults to 1 following a reset.

Clock Recovery circuitry uses the incoming USB data stream to adjust the internal dedicated USB0 oscillator; this allows the internal oscillator to meet the requirements for USB clock tolerance. Clock Recovery should be used in the following configurations:

Table 38.1. USB0 Module Clocking Configurations

Communication Speed	USB Clock
Full Speed	Internal USB Clock (48 MHz)
Low Speed	Internal USB Clock/8 (48 MHz / 8)

When operating USB0 as a Low Speed function with Clock Recovery, software must write 1 to the LSCRMD bit in the CRCONTROL register to enable Low Speed Clock Recovery. Clock Recovery is typically not necessary in Low Speed mode.

Single Step Mode can be used to help the Clock Recovery circuitry to lock when high noise levels are present on the USB network. This mode is not required (or recommended) in typical USB environments.

38.3.1.1. Suspend Mode

The dedicated USB0 oscillator may be placed in Suspend mode by writing 1 to the SUSPEND bit in register OSCCONTROL. In Suspend mode, the dedicated USB0 oscillator is stopped until a non-idle USB event is detected or a rising or falling edge occurs on VBUS. Note that the USB transceiver can still detect USB events when the oscillator is suspended.

38.3.1.2. Dithering

The USB0 oscillator has optional frequency dithering, which improves the average USB clock accuracy by varying the oscillator control value every four oscillator cycles. Dithering can be enabled by setting the DITHEN bit to 1 in the AFADJUST register.

38.3.2. Phase-Locked Loop (PLL)

When the PLL output clock is selected as the USB0 clock source, the USB0 module will clock from the PLL output, regardless of what clock is selected by the core. The output of the PLL must be the correct frequency and tolerance for USB operation, as the Clock Recovery module is not available for the PLL output clock. The PLL does not multiply the reference clock error as a percentage, but it will add some small additional error in the form of jitter that will need to be taken into account when selecting a reference clock appropriate for use with the USB0 module.

38.3.3. External Clock

When the external clock is selected as the USB0 clock source, the USB0 module will clock from the external clock source, regardless of what clock is selected by the core. In this mode, the external clock source is not synchronized to the selected APB clock. The external clock must be the correct frequency and tolerance for USB operation, as the Clock Recovery module is not available for the external clock.

38.4. Endpoints

A total of ten endpoint pipes are available. The control endpoint (Endpoint 0) always functions as a bi-directional IN/OUT endpoint. The other endpoints are implemented as four pairs of IN/OUT endpoint pipes. The registers for Endpoint 1 through Endpoint 4 are identical, while the Endpoint 0 registers are a subset of the other endpoint registers.

38.5. USB Transceiver

The USB Transceiver is configured via the TCONTROL register. This configuration includes Transceiver enable/disable, pull-up resistor enable/disable, and device speed selection (Full or Low Speed). When bit SSEL = 1, USB0 operates as a Full Speed USB module, and the on-chip pull-up resistor (if enabled) appears on the D+ pin. When bit SSEL = 0, USB0 operates as a Low Speed USB module, and the on-chip pull-up resistor (if enabled) appears on the D- pin. The pull-up resistor is enabled only when VBUS is present.

Note: The USB clock should be active before the Transceiver is enabled.

38.6. FIFO Management

2048 bytes of on-chip RAM are used as FIFO space for USB0. This FIFO space is split between Endpoints 0-4 as shown in Figure 38.2. FIFO space allocated for Endpoints 1-4 is configurable as IN, OUT, or both (Split Mode: half IN, half OUT).

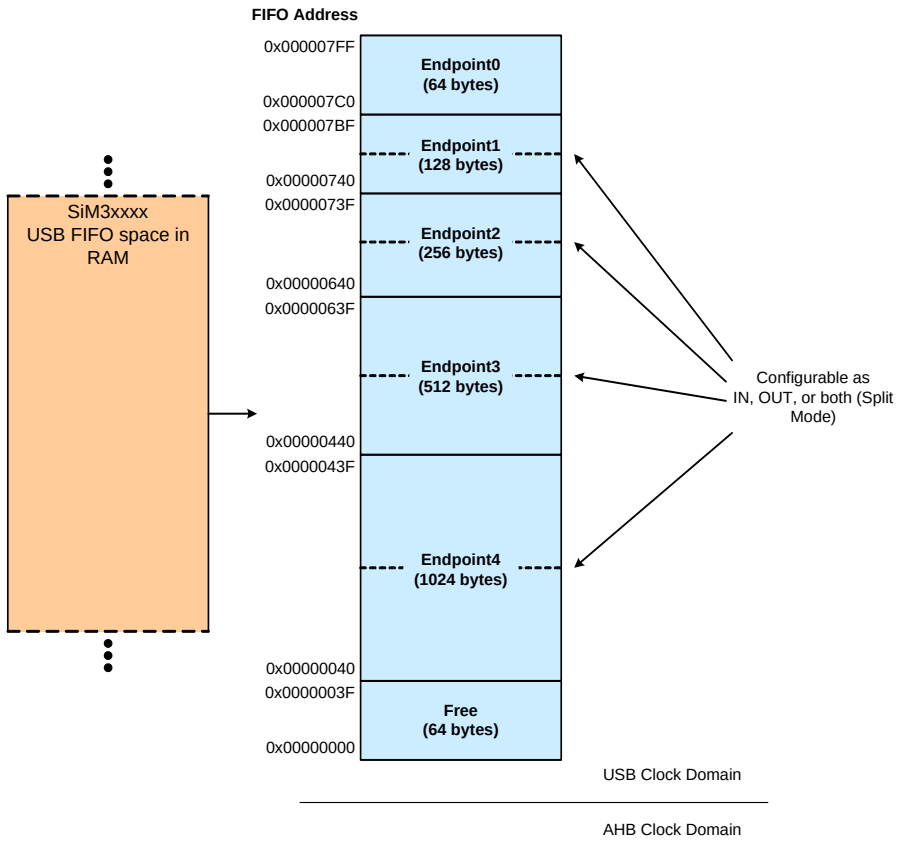


Figure 38.2. USB0 FIFO Allocation

38.6.1. FIFO Split Mode

The FIFO space for Endpoints 1-4 can be split such that the upper half of the FIFO space is used by the IN endpoint and the lower half is used by the OUT endpoint. For example: if the Endpoint 3 FIFO is configured for Split Mode, the upper 256 bytes (FIFO addresses 0x00000540 to 0x0000063F) are used by Endpoint 3 IN and the lower 256 bytes (FIFO addresses 0x00000440 to 0x0000053F) are used by Endpoint 3 OUT.

If an endpoint FIFO is not configured for Split Mode, that endpoint IN/OUT pair's FIFOs are combined to form a single IN or OUT FIFO. In this case, only one direction of the endpoint IN/OUT pair may be used at a time. The endpoint direction (IN/OUT) is determined by the DIRSEL bit in the corresponding endpoint's ECSR register.

38.6.2. FIFO Double Buffering

FIFO slots for Endpoints 1-4 are automatically configured for double-buffered mode if the maximum packet size for the endpoint is set to less than or equal to half the available space. In this mode, the FIFO may contain two packets at a time. See Table 38.2 for a list of maximum memory sizes for each FIFO configuration.

Table 38.2. FIFO Configurations

Endpoint Number	Split Mode Enabled?	Maximum IN Packet Size (Double Buffer Disabled / Enabled)	Maximum OUT Packet Size (Double Buffer Disabled / Enabled)
0	N/A	64	
1	N	128 / 64	
	Y	64 / 32	64 / 32
2	N	256 / 128	
	Y	128 / 64	128 / 64
3	N	512 / 256	
	Y	256 / 128	256 / 128
4	N	1024 / 512	
	Y	512 / 256	512 / 256

38.6.3. FIFO Access

Each endpoint FIFO is accessed through a corresponding FIFOn register. A read of an endpoint FIFOn register unloads a word, half word, or byte from the FIFO; a write of an endpoint FIFOn register loads one word, half word, or byte into the endpoint FIFO. When an endpoint FIFO is configured for Split Mode, a read of the endpoint FIFOn register unloads data from the OUT endpoint FIFO; a write of the endpoint FIFOn register loads data into the IN endpoint FIFO. All reads and writes must be right-justified in the FIFOn register.

For multi-byte reads and writes, the least-significant byte is read from or written to the FIFO first.

38.7. Function Addressing

The FADDR register holds the current USB0 function address. Software should write the host-assigned 7-bit function address to the FADDR register when received as part of a SET_ADDRESS command. A new address written to FADDR will not take effect (USB0 will not respond to the new address) until the end of the current transfer (typically following the status phase of the SET_ADDRESS command transfer). The FADDRUPD bit is set to 1 by hardware when software writes a new address to the FADDR register. Hardware clears the FADDRUPD bit when the new address takes effect as described above.

38.8. Function Configuration and Control

The POWER register is used to configure and control USB0 at the device level (enable/disable, Reset/Suspend/Resume handling, etc.).

38.8.1. USB Reset

The RSTDETF bit in the POWER register is set to 1 by hardware when Reset signaling is detected on the bus. Upon this detection, the following occur:

1. The USB0 Address is reset (FADDR = 0x00).
2. Endpoint FIFOs are flushed.
3. Endpoint data toggles are reset to 0.
4. Control/status registers are reset to 0x00 (EP0CONTROL, EPCONTROL).
5. A USB Reset interrupt is generated if enabled. A USB Reset device reset can also occur if enabled.

Writing a 1 to the RESET bit in the CLKSEL register will generate an asynchronous USB0 reset. All USB registers are reset to their default values following this asynchronous reset.

38.8.2. Suspend Mode

With Suspend Detection enabled (SUSDEN = 1), USB0 will enter Suspend Mode when Suspend signaling is detected on the bus. An interrupt will be generated if enabled (SUSIEN = 1). The Suspend Interrupt Service Routine (ISR) should perform application-specific configuration tasks, such as disabling appropriate peripherals and/or configuring clock sources for low power modes. See Section 38.3.1 for more details on the dedicated USB0 oscillator configuration, including the Suspend mode feature.

USB0 exits Suspend mode when any of the following occur: (1) Resume signaling is detected or generated, (2) Reset signaling is detected, or (3) a device or USB reset occurs. If suspended, the dedicated USB0 oscillator will exit Suspend mode upon any of the above listed events.

38.8.3. Resume Signaling

USB0 will exit Suspend mode if any non-idle signaling is detected on the bus. A Resume interrupt will be generated upon detection if enabled (RESIEN = 1). Software may force a Remote Wakeup by writing 1 to the RESUME bit in the POWER register. When forcing a Remote Wakeup, software should write RESUME = 0 to end Resume signaling 10-15 ms after the Remote Wakeup is initiated (RESUME = 1).

38.8.4. ISO Update

When software writes 1 to the ISOUPDMD bit in the POWER register, the ISO Update function is enabled. With ISO Update enabled, new packets written to an ISO IN endpoint will not be transmitted until a new Start-Of-Frame (SOF) is received. If the ISO IN endpoint receives an IN token before a SOF, USB0 will transmit a zero-length packet. When ISOUPDMD = 1, ISO Update is enabled for all ISO endpoints.

38.8.5. USB Enable

USB0 is disabled following any device reset. USB0 is enabled by clearing the USBINH bit. Once written to 0, the USBINH can only be set to 1 by one of the following: (1) any device reset, or (2) an asynchronous USB0 reset generated by writing 1 to the RESET bit in the CLKSEL register.

Software should perform all USB0 configuration before enabling USB0. The configuration sequence should be performed as follows:

1. Select and enable the USB clock source.
2. Reset USB0 by writing RESET = 1.
3. Configure and enable the USB Transceiver.
4. Perform any USB0 function configuration (interrupts, Suspend detect).
5. Enable USB0 by writing USBINH = 0.

38.9. Interrupts

The USB0 interrupt flags are located in the IOINT and CMINT registers. The associated interrupt enable bits are located in the IOINTE and CMINTEPE registers. A USB0 interrupt is generated when any of the USB interrupt flags is set to 1. The interrupt flag is cleared by writing a 1 back to the flag for that interrupt.

38.10. The Serial Interface Engine

The Serial Interface Engine (SIE) performs all low level USB protocol tasks, interrupting the processor when data has successfully been transmitted or received. When receiving data, the SIE will interrupt the processor when a complete data packet has been received; appropriate handshaking signals are automatically generated by the SIE. When transmitting data, the SIE will interrupt the processor when a complete data packet has been transmitted and the appropriate handshake signal has been received.

The SIE will not interrupt the processor when corrupted/erroneous packets are received. The SIE will also not interrupt if a DMA channel is enabled on an endpoint and a maximum size packet is transmitted or received.

38.11. Endpoint 0

Endpoint 0 is managed through the EP0CONTROL register.

An Endpoint 0 interrupt is generated when the following occurs:

1. A data packet (OUT or SETUP) has been received and loaded into the Endpoint 0 FIFO. The OPRDYI bit is set to 1 by hardware.
2. An IN data packet has successfully been unloaded from the Endpoint 0 FIFO and transmitted to the host; IPRDYI is reset to 0 by hardware.
3. An IN transaction is completed (this interrupt is generated during the status stage of the transaction).
4. Hardware sets the STSTLI bit after a control transaction ended due to a protocol violation.
5. Hardware sets the SUENDI bit because a control transfer ended before firmware sets the DEND bit.

The Endpoint 0 EP0COUNT register holds the number of received data bytes in the Endpoint 0 FIFO.

Hardware will automatically detect protocol errors and send a STALL condition in response. Firmware may force a STALL condition to abort the current transfer. When a STALL condition is generated, the STSTLI bit will be set to 1 and an interrupt generated. The following conditions will cause hardware to generate a STALL condition:

1. The host sends an OUT token during a OUT data phase after the DEND bit has been set to 1.
2. The host sends an IN token during an IN data phase after the DEND bit has been set to 1.
3. The host sends a packet that exceeds the maximum packet size for Endpoint 0.
4. The host sends a non-zero length DATA1 packet during the status phase of an IN transaction.
5. Firmware sets the SDSTL bit to 1.

38.11.1. Endpoint 0 SETUP Transactions

All control transfers must begin with a SETUP packet. SETUP packets are similar to OUT packets, containing an 8-byte data field sent by the host. Any SETUP packet containing a command field of anything other than 8 bytes will be automatically rejected and ignored by USB0. An Endpoint 0 interrupt is generated when the data from a SETUP packet is loaded into the Endpoint 0 FIFO. Software should unload the command from the Endpoint 0 FIFO, decode the command, perform any necessary tasks, and set the OPRDYIS bit to indicate that it has serviced the OUT packet.

38.11.2. Endpoint 0 IN Transactions

When a SETUP request is received that requires USB0 to transmit data to the host, one or more IN requests will be sent by the host. For the first IN transaction, firmware should load an IN packet into the Endpoint 0 FIFO, and set the IPRDYI bit. An interrupt will be generated when an IN packet is transmitted successfully. Note that no interrupt will be generated if an IN request is received before firmware has loaded a packet into the Endpoint 0 FIFO. If the requested data exceeds the maximum packet size for Endpoint 0 (as reported to the host), the data should be split

into multiple packets; each packet should be of the maximum packet size excluding the last (residual) packet. If the requested data is an integer multiple of the maximum packet size for Endpoint 0, the last data packet should be a zero-length packet signaling the end of the transfer. Firmware should set the DEND bit to 1 after loading the last data packet for a transfer into the Endpoint 0 FIFO.

Upon reception of the first IN token for a particular control transfer, Endpoint 0 is said to be in Transmit Mode. In this mode, only IN tokens should be sent by the host to Endpoint 0. The SUENDI bit is set to 1 if a SETUP or OUT token is received while Endpoint 0 is in Transmit Mode.

Endpoint 0 will remain in Transmit Mode until any of the following occur:

1. USB0 receives an Endpoint 0 SETUP or OUT token.
2. Firmware sends a packet less than the maximum Endpoint 0 packet size.
3. Firmware sends a zero-length packet.

Firmware should set the DEND bit to 1 when performing (2) and (3) above.

The SIE will transmit a NAK in response to an IN token if there is no packet ready in the IN FIFO (IPRDYI = 0).

38.11.3. Endpoint 0 OUT Transactions

When a SETUP request is received that requires the host to transmit data to USB0, one or more OUT requests will be sent by the host. When an OUT packet is successfully received by USB0, hardware will set the OPRDYI bit to 1 and generate an Endpoint 0 interrupt. Following this interrupt, firmware should unload the OUT packet from the Endpoint 0 FIFO and set the OPRDYIS bit to 1.

If the amount of data required for the transfer exceeds the maximum packet size for Endpoint 0, the data will be split into multiple packets. If the requested data is an integer multiple of the maximum packet size for Endpoint 0 (as reported to the host), the host will send a zero-length data packet signaling the end of the transfer.

Upon reception of the first OUT token for a particular control transfer, Endpoint 0 is said to be in Receive Mode. In this mode, only OUT tokens should be sent by the host to Endpoint 0. The SUENDI bit is set to 1 if a SETUP or IN token is received while Endpoint 0 is in Receive Mode.

Endpoint 0 will remain in Receive mode until:

1. The SIE receives a SETUP or IN token.
2. The host sends a packet less than the maximum Endpoint 0 packet size.
3. The host sends a zero-length packet.

Firmware should set the DEND bit to 1 when the expected amount of data has been received. The SIE will transmit a STALL condition if the host sends an OUT packet after the DEND bit has been set by firmware. An interrupt will be generated with the STSTLI bit set to 1 after the STALL is transmitted.

38.12. Configuring Endpoints 1-4

Endpoints 1–4 are configured and controlled through their own EPCONTROL control/status registers.

Endpoints 1–4 can be configured as IN, OUT, or both IN/OUT (Split Mode) as described in Section 38.6.1. The endpoint mode (Split/Normal) is selected via the SPLITEN bit in the endpoint's EPCONTROL register.

When SPLITEN = 1, the corresponding endpoint FIFO is split, and both IN and OUT pipes are available.

When SPLITEN = 0, the corresponding endpoint functions as either IN or OUT; the endpoint direction is selected by the DIRSEL bit.

38.13. Controlling Endpoints 1-4 IN

Endpoints 1-4 IN are managed via the corresponding EPCONTROL register. All IN endpoints can be used for Interrupt, Bulk, or Isochronous transfers. Isochronous (ISO) mode is enabled by writing 1 to the IISOEN bit. Bulk and Interrupt transfers are handled identically by hardware.

An Endpoint 1-4 IN interrupt is generated by any of the following conditions:

1. An IN packet is successfully transferred to the host.
2. Software writes 1 to the IFIFOFL bit when the target FIFO is not empty.
3. Hardware generates a STALL condition.

A firmware-controlled USB IN transaction with a double-buffered endpoint is shown in Figure 38.3.

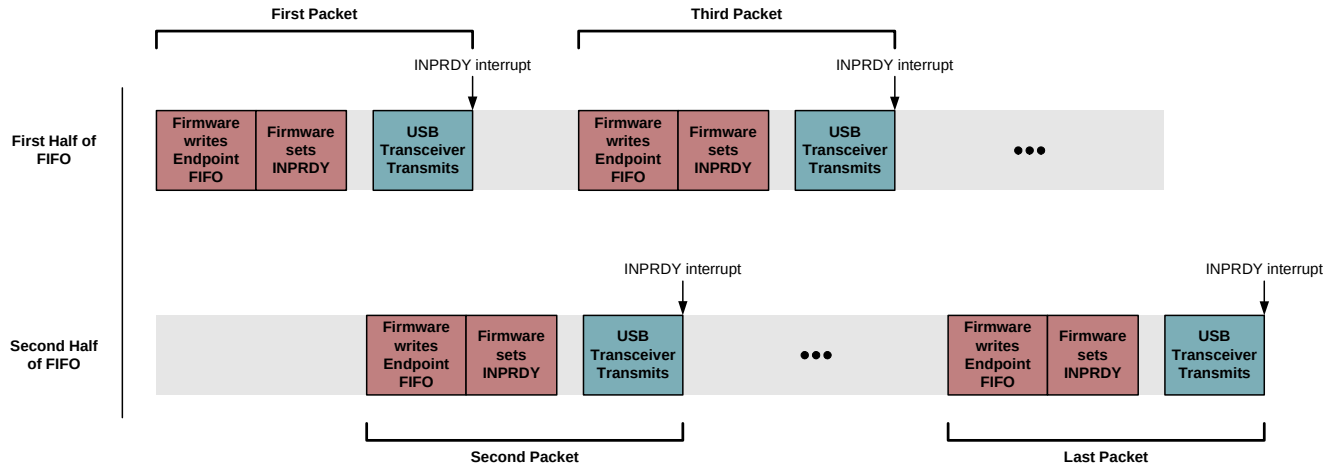


Figure 38.3. USB Firmware-Controlled IN Transfer

38.13.1. Endpoints 1-4 IN Interrupt or Bulk Mode

When the IISOEN bit is cleared to 0, the target endpoint operates in Bulk or Interrupt Mode. Once an endpoint has been configured to operate in Bulk/Interrupt IN mode (typically following an Endpoint 0 SET_INTERFACE command), firmware should load an IN packet into the endpoint IN FIFO and set the IPRDYI bit. The hardware will transmit the data upon reception of an IN token, clear the IPRDYI bit, and also generate an interrupt depending on the DMA configuration for the endpoint.

Writing 1 to IPRDYI without writing any data to the endpoint FIFO will cause a zero-length packet to be transmitted upon reception of the next IN token.

A Bulk or Interrupt pipe can be shut down (or Halted) by writing 1 to the ISDSTL bit. While ISDSTL = 1, hardware will respond to all IN requests with a STALL condition. Each time hardware generates a STALL condition, an interrupt will be generated and the ISTSTLI bit set to 1. The ISTSTLI bit must be reset to 0 by firmware.

Hardware will automatically reset IPRDYI to 0 when a packet slot is open in the endpoint FIFO. If double buffering is enabled for the target endpoint, it is possible for firmware to load two packets into the IN FIFO at a time. In this case, hardware will reset IPRDYI to 0 immediately after firmware loads the first packet into the FIFO and sets IPRDYI to 1. An interrupt will not be generated in this case; an interrupt will only be generated when a data packet is transmitted.

When firmware writes 1 to the FDTEN bit, the data toggle for each IN packet will be toggled continuously, regardless of the handshake received from the host. This feature is typically used by Interrupt endpoints functioning as rate feedback communication for Isochronous endpoints. When FDTEN = 0, the data toggle bit will only be toggled when an ACK is sent from the host in response to an IN packet.

38.13.2. Endpoints 1-4 IN Isochronous Mode

When the IISOEN bit is set to 1, the target endpoint operates in Isochronous (ISO) mode. Once an endpoint has been configured for ISO IN mode, the host will send one IN token (data request) per frame; the location of data within each frame may vary. Because of this, it is recommended that double buffering be enabled for ISO IN endpoints.

Hardware will automatically reset IPRDYI to 0 when a packet slot is open in the endpoint FIFO. Note that if double buffering is enabled for the target endpoint, it is possible for firmware to load two packets into the IN FIFO at a time. In this case, hardware will reset IPRDYI to 0 immediately after firmware loads the first packet into the FIFO and sets IPRDYI to 1. An interrupt will not be generated in this case; an interrupt will only be generated when a data packet is transmitted.

If there is not a data packet ready in the endpoint FIFO when USB0 receives an IN token from the host, USB0 will transmit a zero-length data packet and set the IURF bit to 1.

The ISO Update feature (see Section 38.8) can be useful in starting a double buffered ISO IN endpoint. If the host has already set up the ISO IN pipe (has begun transmitting IN tokens) when firmware writes the first data packet to the endpoint FIFO, the next IN token may arrive and the first data packet sent before firmware has written the second (double buffered) data packet to the FIFO. The ISO Update feature ensures that any data packet written to the endpoint FIFO will not be transmitted during the current frame; the packet will only be sent after a SOF signal has been received.

38.14. Controlling Endpoints 1-4 OUT

Endpoints 1–4 OUT are managed via the corresponding EPCONTROL register. All OUT endpoints can be used for Interrupt, Bulk, or Isochronous transfers. Isochronous (ISO) mode is enabled by writing 1 to the OISOEN bit. Bulk and Interrupt transfers are handled identically by hardware.

An Endpoint 1–4 OUT interrupt may be generated by the following:

1. Hardware sets the OPRDYI bit to 1.
2. Hardware generates a STALL condition.

A firmware-controlled USB OUT transaction with a double-buffered endpoint is shown in Figure 38.4.

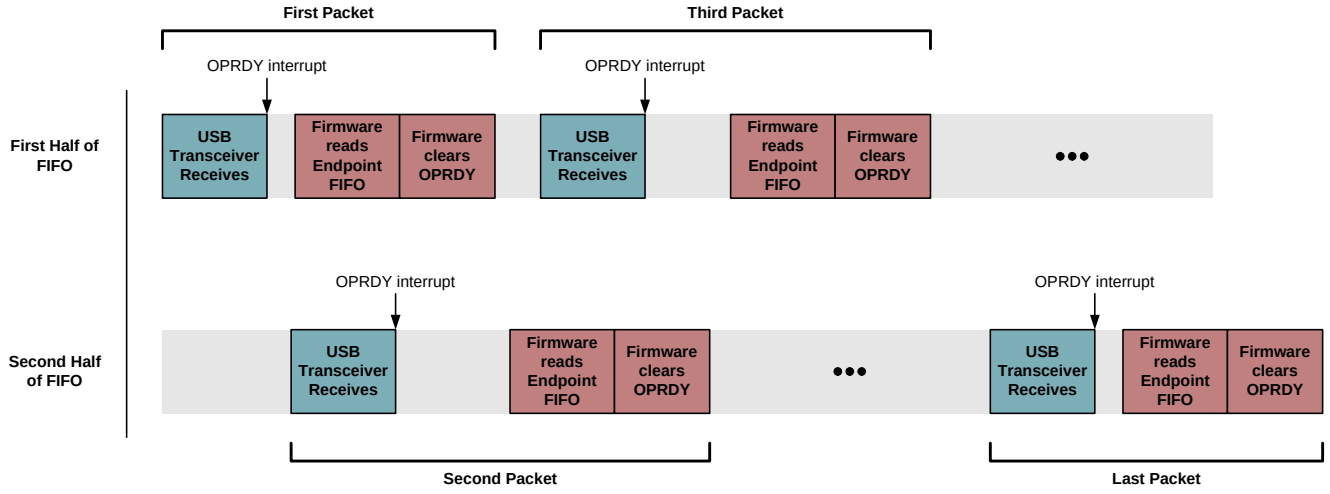


Figure 38.4. USB Firmware-Controlled OUT Transfer

38.14.1. Endpoints 1-4 OUT Interrupt or Bulk Mode

When the OISOEN bit is cleared to 0, the target endpoint operates in Bulk or Interrupt mode. Once an endpoint has been configured to operate in Bulk/Interrupt OUT mode (typically following an Endpoint 0 SET_INTERFACE command), the reception of an OUT token and data packet will cause the hardware to set the OPRDYI bit to 1 and generate an interrupt depending on the DMA configuration for the endpoint. The number of bytes in the current OUT data packet (the packet ready to be unloaded from the FIFO) is given in the EPCOUNT register. In response to this interrupt, firmware should unload the data packet from the OUT FIFO and reset the OPRDYI bit to 0.

A Bulk or Interrupt pipe can be shut down (or Halted) by writing 1 to the OSDSTL bit. While OSDSTL = 1, hardware will respond to all OUT requests with a STALL condition. Each time hardware generates a STALL condition, an interrupt will be generated and the OSTSTLI bit set to 1. The OSTSTLI bit must be reset to 0 by firmware.

Hardware will automatically set OPRDYI when a packet is ready in the OUT FIFO. Note that if double buffering is enabled for the target endpoint, it is possible for two packets to be ready in the OUT FIFO at a time. In this case, hardware will set OPRDYI to 1 immediately after firmware unloads the first packet and resets OPRDYI to 0. A second interrupt will be generated in this case.

38.14.2. Endpoints 1-4 OUT Isochronous Mode

When the OISOEN bit is set to 1, the target endpoint operates in Isochronous (ISO) mode. Once an endpoint has been configured for ISO OUT mode, the host will send exactly one data packet per USB frame; the location of the data packet within each frame may vary, however. Because of this, it is recommended that double buffering be enabled for ISO OUT endpoints, which is discussed in Section 38.6.2 and Section 38.15.5.

Each time a data packet is received, hardware will load the received data packet into the endpoint FIFO, set the OPRDYI bit to 1, and generate an interrupt (if enabled and depending on the DMA configuration for the endpoint). Firmware would typically use this interrupt to unload the data packet from the endpoint FIFO and reset the OPRDYI bit to 0.

If a data packet is received when there is no room in the endpoint FIFO, an interrupt will be generated and the OORF bit set to 1. If USB0 receives an ISO data packet with a CRC error, the data packet will be loaded into the endpoint FIFO, OPRDYI will be set to 1, an interrupt (if enabled) will be generated, and the ODERRF bit will be set to 1. Software should check the ODERRF bit each time a data packet is unloaded from an ISO OUT endpoint FIFO.

38.15. DMA Configuration and Usage

A DMA channel may be used to transfer data for IN and OUT packets between system memory and an endpoint FIFO. The DMA module for Endpoints 1-4 supports only word (32-bit) reads and writes. Each DMA transfer consists of up to 8 words (32 bytes). For an IN endpoint (writes from the device to the host), the DMA will move data from the source location in memory to the internal USB0 DMA buffer, and hardware then transfers this buffer into the appropriate USB FIFO. For an OUT endpoint (writes from the host to the device), hardware will move data from the OUT endpoint FIFO to the internal USB0 DMA buffer and makes a DMA request that causes the DMA channel to transfer the data to the destination location in memory. These sequences repeat until the FIFO is fully loaded or unloaded. For both IN and OUT endpoints, the DMA channel must target the DMAFIFO register to access the internal USB0 DMA buffer. Firmware should never directly access the DMAFIFO register. There is no DMA support for Endpoint 0.

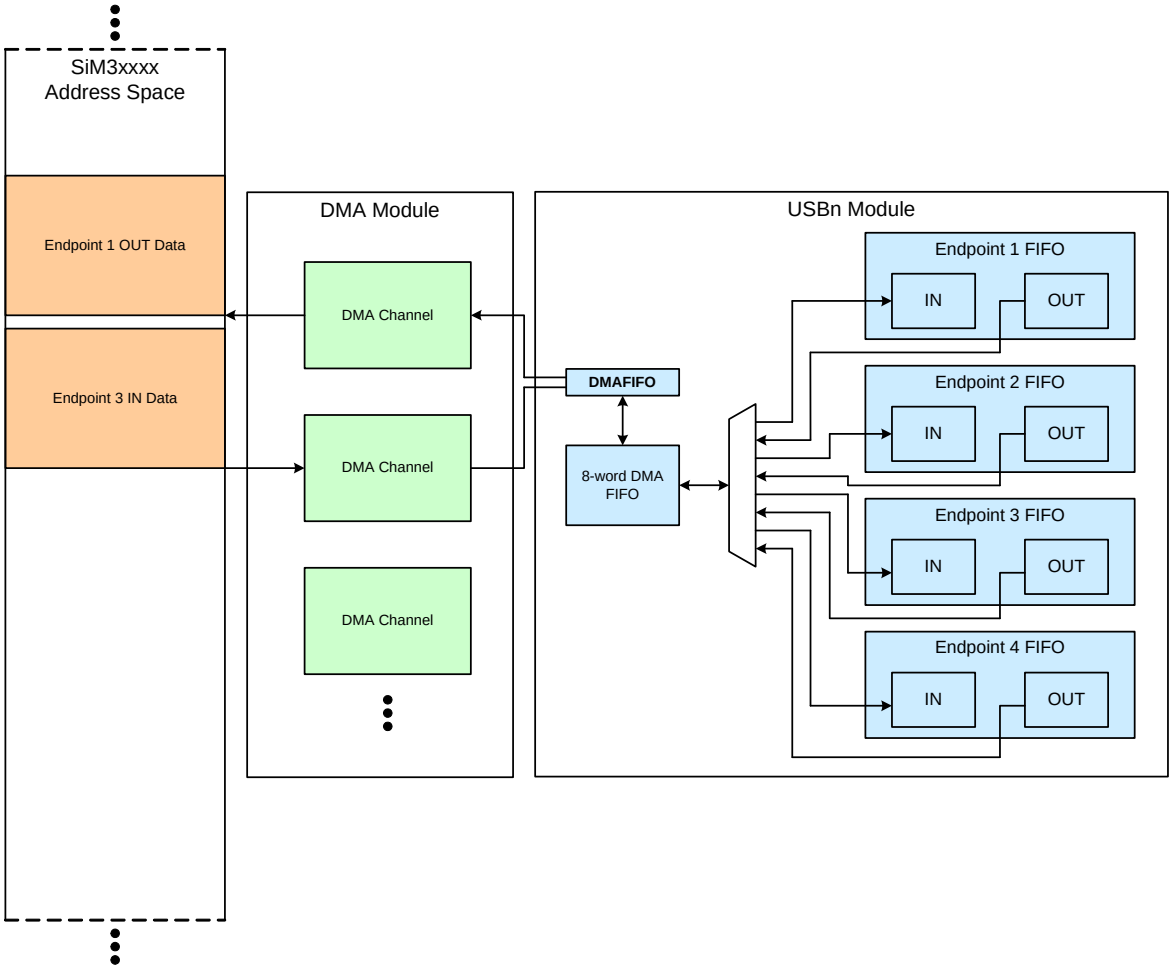


Figure 38.5. USB DMA Configuration

In the case of multiple Endpoint transactions occurring simultaneously, the DMA module will arbitrate between multiple DMA channels and Endpoints after each 8-word transfer. The fixed priority for the arbitration is (in order of highest to lowest priority): Endpoint 4 OUT, Endpoint 3 OUT, Endpoint 2 OUT, Endpoint 1 OUT, Endpoint 4 IN, Endpoint 3 IN, Endpoint 2 IN, and Endpoint 1 IN. Once the DMA module begins an 8-word transfer for an endpoint, a transfer for another endpoint will not begin until the original transfer completes. This allows all DMA channels interfacing with the USB0 module to use the same internal USB0 DMA buffer, which is accessed through the DMAFIFO register.

USB0 FIFOs targeted by the DMA module should not be manually written to or read from. Attempting an unsafe FIFO access during a DMA buffer cycle can delay the cycle by a significant number of clock cycles and potentially cause a loss of data. These unsafe operations include:

1. Manually writing to EPFIFO with an active IN DMA.
2. Manually reading from EPFIFO with an active OUT DMA.
3. Writing to an EPCONTROL register with an active OUT DMA.
4. Writing to an EPCONTROL register with an active IN DMA.

38.15.1. DMA and Interrupts

If a DMA channel is enabled for an endpoint, interrupts (IPRDYI and OPRDYI) will be suppressed except when the DMA encounters the last packet of the transfer (set by OMAXP and IMAXP in the endpoint EPMPSIZE register).

For IN packets, the block length in whole words is programmed into the DMA setup, and the DMA complete interrupt will occur when DMA finishes loading this number of whole words into the endpoint FIFO.

For OUT packets, the number of bytes sent to the device from the host can be variable, so the DMA channel should be configured to expect the largest possible block. The USB module then interrupts when it receives the last packet of the block, which is denoted by a packet that is not the maximum size. At this point, a DMA complete interrupt will not occur if the DMA module is still expecting data.

The USB DMA can be configured to load or unload all whole words from the last packet before issuing an interrupt. Any remaining data must be read or written manually by firmware before IPRDYI is set or OPRDYI is cleared.

38.15.2. Configuring the DMA for an IN Transfer

To use a DMA channel to complete a transfer to a Bulk/Interrupt or Isochronous IN Endpoint:

1. Configure the endpoint appropriately (Bulk/Interrupt/Isochronous mode, IN or Split Mode, etc.). The AUTOSETEN bit must be set to 1.
2. Set the maximum packet size using the endpoint's EPMPSIZE register.
3. Set up the desired DMA channel to source from a location in memory and write to the DMAFIFO register using up to 8-word transfers.
4. Enable the DMA for the IN endpoint.
5. When the DMA interrupt occurs on the last packet, disable the DMA channel, write any remaining bytes to the endpoint EPFIFO (not to the DMAFIFO register), and set the IPRDYI bit.

The IN DMA channel transfer is shown in Figure 38.6.

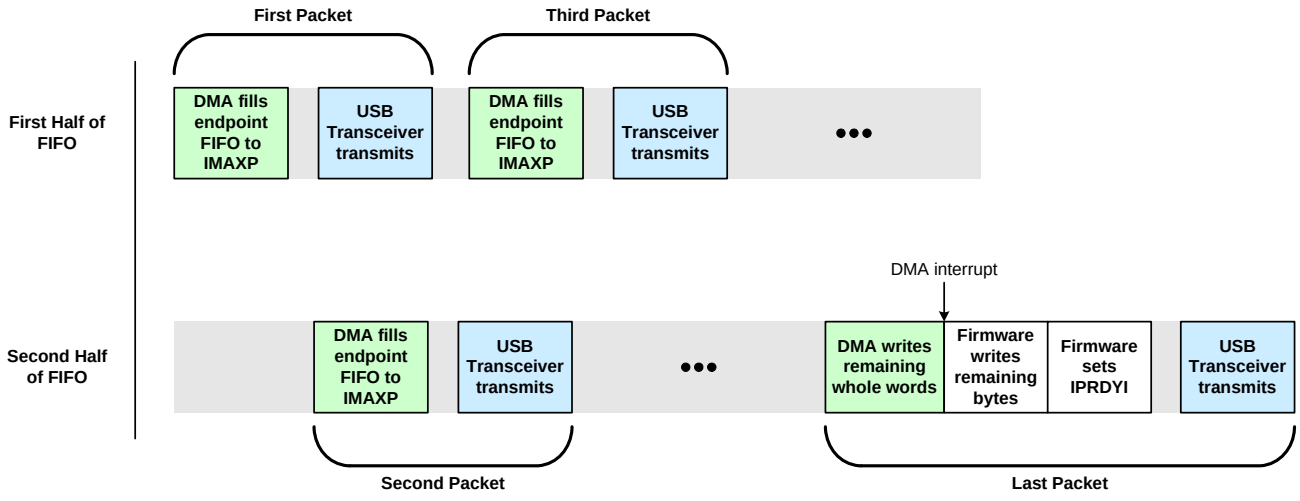


Figure 38.6. USB DMA IN Transfer

38.15.3. Configuring the DMA for an OUT Transfer

To use a DMA channel to complete a transfer from a Bulk/Interrupt or Isochronous OUT Endpoint:

1. Configure the endpoint appropriately (Bulk/Interrupt/Isochronous mode, OUT or Split Mode, etc.). The AUTOCLREN bit must be set to 1.
2. Set the maximum packet size using the endpoint's EPMPSIZE register.
3. Set up the desired DMA channel to source from the DMAFIFO register and write to a location in memory using exactly 8-word transfers.
4. Enable the DMA for the OUT endpoint.
5. When the USB interrupt occurs on the last packet, disable the DMA channel, read any remaining bytes from the endpoint EPFIFO (not from the DMAFIFO register), and clear the OPRDYI bit.

The OUT DMA channel transfer is shown in Figure 38.7.

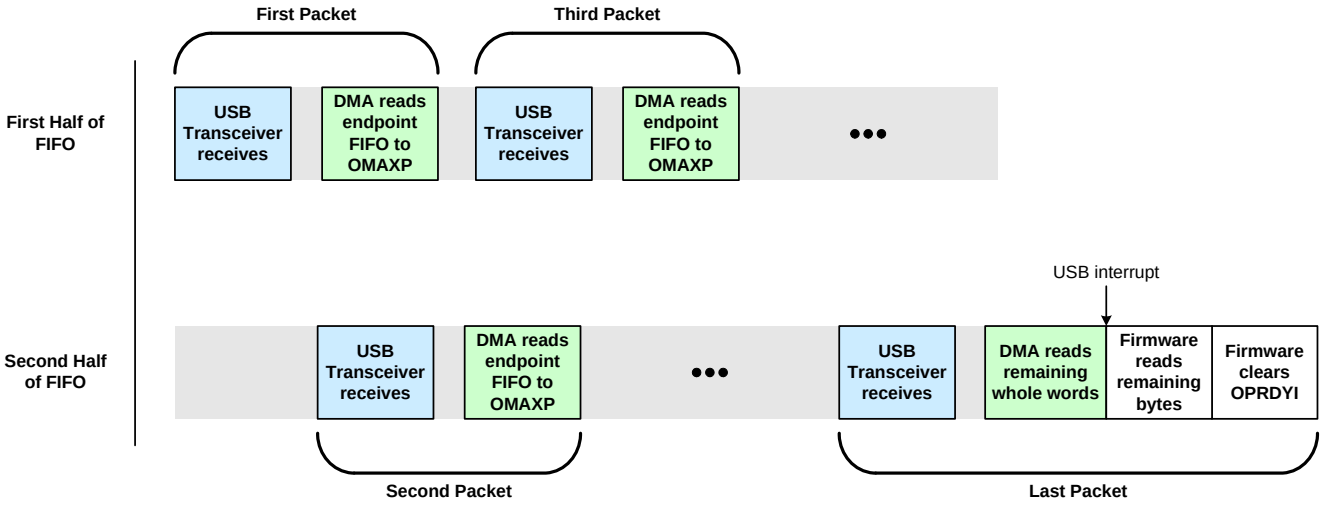


Figure 38.7. USB DMA OUT Transfer

38.15.4. Cancelling DMA Transfers

To cancel a USB0 DMA IN operation:

1. Disable the system DMA channel for the endpoint IN FIFO.
2. Flush the USB0 DMA buffer by writing 1 to DFIFOFL in the DMACONTROL register.
3. Flush the partial IN FIFO by writing 1 to IFIFOFL in the endpoint's EPCONTROL register.

To cancel a USB0 DMA OUT operation:

1. Flush the partial OUT FIFO by writing 1 to OFIFOFL in the endpoint's EPCONTROL register.
2. Flush the USB DMA buffer by writing 1 to DFIFOFL in the DMACONTROL register.
3. Disable the system DMA channel for the endpoint OUT FIFO.

38.15.5. Maximum Packet Size

Endpoints 1-4 have an IN and OUT maximum packet size. This maximum packet size is set as 8 times the value in the EPMPSIZE register (with one exception noted below). For Bulk and Interrupt transfers, the maximum packet size must be a multiple of 8 up to 64. For Isochronous packets, the maximum size must be a multiple of 8 up to 1016, with a final maximum size of 1023. Because of this special case in the USB spec, a maximum packet size value in the EPMPSIZE register of 128 is interpreted as 1023 bytes. If the maximum packet size is set to less than half the available FIFO size for the Endpoint, double-buffering will automatically be enabled.

If the AUTOSETEN bit is set to 1, the Endpoint IPRDYI bit will be automatically set when the last byte of a maximum-size packet is written to the Endpoint IN FIFO. If the AUTOCLREN bit is set to 1, the OPRDYI bit is automatically cleared when the last byte of a maximum-size packet is read from the Endpoint OUT FIFO.

The endpoint's EPMPSIZE register must always be configured.

38.16. USB0 Registers

This section contains the detailed register descriptions for USB0 registers.

Register 38.1. USB0_FADDR: Function Address

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved								FADDRUPD	FADDR						
Type	R								R	RW						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

USB0_FADDR = 0x4001_8000

Table 38.3. USB0_FADDR Register Bit Descriptions

Bit	Name	Function
31:8	Reserved	Must write reset value.
7	FADDRUPD	**Function Address Update.** This bit is set to 1 by the hardware when a value is written to the FADDR bits. This bit is cleared by hardware when the new address takes effect. 0: The last address written to FADDR is in effect. 1: The last address written to FADDR is not yet in effect.
6:0	FADDR	**Function Address.** Holds the 7-bit function address for USB. This address should be written by software when the SET_ADDRESS standard device request is received on Endpoint 0. The new address takes effect when the device request completes.

Notes:

1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Register 38.2. USB0_POWER: Power Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved								ISOUPDMD	Reserved	DITHEN	USBINH	RSTDETF	RESUME	SUSMDF	SUSDEN
Type	R								RW	R	RW	RW	R	RW	R	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
Register ALL Access Address USB0_POWER = 0x4001_8010																

Table 38.4. USB0_POWER Register Bit Descriptions

Bit	Name	Function
31:8	Reserved	Must write reset value.
7	ISOUPDMD	**ISO Update Mode.** This bit affects all IN Isochronous endpoints. 0: When software writes IPRDYI = 1, USB will send the packet when the next IN token is received. 1: When software writes IPRDYI = 1, USB will wait for a SOF token before sending the packet. If an IN token is received before a SOF token, USB will send a zero-length data packet.
6	Reserved	Must write reset value.
5	DITHEN	**USB Dither Enable.** This bit enables automatic dithering of the USB signals on the D+ and D- pins. This will help reduce switching noise for sensitive systems. 0: Disable automatic USB dithering. 1: Enable automatic USB dithering.
4	USBINH	**USB Inhibit.** This bit is set to 1 following any device reset. Software should clear this bit after all USB and transceiver initialization is complete. Software cannot set this bit to 1. 0: Enable the USB module. 1: USB module inhibited. All USB traffic is ignored.

Notes:
1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Table 38.4. USB0_POWER Register Bit Descriptions

Bit	Name	Function
3	RSTDETF	**Reset Detect Flag.** 0: Reset signaling is not present on the bus. 1: Reset signaling detected on the bus.
2	RESUME	**Force Resume.** Writing a 1 to this bit while in suspend mode (SUSMDF = 1) forces the USB module to generate resume signaling on the bus (a remote wakeup event). Software should write RESUME to 0 after 10 to 15 ms to end the resume signaling. An interrupt is generated, and hardware clears SUSMDF, when software clears RESUME.
1	SUSMDF	**Suspend Mode Flag.** This bit is set to 1 by hardware when USB enters suspend mode. This bit is cleared by hardware when software clears RESUME (following a remote wakeup). 0: The USB module is not in suspend mode. 1: The USB module is in suspend mode.
0	SUSDEN	**Suspend Detection Enable.** 0: Disable suspend detection. The USB module will ignore suspend signaling on the bus. 1: Enable suspend detection. The USB module will enter suspend mode if it detects suspend signalling on the bus.

Notes:
1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Register 38.3. USB0_IOINT: IN/OUT Endpoint Interrupt Flags

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved											OUT4I	OUT3I	OUT2I	OUT1I	Reserved
Type	R											RW	RW	RW	RW	R
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved											IN4I	IN3I	IN2I	IN1I	EP0I
Type	R											RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

USB0_IOINT = 0x4001_8020

This register also supports CLR access at (ALL+0x8)

Table 38.5. USB0_IOINT Register Bit Descriptions

Bit	Name	Function
31:21	Reserved	Must write reset value.
20	OUT4I	**OUT Endpoint 4 Interrupt Flag.** This bit must be cleared by software. Read: 0 : OUT Endpoint 4 interrupt has not occurred. 1 : OUT Endpoint 4 interrupt occurred. Write: 0 : No effect. 1 : Clear the interrupt.
19	OUT3I	**OUT Endpoint 3 Interrupt Flag.** This bit must be cleared by software. Read: 0 : OUT Endpoint 3 interrupt has not occurred. 1 : OUT Endpoint 3 interrupt occurred. Write: 0 : No effect. 1 : Clear the interrupt.

Notes:

1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Table 38.5. USB0_IOINT Register Bit Descriptions

Bit	Name	Function
18	OUT2I	**OUT Endpoint 2 Interrupt Flag.** This bit must be cleared by software. Read: 0 : OUT Endpoint 2 interrupt has not occurred. 1 : OUT Endpoint 2 interrupt occurred. Write: 0 : No effect. 1 : Clear the interrupt.
17	OUT1I	**OUT Endpoint 1 Interrupt Flag.** This bit must be cleared by software. Read: 0 : OUT Endpoint 1 interrupt has not occurred. 1 : OUT Endpoint 1 interrupt occurred. Write: 0 : No effect. 1 : Clear the interrupt.
16:5	Reserved	Must write reset value.
4	IN4I	**IN Endpoint 4 Interrupt Flag.** This bit must be cleared by software. Read: 0 : IN Endpoint 4 interrupt has not occurred. 1 : IN Endpoint 4 interrupt occurred. Write: 0 : No effect. 1 : Clear the interrupt.
3	IN3I	**IN Endpoint 3 Interrupt Flag.** This bit must be cleared by software. Read: 0 : IN Endpoint 3 interrupt has not occurred. 1 : IN Endpoint 3 interrupt occurred. Write: 0 : No effect. 1 : Clear the interrupt.
2	IN2I	**IN Endpoint 2 Interrupt Flag.** This bit must be cleared by software. Read: 0 : IN Endpoint 2 interrupt has not occurred. 1 : IN Endpoint 2 interrupt occurred. Write: 0 : No effect. 1 : Clear the interrupt.

Notes:
1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Table 38.5. USB0_IOINT Register Bit Descriptions

Bit	Name	Function
1	IN1I	**IN Endpoint 1 Interrupt Flag.** This bit must be cleared by software. Read: 0 : IN Endpoint 1 interrupt has not occurred. 1 : IN Endpoint 1 interrupt occurred. Write: 0 : No effect. 1 : Clear the interrupt.
0	EP0I	**Endpoint 0 Interrupt Flag.** This bit must be cleared by software. Read: 0 : Endpoint 0 interrupt has not occurred. 1 : Endpoint 0 interrupt occurred. Write: 0 : No effect. 1 : Clear the interrupt.

Notes:
1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Register 38.4. USB0_CMINT: Common Interrupt Flags

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved												SOFI	RSTI	RESI	SUSI
Type	R												RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

USB0_CMINT = 0x4001_8030

This register also supports CLR access at (ALL+0x8)

Table 38.6. USB0_CMINT Register Bit Descriptions

Bit	Name	Function
31:4	Reserved	Must write reset value.
3	SOFI	**Start of Frame Interrupt Flag.** This bit is set by hardware when a SOF token is received. This interrupt event is synthesized by hardware; an interrupt will be generated when hardware expects to receive a SOF event, even if the actual SOF signal is missed or corrupted. This bit must be cleared by software. Read: 0 : SOF interrupt has not occurred. 1 : SOF interrupt occurred. Write: 0 : No effect. 1 : Clear the interrupt.
2	RSTI	**Reset Interrupt Flag.** This bit is set by hardware when Reset signaling is detected on the bus. This bit must be cleared by software. Read: 0 : Reset interrupt has not occurred. 1 : Reset interrupt occurred. Write: 0 : No effect. 1 : Clear the interrupt.

Notes:

1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Table 38.6. USB0_CMINT Register Bit Descriptions

Bit	Name	Function
1	RESI	**Resume Interrupt Flag.** This bit is set by hardware when Resume signaling is detected on the bus while USB is in Suspend mode. This bit must be cleared by software. Read: 0 : Resume interrupt has not occurred. 1 : Resume interrupt occurred. Write: 0 : No effect. 1 : Clear the interrupt.
0	SUSI	**Suspend Interrupt Flag.** When Suspend detection is enabled (SUSDEN = 1), this bit is set by hardware when Suspend signaling is detected on the bus. This bit must be cleared by software. Read: 0 : Suspend interrupt has not occurred. 1 : Suspend interrupt occurred. Write: 0 : No effect. 1 : Clear the interrupt.

Notes:
1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Register 38.5. USB0_IOINTE: IN/OUT Endpoint Interrupt Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved											OUT4IEN	OUT3IEN	OUT2IEN	OUT1IEN	Reserved
Type	R											RW	RW	RW	RW	R
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved											IN4IEN	IN3IEN	IN2IEN	IN1IEN	EP0IEN
Type	R											RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

Register ALL Access Address
USB0_IOINTE = 0x4001_8040

Table 38.7. USB0_IOINTE Register Bit Descriptions

Bit	Name	Function
31:21	Reserved	Must write reset value.
20	OUT4IEN	**OUT Endpoint 4 Interrupt Enable.** 0: Disable the OUT Endpoint 4 interrupt. 1: Enable the OUT Endpoint 4 interrupt.
19	OUT3IEN	**OUT Endpoint 3 Interrupt Enable.** 0: Disable the OUT Endpoint 3 interrupt. 1: Enable the OUT Endpoint 3 interrupt.
18	OUT2IEN	**OUT Endpoint 2 Interrupt Enable.** 0: Disable the OUT Endpoint 2 interrupt. 1: Enable the OUT Endpoint 2 interrupt.
17	OUT1IEN	**OUT Endpoint 1 Interrupt Enable.** 0: Disable the OUT Endpoint 1 interrupt. 1: Enable the OUT Endpoint 1 interrupt.
16:5	Reserved	Must write reset value.
4	IN4IEN	**IN Endpoint 4 Interrupt Enable.** 0: Disable the IN Endpoint 4 interrupt. 1: Enable the IN Endpoint 4 interrupt.

Notes:
1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Table 38.7. USB0_IOINTE Register Bit Descriptions

Bit	Name	Function
3	IN3IEN	**IN Endpoint 3 Interrupt Enable.** 0: Disable the IN Endpoint 3 interrupt. 1: Enable the IN Endpoint 3 interrupt.
2	IN2IEN	**IN Endpoint 2 Interrupt Enable.** 0: Disable the IN Endpoint 2 interrupt. 1: Enable the IN Endpoint 2 interrupt.
1	IN1IEN	**IN Endpoint 1 Interrupt Enable.** 0: Disable the IN Endpoint 1 interrupt. 1: Enable the IN Endpoint 1 interrupt.
0	EP0IEN	**Endpoint 0 Interrupt Enable.** 0: Disable the Endpoint 0 interrupt. 1: Enable the Endpoint 0 interrupt.

Notes:
1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Register 38.6. USB0_CMINTEPE: Common Interrupt and Endpoint Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved											EP4EN	EP3EN	EP2EN	EP1EN	EP0EN
Type	R											RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved												SOFIEN	RSTIEN	RESIEN	SUSIEN
Type	R												RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

Register ALL Access Address

USB0_CMINTEPE = 0x4001_8050

Table 38.8. USB0_CMINTEPE Register Bit Descriptions

Bit	Name	Function
31:21	Reserved	Must write reset value.
20	EP4EN	**Endpoint 4 Enable.** This bit enables and disables Endpoint 4. 0: Disable Endpoint 4 (no NACK, ACK, or STALL on the USB network). 1: Enable Endpoint 4 (normal).
19	EP3EN	**Endpoint 3 Enable.** This bit enables and disables Endpoint 3. 0: Disable Endpoint 3 (no NACK, ACK, or STALL on the USB network). 1: Enable Endpoint 3 (normal).
18	EP2EN	**Endpoint 2 Enable.** This bit enables and disables Endpoint 2. 0: Disable Endpoint 2 (no NACK, ACK, or STALL on the USB network). 1: Enable Endpoint 2 (normal).
17	EP1EN	**Endpoint 1 Enable.** This bit enables and disables Endpoint 1. 0: Disable Endpoint 1 (no NACK, ACK, or STALL on the USB network). 1: Enable Endpoint 1 (normal).

Notes:

1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Table 38.8. USB0_CMINTEPE Register Bit Descriptions

Bit	Name	Function
16	EP0EN	**Endpoint 0 Enable.** This bit enables and disables Endpoint 0. 0: Disable Endpoint 0 (no NACK, ACK, or STALL on the USB network). 1: Enable Endpoint 0 (normal).
15:4	Reserved	Must write reset value.
3	SOFIEN	**Start of Frame Interrupt Enable.** 0: Disable the SOF interrupt. 1: Enable the SOF interrupt.
2	RSTIEN	**Reset Interrupt Enable.** 0: Disable the Reset interrupt. 1: Enable the Reset interrupt.
1	RESIEN	**Resume Interrupt Enable.** 0: Disable the Resume interrupt. 1: Enable the Resume interrupt.
0	SUSIEN	**Suspend Interrupt Enable.** 0: Disable the Suspend interrupt. 1: Enable the Suspend interrupt.

Notes:
1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Register 38.7. USB0_CRCONTROL: Clock Recovery Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved								CREN	CRSSEN	LSCRMD	OLEN	Reserved			
Type	R								RW	RW	RW	RW	R			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

USB0_CRCONTROL = 0x4001_8060

Table 38.9. USB0_CRCONTROL Register Bit Descriptions

Bit	Name	Function
31:8	Reserved	Must write reset value.
7	CREN	**Clock Recovery Enable.** This bit enables and disables the USB clock recovery feature. 0: Disable clock recovery. 1: Enable clock recovery.
6	CRSSEN	**Clock Recovery Single Step Enable.** This bit forces the oscillator control into single step mode during clock recovery. 0: Normal calibration mode. 1: Single step mode.
5	LSCRMD	**Low Speed Clock Recovery Mode.** This bit must be set to 1 if clock recovery is used when operating as a Low Speed USB device. 0: Full Speed Mode. 1: Low Speed Mode.
4	OLEN	**Oscillator Open-Loop Mode Enable.** This bit freezes the USB oscillator (USBnOSC) frequency at its current value. 0: Do not freeze the USB oscillator output frequency (closed loop mode). 1: Freeze the USB oscillator output frequency (open loop mode).
3:0	Reserved	Must write reset value.

Notes:

1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Register 38.8. USB0_FRAME: Frame Number

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved					FRAMENUM										
Type	R					R										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

USB0_FRAME = 0x4001_8070

Table 38.10. USB0_FRAME Register Bit Descriptions

Bit	Name	Function
31:11	Reserved	Must write reset value.
10:0	FRAMENUM	**Frame Number.** This field represents the 11-bit frame number of the last received packet.

Notes:

1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Register 38.9. USB0_TCONTROL: Transceiver Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved								PUEN	PHYEN	SSEL	PHYTST		DFREC	DP	DN
Type	R								RW	RW	RW	RW		R	R	R
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address
USB0_TCONTROL = 0x4001_8200

Table 38.11. USB0_TCONTROL Register Bit Descriptions

Bit	Name	Function
31:8	Reserved	Must write reset value.
7	PUEN	**Internal Pull-up Resistor Enable.** The location of the pull-up resistor (D+ or D-) is determined by the SSEL bit. 0: Disable the internal pull-up resistor (device effectively detached from the USB network). 1: Enable the internal pull-up resistor when VBUS is present (device is attached to the USB network).
6	PHYEN	**Physical Layer Enable.** 0: Disable the USB physical layer Transceiver (suspend). 1: Enable the USB physical layer Transceiver (normal).
5	SSEL	**USB Speed Select.** 0: USB operates as a Low Speed device. If enabled, the internal pull-up resistor appears on the D- line. 1: USB operates as a Full Speed device. If enabled, the internal pull-up resistor appears on the D+ line.
4:3	PHYTST	**Physical Layer Test.** 00: Mode 0: Normal (non-test mode) (D+ = X, D- = X). 01: Mode 1: Differential 1 Forced (D+ = 1, D- = 0). 10: Mode 2: Differential 0 Forced (D+ = 0, D- = 1). 11: Mode 3: Single-Ended 0 Forced (D+ = 0, D- = 0).

Notes:
1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Table 38.11. USB0_TCONTROL Register Bit Descriptions

Bit	Name	Function
2	DFREC	**Differential Receiver State.** The state of this bit indicates the current differential value present on the D+ and D- lines when PHYEN is set to 1. 0: Differential 0 signalling is present on the bus. 1: Differential 1 signalling is present on the bus.
1	DP	**D+ Signal State.** This bit indicates the current logic level of the D+ pin. 0: D+ signal currently at logic 0. 1: D+ signal currently at logic 1.
0	DN	**D- Signal State.** This bit indicates the current logic level of the D- pin. 0: D- signal currently at logic 0. 1: D- signal currently at logic 1.

Notes:
1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Register 38.10. USB0_CLKSEL: Module Clock Select

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved									RESET	CLKDIV		Reserved		CLKSEL	
Type	R								RW	W	RW		R		RW	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address
USB0_CLKSEL = 0x4001_8300

Table 38.12. USB0_CLKSEL Register Bit Descriptions

Bit	Name	Function
31:7	Reserved	Must write reset value.
6	RESET	**USB Reset.** Writing a 1 forces an asynchronous reset of the USB module. This bit is cleared by hardware after the reset operation completes. 0: Do not reset the USB module. 1: Reset the USB module.
5:4	CLKDIV	**USB Clock Divider.** Divides the clock selected by CLKSEL. 00: The USB module uses the selected input clock divided by 1. 01: The USB module uses the selected input clock divided by 2. 10: The USB module uses the selected input clock divided by 4. 11: The USB module uses the selected input clock divided by 8.
3:2	Reserved	Must write reset value.
1:0	CLKSEL	**USB Clock Select.** 00: Select the USB Oscillator as the USB clock. 01: Select the PLL output as the USB clock. 10: Select the External Oscillator output (EXTOSCn) as the USB clock. 11: Reserved.

Notes:
1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Register 38.11. USB0_OSCCONTROL: Oscillator Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved								OSCEN	Reserved	SUSPEND	Reserved				
Type	R								RW	R	RW	R				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

USB0_OSCCONTROL = 0x4001_8310

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 38.13. USB0_OSCCONTROL Register Bit Descriptions

Bit	Name	Function
31:8	Reserved	Must write reset value.
7	OSCEN	**USB Oscillator Enable.** 0: Disable the USB oscillator. 1: Enable the USB oscillator.
6	Reserved	Must write reset value.
5	SUSPEND	**USB Oscillator Suspend.** Setting this bit to 1 places the USB Oscillator in Suspend mode. The oscillator resumes operation when a transition on VBUS or a non-idle USB bus state is detected. 0: The USB oscillator is not suspended. 1: Suspend the USB oscillator.
4:0	Reserved	Must write reset value.

Notes:

1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Register 38.12. USB0_AFADJUST: Oscillator Additional Frequency Adjust

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved									DITHEN	FINEFADJ					
Type	R								RW	RW	RW					
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

Register ALL Access Address

USB0_AFADJUST = 0x4001_8320

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 38.14. USB0_AFADJUST Register Bit Descriptions

Bit	Name	Function
31:7	Reserved	Must write reset value.
6	DITHEN	**USB Oscillator Dithering Enable.** Setting this bit to 1 improves the average USB clock accuracy by dithering the oscillator control value every four oscillator cycles. 0: Disable USB oscillator dithering. 1: Enable USB oscillator dithering.
5:0	FINEFADJ	**USB Oscillator Fine Output Frequency Adjust.** This field controls the output frequency of the USB oscillator in fine steps. The reset value is factory calibrated to generate a USB oscillator frequency of 48 MHz. This register is modified by the clock recovery hardware to fine-tune the USB oscillator to meet the requirements for USB clock tolerance.

Notes:

1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Register 38.13. USB0_FADJUST: Oscillator Frequency Adjust

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved									FADJ						
Type	R									RW						
Reset	0	0	0	0	0	0	0	0	0	X	X	X	X	X	X	X

Register ALL Access Address

USB0_FADJUST = 0x4001_8330

Table 38.15. USB0_FADJUST Register Bit Descriptions

Bit	Name	Function
31:7	Reserved	Must write reset value.
6:0	FADJ	**Oscillator Output Frequency Adjust.** This field controls the output frequency of the USB oscillator in coarse steps. The reset value is factory calibrated to generate a USB oscillator frequency of 48 MHz.

Notes:

1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Register 38.14. USB0_DMAFIFO: DMA Data FIFO Access

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	DMAFIFO[31:16]															
Type	RW															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	DMAFIFO[15:0]															
Type	RW															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Register ALL Access Address

USB0_DMAFIFO = 0x4001_8400

Table 38.16. USB0_DMAFIFO Register Bit Descriptions

Bit	Name	Function
31:0	DMAFIFO	**DMA Data FIFO Access.** These bits provide an access point for the 8-word (32-byte) DMA buffer. This register should be the destination or source for the DMA module channel when using that channel for a USB IN (destination) or OUT (source) endpoint. This register should not be accessed by firmware.

Notes:

1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.
2. Reads of this register modify the state of hardware. Debug logic should take care when reading this register.

Register 38.15. USB0_DMACONTROL: DMA Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved								DFIFOFL	DBUSYF	TERRF	DBGMD	Reserved			
Type	R								RW	R	RW	RW	R			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

USB0_DMACONTROL = 0x4001_8410

Table 38.17. USB0_DMACONTROL Register Bit Descriptions

Bit	Name	Function
31:8	Reserved	Must write reset value.
7	DFIFOFL	**USB DMA Buffer Flush Control.** Writing a 1 to this bit flushes the 32-byte USB DMA buffer.
6	DBUSYF	**USB DMA Busy Flag.** This bit indicates USB FIFO data is being loaded from or unloaded to the 8-word DMA buffer. If software attempts to access certain USB registers while DBUSYF is 1, the APB interface will be stalled until the DMA buffer completes the active operation. The registers that can always be accessed without stalling the APB interface are OSCCONTROL, AFADJUST, FADJUST, DMAFIFO, TCONTROL, and CLKSEL. 0: The DMA buffer is not busy. 1: The DMA buffer is busy reading or writing an 8-word packet.
5	TERRF	**Timeout Error Flag.** This bit is set by hardware when a read or write operation times out. This can occur when access to some USB registers is attempted when the selected USB clock source is not running or is too slow. The registers that can be accessed at any time regardless of the USBnOSC clock are OSCCONTROL, AFADJUST, FADJUST, DMAFIFO, TCONTROL, and CLKSEL. This bit must be cleared by software. 0: A timeout error has not occurred. 1: A timeout error occurred.

Notes:

1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Table 38.17. USB0_DMACONTROL Register Bit Descriptions

Bit	Name	Function
4	DBGMD	**USB DMA Debug Mode.** 0: The USB module will continue to operate while the core is halted in debug mode. 1: A debug breakpoint will prevent the USB DMA buffer from transferring data to and from the USB FIFOs when the core is halted.
3:0	Reserved	Must write reset value.

Notes:
1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Register 38.16. USB0_EP0CONTROL: Endpoint 0 Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved								SUENDIS	OPRDYIS	SDSTL	SUENDI	DEND	STSTLI	IPRDYI	OPRDYI
Type	R								RW	RW	RW	R	RW	RW	RW	R
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

USB0_EP0CONTROL = 0x4001_8810

Table 38.18. USB0_EP0CONTROL Register Bit Descriptions

Bit	Name	Function
31:8	Reserved	Must write reset value.
7	SUENDIS	**Serviced Setup End Interrupt Flag.** Software should set this bit to 1 after servicing a Setup End (bit SUENDI) event. Hardware clears the SUENDI bit when software writes 1 to SUENDIS. This bit always reads 0. 0: Setup end has not been serviced. 1: Setup end has been serviced.
6	OPRDYIS	**Serviced Out Packet Ready Interrupt Flag.** Software should write 1 to this bit after servicing a received Endpoint 0 packet. The OPRDYI bit will be cleared by a write of 1 to OPRDYIS. This bit always reads 0. 0: The out packet has not been processed. 1: The out packet has been received and accepted.
5	SDSTL	**Send Stall.** Software can write 1 to this bit to terminate the current transfer (due to an error condition, unexpected transfer request, etc.). Hardware will clear this bit to 0 when the STALL handshake is transmitted. 0: The STALL handshake has been transmitted or not triggered. 1: Initiate a STALL condition.

Notes:

1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Table 38.18. USB0_EP0CONTROL Register Bit Descriptions

Bit	Name	Function
4	SUENDI	**Setup End Interrupt Flag.** Hardware sets this read-only bit to 1 when a control transaction ends before software has written 1 to the DEND bit. This occurs when the host sends a setup packet before the previous control transaction is complete. Hardware clears this bit when software writes 1 to SSUEND. 0: The current packet is not the last packet of setup. 1: The current packet is the last packet of setup.
3	DEND	**Data End.** Software should write 1 to this bit: 1) When writing 1 to IPRDYI for the last outgoing data packet. 2) When writing 1 to IPRDYI for a zero-length data packet. 3) When writing 1 to OPRDYIS after servicing the last incoming data packet. This bit is automatically cleared by hardware. 0: The current packet is not the last packet of the transfer. 1: The current packet is the last packet of the transfer.
2	STSTLI	**Sent Stall Interrupt Flag.** Hardware sets this bit to 1 after transmitting a STALL handshake signal. This interrupt flag must be cleared by software. Read: 0 : A STALL handshake has not been sent or is cleared. 1 : STALL handshake sent. Write: 0 : Clear the interrupt. 1 : No effect.
1	IPRDYI	**IN Packet Ready Indicator.** Software should write 1 to this bit after loading a data packet into the Endpoint 0 FIFO for transmit. Hardware clears this bit and generates an interrupt under one of the following conditions: 1) The packet is transmitted. 2) The packet is overwritten by an incoming SETUP packet. 3) The packet is overwritten by an incoming OUT packet. 0: A packet is not ready for transmission to host. 1: A packet is ready for transmission to host.
0	OPRDYI	**OUT Packet Ready Interrupt Flag.** Hardware sets this read-only bit to 1 and generates an interrupt when a data packet has been received. This bit is cleared only when software writes 1 to the OPRDYIS bit. 0: A packet is not available. 1: A packet is available.

Notes:
1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Register 38.17. USB0_EP0COUNT: Endpoint 0 Data Count

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved									COUNT						
Type	R									R						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

USB0_EP0COUNT = 0x4001_8820

Table 38.19. USB0_EP0COUNT Register Bit Descriptions

Bit	Name	Function
31:7	Reserved	Must write reset value.
6:0	COUNT	**Endpoint 0 OUT Data Count.** This 7-bit number indicates the number of received data bytes in the Endpoint 0 FIFO. This number is only valid while bit OPRDYI is 1.

Notes:

1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Register 38.18. USB0_EP0FIFO: Endpoint 0 Data FIFO Access

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	FIFO[31:16]															
Type	RW															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	FIFO[15:0]															
Type	RW															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Register ALL Access Address

USB0_EP0FIFO = 0x4001_8830

Table 38.20. USB0_EP0FIFO Register Bit Descriptions

Bit	Name	Function
31:0	FIFO	**Endpoint 0 Data FIFO.** Any bytes written to or read from this register will be loaded into or out of the Endpoint 0 Data FIFO. Any right-justified word, half-word, or byte operation is allowed. Non right-justified accesses will return incorrect data and should not be used. Multi-byte reads and writes occur least-significant byte first.

Notes:

1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.
2. Reads of this register modify the state of hardware. Debug logic should take care when reading this register.

38.17. USB0 Register Memory Map

Table 38.21. USB0 Memory Map

Register Name	USB0_FADDR	USB0_POWER	USB0_IOINT	USB0_CMINT	USB0_IOINTE
ALL Address	0x4001_8000	0x4001_8010	0x4001_8020	0x4001_8030	0x4001_8040
Access Methods	ALL	ALL	ALL | CLR	ALL | CLR	ALL
Bit 31	Reserved	Reserved	Reserved	Reserved	Reserved
Bit 30					
Bit 29					
Bit 28					
Bit 27					
Bit 26					
Bit 25					
Bit 24					
Bit 23					
Bit 22					
Bit 21					
Bit 20			OUT4I		OUT4IEN
Bit 19			OUT3I		OUT3IEN
Bit 18			OUT2I		OUT2IEN
Bit 17			OUT1I		OUT1IEN
Bit 16			Reserved		Reserved
Bit 15					
Bit 14					
Bit 13					
Bit 12					
Bit 11					
Bit 10					
Bit 9					
Bit 8					
Bit 7	FADDRUPD	ISOUPDMD	IN4I	SOFI	IN4IEN
Bit 6	FADDR	Reserved	IN3I	RSTI	IN3IEN
Bit 5		DITHEN	IN2I	RESI	IN2IEN
Bit 4		USBINH	IN1I	SUSI	IN1IEN
Bit 3		RSTDETF	EP0I		EP0IEN
Bit 2		RESUME			
Bit 1		SUSMDF			
Bit 0		SUSDEN			

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

Table 38.21. USB0 Memory Map

Register Name	USB0_CLKSEL	USB0_TCONTROL	USB0_FRAME	USB0_CRCONTROL	USB0_CMINTEPE
ALL Address	0x4001_8300	0x4001_8200	0x4001_8070	0x4001_8060	0x4001_8050
Access Methods	ALL	ALL	ALL	ALL	ALL
Bit 31	Reserved	Reserved	Reserved	Reserved	Reserved
Bit 30					
Bit 29					
Bit 28					
Bit 27					
Bit 26					
Bit 25					
Bit 24					
Bit 23					
Bit 22					
Bit 21					
Bit 20					EP4EN
Bit 19					EP3EN
Bit 18					EP2EN
Bit 17					EP1EN
Bit 16					EP0EN
Bit 15					Reserved
Bit 14					
Bit 13					
Bit 12					
Bit 11					
Bit 10					
Bit 9					
Bit 8					
Bit 7					
Bit 6					
Bit 5				CREN	
Bit 4				CRSSEN	
Bit 3				LSCRMD	SOFIEN
Bit 2				OLEN	RSTIEN
Bit 1			FRAMENUM	Reserved	RESIEN
Bit 0					SUSIEN
	RESET	PUEN			
	CLKDIV	PHYEN			
	Reserved	SSEL			
	CLKSEL	PHYTST			
		DFREC			
		DP			
		DN			

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

Table 38.21. USB0 Memory Map

Register Name	USB0_DMAFIFO	USB0_FADJUST	USB0_AFADJUST	USB0_OSCCONTROL
ALL Address	0x4001_8400	0x4001_8330	0x4001_8320	0x4001_8310
Access Methods	ALL	ALL	ALL | SET | CLR	ALL | SET | CLR
Bit 31	DMAFIFO	Reserved	Reserved	Reserved
Bit 30				
Bit 29				
Bit 28				
Bit 27				
Bit 26				
Bit 25				
Bit 24				
Bit 23				
Bit 22				
Bit 21				
Bit 20				
Bit 19				
Bit 18				
Bit 17				
Bit 16				
Bit 15				
Bit 14				
Bit 13				
Bit 12				
Bit 11				
Bit 10				
Bit 9				
Bit 8				
Bit 7		FADJ	DITHEN	OSCEN
Bit 6				Reserved
Bit 5			FINEFADJ	SUSPEND
Bit 4				Reserved
Bit 3				
Bit 2				
Bit 1				
Bit 0				

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

Table 38.21. USB0 Memory Map

Register Name	USB0_EP0FIFO	USB0_EP0COUNT	USB0_EP0CONTROL	USB0_DMACONTROL
ALL Address	0x4001_8830	0x4001_8820	0x4001_8810	0x4001_8410
Access Methods	ALL	ALL	ALL	ALL
Bit 31	FIFO	Reserved	Reserved	Reserved
Bit 30				
Bit 29				
Bit 28				
Bit 27				
Bit 26				
Bit 25				
Bit 24				
Bit 23				
Bit 22				
Bit 21				
Bit 20				
Bit 19				
Bit 18				
Bit 17				
Bit 16				
Bit 15				
Bit 14				
Bit 13				
Bit 12				
Bit 11				
Bit 10				
Bit 9				
Bit 8				
Bit 7			SUENDIS	DFIFOFL
Bit 6		COUNT	OPRDYIS	DBUSYF
Bit 5			SDSTL	TERRF
Bit 4			SUENDI	DBGMD
Bit 3			DEND	Reserved
Bit 2			STSTLI	
Bit 1			IPRDYI	
Bit 0			OPRDYI	

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

38.18. USB0_EP1-4 Registers

This section contains the detailed register descriptions for USB0_EP1, USB0_EP2, USB0_EP3 and USB0_EP4 registers.

Register 38.19. USBn_EPx_EPMPSIZE: Endpoint Maximum Packet Size

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved								OMAXP							
Type	R								RW							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved								IMAXP							
Type	R								RW							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

USB0_EP1_EPMPSIZE = 0x4001_8880
USB0_EP2_EPMPSIZE = 0x4001_8900
USB0_EP3_EPMPSIZE = 0x4001_8980
USB0_EP4_EPMPSIZE = 0x4001_8A00

Table 38.22. USBn_EPx_EPMPSIZE Register Bit Descriptions

Bit	Name	Function
31:24	Reserved	Must write reset value.
23:16	OMAXP	**OUT Maximum Packet Size.** These bits set the maximum packet size of the OUT endpoint in increments of 8 bytes. A value of 128 (0x80) is interpreted as 1023 bytes. Values above the available FIFO size divided by 8 (which depends on the endpoint settings) are illegal and should not be written. If the maximum packet size is set to less than or equal to half the available FIFO size, double-buffering will automatically be enabled for the endpoint.
15:8	Reserved	Must write reset value.
7:0	IMAXP	**IN Maximum Packet Size.** These bits set the maximum packet size of the IN endpoint in increments of 8 bytes. A value of 128 (0x80) is interpreted as 1023 bytes. Values above the available FIFO size divided by 8 (which depends on the endpoint settings) are illegal and should not be written. If the maximum packet size is set to less than or equal to half the available FIFO size, double-buffering will automatically be enabled for the endpoint.

Notes:
1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Register 38.20. USBn_EPx_EPCONTROL: Endpoint Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	AUTOCLREN	OISOEN	ODMAEN	ODMAMD	Reserved				OCLRDT	OSTSTLI	OSDSTL	OFIFOFL	ODERRF	OORF	OFIFOFF	OPRDYI
Type	RW	RW	RW	RW	R				RW	RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	AUTOSETEN	IISOEN	DIRSEL	IDMAEN	FDTEN	SPLITEN	Reserved			ICLRDT	ISTSTLI	ISDSTL	IFIFOFL	IURF	IFIFONEF	IPRDYI
Type	RW	RW	RW	RW	RW	RW	R			RW	RW	RW	RW	RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

USB0_EP1_EPCONTROL = 0x4001_8890
USB0_EP2_EPCONTROL = 0x4001_8910
USB0_EP3_EPCONTROL = 0x4001_8990
USB0_EP4_EPCONTROL = 0x4001_8A10

Table 38.23. USBn_EPx_EPCONTROL Register Bit Descriptions

Bit	Name	Function
31	AUTOCLREN	**OUT Endpoint OPRDYI Auto-Clear Enable.** Writing a 1 to this bit will cause the hardware to automatically clear the OPRDYI bit when the last byte of a maximum-sized packet is read from the OUT FIFO. This bit must be set to 1 if the OUT Endpoint is the source for a DMA channel. 0: The OPRDYI bit is not automatically cleared by hardware. 1: The OPRDYI bit is automatically cleared by hardware.
30	OISOEN	**OUT Isochronous Transfer Enable.** This bit enables and disables Isochronous transfers on the endpoint. 0: Configure the endpoint for Bulk/Interrupt transfers. 1: Configure the endpoint for Isochronous transfers.

Notes:
1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Table 38.23. USBn_EPx_EPCONTROL Register Bit Descriptions

Bit	Name	Function
29	ODMAEN	**OUT Endpoint DMA Enable.** When enabled, a DMA transaction is requested when a packet is available in the FIFO. 0: Disable the DMA request for the OUT endpoint. 1: Enable the DMA request for the OUT endpoint.
28	ODMAMD	**OUT Endpoint DMA Mode.** This bit has no effect when ODMAEN = 0. An interrupt will be generated for every Isochronous packet received, regardless of the setting of this bit. 0: Automatic DMA service is requested on the last packet of the transfer until less than four bytes remain in the packet. At this time, an interrupt is generated. The firmware must read or write the last few bytes of the packet, if any remain. 1: No DMA service is requested on the last packet of the transfer. When the DMA recognizes the last packet, an interrupt is generated. The firmware must handle the entirety of the last packet.
27:24	Reserved	Must write reset value.
23	OCLRDT	**OUT Clear Data Toggle.** Software should write 1 to this bit to reset the OUT endpoint data toggle to 0. This bit always reads 0. 0: Do not reset the OUT data toggle. 1: Reset the OUT data toggle.
22	OSTSTLI	**OUT Sent Stall Interrupt Flag.** Hardware sets this bit to 1 when a STALL handshake signal is transmitted. This interrupt flag must be cleared by software. Read: 0 : A stall condition has not been sent since this bit was last cleared. 1 : A stall condition has been sent since this bit was last cleared. Write: 0 : Clear the interrupt. 1 : No effect. 0: Read: A stall condition has not been sent since this bit was last cleared. Write: Clear the interrupt. 1: Read: A stall condition has been sent since this bit was last cleared. Write: No effect.
21	OSDSTL	**OUT Send Stall.** Software should write 1 to this bit to generate a STALL handshake. Software should write 0 to this bit to terminate the STALL signal. This bit has no effect in Isochronous mode. 0: Stop sending a stall. 1: Generate a stall.

Notes:
1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Table 38.23. USBn_EPx_EPCONTROL Register Bit Descriptions

Bit	Name	Function
20	OFIFOFL	**OUT FIFO Flush.** Writing a 1 to this bit flushes the next packet to be read from the OUT endpoint FIFO. The FIFO pointer is reset and the OPRDYI bit is cleared. Multiple packets must be flushed individually. Hardware resets the OFIFOFL bit to 0 when the flush is complete. Note: If data for the current packet has already been read from the FIFO, the OFIFOFL bit should not be used to flush the packet. Instead, the FIFO should be read manually.
19	ODERRF	**OUT Data Error Flag.** In Isochronous mode, this bit is set by hardware if a received packet has a CRC or bit stuffing error. It is cleared when software clears OPRDYI. This bit is only valid in Isochronous mode. 0: A CRC or bit-stuff error has not occurred. 1: A CRC or bit-stuff error occurred.
18	OORF	**OUT FIFO Overrun Flag.** This bit is set by hardware when an incoming data packet cannot be loaded into the OUT endpoint FIFO. This bit is only valid in Isochronous mode and must be cleared by software. 0: No data overrun. 1: A data packet was lost because of a full FIFO since this flag was last cleared.
17	OFIFOFF	**OUT FIFO Full.** This bit indicates the contents of the OUT FIFO. If double buffering is enabled, the FIFO is full when the FIFO contains two packets. 0: The OUT endpoint FIFO is not full. 1: The OUT endpoint FIFO is full.
16	OPRDYI	**OUT Packet Ready.** Hardware sets this bit to 1 and, depending on the DMA settings, generates an interrupt when a data packet is available. Software should clear this bit after each data packet is unloaded from the OUT endpoint FIFO. This bit is automatically cleared by hardware if AUTOCLREN is 1 for maximum-size packets. 0: A data packet is not available. 1: A data packet is available.
15	AUTOSETEN	**IN Endpoint IPRDYI Automatic Set Enable.** Writing a 1 to this bit will cause the hardware to automatically set the IPRDYI bit when the last byte of a maximum-size packet is written to the IN FIFO. This bit must be set to 1 if the IN Endpoint is the destination of a DMA channel. 0: The IPRDYI bit is not automatically set by hardware. 1: The IPRDYI bit is automatically set by hardware.
14	IISOEN	**IN Isochronous Transfer Enable.** This bit enables and disables Isochronous transfers on the endpoint. 0: Configure the endpoint for Bulk/Interrupt transfers. 1: Configure the endpoint for Isochronous transfers.

Notes:
1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Table 38.23. USBn_EPx_EPCONTROL Register Bit Descriptions

Bit	Name	Function
13	DIRSEL	**Endpoint Direction Select.** This bit is valid only when the selected FIFO is not split (SPLIT = 0). 0: Select the endpoint direction as OUT. 1: Select the endpoint direction as IN.
12	IDMAEN	**IN Endpoint DMA Enable.** When enabled, a DMA transaction is requested when there is room for a packet in the FIFO. 0: Disable the DMA request for the IN endpoint. 1: Enable the DMA request for the IN endpoint.
11	FDTEN	**Force Data Toggle Enable.** 0: The endpoint data toggle switches only when an ACK is received following a data packet transmission. 1: The endpoint data toggle is forced to switch after every data packet is transmitted, regardless of ACK reception.
10	SPLITEN	**FIFO Split Enable.** Writing a 1 to this bit will split the endpoint FIFO. In this case, the upper half of the FIFO is used by the IN endpoint; the lower half of the FIFO is used by the OUT endpoint. 0: Do not split the endpoint FIFO. 1: Split the endpoint FIFO.
9:7	Reserved	Must write reset value.
6	ICLRDT	**IN Clear Data Toggle.** Software should write 1 to this bit to reset the IN endpoint data toggle to 0. This bit always reads 0. 0: Do not reset the IN data toggle. 1: Reset the IN data toggle.
5	ISTSTLI	**IN Sent Stall Interrupt Flag.** Hardware sets this bit to 1 when a STALL handshake signal is transmitted, the FIFO is flushed, and the IPRDYI bit cleared. This interrupt flag must be cleared by software. Read: 0 : A stall condition has not been sent since this bit was last cleared. 1 : A stall condition has been sent since this bit was last cleared. Write: 0 : Clear the interrupt. 1 : No effect. 0: Read: A stall condition has not been sent since this bit was last cleared. Write: Clear the interrupt. 1: Read: A stall condition has been sent since this bit was last cleared. Write: No effect.

Notes:
1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Table 38.23. USBn_EPx_EPCONTROL Register Bit Descriptions

Bit	Name	Function
4	ISDSTL	**IN Send Stall.** Software should write 1 to this bit to generate a STALL handshake in response to an IN token. Software should write 0 to this bit to terminate the STALL signal. This bit has no effect in Isochronous mode. 0: Stop sending a stall. 1: Generate a stall.
3	IFIFOFL	**IN FIFO Flush.** Writing a 1 to this bit flushes the next packet to be transmitted from the IN Endpoint FIFO. The FIFO pointer is reset and the IPRDYI bit is cleared. If the FIFO contains multiple packets, software must write 1 to IFIFOFL for each packet. Hardware resets the IFIFOFL bit to 0 when the FIFO flush is complete.
2	IURF	**IN FIFO Underrun Flag.** This bit must be cleared by software. The function of this bit depends on the IN Endpoint mode: Isochronous : Set when a zero-length packet is sent after an IN token is received while bit IPRDYI = 0. Interrupt/Bulk : Set when a NAK is returned in response to an IN token. 0: Underrun has not occurred. 1: Underrun occurred.
1	IFIFONEF	**IN FIFO Not Empty Flag.** 0: The IN Endpoint FIFO is empty. 1: The IN Endpoint FIFO contains one or more packets.
0	IPRDYI	**IN Packet Ready Indicator.** Software should write 1 to this bit after loading a data packet into the IN Endpoint FIFO. This bit is automatically set by hardware if AUTOSETEN is 1 for maximum-size packets. Hardware clears IPRDYI due to any of the following: 1) A data packet is transmitted. 2) There is an open FIFO packet slot. 3) If the endpoint is in Isochronous Mode (IISOEN = 1) and ISOUPD = 1, IPRDYI will read 0 until the next SOF is received. Note: Depending on the DMA settings, an interrupt (if enabled) will be generated when hardware clears IPRDYI as a result of a packet being transmitted. 0: The packet has been sent or there is an open FIFO slot. 1: A packet is loaded in the FIFO.

Notes:
1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Register 38.21. USBn_EPx_EPCOUNT: Endpoint Data Count

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved						COUNT									
Type	R						R									
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Addresses

USB0_EP1_EPCOUNT = 0x4001_88A0
USB0_EP2_EPCOUNT = 0x4001_8920
USB0_EP3_EPCOUNT = 0x4001_89A0
USB0_EP4_EPCOUNT = 0x4001_8A20

Table 38.24. USBn_EPx_EPCOUNT Register Bit Descriptions

Bit	Name	Function
31:10	Reserved	Must write reset value.
9:0	COUNT	**Endpoint OUT Data Count.** This 10-bit number indicates the number of data bytes in the last received packet in the OUT endpoint FIFO. This number is only valid while bit OPRDYI is a 1.

Notes:
1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.

Register 38.22. USBn_EPx_EPFIFO: Endpoint Data FIFO Access

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	FIFO[31:16]															
Type	RW															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	FIFO[15:0]															
Type	RW															
Reset	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X	X

Register ALL Access Addresses

USB0_EP1_EPFIFO = 0x4001_88B0
USB0_EP2_EPFIFO = 0x4001_8930
USB0_EP3_EPFIFO = 0x4001_89B0
USB0_EP4_EPFIFO = 0x4001_8A30

Table 38.25. USBn_EPx_EPFIFO Register Bit Descriptions

Bit	Name	Function
31:0	FIFO	**Endpoint Data FIFO.** Any bytes read from this register will be loaded out of the OUT data FIFO and any bytes written to this register will be loaded into the IN data FIFO for the endpoint. Any right-justified word, half-word, or byte operation is allowed. Non right-justified accesses will write or return incorrect data and should not be used. Multi-byte reads and writes occur least-significant byte first.

Notes:
1. Accessing most USB registers takes several system clock cycles, so the APB is stalled while an access is performed.
2. Reads of this register modify the state of hardware. Debug logic should take care when reading this register.

38.19. USBn_EPx Register Memory Map

Table 38.26. USBn_EPx Memory Map

Register Name	ALL Offset	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
USBn_EPx_EPMPSIZE	0x0	ALL	Reserved								OMAXP								Reserved								IMAXP							
USBn_EPx_EPCONTROL	0x10	ALL	AUTOCLREN	OISOEN	ODMAEN	ODMAMD	Reserved				OCLRDT	OSTSTLI	OSDSTL	OFIFOFL	ODERRF	OORF	OFIFOFF	OPRDYI	AUTOSETEN	IISOEN	DIRSEL	IDMAEN	FDTEN	SPLITEN	Reserved		ICLRDT	ISTSTLI	ISDSTL	IFIFOFL	IURF	IFIFONEF	IPRDYI	

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: USB0_EP1 = 0x4001_8880, USB0_EP2 = 0x4001_8900, USB0_EP3 = 0x4001_8980, USB0_EP4 = 0x4001_8A00

Table 38.26. USBn_EPx Memory Map

Register Name	ALL Offset	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
USBn_EPx_EPCOUNT	0x20	ALL	Reserved																						COUNT									
USBn_EPx_EPFIFO	0x30	ALL	FIFO																															

Notes:

1. The "ALL Offset" refers to the address offset of the ALL access method for a register, this offset should be referenced to the base address for the block. For example, if a register block has a base address of 0x4001_0000 and the ALL offset is specified to be 0xA4, the register's absolute ALL access address is located at 0x4001_00A0 in the address map. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. The register with ALL access at 0x4001_00A0 may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.
2. The base addresses for this register block are: USB0_EP1 = 0x4001_8880, USB0_EP2 = 0x4001_8900, USB0_EP3 = 0x4001_8980, USB0_EP4 = 0x4001_8A00

39. Voltage Supply Monitor (VMON0)

This section describes the Voltage Supply Monitor (VMON) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the VMON block, which is used by all device families covered in this document.

39.1. Voltage Supply Monitor Features

The Voltage Supply Monitor module includes the following features:

- Main supply “VDD Low” (VDD below the early warning threshold) notification.
- Holds the device in reset if the main VDD supply drops below the VDD Reset threshold.
- VREGIN divided by 4 (VREGIN/4) supply “VREGIN Low” notification.

The Voltage Supply Monitor allows devices to function in known, safe operating conditions without the need for external hardware.

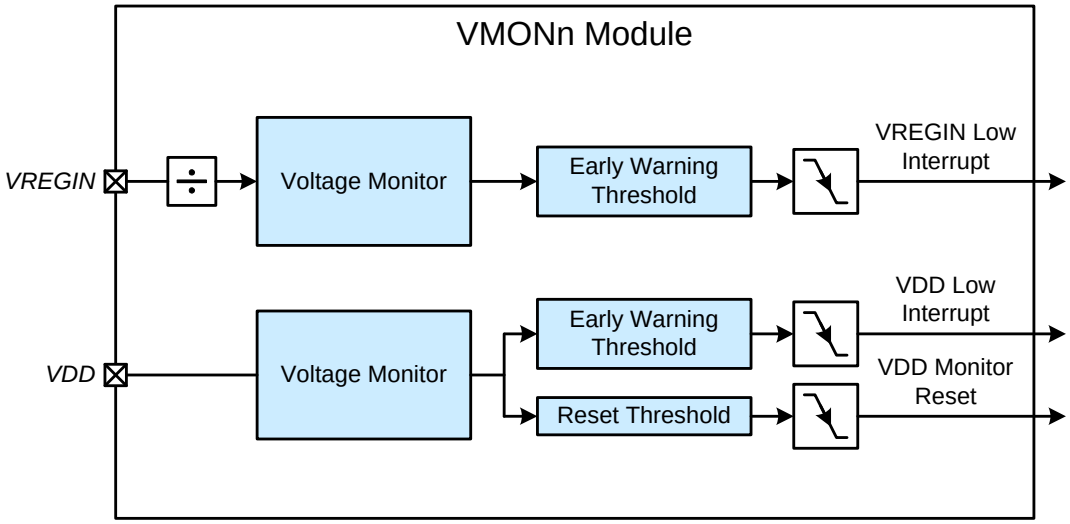


Figure 39.1. Voltage Supply Monitor Block Diagram

39.2. VDD Supply Monitoring

The VDD supply monitor senses the voltage on the device VDD supply and can generate an interrupt or reset if the supply drops below the corresponding thresholds. This monitor is enabled and enabled as a reset source after initial power-on to protect the device until VDD is an adequate and stable voltage.

When enabled and selected as a reset source, any power down transition or power irregularity that causes VDD to drop below the reset threshold will drive the RESET pin low and hold the core in a reset state. When VDD returns to a level above the reset threshold, the monitor will release the core from the reset state. The reset status can then be read using the device reset sources module. The VDD monitor reset threshold status can be read using the VDDRSTF flag. The power-on reset delay (t_{POR}) is not incurred after a supply monitor reset.

The contents of standard RAM is invalid after a VDD monitor reset. Any data stored in retention RAM will be valid unless the supply drops enough for the device to detect a power-on reset. Firmware should check and interpret the flags in the device reset sources module before relying on retention RAM, to ensure that a power-on reset did not occur.

The enable state of the VDD supply monitor and its selection as a reset source is not altered by device resets. For example, if the VDD supply monitor is de-selected as a reset source and disabled by software, and then firmware performs a software reset, the VDD supply monitor will remain disabled and de-selected after the reset.

To protect the integrity of flash contents, the VDD supply monitor must be enabled and selected as a reset source if software contains routines that erase or write flash memory. If the VDD supply monitor is not enabled, any erase or write performed on flash memory will be ignored.

The VDD monitor also includes a VDD-is-low interrupt. Hardware clears the VDDLI interrupt flag when VDD drops below the early warning threshold. This 1-to-0 transition on VDDLI can generate an interrupt when the VDD low interrupt enable (VDDLIEN) bit is set to 1. The early warning interrupt can be used to save any data in the retention RAM, if supported on the device, and otherwise prepare the system for power down.

The high threshold enable (VDDHITHEN) bit can increase the VDD monitor reset and early warning thresholds by approximately 300 mV, if appropriate for the system. This setting is recommended when operating at faster AHB clock speeds. The device data sheet contains more information.

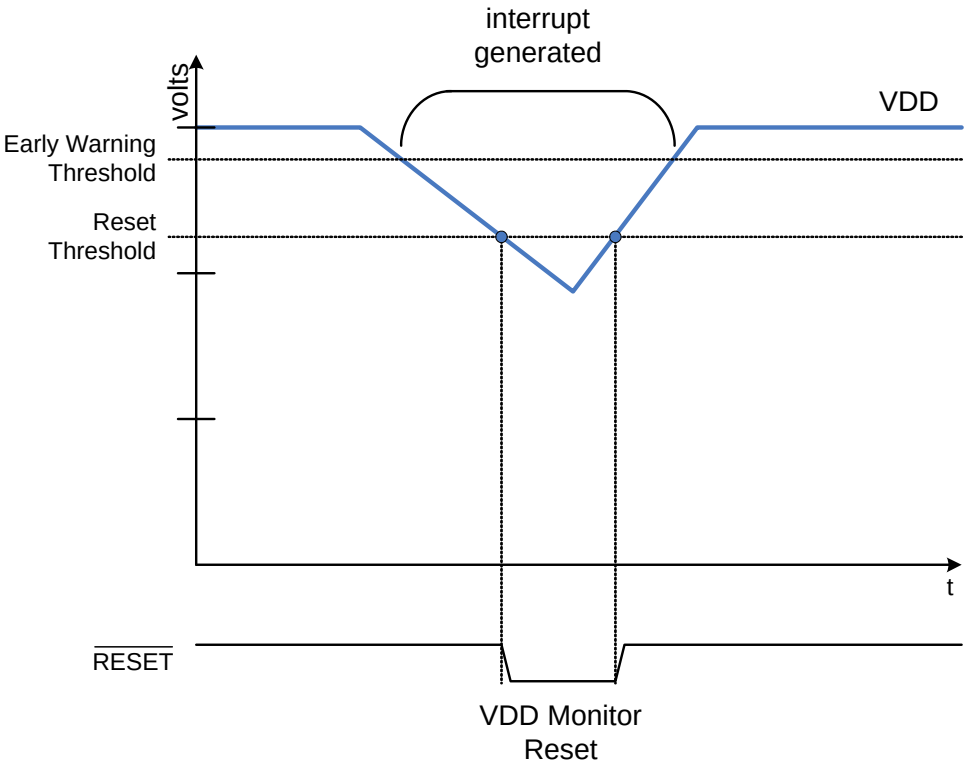


Figure 39.2. VDD Supply Monitor Thresholds

39.2.1. Enabling the VDD Monitor

The VDD supply monitor must be enabled before selecting it as a reset source. Selecting the VDD supply monitor as a reset source before it has stabilized may generate a system reset. In systems where this reset would be undesirable, a delay should be introduced between enabling the VDD supply monitor and selecting it as a reset source. No delay should be introduced in systems where software contains routines that erase or write flash memory. The procedure for enabling the VDD supply monitor and selecting it as a reset source is:

1. Enable the VDD supply monitor (VMONEN = 1).
2. Wait for the VDD supply monitor to stabilize (optional).
3. Select the VDD monitor as a reset source in the device reset sources module.

39.3. VREGIN Pin Monitoring

In addition to monitoring the VDD supply, the VMON module can also monitor the voltage on the VREGIN pin. The VREGIN pin sense enable (VREGINSEN) bit enables this feature in the VMON module. The VREGIN voltage sensing feature of the voltage regulator module (VREGn) must also be enabled in order to monitor the voltage in the VMON module.

When VREGIN pin voltage sensing is enabled, hardware clears the VREGINLI flag when the VREGIN pin voltage divided by 4 drops below the early warning threshold specified in the device data sheet. This 1-to-0 transition on the VREGINLI flag can also generate an interrupt if VREGINLIEN is set to 1.

Firmware can use this feature on devices that support the USB module to also monitor the voltage on the VBUS pin in bus-powered systems.

39.4. VMON0 Registers

This section contains the detailed register descriptions for VMON0 registers.

Register 39.1. VMON0_CONTROL: Module Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	VMONEN	Reserved														
Type	RW	R														
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved								VREGINLIEN	VDDLIEN	Reserved	VDDHITHEN	VDDLI	VDDRSTF	VREGINLI	VREGINSEN
Type	R								RW	RW	R	RW	R	R	R	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	X	X	X	0

Register ALL Access Address

VMON0_CONTROL = 0x4002_F000

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 39.1. VMON0_CONTROL Register Bit Descriptions

Bit	Name	Function
31	VMONEN	**VDD Supply Monitor Enable.** This bit enables the voltage supply monitor circuitry to monitor the VDD supply. 0: Disable the VDD supply monitor. 1: Enable the VDD supply monitor.
30:8	Reserved	Must write reset value.
7	VREGINLIEN	**VREGIN Low Interrupt Enable.** Enables the VREGIN low interrupt. The VREGIN supply monitor and VREGIN sensing must be enabled before setting this bit. 0: Disable the VREGIN low interrupt. 1: Enable the VREGIN low interrupt.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 39.1. VMON0_CONTROL Register Bit Descriptions

Bit	Name	Function
6	VDDLIEN	**VDD Low Interrupt Enable.** Enables the VDD low (early warning) interrupt. 0: Disable the VDD low interrupt. 1: Enable the VDD low interrupt.
5	Reserved	Must write reset value.
4	VDDHITHEN	**VDD High Threshold Enable.** Setting this bit to 1 will raise both the VDD low (early warning) and VDD reset thresholds by approximately 300 mV. 0: Use the standard VDD thresholds. 1: Use the high VDD thresholds.
3	VDDLI	**VDD Low Interrupt Flag.** This bit indicates whether the VDD supply voltage is above or below the early warning threshold. Falling below this threshold will generate an interrupt, if the VDD low interrupt is enabled. 0: The VDD voltage is below the early warning threshold. 1: The VDD voltage is above the early warning threshold.
2	VDDRSTF	**VDD Reset Threshold Status Flag.** This bit indicates whether the VDD supply voltage is above or below the reset threshold. If the VDD supply monitor is enabled as a reset source and VDD falls below the VDD reset threshold, the device will be reset. 0: The VDD voltage is below the VDD reset threshold. 1: The VDD voltage is above the VDD reset threshold.
1	VREGINLI	**VREGIN Low Interrupt Flag.** When the VREGIN supply monitor and VREGIN sensing are enabled, this bit indicates whether the input voltage is above or below the interrupt threshold. When the voltage is low, this bit will read 0, and if enabled, a VREGIN low interrupt will be generated. 0: VREGIN is not above the interrupt threshold. 1: VREGIN is above the interrupt threshold.
0	VREGINSEN	**VREGIN Supply Monitor Enable.** This bit enables the voltage supply monitor circuitry to monitor the VREGIN supply. VREGIN sensing must also be enabled in the voltage regulator. 0: Disable the VREGIN supply monitor. 1: Enable the VREGIN supply monitor.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

39.5. VMON0 Register Memory Map

Table 39.2. VMON0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
VMON0_CONTROL	0x4002_F000	ALL | SET | CLR	VMONEN	Reserved																							VREGINLIEN	VDDLIEN	Reserved	VDDHITHEN	VDDLI	VDDRSTF	VREGINLI	VREGINSEN

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

40. Voltage Reference and Temperature Sensor (VREF0)

This section describes the Voltage Reference and Temperature Sensor (VREF) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the VREF block, which is used by all device families covered in this document.

40.1. Voltage Reference Features

The Voltage Reference module includes the following features:

- Two programmable settings: 1.2 V and 2.4 V.
- The voltage reference can be used internally while driven on the VREF pin.
- Temperature sensor provides a voltage output that can be measured by an ADC module.

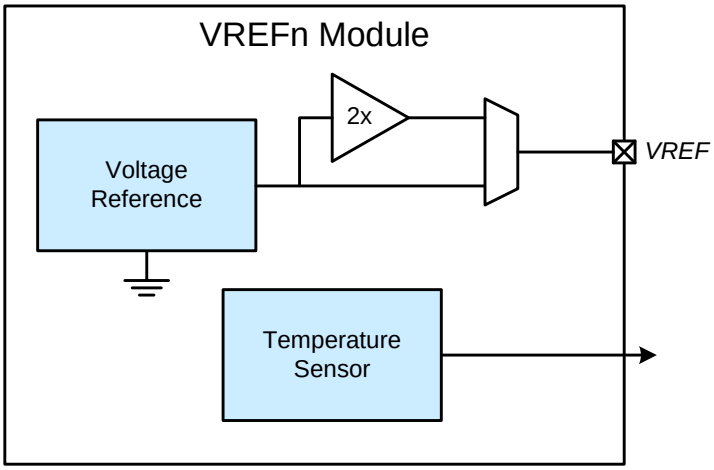


Figure 40.1. Voltage Reference Block Diagram

40.2. Functional Description

The Voltage Reference module contains both a precision bandgap voltage reference generator and a temperature sensor. The reference voltage output is available on the VREF pin. It can be used internally by analog peripherals such as ADCs, and it can also serve as a reference for external circuitry. There are two nominal output levels: 1.2 V and 2.4 V, which allows the VREF to be used with different supply voltages.

40.2.1. Voltage Reference Operation

The internal voltage reference can be enabled using the VREFEN bin in the CONTROL register. When enabled, the internal voltage reference node is available to any analog peripherals which have a VREF input. When disabled, the reference supply is completely shut down.

The VREF2X bit in the CONTROL register can be used to double the output voltage of the reference from 1.2 V to 2.4 V (nominal). The lower setting is ideal for applications where the supply voltage may be at the lower end of the valid supply range.

The reference voltage output must be sent to the VREF pin on the device when the reference is enabled. When sending the reference voltage to the VREF pin, the VREF pin should be configured for analog mode, and a minimum bypass capacitance of 0.1uF should be used between VREF and VSS.

40.2.2. Temperature Sensor Operation

The temperature sensor is also controlled from within the VREF module. The temperature sensor is enabled using the TEMPEN bit in the CONTROL register. When enabled, the temperature sensor output is available to any analog peripherals which have a temperature sensor input.

The temperature sensor output is linear with a positive slope; when temperature increases, the output voltage of the sensor increases. Refer to the electrical characteristics section of the specific device data sheet for temperature sensor parameters. Figure 40.2 illustrates the temperature sensor transfer function.

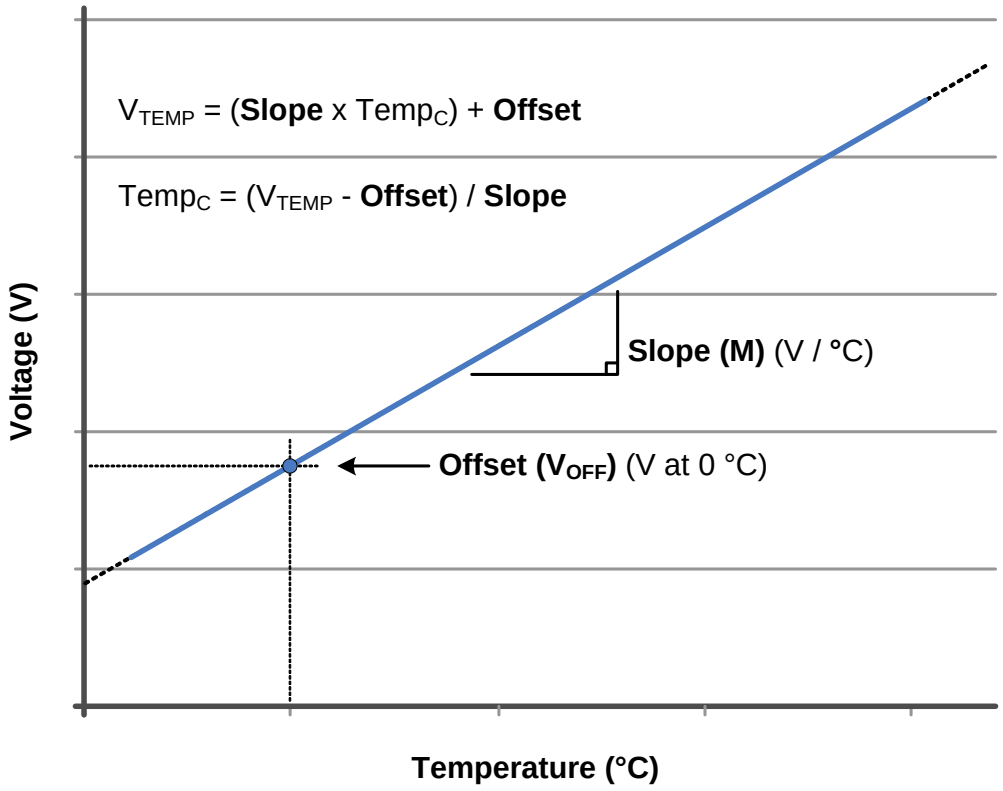


Figure 40.2. Temperature Sensor Transfer Function

40.3. VREF0 and Temperature Sensor Registers

This section contains the detailed register descriptions for VREF0 registers.

Register 40.1. VREF0_CONTROL: Voltage Reference Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	VREFEN	Reserved														
Type	RW	R														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved														TEMPEN	VREF2X
Type	R													RW	RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

VREF0_CONTROL = 0x4003_9010

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 40.1. VREF0_CONTROL Register Bit Descriptions

Bit	Name	Function
31	VREFEN	**Voltage Reference Enable.** 0: Disable the Voltage Reference. 1: Enable the Voltage Reference.
30:2	Reserved	Must write reset value.
1	TEMPEN	**Temperature Sensor Enable.** 0: Disable the temperature sensor. 1: Enable the temperature sensor.
0	VREF2X	**Voltage Reference Doubler.** Enables a 2x buffer on the reference to double the output voltage. 0: VREF output is nominally 1.2 V 1: VREF output is nominally 2.4 V

40.4. VREF0 Register Memory Map

Table 40.2. VREF0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
VREF0_CONTROL	0x4003_9010	ALL | SET | CLR	VREFEN	Reserved																													TEMPEN	VREF2X

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

41. Voltage Regulator (VREG0)

This section describes the Voltage Regulator (VREG) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the VREG block, which is used by all device families covered in this document. Note that features related to the USB VBUS detection are only available on the SiM3U1xx device family.

41.1. Voltage Regulator Features

The Voltage Regulator module includes the following features:

- 5 V input regulated down to 3.3 V and output on the VDD pin.
- VBUS invalid voltage sensing and interrupt for devices with USB.

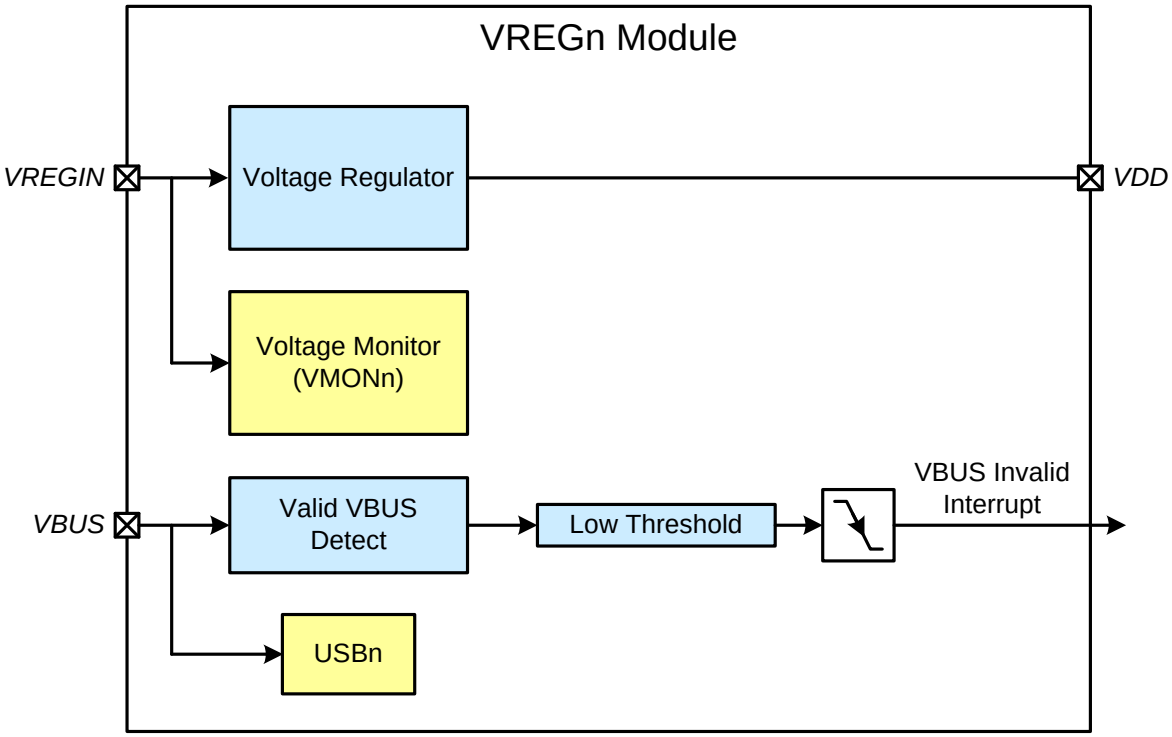


Figure 41.1. Voltage Regulator Block Diagram

41.2. Operational Modes

The VREG module has four operating modes: normal, suspend, sleep, and off. Table 41.1 shows a summary of each of these modes.

Table 41.1. Voltage Regulator Operational Modes

Regulator Mode	SUSEN Bit Value	BGDIS Bit Value	VREGDIS Bit Value	Power Consumption
Normal	0	0	0	highest
Suspend	1	0	0	low
Sleep	X	1	0	extremely low
Disabled	X	1	1	off

41.2.1. Normal Operation

The VREG voltage regulator is enabled by default. The regulator takes a 5 V input voltage and regulates down to 3.3 V.

The module also has the following additional features:

1. The VREGIN sense enable (SENSEEN) bit enables the internal VREGIN pin voltage sensing. VREGIN voltage sensing must be enabled in order to monitor or measure the VREGIN voltage in other modules, such as measuring the current voltage with the SARADCn module or using the VREGIN supply monitor and the VREGIN low interrupt flag in the VMONn module.
2. The VBUS invalid interrupt enable (VBUSIVLDIEN) bit causes the VREG module to generate an interrupt when the VBUS invalid interrupt (VBUSIVLDI) flag is set.

Figure 41.2 shows the typical connection diagram for the voltage regulator in normal, sleep, or suspend modes.

When the voltage regulator is not used, connect both the VREGIN and VDD pins to an external 1.8-3.6 V supply as shown in Figure 41.3.

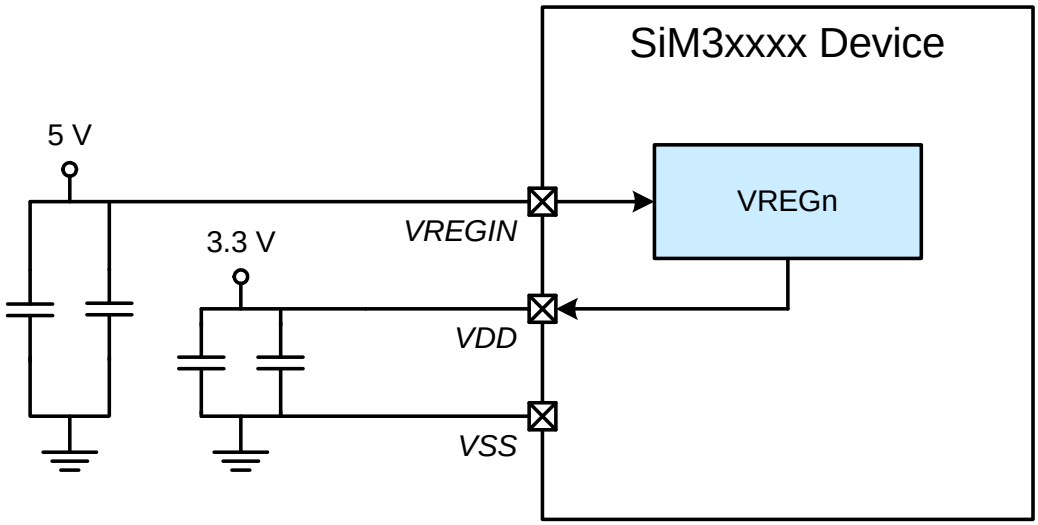


Figure 41.2. Typical Voltage Regulator Connection Diagram (Normal, Sleep, or Suspend Modes)

41.2.2. Suspend Mode

The VREG module supports a low quiescent current suspend mode for USB suspend or other applications that can tolerate a lower regulator bandwidth. Firmware can set the voltage regulator suspend enable (SUSEN) bit to enable suspend mode. When operating in this mode, the module lowers its internal bias currents to reduce the overall power consumed by the voltage regulator. These lower bias currents result in reduced dynamic performance, so the regulator will not respond as quickly to rapidly changing power requirements.

41.2.3. Sleep Mode

For low power applications that can tolerate reduced output voltage accuracy and load regulation, the VREG module offers a sleep mode. If firmware sets the BGDIS bit, the VREG module will regulate the voltage using a method that is more susceptible to process and temperature variations. In addition, the actual output voltage may drop substantially under heavy loads. The band gap should only be disabled for light loads (5 mA or less) or when the voltage regulator is disabled (VREGDIS = 1).

41.2.4. EXTREG0 Current Sensing

The current sensing feature in the EXTREG0 block relies on the VREG0 bandgap circuit to operate. It is necessary to clear the BGDIS bit in VREG0_CONTROL to 0 before using the EXTREG0 current sense feature. In this case, lower supply current will be achieved by also enabling the VREG0 block (even if it is not used).

41.2.5. Disabling the Voltage Regulator

If the regulator is not used at all by the application and the VDD supply is powered from an external 1.8-3.6 V supply, the VREG module should be disabled using the following steps:

1. Set the voltage regulator disable (VREGDIS) bit to disable the regulator.
2. Set the band gap disable (BGDIS) bit to disable the regulator band gap.

Figure 41.3 illustrates the typical connection diagram when the voltage regulator is not used.

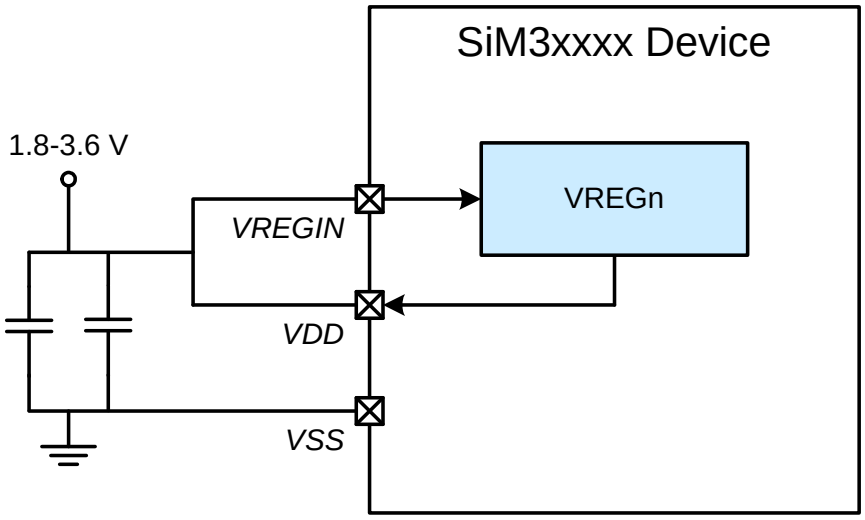


Figure 41.3. Typical Connection Diagram for Unused Voltage Regulator

41.3. Interrupts and Flags

For SiM3U1xx device, which also have the USB0 module, the VREG module includes VBUS detection functionality for determining whether the USB bus is connected to the device. The threshold for the VBUS detection is specified in the device data sheet. This flag (VBUSVLDF) indicates the instantaneous status of the VBUS voltage with regards to the valid threshold.

The VBUS invalid interrupt (VBUSIVLDI) flag indicates that the voltage on the VBUS pin dropped lower than the valid threshold since the last time the flag was cleared. The VREG module will generate an interrupt when VBUSIVLDI is set if this flag is enabled as an interrupt source (VBUSIVLDIEN = 1).

41.4. VREG0 Registers

This section contains the detailed register descriptions for VREG0 registers.

Register 41.1. VREG0_CONTROL: Module Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	VREGDIS	Reserved														
Type	RW	RW	R													
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved							VBUSIVLDIEN	VBUSIVLDI	SENSEEN	BGDIS	Reserved			SUSEN	VBUSVLDF
Type	R							RW	RW	RW	RW	R			RW	R
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	X

Register ALL Access Address

VREG0_CONTROL = 0x4004_0000

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 41.2. VREG0_CONTROL Register Bit Descriptions

Bit	Name	Function
31	VREGDIS	**Voltage Regulator Disable.** 0: Enable the voltage regulator. 1: Disable the voltage regulator.
30:9	Reserved	Must write reset value.
8	VBUSIVLDIEN	**VBUS Invalid Interrupt Enable.** 0: Disable the VBUS invalid interrupt. 1: Enable the VBUS invalid interrupt.
7	VBUSIVLDI	**VBUS Invalid Interrupt Flag.** Hardware sets this bit to 1 when the voltage on VBUS drops below the valid threshold. This flag may trigger an interrupt, if enabled (VBUSIVLDIEN = 1). This flag must be cleared by firmware. This flag still reports the VBUS valid status when the voltage regulator is disabled (VREGDIS = 1).

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

Table 41.2. VREG0_CONTROL Register Bit Descriptions

Bit	Name	Function
6	SENSEEN	**VREGIN Sense Enable.** This bit enables the VREGIN sense circuitry. Voltage sensing must be enabled in order to monitor or measure the VREGIN supply in other modules. 0: Disable VREGIN voltage sensing. 1: Enable VREGIN voltage sensing.
5	BGDIS	**Band Gap Disable.** When the band gap is enabled, the voltage regulator regulates closed-loop to an accurate voltage derived from the band gap. When the band gap is disabled (BGDIS = 1), the voltage regulator regulates open-loop and is dependent on process and temperature variations. The regulator output with the band gap disabled may drop significantly under heavy loads. 0: Enable the voltage regulator band gap. 1: Disable the voltage regulator band gap.
4:2	Reserved	Must write reset value.
1	SUSEN	**Voltage Regulator Suspend Enable.** 0: Disable regulator suspend mode. 1: Enable regulator suspend mode.
0	VBUSVLDF	**VBUS Valid Flag.** This flag still reports the VBUS status when the voltage regulator is disabled (VREGDIS = 1). 0: The current voltage on the VBUS pin is below the valid threshold. 1: The current voltage on the VBUS pin is above the valid threshold.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.

41.5. VREG0 Register Memory Map

Table 41.3. VREG0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
VREG0_CONTROL	0x4004_0000	ALL | SET | CLR	VREGDIS	Reserved																						VBUSIVLDIEN	VBUSIVLDI	SENSEEN	BGDIS	Reserved			SUSEN	VBUSVLDF

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

42. Watchdog Timer (WDTIMER0)

This section describes the Watchdog Timer (WDTIMER) module, and is applicable to all products in the following device families, unless otherwise stated:

- SiM3U1xx
- SiM3C1xx

This section describes version “A” of the WDTIMER block, which is used by all device families covered in this document.

42.1. Watchdog Timer Features

The Watchdog Timer module includes the following features:

- Programmable timeout interval.
- Optional interrupt to warn when the watchdog timer is nearing the reset trip value.
- Lock-out feature to prevent any modification until a system reset.

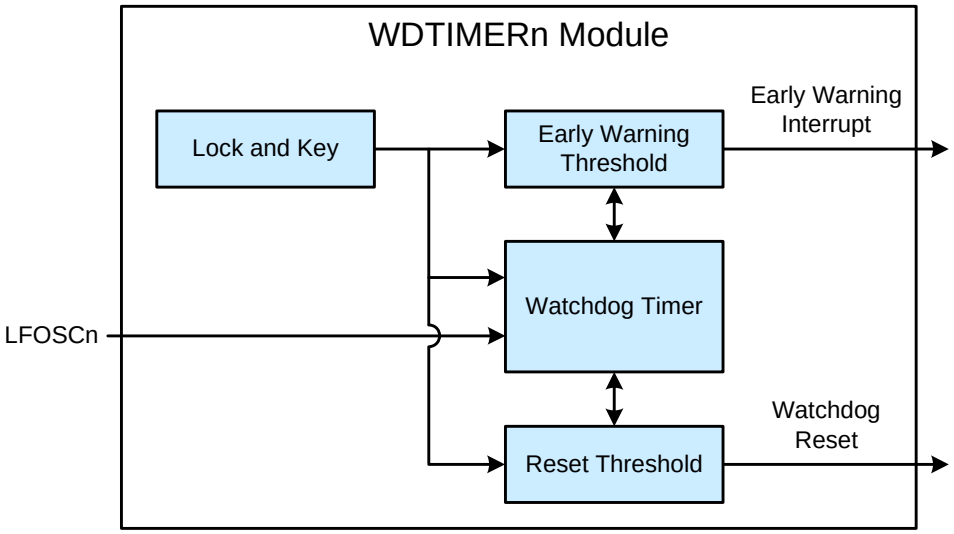


Figure 42.1. Watchdog Timer Block Diagram

42.2. Overview

The WDTIMER module includes a 16-bit timer, a programmable early warning interrupt, and a programmable reset period. The timer registers are protected from inadvertent access by an independent lock and key interface.

The watchdog timer runs from a low frequency oscillator (LFOSC0). Writes to the COMPARE register takes several clock cycles of the associated low frequency oscillator to complete.

42.3. Lock and Key Interface

The lock and key interface locks all WDTIMER register writes to protect firmware from inadvertently modifying any of the timer settings. The WDTKEY register contains the 8-bit KEY field. Firmware must write the attention value followed by one of the valid command values to this key field to interact with the module registers. The KEYSTS bit reports whether firmware must write the attention key or if the module is waiting for a command. Table 42.1 lists the valid key values.

Table 42.1. Valid Key Values

Command	KEY Value
Attention (ATTN)	0xA5
Write (WRITE)	0xF1
Reset (RESET)	0xCC
Disable (DISABLE)	0xDD
Start (START)	0xEE
Lock (LOCK)	0xFF

42.3.1. Write

To issue a write command, firmware must write the attention key followed by a write command to the KEY field. This command allows one write access to one of the watchdog timer registers. Once firmware completes the write, the interface relocks.

The PRIVSTS flag indicates whether the watchdog timer registers are unlocked for a write operation. The WDTIMER registers can be read at any time, regardless of the write status.

42.3.2. Reset

The reset command resets the timer back to zero. This action must be performed periodically to ensure the timer does not cause a system reset. The early warning interrupt can be used as a method to force the firmware to reset the timer.

Firmware must first write the attention key followed by the reset command to reset the timer.

42.3.3. Disable

The disable command stops the watchdog timer until the next system reset or until the module receives a Start command.

Firmware must write the attention key followed by the disable command to disable the timer.

42.3.4. Start

To issue a start command, firmware must write the attention key followed by a start command. The start command re-starts the watchdog timer if it has been stopped by a disable command.

42.3.5. Lock

The lock command prevents any disable commands from taking effect until the next system reset. This command ensures that firmware will not accidentally disable the timer. Firmware can still reset the timer and update the thresholds while the timer is locked.

Firmware can issue a lock command by writing the attention key followed by the lock command to the KEY field.

Note that the watchdog timer can still be disabled as a reset source by firmware, even if the timer is locked. To prevent this disable function, firmware should lock the RSTSRC0 registers by setting the CLKRSTL bit in the LOCK0_PERIPHLOCK0 register to 1.

42.4. Setting the Early Warning and Reset Thresholds

The early warning interrupt (EWTH) and reset (RTH) thresholds contain the values that can cause an interrupt or system reset. These 16-bit values are directly compared to the 16-bit timer. The early warning threshold can be used to remind firmware to periodically reset the timer.

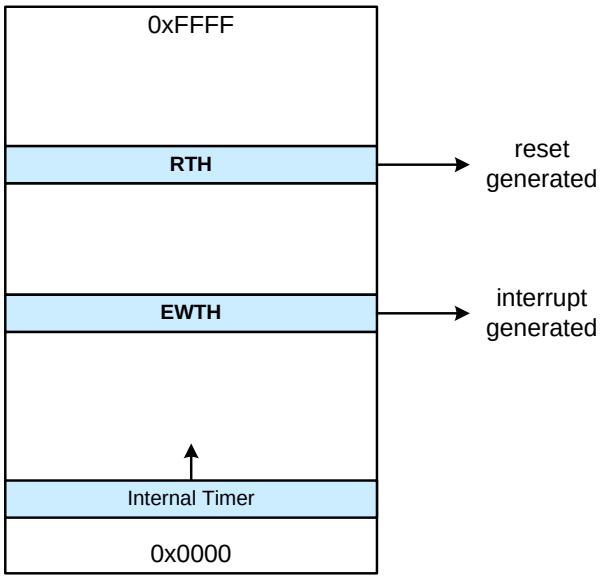


Figure 42.2. Early Warning and Reset Thresholds

Firmware can update these thresholds using the following procedure:

1. Write the attention key (0xA5) to the KEY field.
2. Write the write command (0xF1) to the KEY field.
3. Write the new EWTH and RTH values to the THRESHOLD register in one write operation.
4. (Optional) Poll on the UPDSTS flag to determine when the register update completes.

To write the threshold fields separately, firmware should use the following procedure:

1. Write the attention key (0xA5) to the KEY field.
2. Write the write command (0xF1) to the KEY field.
3. Write the new EWTH value to the THRESHOLD register.
4. Poll on the UPDSTS flag to determine when the register update completes.
5. Write the attention key (0xA5) to the KEY field.
6. Write the write command (0xF1) to the KEY field.
7. Write the new RTH value to the THRESHOLD register.

42.5. Interrupts and Flags

Hardware sets the early warning interrupt (EWI) flag when the timer reaches the value in the EWTH field. This flag can also cause an interrupt, if enabled (EWIEN = 1).

The reset threshold status (RTHF) flag indicates whether the timer has passed the RTH field value. A system reset can also occur when the timer passes this threshold, if enabled in the device reset sources module.

42.6. Debug Mode

Firmware can set the DBGMD bit to force the WDTIMER module to halt on a debug breakpoint. Clearing the DBGMD bit forces the WDTIMER module to continue operating while the core halts in debug mode.

42.7. WDTIMER0 Registers

This section contains the detailed register descriptions for WDTIMER0 registers.

Register 42.1. WDTIMER0_CONTROL: Module Control

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved														DBGMD	EWIEN
Type	R														RW	RW
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

Register ALL Access Address

WDTIMER0_CONTROL = 0x4003_0000

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 42.2. WDTIMER0_CONTROL Register Bit Descriptions

Bit	Name	Function
31:2	Reserved	Must write reset value.
1	DBGMD	**Watchdog Timer Debug Mode.** This bit determines the behavior of the watchdog timer when the device is halted during a debug operation. When cleared to 0, the watchdog timer continues to run during a debug halt (and will reset the device). When set to 1, the watchdog timer is halted during a debug halt operation.
0	EWIEN	**Early Warning Interrupt Enable.** 0: Disable the early warning interrupt (EWI). 1: Enable the early warning interrupt (EWI).

Notes:

1. Accessing any WDTIMER register takes several clock cycles of the associated low frequency oscillator. It will take at least 4 low frequency oscillator clock cycles to modify a bit and read back the modified value.
2. WDTIMER registers must be unlocked using the WDTKEY register in order to enable write access.

Register 42.2. WDTIMER0_STATUS: Module Status

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved											UPDSTS	RTHF	EWI	PRIVSTS	KEYSTS
Type	R											R	R	RW	R	R
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

WDTIMER0_STATUS = 0x4003_0010

This register also supports SET access at (ALL+0x4) and CLR access at (ALL+0x8)

Table 42.3. WDTIMER0_STATUS Register Bit Descriptions

Bit	Name	Function
31:5	Reserved	Must write reset value.
4	UPDSTS	**Watchdog Timer Threshold Update Status.** 0: An update completed or is not pending. The EWTH and RTH fields can be written. 1: An update of the threshold register is occurring. The EWTH and RTH fields should not be modified until hardware clears UPDSTS to 0.
3	RTHF	**Reset Threshold Flag.** 0: The counter is currently less than the reset threshold (RTH) value. 1: The counter is currently greater than or equal to the reset threshold (RTH) value.
2	EWI	**Early Warning Interrupt Flag.** This bit is set to 1 by hardware when an early warning match has occurred. Firmware may write the bit to 1 to manually trigger this interrupt. This bit must be cleared by firmware.
1	PRIVSTS	**Register Access Status.** 0: The watchdog timer registers are currently read-only. 1: A write transaction can be performed on the module registers.

Notes:

1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.
2. Accessing any WDTIMER register takes several clock cycles of the associated low frequency oscillator. It will take at least 4 low frequency oscillator clock cycles to modify a bit and read back the modified value.
3. WDTIMER registers must be unlocked using the WDTKEY register in order to enable write access.

Table 42.3. WDTIMER0_STATUS Register Bit Descriptions

Bit	Name	Function
0	KEYSTS	**Key Status.** 0: No keys have been processed by the interface. 1: The attention key has been received and the module is awaiting a command.

Notes:
1. This register contains interrupt flags. Firmware should only use the SET and CLR addresses when modifying interrupt flags to avoid conflicts with hardware.
2. Accessing any WDTIMER register takes several clock cycles of the associated low frequency oscillator. It will take at least 4 low frequency oscillator clock cycles to modify a bit and read back the modified value.
3. WDTIMER registers must be unlocked using the WDTKEY register in order to enable write access.

Register 42.3. WDTIMER0_THRESHOLD: Threshold Values

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	RTH															
Type	RW															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	EWTH															
Type	RW															
Reset	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Register ALL Access Address

WDTIMER0_THRESHOLD = 0x4003_0020

Table 42.4. WDTIMER0_THRESHOLD Register Bit Descriptions

Bit	Name	Function
31:16	RTH	**Reset Threshold.** When the counter value matches the value in RTH, the module will initiate a system reset if the watchdog timer is enabled as a reset source.
15:0	EWTH	**Early Warning Threshold.** When the counter value matches EWTH, the module will interrupt the core if watchdog timer early warning interrupt is enabled.

Notes:

1. Accessing any WDTIMER register takes several clock cycles of the associated low frequency oscillator. It will take at least 4 low frequency oscillator clock cycles to modify a bit and read back the modified value.
2. WDTIMER registers must be unlocked using the WDTKEY register in order to enable write access.

Register 42.4. WDTIMER0_WDTKEY: Module Key

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Name	Reserved															
Type	R															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Name	Reserved								KEY							
Type	R								W							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Register ALL Access Address

WDTIMER0_WDTKEY = 0x4003_0030

Table 42.5. WDTIMER0_WDTKEY Register Bit Descriptions

Bit	Name	Function
31:8	Reserved	Must write reset value.
7:0	KEY	**Watchdog Timer Key.** The watchdog timer key is used to perform certain functions on the watchdog timer, such as resetting the counter and locking the timer registers. The attention key (ATTN) must be written first, followed by a command key. The available commands keys are: WRITE, RESET, DISABLE, START, and LOCK. If the command key is not written within 32 APB clocks of writing the ATTN key, then the sequence must be started over with a new ATTN key write. 0xA5 : ATTN: Attention key to start the command sequence. 0xF1 : WRITE: Allow one write access to the WDT registers. 0xCC : RESET: Reset the WDT counter. 0xDD : DISABLE: Disable the WDT counter. 0xEE : START: Start the WDT counter. 0xFF : LOCK: Lock the WDT from any other writes until the next system reset. All other values are reserved.

Notes:

1. Accessing any WDTIMER register takes several clock cycles of the associated low frequency oscillator. It will take at least 4 low frequency oscillator clock cycles to modify a bit and read back the modified value.
2. WDTIMER registers must be unlocked using the WDTKEY register in order to enable write access.

42.8. WDTIMER0 Register Memory Map

Table 42.6. WDTIMER0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
WDTIMER0_CONTROL	0x4003_0000	ALL | SET | CLR	Reserved																														DBGMD	EWIEN
WDTIMER0_STATUS	0x4003_0010	ALL | SET | CLR	Reserved																											UPDSTS	RTHF	EWI	PRIVSTS	KEYSTS

Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

Table 42.6. WDTIMER0 Memory Map

Register Name	ALL Address	Access Methods	Bit 31	Bit 30	Bit 29	Bit 28	Bit 27	Bit 26	Bit 25	Bit 24	Bit 23	Bit 22	Bit 21	Bit 20	Bit 19	Bit 18	Bit 17	Bit 16	Bit 15	Bit 14	Bit 13	Bit 12	Bit 11	Bit 10	Bit 9	Bit 8	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
WDTIMER0_THRESHOLD	0x4003_0020	ALL	RTH																EWTH															
WDTIMER0_WDTKEY	0x4003_0030	ALL	Reserved																								KEY							

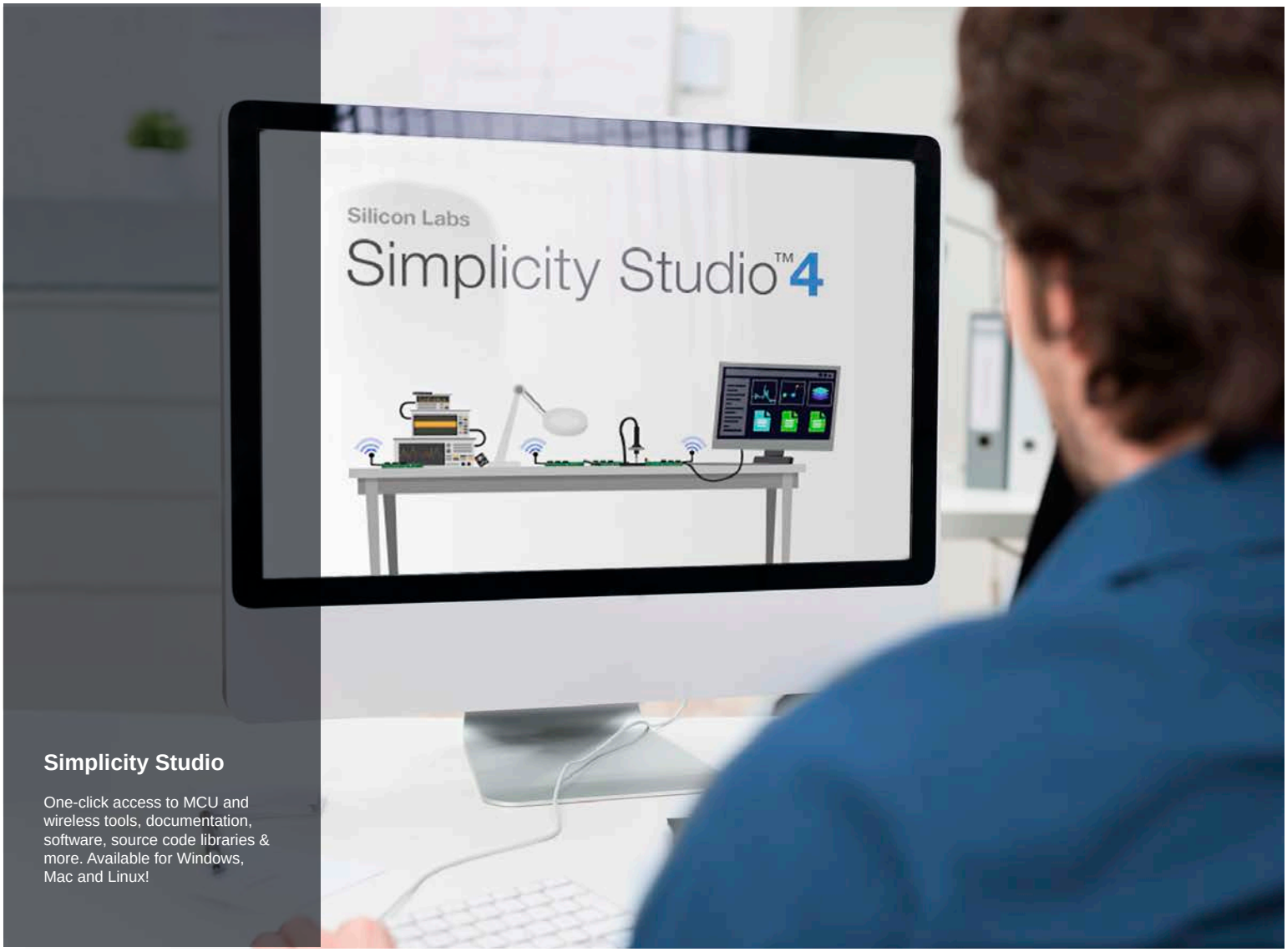
Notes:

1. The "ALL Address" refers to the absolute address of the ALL access method for a register. A register may also support SET, CLR, and MSK access methods, as indicated by the "Access Methods" column. SET, CLR and MSK addresses are offset from the ALL address by 4, 8 and 12 bytes, respectively. For example, a register whose ALL address is located at 0x4001_00A0 in the address map may have a SET address at 0x4001_00A4, a CLR address at 0x4001_00A8, and a MSK address at 0x4001_00AC.

Document Change List

Revision 0.8 to Revision 1.0

- Front Page: Updated block diagram.
- “2. Memory Organization”: Clarified flash lock details.
- Updated and clarified RTC timer clock output. The RTC output is now referred to as "RTC0TCLK".
- Register bit descriptions updated.
- “8. Port I/O Configuration”:
  - Corrected bit names: PBxPOEN = "PBPDEN.x", PBxNOEN = "PBNDEN.x"
  - Corrected table headers to include SiM3U1xx and SiM3C1xx devices.
  - Clarified EMIF pin configuration.
  - Added procedure for accessing high-drive ports.
- “9. Power”: Clarified power modes and updated descriptions.
- “11. Device Identification (DEVICEID0) and Universally Unique Identifier”: Updated register descriptions to describe enhanced device identification capabilities.
- “12. Advanced Encryption Standard (AES0)”:
  - Corrected bit names: EINTEN = ERRIEN, BYPASS = BEN
  - Updated description to reflect correct bit names.
- “18. External Memory Interface (EMIF0)”:
  - Corrected Address ranges and accessible memory size.
  - Made diagrams and descriptions more specific to device.
  - Clarified EMIF operating modes.
- “19. External Oscillator (EXTOSC0)”:
  - Clarified external crystal load cap description.
  - Corrected maximum external crystal frequency to 30 MHz.
- “20. External Regulator (EXTVREG0)”: Added note about current sensing function.
- “21. Flash Controller (FLASHCTRL0)”:
  - Removed references to VDD in favor of "Supply".
  - Corrected number of writes between erases to one.
- “22. Inter-Integrated Circuit Bus (I2C0 and I2C1)”: Clarifications to I2C operation throughout this section.
- “31. Real Time Clock and Low Frequency Oscillator (RTC0)”: Extensive updates to clarify RTC output functions.
- “32. SAR Analog-to-Digital Converter (SARADC0 and SARADC1)”: Added usage cases and clarified operational descriptions.
- “35. Timers (TIMER0 and TIMER1)”: Added descriptions for One Shot Mode.
- “41. Voltage Regulator (VREG0)”: Added text about EXTREG current sensing.



Simplicity Studio

One-click access to MCU and wireless tools, documentation, software, source code libraries & more. Available for Windows, Mac and Linux!

IoT Portfolio
www.silabs.com/IoT

SW/HW
www.silabs.com/simplicity

Quality
www.silabs.com/quality

Support and Community
community.silabs.com

Disclaimer
Silicon Labs intends to provide customers with the latest, accurate, and in-depth documentation of all peripherals and modules available for system and software implementers using or intending to use the Silicon Labs products. Characterization data, available modules and peripherals, memory sizes and memory addresses refer to each specific device, and "Typical" parameters provided can and do vary in different applications. Application examples described herein are for illustrative purposes only. Silicon Labs reserves the right to make changes without further notice and limitation to product information, specifications, and descriptions herein, and does not give warranties as to the accuracy or completeness of the included information. Silicon Labs shall have no liability for the consequences of use of the information supplied herein. This document does not imply or express copyright licenses granted hereunder to design or fabricate any integrated circuits. The products are not designed or authorized to be used within any Life Support System without the specific written consent of Silicon Labs. A "Life Support System" is any product or system intended to support or sustain life and/or health, which, if it fails, can be reasonably expected to result in significant personal injury or death. Silicon Labs products are not designed or authorized for military applications. Silicon Labs products shall under no circumstances be used in weapons of mass destruction including (but not limited to) nuclear, biological or chemical weapons, or missiles capable of delivering such weapons.

Trademark Information
Silicon Laboratories Inc.® , Silicon Laboratories®, Silicon Labs®, SiLabs® and the Silicon Labs logo®, Bluegiga®, Bluegiga Logo®, Clockbuilder®, CMEMS®, DSPLL®, EFM®, EFM32®, EFR, Ember®, Energy Micro, Energy Micro logo and combinations thereof, "the world's most energy friendly microcontrollers", Ember®, EZLink®, EZRadio®, EZRadioPRO®, Gecko®, ISOmodem®, Precision32®, ProSLIC®, Simplicity Studio®, SiPHY®, Telegesis, the Telegesis Logo®, USBXpress® and others are trademarks or registered trademarks of Silicon Labs. ARM, CORTEX, Cortex-M3 and THUMB are trademarks or registered trademarks of ARM Holdings. Keil is a registered trademark of ARM Limited. All other products or brand names mentioned herein are trademarks of their respective holders.

SILICON LABS

Silicon Laboratories Inc.
400 West Cesar Chavez
Austin, TX 78701
USA

http://www.silabs.com